

ALTO
Internet-Draft
Intended status: Standards Track
Expires: 16 September 2020

J. Zhang
Tongji University
K. Gao
Sichuan University
J. Wang

Q. Xiang
Y.R. Yang
Yale University
15 March 2020

ALTO Extension: Flow-based Cost Query
draft-gao-alto-fcs-07

Abstract

ALTO cost maps and endpoint cost services map a source-destination pair into a cost value. However, current filter specifications, which define the set of source-destination pairs in an ALTO query, have two limitations: 1) Only very limited address types are supported (IPv4 and IPv6), which is not sufficient to uniquely identify a flow in networks with fine-grained routing, such as the emerging Software Defined Networks; 2) The base ALTO protocol only defines filters enumerating all sources and all destinations, leading to redundant information in the response; 3) Cannot distinguish transmission types of flows in the query, which makes the server hard to respond the accurate resource consumption. To address these three issues, this document extends the base ALTO protocol with a more fine-grained filter type which allows ALTO clients to select only the concerned source-destination pairs and announce the flow-specific information like data transmission type, and a more expressive address space which allows ALTO clients to make queries beyond the limited IP addresses.

Status of This Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any

time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 16 September 2020.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	3
1.1. Requirement Language	5
1.2. Terminology	5
1.2.1. Flow	5
1.2.2. Data Transmission Type	5
2. Overview of Approaches	5
2.1. Extended Endpoint Address	6
2.2. Flow-based Filter	6
2.3. Flow-specific Announcement	6
3. Extended Endpoint Address	7
3.1. Address Type	7
3.2. Endpoint Address	8
3.2.1. MAC Address	8
3.2.2. Internet Domain Name	8
3.2.3. IPv4 Socket Address	8
3.2.4. IPv6 Socket Address	8
3.3. Address Type Compatibility	9
3.4. Examples	9
4. Flow-specific Announcement	9
4.1. Flow-specific Announcement Type	9
4.2. Data Transmission Type Announcement	10
5. Extended Cost Query Filters	10
5.1. Filtered Cost Map Extension	10
5.1.1. Capabilities	10
5.1.2. Accept Input Parameters	11
5.2. Response	12
5.3. Endpoint Cost Service Extension	12

5.3.1. Capabilities	12
5.3.2. Accept Input Parameters	13
5.4. Response	14
5.5. Examples	15
5.5.1. Information Resource Directory	15
5.5.2. Flow-based Filtered Cost Map Example	17
5.5.3. Flow-based Endpoint Cost Service Example #1	18
5.5.4. Flow-based Endpoint Cost Service Example #2	19
6. Security Considerations	21
7. IANA Considerations	21
7.1. ALTO Address Type Registry	21
7.2. ALTO Address Type Compatibility Registry	22
7.3. ALTO Flow-specific Announcement Registry	23
8. References	24
8.1. Normative References	24
8.2. Informative References	24
Appendix A. Acknowledgment	25
Appendix B. Change Logs	25
Authors' Addresses	27

1. Introduction

Application-Layer Traffic Optimization (ALTO) protocol [RFC7285] defines several cost query services, like Filtered Cost Map and Endpoint Cost Service, to allow applications to query path costs. Generally, an ALTO cost query service can be regarded as a function transforming a given subset of a specific query space into a network view abstract, where the subset is specified by a PID filter or an endpoint filter. However, the current specification has some limitations.

First, in the base ALTO protocol [RFC7285], the endpoint filter only contains the source and destination IP addresses. In practice, both Internet Service Providers (ISP) and local network administrators may conduct policy-based routing, e.g., P2P traffic may be constrained and has a smaller bandwidth than HTTP traffic. Also, web services with different QoS requirements may be hosted on the same machine with the same IP address but different paths with different QoS metrics.

Second, in the base ALTO protocol [RFC7285], the query space is defined by a source list and a destination list. For a query with N sources and M destinations, the response contains N*M entries. While such a query schema is well suited for peer-to-peer (P2P) applications where files of the same seed are stored on all hosts, it may lead to a lot of redundancy in use cases such as modern data analytics systems where replicas of the same dataset are stored on only a small subset of servers. Consider a system where the number

of replicas is 3 (the default in HDFS), jointly scheduling N concurrent transfers only needs a maximum of $3N$ entries but the base ALTO protocol may return up to N^2 entries.

Third, in the base ALTO protocol [RFC7285], the query does not distinguish among the different transmission types like unicast and multicast. For some use cases like the multi-flow scheduling demonstrated by [I-D.ietf-alto-path-vector], the data transmission between endpoints could be beyond unicast. And in those cases, different transmission types may affect the network resource consumption. If applications can receive the path costs distinguishing the different transmission types, it can help applications perform their data transmission decision better.

Thus, we conclude that the following additional requirements (AR) MUST be satisfied to allow ALTO clients make more accurate and efficient cost queries.

AR-1: The ALTO server SHOULD allow the ALTO client to specify accurate query space in cost query services.

The base ALTO protocol only includes IPv4 and IPv6 addresses as endpoint address types, which may not be sufficient to accurately identify an endpoint with emerging flow-based routing mechanisms. ALTO clients MAY suffer from suboptimal decisions because of such inaccuracy. Thus, the ALTO protocol SHOULD be extended so that clients are able to specify accurate query space, i.e., using more fine-grained endpoint address types.

AR-2: The ALTO server SHOULD allow the ALTO client to specify only the essential query space.

Existing PIDFilter (see Sec 11.3.2.3 in [RFC7285]) and EndpointFilter (see Sec 11.5.1.3 in [RFC7285]) represent the cross-product of sources and destinations, and can introduce a lot of redundancy in certain use cases. This limitation greatly harms the scalability of the ALTO protocol. Thus, the ALTO protocol SHOULD be extended so that ALTO clients are able to specify only the essential cost query space, i.e., the concerned source-destination pairs.

AR-3: The ALTO server SHOULD allow the ALTO client to specify different data transmission types for transmissions in the query space.

The input parameters of existing ALTO cost query services only allow the ALTO client to specify the queried transmissions by sources and destinations. The transmission between each source

and destination will always be considered as the unicast. This limitation may make the ALTO client lose the accurate available resources. Thus, the ALTO protocol SHOULD be extended so that ALTO clients are able to specify different transmission types.

In this document, we describe an ALTO extension specifying flow-based cost queries. The rest of this document is organized as follows. Section 3 introduces several new address types that extend the query space of ALTO cost services. Section 5 describes the extended schema on Filtered Cost Map (FCM) and Endpoint Cost Service (ECS) to support cost queries of arbitrary source-destination combinations with the optional flow-specific information. Section 6 and Section 7 discuss security and IANA considerations.

1.1. Requirement 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 [RFC2119].

1.2. Terminology

This document uses the same terms as defined in [RFC7285] and [RFC8189] with the following additional term:

1.2.1. Flow

In this document, a flow refers to all communications between two endpoints. A flow is "valid" if and only if there CAN be valid communications between the two endpoints, which oftentimes requires that that two endpoint addresses have compatible address types.

1.2.2. Data Transmission Type

This document use the term "Data Transmission Type" or "Transmission Type" to indicate how an application sends the network flows. See Section 4.2.

2. Overview of Approaches

This section presents a non-normative overview of the extension to support flow-based cost query. It assumes the readers are familiar with Filtered Cost Map and Endpoint Cost Service defined in [RFC7285] and their extensions defined in [RFC8189].

2.1. Extended Endpoint Address

To allow ALTO clients specify accurate query space in cost query services (AR-1), this document defines several new endpoint address types. An endpoint address with a new type is referred to as an extended endpoint address.

Since the address types of both the source and the destination correspond to the same network flow, they MUST NOT conflict. This document defines an address type conflict table to indicate conflicts. If some source and destination address types in a query conflict with each others, an ALTO server MUST return the corresponding error.

2.2. Flow-based Filter

To allow ALTO clients specify only the essential query space in cost query services (AR-2), both PIDFilter and EndpointFilter in the base protocol MUST be extended. The extended filters are referred to as flow-based filters.

A straight-forward way of satisfying AR-2 is to have an ALTO client list all its concerned flows. Despite its simplicity, it MAY be too large in size, especially when many flows have common sources or common destinations in the query. Also from the implementation's perspective, it cannot reuse the functionality to parse a PIDFilter/EndpointFilter.

Thus, the flow-based filters defined in this document allow ALTO clients to include multiple PIDFilter/EndpointFilter objects in the same query. Apparently, if we replace each PIDFilter/EndpointFilter of N sources and M destinations with NM filters that have exactly one source and destination, the two representations refer to the same set of flows. As a result, one can aggregate flows with common sources or destinations in one PIDFilter/EndpointFilter object without introducing redundant flows.

From the implementation's perspective, one MAY reuse an ALTO library which parses PIDFilter/EndpointFilter and/or converts them into a set of source-destination pairs.

2.3. Flow-specific Announcement

Some properties are only related to the transmission and cannot be encoded into endpoints, e.g., the data transmission type of a flow. However, these properties may help the ALTO client get more accurate costs.

To allow the ALTO client to specify these flow-specific properties (AR-3), this document introduces an extensible field in the flow-based filter. The ALTO client can announce the flow-specific information in this field. An announcement, whose type is encoded as a FlowSpecAnnounceType (Section 4.1), can represent transmission type, equal cost multipath assumption and other kinds of flow-specific information.

This document adopts an extensible design for this announcement field. Although only the data transmission type is defined in this document, other announcement properties can be defined in future documents and are not in the scope of this document.

3. Extended Endpoint Address

This document registers new address types and defines the corresponding formats for endpoint addresses of each new address type.

3.1. Address Type

The new AddressType identifiers defined in this document are as follows:

eth: An endpoint address with type "eth" is the address of an Ethernet interface. It is used to uniquely identify an endpoint in the data link layer.

domain: An endpoint address with type "domain" is the domain name of a web service. It is used to uniquely identify a web service which MAY be translated to one or more IPv4 address(es).

domain6: An endpoint address with type "domain6" is the domain name of a web service. It is used to uniquely identify a web service which MAY be translated to one or more IPv6 address(es).

tcp: An endpoint address with type "tcp" is the address of a TCP socket. It is used to uniquely identify an IPv4 TCP socket in the transport layer.

tcp6: An endpoint address with type "tcp6" is the address of a TCP socket. It is used to uniquely identify an IPv6 TCP socket in the transport layer.

udp: An endpoint address with type "udp" is the address of a UDP socket. It is used to uniquely identify an IPv4 UDP socket in the transport layer.

udp6: An endpoint address with type "udp6" is the address of a UDP socket. It is used to uniquely identify an IPv6 UDP socket in the transport layer.

3.2. Endpoint Address

This document defines EndpointAddr when AddressType is in Section 7.1.

3.2.1. MAC Address

An Endpoint Address of type "eth" is encoded as a MAC address, whose format is encoded as specified by either format EUI-48 in [EUI48] or EUI-64 in [EUI64].

3.2.2. Internet Domain Name

An Endpoint Address of type "domain" or "domain6" is encoded as a domain name in the Internet, as specified in Section 11 of [RFC2181]. It MUST have at least one corresponding A ("domain") or AAAA ("domain6") record in the DNS.

3.2.3. IPv4 Socket Address

An Endpoint Address of type "tcp" or "udp" is encoded as an IPv4 socket address. It is encoded as a string of the format Host:Port with the ":" character as a separator. The Host component of an IPv4 socket address is encoded as specified by either an IPv4 address (see Section 10.4.3.1 of [RFC7285]) or an IPv4-compatible domain name (see Section 3.2.2). The Port component of an IPv4 socket address is encoded as an integer between 1 and 65535.

3.2.4. IPv6 Socket Address

An Endpoint Address of type "tcp6" or "udp6" is encoded as an IPv6 socket address. It is also encoded as a string of the format Host:Port with the ":" character as a separator. The Host component of an IPv6 socket address is encoded as specified by either an IPv6 address (see Section 10.4.3.2 of [RFC7285]) enclosed in the "[" and "]" characters as recommended in [RFC2732] or an IPv6-compatible domain name (see Section 3.2.2). The Port component of IPv6 socket address is encoded as an integer between 1 and 65535.

3.3. Address Type Compatibility

In practice, a flow with endpoint addresses with different types MAY NOT be valid. For example, a source endpoint with an IPv4 address CANNOT establish a network connection with a destination endpoint with an IPv6 address. Neither can a source with a TCP socket address and a destination with a UDP socket address.

Thus, to explicitly define the compatibility between AddressType identifiers, every ALTO AddressType identifier MUST provide a list of AddressType identifiers that are compatible with it in the "ALTO Address Type Compatibility Registry" Section 7.2. For all sources and destinations in a PIDFilter/EndpointFilter, if the AddressType identifiers of a given pair DO NOT appear in the ALTO Address Type Compatibility Registry, an ALTO server MUST return an ALTO error response with the error code "E_INVALID_FIELD_VALUE" with optional information to help diagnose the incompatibility.

3.4. Examples

Some valid endpoint addresses are demonstrated as follows:

```
"eth:98-e0-d9-9c-df-81"  
"domain:www.example.com"  
"tcp:198.51.100.34:5123"  
"udp6:[2000::1:2345:6789:abcd]:8080"
```

4. Flow-specific Announcement

Each flow-specific announcement refers to a property of the traffic between a set of source and destination pairs. The interpretation of a property, however, depends on the meaning, i.e., the type, of the property. This document uses flow-specific announcement type as an indicator of the "type" information, and requires the flow-specific announcement types be registered in an IANA registry called "ALTO Flow-specific Announcement Registry".

4.1. Flow-specific Announcement Type

It is encoded as a JSONString and has the same format as a PID Name defined in Section 10.1 in [RFC7285]. The type FlowSpecAnnounceType is used in this document to indicate a string of that format.

4.2. Data Transmission Type Announcement

This document defines a flow-specific announcement called the data transmission type. The type of this announcement is indicated by the string "transmission-type", and the value of this announcement is a JSONString of either "unicast", "anycast", "broadcast" or "multicast".

5. Extended Cost Query Filters

This section describes extensions to [RFC7285] and [RFC8189] to support flow-based cost queries.

This document uses the notation rules specified in Section 8.2 of [RFC7285] and also the notation rules for optional fields in Section 4 of [RFC8189].

5.1. Filtered Cost Map Extension

This document extends the Filtered Cost Map as defined in Section 11.3.2 of [RFC7285] and Section 4.1 of [RFC8189], by adding a new capability and new input parameters.

The media type, HTTP method, and "uses" specifications (described in Sections 11.3.2.1, 11.3.2.2, and 11.3.2.5 of [RFC7285], respectively) are not changed.

The format of the response is the same as defined in Section 4.1.3 of [RFC8189]. But this document recommends how to generate the response based on the extended input parameters.

5.1.1. Capabilities

The Filtered Cost Map capabilities are extended with two additional members:

- * flow-based-filter
- * flow-spec-announce

The capability "flow-based-filter" indicates whether this resource supports flow-based cost queries, and the capability "flow-spec-announce" indicates which flow-specific announcements are supported. The FilteredCostMapCapabilities object in Section 4.1.1 of [RFC8189] is extended as follows:

```
object {
  JSONString cost-type-names<1..*>;
  [JSONBool cost-constraints;]
  [JSONNumber max-cost-types;]
  [JSONString testable-cost-type-names<1..*>;]
  [JSONBool flow-based-filter;]
  [FlowSpecAnnounceType flow-spec-announce<1..*>;]
} FilteredCostMapCapabilities;
```

cost-type-names and cost-constraints: As defined in Section 11.3.2.4 of [RFC7285].

max-cost-types and testable-cost-type-names: As defined in Section 4.1.1 of [RFC8189].

flow-based-filter: If true, an ALTO Server allows a field "pid-flows" to be included in the requests. If not present, this field MUST be interpreted as if it is false.

flow-spec-announce: It MUST NOT be present if "flow-based-filter" is not true. If present, the value is the an array of supported flow-specific announcement types.

5.1.2. Accept Input Parameters

The ReqFilteredCostMap object in Section 4.1.2 of [RFC8189] is extended as follows:

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

object {
  [FlowSpecAnnounceMap flow-spec-announce;]
} ExtPIDFilter : PIDFilter;

object-map {
  FlowSpecAnnounceType -> JSONValue;
} FlowSpecAnnounceMap;
```

cost-type, multi-cost-types, testable-cost-types, constraints, or-constraints: As defined in Section 4.1.2 of [RFC8189].

pids: As defined in Section 11.3.2.3 of [RFC7285].

pid-flows: Defined as a list of ExtPIDFilter objects. The ALTO server MUST interpret PID pairs appearing in multiple ExtPIDFilter objects as if they appeared only once. If the capability "flow-spec-announce" is present, the "flow-spec-announce" input parameter can be specified. The value of this field is of type FlowSpecAnnounceMap, which maps a FlowSpecAnnounceType to its corresponding value. The interpretation of the value MUST follow the definition of the flow-specific announcement in the ALTO Flow-specific Announcement Registry.

An ALTO client MUST include either "pids" or "pid-flows" in a query but MUST NOT include both at the same time.

5.2. Response

This document does not change the format of the response entity. But the ALTO server responds the request with "pid-flows" filter as follows:

The ALTO server MUST include the path costs of pairs in each ExtPIDFilter in the "pid-flows" filter. If the "flow-spec-announce" field is specified in some ExtPIDFilter, the path costs for flows in this ExtPIDFilter SHOULD respond the flow-specific information announced by this field.

5.3. Endpoint Cost Service Extension

This document extends the Endpoint Cost Service as defined in Section 11.5.1 of [RFC7285] and Section 4.2 of [RFC8189], by adding a new capability and input parameters.

The media type, HTTP method, and "uses" specifications (described in Sections 11.5.1.1, 11.5.1.2, and 11.5.1.5 of [RFC7285], respectively) are unchanged.

The format of the response is the same as defined in Section 4.2.3 of [RFC8189]. But this document recommends how to generate the response based on the extended input parameters.

5.3.1. Capabilities

The extension to EndpointCostCapabilities includes three additional members:

- * flow-based-filter

- * address-types
- * flow-spec-announce

Only if the capability "flow-based-filter" is present and its value is "true", the ALTO server supports the flow-based extension for this endpoint cost service. The capability "address-types" indicates which endpoint address types are supported by this resource, it MUST NOT be specified if "flow-based-filter" is absent or the value is false. The capability "flow-spec-announce" indicates which flow-specific announcements are supported, just like it works in the Filtered Cost Map resource.

```
object {  
  [JSONBool    flow-based-filter;]  
  [JSONString  address-types<0..*>;]  
  [FlowSpecAnnounceType flow-spec-announce<1..*>;]  
} EndpointCostCapabilities : FilteredCostMapCapabilities;
```

flow-based-filter: If true, an ALTO Server MUST accept field "endpoint-flows" in the requests. If not present, this field MUST be interpreted as if it is specified false.

address-types: Defines a list of AddressType identifiers encoded as a JSONArray of JSONString. All AddressType identifiers MUST be registered in the "ALTO Address Type Registry" (see Section 14.4 of [RFC7285]). An ALTO server SHOULD NOT claim "ipv4" and "ipv6" in this field explicitly, because they are supported by default. If not present, this field MUST be interpreted as if it is an empty array, i.e., the ALTO server only supports the default "ipv4" and "ipv6" address types.

flow-spec-announce: It MUST NOT be present if "flow-based-filter" is not true. If present, the value is the an array of supported flow-specific announcement types.

5.3.2. Accept Input Parameters

The ReqEndpointCostMap object in Section 4.2.2 of [RFC8189] is extended as follows:

```
object {  
  [CostType cost-type;]  
  [CostType multi-cost-types<1..*>;]  
  [CostType testable-cost-types<1..*>;]  
  [JSONString constraints<0..*>;]  
  [JSONString or-constraints<1..*><1..*>;]  
  [EndpointFilter endpoints;]  
  [ExtEndpointFilter endpoint-flows<1..*>;]  
} ReqEndpointCostMap;  
  
object {  
  [FlowSpecAnnounceMap flow-spec-announce;]  
} ExtEndpointFilter : EndpiontFilter;
```

cost-type, multi-cost-types, testable-cost-types, constraints, or-constraints:

As defined in Section 4.1.2 of [RFC8189].

endpoints: As defined in Section 11.5.1.3 of [RFC7285].

endpoint-flows: Defined as a list of ExtEndpointFilter objects. The ALTO server MUST interpret endpoint pairs appearing in multiple ExtEndpointFilter objects as if they appeared only once. If the capability "flow-spec-announce" is present, the "flow-spec-announce" input parameter can be specified. The interpretation of this field is the same as in Section 5.1.2.

If the AddressType of the source and destination in the same EndpointFilter do not conform the compatibility rule defined in Table 1 of Section 7.1, an ALTO server MUST return an ALTO error response with the error code "E_INVALID_FIELD_VALUE".

An ALTO client MUST specify either "endpoints" or "endpoint-flows", but MUST NOT specify both in the same query.

5.4. Response

This document does not change the format of the response entity. But the ALTO server responds the request with "pid-flows" filter as follows:

The ALTO server MUST include the path costs of pairs in each ExtPIDFilter in the "pid-flows" filter. If the "flow-spec-announce" field is specified in some ExtPIDFilter, the path costs for flows in this ExtPIDFilter SHOULD respond the flow-specific information announced by this field. Especially, if "transmission-type" is specified as "multicast", the ALTO server SHOULD expose all the destination address as a multicast group address, and append the

shared trees to the multicast destination addresses into the response if possible.

5.5. Examples

5.5.1. Information Resource Directory

The following is an example of IRD with relevant resources of the ALTO server. It provides a default network map, a property map of "ane" domain, a filtered cost map and two endpoint cost resources. All of three cost query resources (filtered cost map and endpoint cost resources) support "flow-based-filter". One endpoint cost resource support "flow-spec-announce" and the compound query extension defined in I-D.ietf-alto-path-vector.

Examples followed this section use the same IRD in this document.

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

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

```
{
  "meta" : {
    "default-alto-network-map" : "my-default-network-map",
    "cost-types" : {
      "num-hopcount" : {
        "cost-mode" : "numerical",
        "cost-metric" : "hopcount"},
      "num-routingcost" : {
        "cost-mode" : "numerical",
        "cost-metric" : "routingcost"},
      "ord-routingcost" : {
        "cost-mode" : "ordinal",
        "cost-metric" : "routingcost"},
      "path-vector" : {
        "cost-mode" : "array",
        "cost-metric" : "ane-path"}
    },
    .....,
    Other ALTO cost types as described in RFC7285
    .....,
  },
  "resources" : {
```

```

    "my-default-network-map" : {
      "uri" : "http://alto.example.com/networkmap",
      "media-type" : "application/alto-networkmap+json"
    },
    "flow-based-cost-map" : {
      "uri" : "http://alto.example.com/costmap/multi/filtered",
      "media-type" : "application/alto-costmap+json",
      "accepts" : "application/alto-costmapfilter+json",
      "uses" : [ "my-default-network-map" ],
      "capabilities" : {
        "max-cost-types" : 2,
        "flow-based-filter" : true,
        "cost-type-names" : [ "num-hopcount",
                              "num-routingcost" ]
      }
    },
    "flow-based-endpoint-cost" : {
      "uri" : "http://alto.example.com/endpointcost/lookup",
      "media-type" : "application/alto-endpointcost+json",
      "accepts" : "application/alto-endpointcostparams+json",
      "capabilities" : {
        "address-types": ["tcp", "udp"],
        "flow-based-filter" : true,
        "cost-type-names" : [ "ord-routingcost",
                              "num-routingcost" ]
      }
    },
    "path-vector-endpoint-cost" : {
      "uri" : "http://alto.example.com/pathvector/lookup",
      "media-type": "multipart/related;
                    type=application/alto-costmap+json",
      "accepts" : "application/alto-endpointcostparams+json",
      "capabilities" : {
        "address-types": ["tcp", "tcp6"],
        "flow-based-filter" : true,
        "flow-spec-announce" : [ "transmission-type" ]
        "cost-type-names" : [ "path-vector" ],
        "ane-property-names": [ "maxresbw" ],
      }
    }
  }
}

```


5.5.2. Flow-based Filtered Cost Map Example

This example shows how an ALTO client requests a filtered cost map using the "pid-flows" filter. In this case, the ALTO client receives a sparse cost map, which cuts 50% useless cost values from the full mesh.

```
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

{
  "cost-type": {
    "cost-mode": "numerical",
    "cost-metric": "routingcost"
  },
  "pid-flows": [
    { "srcs": ["PID1"], "dsts": ["PID2", "PID3"] },
    { "srcs": ["PID3"], "dsts": ["PID4"] }
  ]
}

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

{
  "meta": {
    "dependent-vtags": [
      {
        "resource-id": "my-default-network-map",
        "tag": "75ed013b3cb58f896e839582504f622838ce670f"
      }
    ],
    "cost-type": {
      "cost-mode": "numerical",
      "cost-metric": "hopcount"
    }
  },
  "cost-map": {
    "PID1": { "PID2": 6, "PID3": 2 },
    "PID3": { "PID4": 1 }
  }
}
```

5.5.3. Flow-based Endpoint Cost Service Example #1

This example shows how the ALTO client requests endpoint cost using "flow-based-filter" and extended endpoint addresses. In this case, the ALTO client specifies tcp socket address to get more accurate path cost.

```
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

{
  "cost-type": {
    "cost-mode": "numerical",
    "cost-metric": "hopcount"
  },
  "endpoint-flows": [
    { "srcs": ["ipv4:192.0.2.2"],
      "dsts": ["ipv4:192.0.2.89", "tcp:cdn1.example.com:21"] },
    { "srcs": ["tcp:203.0.113.45:54321"],
      "dsts": ["tcp:cdn1.example.com:21"] }
  ]
}

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

{
  "meta": {
    "cost-type": {
      "cost-mode": "numerical",
      "cost-metric": "routingcost"
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89": 100,
      "tcp:cdn1.example.com:21": 20
    },
    "tcp:203.0.113.45:54321": {
      "tcp:cdn1.example.com:21": 80
    }
  }
}
```

5.5.4. Flow-based Endpoint Cost Service Example #2

This example shows the integration of the path vector extension and the flow-based query. And in this example, the ALTO client specifies the flow from "tcp6:203.0.113.45:54321" to "tcp6:group1.example.com:21" is multicast. So the ALTO server will expose the destination IP as a multicast group IP, and find the multicast destinations "fe80::40e:9594:da3d:34b" and "fe80::826:daff:feb8:1bb". Then the ALTO server will append the cost for the shared tree into the "endpoint-cost-map".

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

```
{
  "cost-type": {
    "cost-mode": "array",
    "cost-metric": "ane-path"
  },
  "endpoint-flows": [
    { "srcs": ["ipv4:192.0.2.2"],
      "dsts": ["tcp:192.0.2.89:21",
               "tcp:cdn1.example.com:21"] },
    { "srcs": ["tcp6:203.0.113.45:54321"],
      "dsts": ["tcp6:group1.example.com:21"],
      "flow-spec-announce": {
        "transmission-type": "multicast" } }
  ],
  "ane-property-names": ["maxresbw"]
}
```

```
HTTP/1.1 200 OK
Content-Length: [TBD]
Content-Type: multipart/related; boundary=example;
             type=application/alto-endpointcost+json
```

```
--example
Resource-Id: ecs
Content-Type: application/alto-endpointcost+json
```

```
{
  "meta": {
    "vtags": {
```

```

        "resource-id": "path-vector-endpoint-cost.ecs",
        "tag": "bb6bb72eafe8f9bdc4f335c7ed3b10822a391cef"
    },
    "cost-type": {
        "cost-mode": "array",
        "cost-metric": "ane-path"
    }
},
"endpoint-cost-map": {
    "ipv4:192.0.2.2": {
        "tcp:192.0.2.89:21": [ "ane:S1", "ane:D1" ],
        "tcp:cdn1.example.com:21": [ "ane:S1", "ane:D2", "ane:D3" ]
    },
    "tcp6:203.0.113.45:54321": {
        "tcp6:group1.example.com:21": [ "ane:S2", "ane:D3" ]
    },
    "tcp6:group1.example.com:21": {
        "tcp6:[fe80::40e:9594:da3d:34b]:21": [ "ane:G1" ],
        "tcp6:[fe80::826:daff:feb8:1bb]:21": [ "ane:G2" ],
    }
}
}

--example
Resource-Id: propmap
Content-Type: application/alto-propmap+json

{
  "meta": {
    "dependent-vtags": [
      {
        "resource-id": "path-vector-endpoint-cost.ecs",
        "tag": "bb6bb72eafe8f9bdc4f335c7ed3b10822a391cef"
      }
    ]
  },
  "property-map": {
    "ane:S1": { "maxresbw": 100 },
    "ane:S2": { "maxresbw": 100 },
    "ane:D1": { "maxresbw": 150 },
    "ane:D2": { "maxresbw": 80 },
    "ane:D3": { "maxresbw": 150 },
    "ane:G1": { "maxresbw": 100 },
    "ane:G2": { "maxresbw": 100 }
  }
}

```

6. Security Considerations

As discussed in Section 15.4 of [RFC7285], an ALTO server or a third party who is able to intercept the flow-based cost query messages MAY store and process the obtained information in order to analyze user behaviors and communication patterns. Since flow-based cost queries MAY potentially provide more accurate information, an ALTO client should be cognizant about the trade-off between redundancy and privacy.

7. IANA Considerations

This document defines new address types to be registered to an existing ALTO registry, and a new registry for their compatible address types.

7.1. ALTO Address Type Registry

This document defines several new address types to be registered to "ALTO Address Type Registry", listed in Table 1.

Identifier	Address Encoding	Prefix Encoding	Mapping to/from IPv4/v6
eth	See Section 3.2.1	None	Mapping to/from IPv4 by [RFC0903] and [RFC0826]; Mapping to/from IPv6 by [RFC3122] and [RFC4861]
domain	See Section 3.2.2	None	Mapping to/from IPv4 by [RFC1034]
domain6	See Section 3.2.2	None	Mapping to/from IPv6 by [RFC3596]
tcp	See Section 3.2.3	None	No mapping
tcp6	See Section 3.2.4	None	No mapping
udp	See Section 3.2.3	None	No mapping
udp6	See Section 3.2.4	None	No mapping

Table 1: ALTO Address Type Registry

7.2. ALTO Address Type Compatibility Registry

This document proposes to create a new registry called "ALTO Address Type Compatibility Registry", whose purpose is stated in Section 3.3.

The compatible address type identifiers of the ones registered in the ALTO Address Type Registry are listed in Table 2.

Identifier	Compatible Identifiers
eth	ipv4, ipv6
domain	eth, ipv4
domain6	eth, ipv6
tcp	eth, ipv4, domain
tcp6	eth, ipv6, domain6
udp	eth, ipv4, domain
udp6	eth, ipv6, domain6

Table 2: ALTO Address Type
Compatibility Registry

The entry of an address type identifier SHOULD only include the identifiers registered before it. The compatibility between address types are bidirectional. For example, although "eth" does not register "tcp" as its compatible identifier, an ALTO server MUST recognize them as compatible because "eth" is registered as a compatible identifier of "tcp".

Any new ALTO address type identifier registered after this document MUST register their compatible identifiers in this registry simultaneously.

7.3. ALTO Flow-specific Announcement Registry

This document proposes to create a new registry called "ALTO Flow-specific Announcement Registry", whose purpose is stated in Section 4. An initial registry entry is also documented as shown in Table 3.

Type	Interpretation
transmission-type	See Section 4.2

Table 3: ALTO Flow-specific
Announcement Registry

8. References

8.1. Normative References

- [EUI48] IEEE, ., "Guidelines for use of a 48-bit Extended Unique Identifier (EUI-48)", 2012.
- [EUI64] IEEE, ., "Guidelines for use of a 64-bit Extended Unique Identifier (EUI-64)", 2012.
- [I-D.ietf-alto-path-vector]
Gao, K., Randriamasy, S., Yang, Y., and J. Zhang, "ALTO Extension: Path Vector", Work in Progress, Internet-Draft, draft-ietf-alto-path-vector-10, 9 March 2020, <<http://www.ietf.org/internet-drafts/draft-ietf-alto-path-vector-10.txt>>.
- [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>>.

8.2. Informative References

- [RFC0826] Plummer, D., "An Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware", STD 37, RFC 826, DOI 10.17487/RFC0826, November 1982, <<https://www.rfc-editor.org/info/rfc826>>.
- [RFC0903] Finlayson, R., Mann, T., Mogul, J.C., and M. Theimer, "A Reverse Address Resolution Protocol", STD 38, RFC 903, DOI 10.17487/RFC0903, June 1984, <<https://www.rfc-editor.org/info/rfc903>>.
- [RFC1034] Mockapetris, P.V., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <<https://www.rfc-editor.org/info/rfc1034>>.
- [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>>.
- [RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, DOI 10.17487/RFC2181, July 1997,
<<https://www.rfc-editor.org/info/rfc2181>>.
- [RFC2732] Hinden, R., Carpenter, B., and L. Masinter, "Format for Literal IPv6 Addresses in URL's", RFC 2732, DOI 10.17487/RFC2732, December 1999,
<<https://www.rfc-editor.org/info/rfc2732>>.
- [RFC3122] Conta, A., "Extensions to IPv6 Neighbor Discovery for Inverse Discovery Specification", RFC 3122, DOI 10.17487/RFC3122, June 2001,
<<https://www.rfc-editor.org/info/rfc3122>>.
- [RFC3596] Thomson, S., Huitema, C., Ksinant, V., and M. Souissi, "DNS Extensions to Support IP Version 6", STD 88, RFC 3596, DOI 10.17487/RFC3596, October 2003,
<<https://www.rfc-editor.org/info/rfc3596>>.
- [RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007,
<<https://www.rfc-editor.org/info/rfc4861>>.

Appendix A. Acknowledgment

The authors would like to thank Dawn Chen, Haizhou Du, Sabine Randriamasy and Wendy Roome for their fruitful discussions and feedback on this document. Shawn Lin also gave substantial review feedback and suggestions on the protocol design.

Appendix B. Change Logs

Note to Editor: Please remove this section prior to publication.

This section records the change logs of the draft updates.

Changes since -06 versions:

- * Clarify the type of "flow-spec-announcement" in both the request and the response
- * Modify examples to be compatible with the latest path vector document

- * Add a new registry for flow-specific announcements

- * Fix some typos

Changes since -05 versions:

- * Add flow-specific information announcement in the flow-based filter.
- * Modify examples and add descriptions to Make them clear.
- * Rename the address type "Domain Name" to "Internet Domain Name" to distinguish it with the "Domain Name" in the unified properties draft.

Changes since older versions:

Changes since -04 revision:

- * Improve the clarity of the document by explicitly stating the problems.
- * Keep only "flow" in the terminology section.
- * Move section 6 "Advanced Flow-based Query" out of this document.
- * Change "ALTO Address Type Conflicts Registry" to "ALTO Address Type Compatibility Registry".

Since -03 revision:

- * Remove some irrelevant content from the draft.
- * Improve the description of the new endpoint address type identifier registry. And add a new registry to declare the conflicting address type identifiers.

Since -02 revision:

- * Change "EndpointURI" to "AddressType::EndpointAddr" for consistency.
- * Replace "Cost Confidence" by "Cost Statistics" for compatibility.

Since -01 revision:

- * Define the basic flow-based query extensions for Filtered Cost Map and Endpoint Cost service. The basic flow-based query is downward

compatible with the legacy ALTO service. It does not introduce any new media types.

- * Move the service of media-type "application/alto-flowcost+json" to the advanced flow-based query extension. It will ask ALTO server to support the new media type.

Since -00 revision:

- * Change the schema of "pid-flows" and "endpoint-flows" fields from pair list to pair mesh list.

Authors' Addresses

Jingxuan Jensen Zhang
China
201804
Shanghai
4800 Caoan Road
Tongji University

Email: jingxuan.n.zhang@gmail.com

Kai Gao
China
610000
Chengdu
No.24 South Section 1, Yihuan Road
Sichuan University

Email: kaigao@scu.edu.cn

Junzhuo Wang

Qiao Xiang
Yale University
51 Prospect Street
New Haven, CT
United States of America

Email: qiao.xiang@cs.yale.edu

Yang Richard Yang
Yale University

51 Prospect Street
New Haven, CT
United States of America

Email: yry@cs.yale.edu

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 18, 2020

S. Randriamasy
Nokia Bell Labs
R. Yang
Yale University
Q. Wu
Huawei
L. Deng
China Mobile
N. Schwan
Thales Deutschland
March 17, 2020

Application-Layer Traffic Optimization (ALTO) Cost Calendar
draft-ietf-alto-cost-calendar-21

Abstract

This document is an extension to the base Application-Layer Traffic Optimization (ALTO) protocol. It extends the ALTO cost information service so that applications decide not only 'where' to connect, but also 'when'. This is useful for applications that need to perform bulk data transfer and would like to schedule these transfers during an off-peak hour, for example. This extension introduces ALTO Cost Calendar, with which an ALTO Server exposes ALTO cost values in JSON arrays where each value corresponds to a given time interval. The time intervals as well as other Calendar attributes, are specified in the Information Resources Directory and ALTO Server responses.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 18, 2020.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	3
1.1. Some recent known uses	4
1.2. Terminology	5
2. Requirements Language	5
3. Overview of ALTO Cost Calendars and terminology	5
3.1. ALTO Cost Calendar overview	6
3.2. ALTO Cost Calendar information features	6
3.3. ALTO Calendar design characteristics	7
3.3.1. ALTO Cost Calendar for all cost modes	9
3.3.2. Compatibility with legacy ALTO Clients	10
4. ALTO Calendar specification: IRD extensions	10
4.1. Calendar attributes in the IRD resource capabilities . .	11
4.2. Calendars in a delegate IRD	12
4.3. Example IRD with ALTO Cost Calendars	13
5. ALTO Calendar specification: Service Information Resources .	17
5.1. Calendar extensions for Filtered Cost Maps (FCM)	17
5.1.1. Calendar extensions in Filtered Cost Map requests . .	17
5.1.2. Calendar extensions in Filtered Cost Map responses .	18
5.1.3. Use case and example: FCM with a bandwidth Calendar .	21
5.2. Calendar extensions in the Endpoint Cost Service	23
5.2.1. Calendar specific input in Endpoint Cost requests . .	23
5.2.2. Calendar attributes in the Endpoint Cost response . .	24
5.2.3. Use case and example: ECS with a routingcost Calendar	25
5.2.4. Use case and example: ECS with a multi-cost Calendar	
for routingcost and owdelay	27
6. IANA Considerations	29
7. Security Considerations	29
8. Operational Considerations	31
9. Acknowledgements	32
10. References	32
10.1. Normative References	32

10.2. Informative References	33
Authors' Addresses	35

1. Introduction

The base Application-Layer Traffic Optimization (ALTO) protocol specified in [RFC7285] provides guidance to overlay applications that need to select one or several hosts from a set of candidates able to provide a desired resource. This guidance is based on parameters that affect performance and efficiency of the data transmission between the hosts such as the topological distance. The goal of ALTO is to improve the Quality of Experience (QoE) in the application while optimizing resource usage in the underlying network infrastructure.

The ALTO protocol in [RFC7285] specifies a network map which defines groupings of endpoints in provider-defined network regions identified by Provider-defined Identifiers (PIDs). The Cost Map Service, Endpoint Cost Service (ECS) and Endpoint Ranking Service then provide ISP-defined costs and rankings for connections among the specified endpoints and PIDs and thus incentives for application clients to connect to ISP preferred locations, for instance, to reduce their costs. For the reasons outlined in the ALTO problem statement [RFC5693] and requirement AR-14 of [RFC6708], ALTO does not disseminate network metrics that change frequently. In a network, the costs can fluctuate for many reasons having to do with instantaneous traffic load or due to diurnal patterns of traffic demand or planned events such as network maintenance, holidays or highly publicized events. Thus, an ALTO application wishing to use the Cost Map and Endpoint Cost Service at some future time will have to estimate the state of the network at that time, a process that is, at best, fragile and brittle since the application does not have any visibility into the state of the network. Providing network costs for only the current time thus may not be sufficient, in particular for applications that can schedule their traffic in a span of time, for example by deferring backups or other background traffic to off-peak hours.

In case the ALTO Cost value changes are predictable over a certain period of time and the application does not require immediate data transfer, it can save time to get the whole set of cost values over this period in one single ALTO response. Using this set to schedule data transfers allows optimizing the network resources usage and QoE. ALTO Clients and Servers can also minimize their workload by reducing and accordingly scheduling their data exchanges.

This document extends [RFC7285] to allow an ALTO Server to provide network costs for a given duration of time. A sequence of network

costs across a time span for a given pair of network locations is named an "ALTO Cost Calendar". The Filtered Cost Map Service and Endpoint Cost Service are extended to provide Cost Calendars. In addition to this functional ALTO enhancement, we expect to further save network and storage resources by gathering multiple Cost Values for one cost type into one single ALTO Server response.

In this document, an "ALTO Cost Calendar" is specified in terms of information resource capabilities that are applicable to time-sensitive ALTO metrics. An ALTO Cost Calendar exposes ALTO Cost Values in JSON arrays, see [RFC8259], where each value corresponds to a given time interval. The time intervals as well as other Calendar attributes are specified in the Information Resources Directory (IRD) and in the Server response to allow the ALTO Client to interpret the received ALTO values. Last, the extensions for ALTO Calendars are applicable to any Cost Mode and they ensure backwards compatibility with legacy ALTO Clients - those that only support [RFC7285].

In the rest of this document, Section 3 provides the design characteristics. Sections Section 4 and Section 5 define the formal specifications for the IRD and the information resources. IANA, security and operational considerations are addressed respectively in sections Section 6, Section 7 and Section 8.

1.1. Some recent known uses

A potential use case is implementing smart network services that allow applications to dynamically build end-to-end, virtual networks, to satisfy given demands, with no manual intervention. For example, data-transfer automation applications may need a network service to determine on the availability of bandwidth resources, to decide when to transfer their data sets. The SENSE project [SENSE-sdn-e2e-net] supports such applications by requiring that a network provides services such as the Time-Bandwidth-Product (TBP) service, which informs applications of bandwidth availability during a specific time period. ALTO Calendars can support this service if the Calendar start date and duration cover the period of interest of the requesting application.

The need of future scheduling of large scale traffic that can be addressed by the ALTO protocol is also motivated by Unicorn, a unified resource orchestration framework for multi-domain, geo-distributed data analytics, see [Unicorn-fgcs].

1.2. Terminology

- o ALTO transaction: A request/response exchange between an ALTO Client and an ALTO Server.
- o Client: When used with a capital "C", this term refers to an ALTO Client.
- o Calendar, Cost Calendar: When used with capitalized words, these terms refer to an ALTO Cost Calendar.
- o Calendared: this adjective qualifies information resources providing Cost Calendars and information on costs that are provided in the form of a Cost Calendar.
- o Endpoint (EP): An endpoint is defined as in Section 2.1 of [RFC7285]. It can be, for example, a peer, a CDN storage location, a physical server involved in a virtual server-supported application, a party in a resource-sharing swarm such as a computation grid, or an online multi-party game.
- o ECM: Is an abbreviation for Endpoint Cost Map.
- o FCM: Is an abbreviation for Filtered Cost Map.
- o Server: When used with a capital "S", this term refers to an ALTO Server.

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.

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

3. Overview of ALTO Cost Calendars and terminology

This section gives a high-level overview of the design. It assumes the reader is familiar with the ALTO protocol [RFC7285] and its Multi-Cost ALTO extension [RFC8189].

3.1. ALTO Cost Calendar overview

An ALTO Cost Calendar provided by the ALTO Server provides 2 information items:

- o an array of values for a given metric, where each value specifies the metric corresponding to a time interval, where the value array can sometimes be a cyclic pattern that repeats a certain number of times.
- o attributes describing the time scope of the Calendar, including the size and number of the intervals and the date of the starting point of the Calendar, allowing an ALTO Client to interpret the values properly.

An ALTO Cost Calendar can be used like a "time table" to figure out the best time to schedule data transfers and also to proactively manage application traffic given predictable events such as expected spike in traffic due to crowd gathering (concerts, sports, etc.), traffic-intensive holidays and network maintenance. A Calendar may be viewed as a synthetic abstraction of, for example, real measurements gathered over previous periods on which statistics have been computed. However, like for any schedule, unexpected network incidents may require the current ALTO Calendar to be updated and re-sent to the ALTO Clients needing it. The "ALTO Incremental Updates Using Server-Sent Events (SSE)" Service [I-D.ietf-alto-incr-update-sse] can be used to directly update the Calendar upon value changes, if supported by both the Server and the Client.

Most likely, the ALTO Cost Calendar would be used for the Endpoint Cost Service, assuming that a limited set of feasible Endpoints for a non-real time application is already identified, that they do not need to be accessed immediately and that their access can be scheduled within a given time period. The Filtered Cost Map Service is also applicable as long as the size of the Map allows it.

3.2. ALTO Cost Calendar information features

The Calendar attributes are provided in the Information Resources Directory (IRD) and in ALTO Server responses. The IRD announces attributes without date values in its information resources capabilities, whereas attributes with time dependent values are provided in the "meta" section of Server responses. The ALTO Cost Calendar attributes provide the following information:

- o attributes to describe the time scope of the Calendar value array:

- * "time-interval-size": the applicable time interval size for each Calendar value, defined in seconds, that can cover a wide range of values.
- * "number-of-intervals": the number of intervals provided in the Calendar.
- o "calendar-start-time": specifying when the Calendar starts, that is to which date the first value of the Cost Calendar is applicable.
- o "repeated": an optional attribute indicating how many iterations of the provided Calendar will have the same values. The Server may use it to allow the Client to schedule its next request and thus save its own workload by reducing processing of similar requests.

Attribute "repeated" may take a very high value if a Calendar represents a cyclic value pattern that the Server considers valid for a long period. In this case, the Server will only update the Calendar values once this period has elapsed or if an unexpected event occurs on the network. See Section 8 for more discussion.

3.3. ALTO Calendar design characteristics

The present document uses the notations defined in Section "8.2 Notation" of [RFC7285].

The extensions in this document encode requests and responses using JSON [RFC8259].

In the base protocol [RFC7285] section 11.2.3.6, an ALTO cost is specified as a generic JSONValue [RFC8259], to allow extensions. However, that section 11.2.3.6 states: "An implementation of the protocol in this document ([RFC7285]) SHOULD assume that the cost is a JSONNumber and fail to parse if it is not, unless the implementation is using an extension to this document that indicates when and how costs of other data types are signaled".

The present document extends the definition of a legacy cost map given in [RFC7285] to allow a cost entry to be an array of values, with one value per time interval, instead of being just one number, when the Cost Calendar functionality is activated on this cost. Therefore the implementor of this extension MUST consider that a cost entry is an array of values if this cost has been queried as a Calendar.

Specifically, an implementation of this extension MUST parse the "number-of-intervals" attribute of the Calendar attributes in an IRD entry announcing a service providing a Cost Calendar for a given cost type. The implementation then will know that a cost entry of the service will be an array of values, and the expected size of the array is that specified by the "number-of-intervals" attribute. The following rules attempt to ensure consistency between the array size announced by the Server and the actual size of the array received by the Client:

- o The size of the array of values conveyed in a Cost Calendar and received by the Client MUST be equal to the value of attribute "number-of-intervals" indicated in the IRD for the requested cost type.
- o When the size of the array received by the Client is different from the expected size, the Client SHOULD ignore the received array.

To realize an ALTO Calendar, this document extends the IRD and the ALTO requests and responses for Cost Calendars.

This extension is designed to be lightweight and to ensure backwards compatibility with base protocol ALTO Clients and with other extensions. It relies on section 8.3.7 "Parsing of Unknown Fields" of [RFC7285] that writes: "Extensions may include additional fields within JSON objects defined in this document. ALTO implementations MUST ignore unknown fields when processing ALTO messages."

The Calendar-specific capabilities are integrated in the information resources of the IRD and in the "meta" member of ALTO responses to Cost Calendars requests. A Calendar and its capabilities are associated with a given information resource and within this information resource, with a given cost type. This design has several advantages:

- o it does not introduce a new mode,
- o it does not introduce new media types,
- o it allows an ALTO Server to offer, for a cost type, different Calendars with attributes that are specific to the information resources providing a Calendar for this cost type, instead of being globally specific to the cost type.

The applicable Calendared information resources are:

- o the Filtered Cost Map,

- o the Endpoint Cost Map.

The ALTO Server can choose in which frequency it provides cost Calendars to ALTO Clients. It may either provide Calendar updates starting at the request date, or carefully schedule its updates so as to take profit from a potential repetition/periodicity of Calendar values.

Since Calendar attributes are specific to an information resource, a Server may adapt the granularity of the calendared information so as to moderate the volume of exchanged data. For example: suppose a Server provides a Calendar for cost type name "routingcost". The Server may offer a Calendar in a Cost Map resource, which may be a voluminous resource, as an array of 6 intervals lasting each 4 hours. It may also offer a Calendar in an Endpoint Cost Map resource, which is potentially less voluminous, as a finer-grained array of 24 intervals lasting 1 hour each.

The ALTO Server does not support constraints on Calendars, provided Calendars are requested for numerical values, for two main reasons:

- o constraints on an array of values may be various: for instance, some Clients may refuse Calendars with one single value violating a constraint, where as other ones may tolerate Calendars with values violating constraints for example at given times. Therefore, expressing constraints in a way that covers all possible Client preferences is challenging,
- o if constraints were to be supported, the processing overhead would be substantial for the Server as it would have to parse all the values of the Calendar array before returning a response.

As providing the constraint functionality in conjunction with the Calendar functionality is not feasible for the reasons described above, the two features are mutually exclusive. The absence of constraints on Filtered Cost Map and Endpoint Cost Map Calendars reflects a divergence from the non-calendared information resources defined in [RFC7285] and extended in [RFC8189], that support optional constraints.

3.3.1. ALTO Cost Calendar for all cost modes

An ALTO Cost Calendar is well-suited for values encoded in the "numerical" mode. Actually, a Calendar can also represent metrics in other modes considered as compatible with time-varying values. For example, types of Cost values such as JSONBool can also be calendared, as their value may be 'true' or 'false' depending on

given time periods or likewise, values represented by strings, such as "medium", "high", "low", "blue", "open".

Note also that a Calendar is suitable as well for time-varying metrics provided in the "ordinal" mode, if these values are time-varying and the ALTO Server provides updates of cost value based preferences.

3.3.2. Compatibility with legacy ALTO Clients

The ALTO protocol extensions for Cost Calendars have been defined so as to ensure that Calendar-capable ALTO Servers can provide legacy ALTO Clients with legacy information resources as well. That is, a legacy ALTO Client can request resources and receive responses as specified in [RFC7285].

A Calendar-aware ALTO Server MUST implement the base protocol specified in [RFC7285].

A Calendar-aware ALTO Client MUST implement the base protocol specified in [RFC7285].

As a consequence, when a metric is available as a Calendar array, it also MUST be available as a single value as required by [RFC7285]. The Server, in this case, provides the current value of the metric to either Calendar-aware Clients not interested in future or time-based values, or Clients implementing [RFC7285] only.

For compatibility with legacy ALTO Clients specified in [RFC7285], calendared information resources are not applicable for full cost maps for the following reason: a legacy ALTO Client would receive a calendared cost map via an HTTP 'GET' command. As specified in section 8.3.7 of [RFC7285], it will ignore the Calendar Attributes indicated in the "meta" of the responses. Therefore, lacking information on Calendar attributes, it will not be able to correctly interpret and process the values of the received array of Calendar cost values.

Therefore, calendared information resources MUST be requested via the Filtered Cost Map Service or the Endpoint Cost Service, using a POST method.

4. ALTO Calendar specification: IRD extensions

The Calendar attributes in the IRD information resources capabilities carry dateless values. A Calendar is associated with an information resource rather than a cost type. For example, a Server can provide a "routingcost" Calendar for the Filtered Cost Map Service at a

granularity of one day and a "routingcost" Calendar for the Endpoint Cost Service at a finer granularity but for a limited number of endpoints. An example IRD with Calendar specific features is provided in Section 4.3.

4.1. Calendar attributes in the IRD resource capabilities

A Cost Calendar for a given cost type MUST be indicated in the IRD by an object of type CalendarAttributes. A CalendarAttributes object is represented by the "calendar-attributes" member of a resource entry. Member "calendar-attributes" is an array of CalendarAttributes objects. Each CalendarAttributes object lists a set of one or more cost types it applies to. A cost type name MUST NOT appear more than once in the "calendar-attributes" member of a resource entry; multiple appearances of a cost type name in the CalendarAttributes object of the "calendar-attributes" member MUST cause the ALTO Client to ignore any occurrences of this name beyond the first encountered occurrence. The Client SHOULD consider the CalendarAttributes object in the array containing the first encountered occurrence of a cost type as the valid one for this cost type. As an alternative, the Client may want to avoid the risks of erroneous guidance associated to the use of potentially invalid Calendar values. In this case, the Client MAY ignore the totality of occurrences of CalendarAttributes objects containing the cost type name and query the cost type using [RFC7285].

The encoding format for object CalendarAttributes, using JSON [RFC8259], is as follows:

```
CalendarAttributes calendar-attributes <1..*>;
```

```
object{
  JSONString cost-type-names <1..*>;
  JSONNumber time-interval-size;
  JSONNumber number-of-intervals;
} CalendarAttributes;
```

o "cost-type-names":

- * An array of one or more elements indicating the cost type names in the IRD entry to which the values of "time-interval-size" and "number-of-intervals" apply.

o "time-interval-size":

- * is the duration of an ALTO Calendar time interval in a unit of seconds. A "time-interval-size" value contains a non-negative JSONNumber. Example values are: 300 and 7200, meaning that

each Calendar value applies on a time interval that lasts 5 minutes and 2 hours, respectively. Since an interval size (e.g., 100 ms) can be smaller than the unit, the value specified may be a floating point (e.g., 0.1). Both ALTO Clients and Servers should be aware of potential precision issues caused by using floating point numbers; for example, the floating number 0.1 cannot be represented precisely using a finite number of binary bits. To improve interoperability and be consistent with [RFC7285] on the use of floating point numbers, the Server and the Client SHOULD use IEEE 754 double-precision floating point [IEEE.754.2008] to store this value.

- o "number-of-intervals":

- * is a strictly positive integer (greater or equal to 1), that indicates the number of values of the Cost Calendar array.

- An ALTO Server SHOULD specify the "time-interval-size" in the IRD as the smallest it is able to provide. A Client that needs a longer interval can aggregate multiple cost values to obtain it.

- Attribute "cost-type-names" is associated with "time-interval-size" and "number-of-intervals", because multiple cost types may share the same values for attributes "time-interval-size" and "number-of-intervals". To avoid redundancies, cost type names sharing the same values for "time-interval-size" and "number-of-intervals" are grouped in the "cost-type-names" attribute. In the example IRD provided in Section 4.3, the information resource "filtered-cost-map-calendar" provides a Calendar for cost type names "num-routingcost", "num-throughputrating" and "string-servicestatus". Cost type names "num-routingcost" and "num-throughputrating" are grouped in the "cost-type-names" attribute because they share the same values for "time-interval-size" and "number-of-intervals", which are respectively 7200 and 12.

- Multiplying "time-interval-size" by "number-of-intervals" provides the duration of the provided Calendar. For example, an ALTO Server may provide a Calendar for ALTO values changing every "time-interval-size" equal to 5 minutes. If "number-of-intervals" has the value 12, then the duration of the provided Calendar is 1 hour.

4.2. Calendars in a delegate IRD

It may be useful to distinguish IRD resources supported by the base ALTO protocol from resources supported by its extensions. To achieve this, one option, is that a "root" ALTO Server implementing [RFC7285] resources and running at a given domain, delegates "specialized" information resources such as the ones providing Cost Calendars, to

another ALTO Server running in a subdomain. The "root" ALTO Server can provide a Calendar-specific resource entry, that has a media-type of "application/alto-directory+json" and that specifies the URI allowing to retrieve the location of a Calendar-aware Server and discover its resources. This option is described in Section 9.2.4 "Delegation using IRDs" of [RFC7285].

This document provides an example, where a "root" ALTO Server runs in a domain called "alto.example.com". It delegates the announcement of Calendars capabilities to an ALTO Server running in a subdomain called "custom.alto.example.com". The location of the "delegate Calendar IRD" is assumed to be indicated in the "root" IRD by the resource entry: "custom-calendared-resources".

Another benefit of delegation is that some cost types for some resources may be more advantageous as Cost Calendars and it makes little sense to get them as a single value. For example, if a cost type has predictable and frequently changing values, calendared in short time intervals such as a minute, it saves time and network resources to track the cost values via a focused delegate Server rather than the more general "root" Server.

4.3. Example IRD with ALTO Cost Calendars

This section provides an example ALTO Server IRD that supports various cost metrics and cost modes. In particular, since [RFC7285] makes it mandatory, the Server uses metric "routingcost" in the "numerical" mode.

For illustrative purposes, this section introduces 3 other fictitious example metrics and modes that should be understood as examples and should not be used or considered as normative.

The cost type names used in the example IRD are as follows:

- o "num-routingcost": refers to metric "routingcost" in the numerical mode as defined in [RFC7285] and registered with IANA.
- o "num-owdelay": refers to fictitious performance metric "owdelay" in the "numerical" mode, to reflect the one-way packet transmission delay on a path. A related performance metric is currently under definition in [I-D.ietf-alto-performance-metrics].
- o "num-throughputrating": refers to fictitious metric "throughputrating" in the "numerical" mode, to reflect the provider preference in terms of end to end throughput.

- o "string-servicestatus": refers to fictitious metric "servicestatus" containing a string, to reflect the availability, defined by the provider, of for instance path connectivity.

The example IRD includes 2 particular URIs providing Calendars:

- o "https://custom.alto.example.com/calendar/costmap/filtered": a Filtered Cost Map in which Calendar capabilities are indicated for cost type names: "num-routingcost", "num-throughputrating" and "string-servicestatus",
- o "https://custom.alto.example.com/calendar/endpointcost/lookup": an Endpoint Cost Map in which Calendar capabilities are indicated for cost type names: "num-routingcost", "num-owdelay", "num-throughputrating", "string-servicestatus".

The design of the Calendar capabilities allows some Calendars with the same cost type name to be available in several information resources with different Calendar Attributes. This is the case for Calendars on "num-routingcost", "num-throughputrating" and "string-servicestatus", available in both the Filtered Cost map and Endpoint Cost Service, but with different time interval sizes for "num-throughputrating" and "string-servicestatus".

--- Client to Server request for IRD -----

```
GET /calendars-directory HTTP/1.1
Host: custom.alto.example.com
Accept: application/alto-directory+json,application/alto-error+json
```

--- Server response to Client -----

```
HTTP/1.1 200 OK
Content-Length: 2622
Content-Type: application/alto-directory+json
```

```
{
  "meta" : {
    "default-alto-network-map" : "my-default-network-map",
    "cost-types": {
      "num-routingcost": {
        "cost-mode" : "numerical",
        "cost-metric" : "routingcost"
      },
      "num-owdelay": {
        "cost-mode" : "numerical",
        "cost-metric": "owdelay"
      },
    },
  },
}
```

```
    "num-throughputrating": {
      "cost-mode" : "numerical",
      "cost-metric": "throughputrating"
    },
    "string-servicestatus": {
      "cost-mode" : "string",
      "cost-metric": "servicestatus"
    }
  }
},
"resources" : {
  "filtered-cost-map-calendar" : {
    "uri" :
      "https://custom.alto.example.com/calendar/costmap/filtered",
    "media-type" : "application/alto-costmap+json",
    "accepts" : "application/alto-costmapfilter+json",
    "capabilities" : {
      "cost-constraints" : true,
      "cost-type-names" : [ "num-routingcost",
                            "num-throughputrating",
                            "string-servicestatus" ],
      "calendar-attributes" : [
        { "cost-type-names" : [ "num-routingcost",
                                "num-throughputrating" ],
          "time-interval-size" : 7200,
          "number-of-intervals" : 12
        },
        { "cost-type-names" : [ "string-servicestatus" ],
          "time-interval-size" : 1800,
          "number-of-intervals" : 48
        }
      ],
    },
    "uses": [ "my-default-network-map" ]
  },
  "endpoint-cost-map-calendar" : {
    "uri" :
      "https://custom.alto.example.com/calendar/endpointcost/lookup",
    "media-type" : "application/alto-endpointcost+json",
    "accepts" : "application/alto-endpointcostparams+json",
    "capabilities" : {
      "cost-constraints" : true,
      "cost-type-names" : [ "num-routingcost",
                            "num-owdelay",
                            "num-throughputrating",
                            "string-servicestatus" ],
      "calendar-attributes" : [
        { "cost-type-names" : [ "num-routingcost" ],
          "time-interval-size" : 7200,
          "number-of-intervals" : 12
        }
      ],
    },
    "uses": [ "my-default-network-map" ]
  }
}
```

```

        "time-interval-size" : 3600,
        "number-of-intervals" : 24
      },
      { "cost-type-names" : [ "num-owdelay" ],
        "time-interval-size" : 300,
        "number-of-intervals" : 12
      },
      { "cost-type-names" : [ "num-throughputrating" ],
        "time-interval-size" : 60,
        "number-of-intervals" : 60
      },
      { "cost-type-names" : [ "string-servicestatus" ],
        "time-interval-size" : 120,
        "number-of-intervals" : 30
      }
    ]
  }
}
}

```

In this example IRD, for the Filtered Cost Map Service:

- o the Calendar for "num-routingcost" and "num-throughputrating" is an array of 12 values each provided on a time interval lasting 7200 seconds (2 hours).
- o the Calendar for "string-servicestatus": "is an array of 48 values each provided on a time interval lasting 1800 seconds (30 minutes).

For the Endpoint Cost Service:

- o the Calendar for "num-routingcost": is an array of 24 values each provided on a time interval lasting 3600 seconds (1 hour).
- o the Calendar for "num-owdelay": is an array of 12 values each provided on a time interval lasting 300 seconds (5 minutes).
- o the Calendar for "num-throughputrating": is an array of 60 values each provided on a time interval lasting 60 seconds (1 minute).
- o the Calendar for "string-servicestatus": "is an array of 30 values each provided on a time interval lasting 120 seconds (2 minutes).

Note that in this example IRD, member "cost-constraints" is present with a value set to "true" in both information resources "filtered-cost-map-calendar" and "endpoint-cost-map-calendar". Although a

Calendar-aware ALTO Server does not support constraints for the reasons explained in section Section 3.3, it MUST support constraints on cost types that are not requested as Calendars but are requested as specified in [RFC7285] and [RFC8189].

5. ALTO Calendar specification: Service Information Resources

This section documents extensions to two basic ALTO information resources (Filtered Cost Maps and Endpoint Cost Service) to provide calendared information services for them.

Both extensions return calendar start time (calendar-start-time, a point in time), which MUST be specified as an HTTP "Date" header field using the IMF-fixdate format specified in Section 7.1.1.1 of [RFC7231]. Note that the IMF-fixdate format uses "GMT", not "UTC", to designate the time zone, as in this example:

Date: Tue, 15 Nov 2019 08:12:31 GMT

5.1. Calendar extensions for Filtered Cost Maps (FCM)

A legacy ALTO Client requests and gets Filtered Cost Map responses as specified in [RFC7285].

5.1.1. Calendar extensions in Filtered Cost Map requests

The input parameters of a "legacy" request for a Filtered Cost Map, defined by object ReqFilteredCostMap in section 11.3.2 of [RFC7285], are augmented with one additional member. The same augmentation applies to object ReqFilteredCostMap defined in section 4.1.2 of [RFC8189].

A Calendar-aware ALTO Client requesting a Calendar on a given Cost Type for a Filtered Cost Map resource having Calendar capabilities MUST add the following field to its input parameters:

JSONBoolean calendared<1..*>;

This field is an array of 1 to N boolean values, where N is the number of requested metrics. N is greater than 1 when the Client and the Server also implement [RFC8189].

Each entry corresponds to the requested metric at the same array position. Each boolean value indicates whether or not the ALTO Server should provide the values for this cost type as a Calendar. The array MUST contain exactly N boolean values, otherwise, the Server returns an error.

This field MUST NOT be included if no member "calendar-attributes" is specified in this information resource.

If a value of field "calendared" is 'true' for a cost type name for which no Calendar attributes have been specified: an ALTO Server, whether it implements the extensions of this document or only implements [RFC7285], MUST ignore it and return a response with a single cost value as specified in [RFC7285].

If this field is not present, it MUST be assumed to have only values equal to 'false'.

A Calendar-aware ALTO Client that supports requests for only one cost type at a time and wants to request a Calendar MUST provide an array of 1 element:

```
"calendared" : [true],
```

A Calendar-aware ALTO Client that supports requests for more than one cost types at a time, as specified in [RFC8189] MUST provide an array of N values set to 'true' or 'false', depending whether it wants the applicable cost type values as a single or calendared value.

5.1.2. Calendar extensions in Filtered Cost Map responses

In a calendared ALTO Filtered Cost Map, a cost value between a source and a destination is a JSON array of JSON values. An ALTO Calendar values array has a number of values equal to the value of member "number-of-intervals" of the Calendar attributes that are indicated in the IRD. These attributes will be conveyed as metadata in the Filtered Cost Map response. Each element of the array is valid for the time-interval that matches its array position.

The FCM response conveys metadata among which:

- o some are not specific to Calendars and ensure compatibility with [RFC7285] and [RFC8189]
- o some are specific to Calendars.

The non Calendar-specific "meta" fields of a calendared Filtered Cost Map response MUST include at least:

- o if the ALTO Client requests cost values for one cost type at a time only: the "meta" fields specified in [RFC7285] for these information service responses:
 - * "dependent-vtags ",

- * "cost-type" field.
- o if the ALTO Client implements the Multi-Cost ALTO extension specified in [RFC8189] and requests cost values for several cost types at a time: the "meta" fields specified in [RFC8189] for these information service responses:
 - * "dependent-vtags ",
 - * "cost-type" field with value set to '{}', for backwards compatibility with [RFC7285].
 - * "multi-cost-types" field.

If the Client request does not provide member "calendared" or if it provides it with a value equal to 'false', for all the requested cost types, then the ALTO Server response is exactly as specified in [RFC7285] and [RFC8189].

If the value of member "calendared" is equal to 'false' for a given requested cost type, the ALTO Server MUST return, for this cost type, a single cost value as specified in [RFC7285].

If the value of member "calendared" is equal to 'true' for a given requested cost type, the ALTO Server returns, for this cost type, a cost value Calendar as specified above in this section. In addition to the above cited non Calendar specific "meta" members, the Server MUST provide a Calendar specific metadata field.

The Calendar-specific "meta" field that a calendared Filtered Cost Map response MUST include is a member called "calendar-response-attributes", that describes properties of the Calendar and where:

- o member "calendar-response-attributes" is an array of one or more objects of type "CalendarResponseAttributes".
- o each "CalendarResponseAttributes" object in the array is specified for one or more cost types for which the value of member "calendared", in object ReqFilteredCostMap provided in the Client request, is equal to 'true' and for which a Calendar is provided for the requested information resource.
- o the "CalendarResponseAttributes" object that applies to a cost type name has a corresponding "CalendarAttributes" object defined for this cost type name in the IRD capabilities of the requested information resource. This object is the entry, in the "calendar-attributes" array member of the IRD resource entry, that includes the name of the requested cost type. This corresponding

"CalendarAttributes" object has the same values as object "CalendarResponseAttributes" for members "time-interval-size" and "number-of-intervals". The members of the "CalendarResponseAttributes" object include all the members of the corresponding "CalendarAttributes" object.

The format of member "CalendarResponseAttributes" is defined as follows:

```
CalendarResponseAttributes calendar-response-attributes <1..*>;
```

```
object{  
  [JSONString cost-type-names <1..*>;]  
  JSONString calendar-start-time;  
  JSONNumber time-interval-size;  
  JSONNumber number-of-intervals;  
  [JSONNumber repeated;]  
} CalendarResponseAttributes;
```

Object CalendarResponseAttributes has the following attributes:

- o "cost-type-names": is an array of one or more cost-type-names to which the value of the other members of CalendarResponseAttributes apply and for which a Calendar has been requested. The value of this member is a subset of the "cost-type-names" member of the abovementioned corresponding "CalendarAttributes" object in the "calendar-attributes" array member in the IRD. This member MUST be present when Cost Calendars are provided for more than one cost type.
- o "calendar-start-time": indicates the date at which the first value of the Calendar applies. The value is a string that, as specified in Section 5, contains an HTTP "Date" header field using the IMF-fixdate format specified in Section 7.1.1.1 of [RFC7231]. The value provided for attribute "calendar-start-time" SHOULD NOT be later than the request date.
- o "time-interval-size": as specified in Section 4.1 and with the same value as in the abovementioned corresponding "CalendarAttributes" object.
- o "number-of-intervals": as specified in Section 4.1 and with the same value as in the abovementioned corresponding "CalendarAttributes" object.
- o "repeated": is an optional field provided for Calendars. It is an integer N greater or equal to '1' that indicates how many iterations of the Calendar value array starting at the date

indicated by "calendar-start-time" have the same values. The number N includes the iteration provided in the returned response.

For example: suppose the "calendar-start-time" member has value "Mon, 30 Jun 2019 00:00:00 GMT", the "time-interval-size" member has value '3600', the "number-of-intervals" member has value '24' and the value of member "repeated" is equal to '4'. This means that the Calendar values are the same on Monday, Tuesday, Wednesday and Thursday on a period of 24 hours starting at 00:00:00 GMT. The ALTO Client thus may use the same Calendar for the next 4 days starting at "calendar-start-time" and will only need to request a new one for Friday July 4th at 00:00:00 GMT.

Attribute "repeated" may take a very high value if a Calendar represents a cyclic value pattern that the Server considers valid for a long period and hence will only update once this period has elapsed or if an unexpected event occurs on the network. In the latter case, the Client will be notified if it uses the "ALTO Incremental Updates Using Server-Sent Events (SSE)" Service, specified in [I-D.ietf-alto-incr-update-sse]. To this end, it is RECOMMENDED that ALTO Servers providing ALTO Calendars also provide the "ALTO Incremental Updates Using Server-Sent Events (SSE)" Service that is specified in [I-D.ietf-alto-incr-update-sse]. Likewise, ALTO Clients capable of using ALTO Calendars SHOULD also use the SSE Service. See also discussion in Section 8 "Operational Considerations".

5.1.3. Use case and example: FCM with a bandwidth Calendar

An example of non-real-time information that can be provisioned in a Calendar is the expected path throughput. While the transmission rate can be measured in real time by end systems, the operator of a data center is in the position of formulating preferences for given paths, at given time periods to avoid traffic peaks due to diurnal usage patterns. In this example, we assume that an ALTO Client requests a Calendar of network-provider-defined throughput ratings, as specified in the IRD, to schedule its bulk data transfers as described in the use cases.

In the example IRD, Calendars for cost type name "num-throughputrating" are available for the information resources: "filtered-cost-calendar-map" and "endpoint-cost-map-calendar". The ALTO Client requests a Calendar for "num-throughputrating" via a POST request for a Filtered Cost Map.

We suppose in the present example that the ALTO Client sends its request on Tuesday July 1st 2019 at 13:15. The Server returns Calendars with arrays of 12 numbers for each source and destination

pair. The values for metric "throughputrating", in this example, are assumed to be encoded in 2 digits.

```
POST /calendar/costmap/filtered HTTP/1.1
Host: custom.alto.example.com
Content-Length: 217
Content-Type: application/alto-costmapfilter+json
Accept: application/alto-costmap+json,application/alto-error+json
```

```
{
  "cost-type" : {"cost-mode" : "numerical",
                 "cost-metric" : "throughputrating"},
  "calendared" : [true],
  "pids" : {
    "srcs" : [ "PID1", "PID2" ],
    "dsts" : [ "PID1", "PID2", "PID3" ]
  }
}
```

```
HTTP/1.1 200 OK
Content-Length: 1043
Content-Type: application/alto-costmap+json
```

```
{
  "meta" : {
    "dependent-vtags" : [
      {"resource-id": "my-default-network-map",
       "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"}
    ],
    "cost-type" : {"cost-mode" : "numerical",
                   "cost-metric" : "throughputrating"},
    "calendar-response-attributes" : [
      {"calendar-start-time" : "Tue, 1 Jul 2019 13:00:00 GMT",
       "time-interval-size" : 7200,
       "number-of-intervals" : 12}
    ]
  },
  "cost-map" : {
    "PID1": { "PID1": [ 1, 12, 14, 18, 14, 14,
                       14, 18, 19, 20, 11, 12],
              "PID2": [13,  4, 15, 16, 17, 18,
                       19, 20, 11, 12, 13, 14],
              "PID3": [20, 20, 18, 14, 12, 12,
                       14, 14, 12, 12, 14, 16] },
    "PID2": { "PID1": [17, 18, 19, 10, 11, 12,
                       13, 14, 15, 16, 17, 18],
              "PID2": [20, 20, 18, 16, 14, 14,
```

```

        14, 16, 16, 16, 14, 16],
    "PID3": [20, 20, 18, 14, 12, 12,
             14, 14, 12, 12, 14, 16] }
  }
}
```

5.2. Calendar extensions in the Endpoint Cost Service

This document extends the Endpoint Cost Service, as defined in {11.5.1} of [RFC7285], by adding new input parameters and capabilities, and by returning JSONArrays instead of JSONNumbers as the cost values. The media type {11.5.1.1} and HTTP method {11.5.1.2} are unchanged.

5.2.1. Calendar specific input in Endpoint Cost requests

The extensions to the requests for calendared Endpoint Cost Maps are the same as for the Filtered Cost Map Service, specified in Section 5.1.1 of this document. Likewise, the rules defined around the extensions to ECM requests are the same as those defined in Section 5.1.1 for FCM requests.

The ReqEndpointCostMap object for a calendared ECM request will have the following format:

```

object {
  [CostType cost-type;]
  [CostType multi-cost-types<1..*>;]
  [JSONBoolean calendared<1..*>;]
  EndpointFilter endpoints;
} ReqEndpointCostMap;

object {
  [TypedEndpointAddr srcs<0..*>;]
  [TypedEndpointAddr dsts<0..*>;]
} EndpointFilter;
```

Member "cost-type" is optional because, in the ReqEndpointCostMap object definition of this document, it is jointly present with member "multi-cost-types", to ensure compatibility with RFC 8189. In RFC8189, members "cost-type" and "multi-cost-types" are both optional and have to obey the rule specified in section 4.1.2 of 8189 saying that: "the Client MUST specify either "cost-type" or "multi-cost-types" but MUST NOT specify both".

The interpretation of member "calendared" is the same as for the ReqFilteredCostMap object defined in Section 5.1.1 of this document. The interpretation of the other members is the same as for object

ReqEndpointCostMap is defined in [RFC7285] and [RFC8189]. The type TypedEndpointAddr is defined in section 10.4.1 of [RFC7285].

For the reasons explained in section Section 3.3, a Calendar-aware ALTO Server does not support constraints. Therefore, member "[constraints]" is not present in the ReqEndpointCostMap object and member "constraints" MUST NOT be present in the input parameters of a request for an Endpoint Cost Calendar. If this member is present, the Server MUST ignore it.

5.2.2. Calendar attributes in the Endpoint Cost response

The "meta" field of a calendared Endpoint Cost response MUST include at least:

- o if the ALTO Client supports cost values for one cost type at a time only: the "meta" fields specified in {11.5.1.6} of [RFC7285] for the Endpoint Cost response:
 - * "cost-type" field.
- o if the ALTO Client supports cost values for several cost types at a time, as specified in [RFC8189] : the "meta" fields specified in [RFC8189] for the the Endpoint Cost response:
 - * "cost-type" field with value set to '{}', for backwards compatibility with [RFC7285].
 - * "multi-cost-types" field.

If the Client request does not provide member "calendared" or if it provides it with a value equal to 'false', for all the requested Cost Types, then the ALTO Server response is exactly as specified in [RFC7285] and [RFC8189].

If the ALTO Client provides member "calendared" in the input parameters with a value equal to 'true' for given requested Cost Types, the "meta" member of a calendared Endpoint Cost response MUST include, for these cost types, an additional member "calendar-response-attributes", the contents of which obey the same rules as for the Filtered Cost Map Service, specified in Section 5.1.2. The Server response is thus changed as follows, with respect to [RFC7285] and [RFC8189]:

- o the "meta" member has one additional field "CalendarResponseAttributes", as specified for the Filtered Cost Map Service,

- o the calendared costs are JSONArrays instead of the JSONNumbers format used by legacy ALTO implementations. All arrays have a number of values equal to 'number-of-intervals'. Each value corresponds to the cost in that interval.

If the value of member "calendared" is equal to 'false' for a given requested cost type, the ALTO Server MUST return, for this cost type, a single cost value as specified in [RFC7285].

5.2.3. Use case and example: ECS with a routingcost Calendar

Let us assume an Application Client is located in an end system with limited resources and having access to the network that is either intermittent or provides an acceptable quality in limited but predictable time periods. Therefore, it needs to schedule both its resource-greedy networking activities and its ALTO transactions.

The Application Client has the choice to trade content or resources with a set of Endpoints and needs to decide with which one it will connect and at what time. For instance, the Endpoints are spread in different time-zones, or have intermittent access. In this example, the 'routingcost' is assumed to be time-varying, with values provided as ALTO Calendars.

The ALTO Client associated with the Application Client queries an ALTO Calendar on 'routingcost' and will get the Calendar covering the 24 hours time period "containing" the date and time of the ALTO Client request.

For cost type "num-routingcost", the solicited ALTO Server has defined 3 different daily patterns each represented by a Calendar, to cover the week of Monday June 30th at 00:00 to Sunday July 6th 23:59:

- o C1 for Monday, Tuesday, Wednesday, Thursday, (weekdays)
- o C2 for Saturday, Sunday, (weekend)
- o C3 for Friday (maintenance outage on July 4, 2019 from 02:00:00 GMT to 04:00:00 GMT, or big holiday such as New Year evening).

In the following example, the ALTO Client sends its request on Tuesday July 1st 2019 at 13:15.

The "routingcost" values are assumed to be encoded in 3 digits.

```
POST /calendar/endpointcost/lookup HTTP/1.1
Host: custom.alto.example.com
Content-Length: 304
```

Content-Type: application/alto-endpointcostparams+json
Accept: application/alto-endpointcost+json,application/alto-error+json

```
{
  "cost-type" : {"cost-mode" : "numerical",
                 "cost-metric" : "routingcost"},
  "calendared" : [true],
  "endpoints" : {
    "srcs": [ "ipv4:192.0.2.2" ],
    "dsts": [
      "ipv4:192.0.2.89",
      "ipv4:198.51.100.34",
      "ipv4:203.0.113.45",
      "ipv6:2001:db8::10"
    ]
  }
}
```

HTTP/1.1 200 OK

Content-Length: 1351
Content-Type: application/alto-endpointcost+json

```
{
  "meta" : {
    "cost-type" : {"cost-mode" : "numerical",
                   "cost-metric" : "routingcost"},
    "calendar-response-attributes" : [
      {"calendar-start-time" : "Mon, 30 Jun 2019 00:00:00 GMT",
       "time-interval-size" : 3600,
       "number-of-intervals" : 24,
       "repeated": 4
      }
    ]
  },
  "endpoint-cost-map" : {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89" : [100, 100, 100, 100, 100, 150,
                           200, 300, 300, 300, 300, 250,
                           250, 300, 300, 300, 300, 300,
                           400, 250, 250, 200, 150, 150],
      "ipv4:198.51.100.34" : [ 80, 80, 80, 80, 150, 150,
                               250, 400, 400, 450, 400, 200,
                               200, 350, 400, 400, 400, 350,
                               500, 200, 200, 200, 100, 100],
      "ipv4:203.0.113.45" : [300, 400, 250, 250, 200, 150,
                              150, 100, 100, 100, 100, 100,
                              100, 100, 100, 100, 100, 150,

```

```

        200, 300, 300, 300, 300, 250],
    "ipv6:2001:db8::10" : [200, 250, 300, 300, 300, 300,
        250, 300, 300, 300, 300, 350,
        300, 400, 250, 150, 100, 100,
        100, 150, 200, 250, 250, 300]
    }
}
}

```

When the Client gets the Calendar for "routingcost", it sees that the "calendar-start-time" is Monday at 00h00 GMT and member "repeated" is equal to '4'. It understands that the provided values are valid until Thursday included and will only need to get a Calendar update on Friday.

5.2.4. Use case and example: ECS with a multi-cost Calendar for routingcost and owdelay

In this example, it is assumed that the ALTO Server implements multi-cost capabilities, as specified in [RFC8189]. That is, an ALTO Client can request and receive values for several cost types in one single transaction. An illustrating use case is a path selection done on the basis of 2 metrics: routing cost and owdelay.

As in the previous example, the IRD indicates that the ALTO Server provides "routingcost" Calendars in terms of 24 time intervals of 1 hour (3600 seconds) each.

For metric "owdelay", the IRD indicates that the ALTO Server provides Calendars in terms of 12 time intervals values lasting each 5 minutes (300 seconds).

In the following example transaction, the ALTO Client sends its request on Tuesday July 1st 2019 at 13:15.

This example assumes that the values of metric "owdelay" and "routingcost" are encoded in 3 digits.

```

POST calendar/endpointcost/lookup HTTP/1.1
Host: custom.alto.example.com
Content-Length: 390
Content-Type: application/alto-endpointcostparams+json
Accept: application/alto-endpointcost+json,application/alto-error+json

{
  "cost-type" : {},
  "multi-cost-types" : [
    {"cost-mode" : "numerical", "cost-metric" : "routingcost"},

```

```

    {"cost-mode" : "numerical", "cost-metric" : "owdelay"}
  ],
  "calendared" : [true, true],
  "endpoints" : {
    "srcs": [ "ipv4:192.0.2.2" ],
    "dsts": [
      "ipv4:192.0.2.89",
      "ipv4:198.51.100.34",
      "ipv4:203.0.113.45",
      "ipv6:2001:db8::10"
    ]
  }
}

```

HTTP/1.1 200 OK

Content-Length: 2165

Content-Type: application/alto-endpointcost+json

```

{
  "meta" : {
    "multi-cost-types" : [
      {"cost-mode" : "numerical", "cost-metric" : "routingcost"},
      {"cost-mode" : "numerical", "cost-metric" : "owdelay"}
    ],
    "calendar-response-attributes" : [
      {"cost-type-names" : [ "num-routingcost" ],
        "calendar-start-time" : "Mon, 30 Jun 2019 00:00:00 GMT",
        "time-interval-size" : 3600,
        "number-of-intervals" : 24,
        "repeated": 4 },
      {"cost-type-names" : [ "num-owdelay" ],
        "calendar-start-time" : "Tue, 1 Jul 2019 13:00:00 GMT",
        "time-interval-size" : 300,
        "number-of-intervals" : 12}
    ]
  },
  "endpoint-cost-map" : {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89" : [[100, 100, 100, 100, 100, 150,
        200, 300, 300, 300, 300, 250,
        250, 300, 300, 300, 300, 300,
        400, 250, 250, 200, 150, 150],
        [ 20, 400, 20, 80, 80, 90,
        100, 90, 60, 40, 30, 20]],
      "ipv4:198.51.100.34" : [[ 80, 80, 80, 80, 150, 150,
        250, 400, 400, 450, 400, 200,
        200, 350, 400, 400, 400, 350,
        500, 200, 200, 200, 100, 100],

```



```

        [ 20, 20, 50, 30, 30, 30,
          30, 40, 40, 30, 20, 20]],
    "ipv4:203.0.113.45" : [[300, 400, 250, 250, 200, 150,
        150, 100, 100, 100, 100, 100,
        100, 100, 100, 100, 100, 150,
        200, 300, 300, 300, 300, 250],
        [100, 90, 80, 60, 50, 50,
          40, 40, 60, 90, 100, 80]],
    "ipv6:2001:db8::10" : [[200, 250, 300, 300, 300, 300,
        250, 300, 300, 300, 300, 350,
        300, 400, 250, 150, 100, 100,
        100, 150, 200, 250, 250, 300],
        [ 40, 40, 40, 40, 50, 50,
          50, 20, 10, 15, 30, 40]]
    }
}
}

```

When receiving the response, the Client sees that the Calendar values for metric "routingcost" are repeated for 4 iterations. Therefore, in its next requests until the "routingcost" Calendar is expected to change, the Client will only need to request a Calendar for "owdelay".

Without the ALTO Calendar extensions, the ALTO Client would have no clue on the dynamicity of the metric value change and would spend needless time requesting values at an inappropriate pace. In addition, without the Multi-Cost ALTO capabilities, the ALTO Client would duplicate this waste of time as it would need to send one request per cost metric.

6. IANA Considerations

This document does not define any new media types or introduce any new IANA considerations.

7. Security Considerations

As an extension of the base ALTO protocol [RFC7285], this document fits into the architecture of the base protocol, and hence the Security Considerations (Section 15) of [RFC7285] fully apply when this extension is provided by an ALTO Server. For example, the same authenticity and integrity considerations (Section 15.1 of [RFC7285]) still fully apply; the same considerations for the privacy of ALTO users (Section 15.4 of [RFC7285]) also still fully apply.

The calendaring information provided by this extension requires additional considerations on three security considerations discussed

in [RFC7285]: potential undesirable guidance to Clients (Section 15.2 of [RFC7285]), confidentiality of ALTO information (Section 15.2 of [RFC7285]), and availability of ALTO (Section 15.5 of [RFC7285]). For example, by providing network information in the future in a Calendar, this extension may improve availability of ALTO, when the ALTO Server is unavailable but related information is already provided in the Calendar.

For confidentiality of ALTO information, an operator should be cognizant that this extension may introduce a new risk: a malicious ALTO Client may get information for future events that are scheduled through Calendaring. Possessing such information, the malicious Client may use it to generate massive connections to the network at times where its load is expected to be high.

To mitigate this risk, the operator should address the risk of ALTO information being leaked to malicious Clients or third parties. As specified in Section 15.3.2 ("Protection Strategies") of [RFC7285], the ALTO Server should authenticate ALTO Clients and use the Transport Layer Security (TLS) protocol so that Man In The Middle (MITM) attacks to intercept an ALTO Calendar are not possible. [RFC7285] ensures the availability of such a solution in its Section 8.3.5. "Authentication and Encryption", which specifies that: "ALTO Server implementations as well as ALTO Client implementations MUST support the "https" URI scheme of [RFC2818] and Transport Layer Security (TLS) of [RFC5246]".

[RFC8446] specifies TLS 1.3 and writes in its section 1: "While TLS 1.3 is not directly compatible with previous versions, all versions of TLS incorporate a versioning mechanism which allows Clients and Servers to interoperably negotiate a common version if one is supported by both peers". ALTO Clients and Servers SHOULD support both TLS 1.3 [RFC8446] and TLS 1.2 [RFC5246], and MAY support and use newer versions of TLS as long as the negotiation process succeeds.

The operator should be cognizant that the preceding mechanisms do not address all security risks. In particular, they will not help in the case of "malicious Clients" possessing valid authentication credentials. The threat here is that legitimate Clients have become subverted by an attacker and are now 'bots' being asked to participate in a DDoS attack. The Calendar information now becomes valuable in knowing exactly when to perpetrate a DDoS attack. A mechanism such as a monitoring system that detects abnormal behaviors may still be needed.

To avoid malicious or erroneous guidance from ALTO information, an ALTO Client should be cognizant that using calendaring information can have risks: (1) Calendar values, especially in "repeated"

Calendars may be only statistical, and (2) future events may change. Hence, a more robust ALTO Client should adapt and extend protection strategies specified in Section 15.2 of [RFC7285]. For example, to be notified immediately when a particular ALTO value that the Client depends on changes, it is RECOMMENDED that both the ALTO Client and ALTO Server using this extension support "ALTO Incremental Updates Using Server-Sent Events(SSE)" Service [I-D.ietf-alto-incr-update-sse].

Another risk of erroneous guidance appears when the Server exposes an occurrence of a same cost type name in different elements of the Calendar objects array associated to an information resource. In this case, there is no way for the Client to figure out which Calendar object in the array is valid. The specification in this document recommends, in this case, that the Client uses the first encountered Calendar object occurrence containing the cost type name. However, the Client may want to avoid the risks of erroneous guidance associated to the use of potentially invalid Calendar values. To this end, as an alternative to the recommendation in this document, the Client MAY ignore the totality of occurrences of CalendarAttributes objects containing the cost type name and query this cost type using [RFC7285].

8. Operational Considerations

It is important that both the operator of the network and the operator of the applications consider both the feedback aspect and the prediction-based (uncertainty) aspect of using the Cost Calendar.

First consider the feedback aspect and consider the Cost Calendar as a traffic-aware map service (e.g., Google Maps). Using the service without considering its own effect, a large fleet can turn a not-congested road into a congested one; a large number of individual cars each choosing a road with light traffic ("cheap link") can also result in congestion or result in a less optimal global outcome (e.g., the Braess' Paradox [Braess-paradox]).

Next consider the prediction aspect. Conveying ALTO Cost Calendars tends to reduce the on-the-wire data exchange volume compared to multiple single cost ALTO transactions. An application using Calendars has a set of time-dependent values upon which it can plan its connections in advance with no need for the ALTO Client to query information at each time. Additionally, the Calendar response attribute "repeated", when provided, saves additional data exchanges in that it indicates that the ALTO Client does not need to query Calendars during a period indicated by this attribute. The preceding is true only when "accidents" do not happen.

Although individual network operators and application operators can choose their own approaches to address the aforementioned issues, this document recommends the following considerations. First, a typical approach to reducing instability and handling uncertainty is to ensure timely update of information. The SSE Service as discussed in Section 7 can handle updates, if supported by both the Server and the Client. Second, when a network operator updates the Cost Calendar and when an application reacts to the update, they should consider the feedback effects. This is the best approach even though there is theoretical analysis [Selfish-routing-Roughgarden-thesis] and Internet based evaluation [Selfish-routing-Internet-eval] showing that uncoordinated behaviors do not always cause substantial sub-optimal results.

High-resolution intervals may be needed when values change, sometimes during very small time intervals but in a significant manner. A way to avoid conveying too many entries is to leverage on the "repeated" feature. A Server can smartly set the Calendar start time and number of intervals so as to declare them "repeated" for a large number of periods, until the Calendar values change and are conveyed to requesting Clients.

The newer JSON Data Interchange Format specification [RFC8259] used in ALTO Calendars replaces the older one [RFC7159] used in the base ALTO protocol [RFC7285]. The newer JSON mandates UTF-8 text encoding to improve interoperability. Therefore, ALTO Clients and Servers implementations using UTF-{16,32} need to be cognizant of the subsequent interoperability risks and MUST switch to UTF-8 encoding, if they want to interoperate with Calendar-aware Servers and Clients.

9. Acknowledgements

The authors would like to thank Fred Baker, Li Geng, Diego Lopez, He Peng and Haibin Song for fruitful discussions and feedback on earlier draft versions. Dawn Chan, Kai Gao, Vijay Gurbani, Yichen Qian, Juergen Schoenwaelder, and Brian Weis and Jensen Zhang provided substantial review feedback and suggestions to the protocol design.

10. References

10.1. 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>>.

- [RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <<https://www.rfc-editor.org/info/rfc7231>>.
- [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>>.
- [RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <<https://www.rfc-editor.org/info/rfc8259>>.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <<https://www.rfc-editor.org/info/rfc5246>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.
- [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-20 (work in progress), February 2020.
- [IEEE.754.2008]
"Standard for Binary Floating-Point Arithmetic, IEEE Standard 754", August 2008.

10.2. Informative References

- [RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <<https://www.rfc-editor.org/info/rfc2818>>.

- [RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement", RFC 5693, DOI 10.17487/RFC5693, October 2009, <<https://www.rfc-editor.org/info/rfc5693>>.
- [RFC6708] Kiesel, S., Ed., Previdi, S., Stiemerling, M., Woundy, R., and Y. Yang, "Application-Layer Traffic Optimization (ALTO) Requirements", RFC 6708, DOI 10.17487/RFC6708, September 2012, <<https://www.rfc-editor.org/info/rfc6708>>.
- [RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <<https://www.rfc-editor.org/info/rfc7159>>.
- [I-D.ietf-alto-performance-metrics]
WU, Q., Yang, Y., Dhody, D., Randriamasy, S., and L. Contreras, "ALTO Performance Cost Metrics", draft-ietf-alto-performance-metrics-09 (work in progress), March 2020.
- [I-D.xiang-alto-multidomain-analytics]
Xiang, Q., Zhang, J., Le, F., Yang, Y., and H. Newman, "Resource Orchestration for Multi-Domain, Exascale, Geo-Distributed Data Analytics", draft-xiang-alto-multidomain-analytics-03 (work in progress), March 2020.
- [SENSE-sdn-e2e-net]
"SDN for End-to-End Networked Science at the Exascale (SENSE)", <http://sense.es.net/overview>".
- [Braess-paradox]
Steinberg, R. and W. Zangwill, "The Prevalence of Braess' Paradox", Transportation Science Vol. 17 No. 3, August 1983.
- [Unicorn-fgcs]
Xiang, Q., Wang, T., Zhang, J., Newman, H., and Y. Liu, "Unicorn: Unified resource orchestration for multi-domain, geo-distributed data analytics", Future Generation of Computer Systems (FGCS) Volume 93, Pages 188-197, April 2019.
- [Selfish-routing-Roughgarden-thesis]
Roughgarden, T., "Selfish Routing", Dissertation Thesis, Cornell 2002, May 2002.

[Selfish-routing-Internet-eval]

Qiu, L., Yang, Y., Zhang, Y., and S. Shenker, "Selfish Routing in Internet-Like Environments", Proceedings of ACM SIGCOMM 2001, August 2001.

Authors' Addresses

Sabine Randriamasy
Nokia Bell Labs
Route de Villejust
NOZAY 91460
FRANCE

Email: Sabine.Randriamasy@nokia-bell-labs.com

Richard Yang
Yale University
51 Prospect st
New Haven, CT 06520
USA

Email: yry@cs.yale.edu

Qin Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: sunseawq@huawei.com

Lingli Deng
China Mobile
China

Email: denglingli@chinamobile.com

Nico Schwan
Thales Deutschland
Lorenzstrasse 10
Stuttgart 70435
Germany

Email: nico.schwan@thalesgroup.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 29, 2017

S. Randriamasy
W. Roome
Nokia Bell Labs
N. Schwan
Thales Deutschland
April 27, 2017

Multi-Cost ALTO
draft-ietf-alto-multi-cost-10

Abstract

The ALTO (Application Layer-Traffic Optimization) Protocol ([RFC7285]) defines several services that return various metrics describing the costs between network endpoints.

This document defines a new service that allows an ALTO Client to retrieve several cost metrics in a single request for an ALTO Filtered Cost Map and Endpoint Cost Map. In addition, it extends the constraints to further filter those maps by allowing a client to specify a logical combination of tests on several cost metrics.

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 October 29, 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	3
1.1. Requirements Language	4
2. Terminology	4
3. Overview Of Approach	5
3.1. Multi-Cost Data Format	5
3.2. Compatibility With Legacy ALTO Clients	5
3.3. Filtered Multi Cost Map Resources	6
3.4. Endpoint Cost Service Resources	6
3.5. Full Cost Map Resources	7
3.6. Extended Constraint Tests	7
3.6.1. Extended constraint predicates	7
3.6.2. Extended logical combination of predicates	7
3.6.3. Testable Cost Types in constraints	8
3.6.4. Testable Cost Type Names in IRD capabilities	9
3.6.5. Legacy ALTO Client issues	9
4. Protocol Extensions for Multi-Cost ALTO Transactions	11
4.1. Filtered Cost Map Extensions	11
4.1.1. Capabilities	11
4.1.2. Accept Input Parameters	12
4.1.3. Response	15
4.2. Endpoint Cost Service Extensions	15
4.2.1. Capabilities	16
4.2.2. Accept Input Parameters	16
4.2.3. Response	17
5. Examples	17
5.1. Information Resource Directory	17
5.2. Multi-Cost Filtered Cost Map: Example #1	19
5.3. Multi-Cost Filtered Cost Map: Example #2	20
5.4. Multi-Cost Filtered Cost Map: Example #3	22
5.5. Multi-Cost Filtered Cost Map: Example #4	23
5.6. Endpoint Cost Service	24
6. IANA Considerations	25
7. Privacy And Security Considerations	26
8. Acknowledgements	26
9. References	26
9.1. Normative References	26
9.2. Informative References	26
Authors' Addresses	27

1. Introduction

IETF has defined ALTO services in [RFC7285] to provide guidance to overlay applications, which have to select one or several hosts from a set of candidates that are able to provide a desired resource. This guidance is based on parameters such as the topological distance, that affect performance of the data transmission between the hosts. The purpose of ALTO is to improve Quality of Experience (QoE) in the application while reducing resource consumption in the underlying network infrastructure. The ALTO protocol conveys a view of the Internet called a Network Map and composed of Provider defined locations spanning from subnets to several Autonomous Systems (AS). ALTO may also convey the Provider determined Costs between Network Map locations or between groups of individual endpoints.

Current ALTO Cost Types provide values such as hopcount and administrative routing cost to reflect ISP routing preferences. Recently, new use cases have extended the usage scope of ALTO to Content Delivery Networks (CDN), Data Centers and applications that need additional information to select their endpoints or network locations. Thus a multitude of new Cost Types that better reflect the requirements of these applications are expected to be specified.

The ALTO protocol [RFC7285], which this document refers to as the base protocol, restricts ALTO Cost Maps and Endpoint Cost Services to only one Cost Type per ALTO request. To retrieve information for several Cost Types, an ALTO Client must send several separate requests to the Server.

It is far more efficient, in terms of Round Trip Time (RTT), traffic, and processing load on the ALTO Client and Server, to get all costs with a single query/response transaction. One Cost Map reporting on N Cost Types is less bulky than N Cost Maps containing one Cost Type each. This is valuable for both the storage of these maps and their transmission. Additionally, for many emerging applications that need information on several Cost Types, having them gathered in one map will save time. Another advantage is consistency: providing values for several Cost Types in one single batch is useful for ALTO Clients needing synchronized ALTO information updates. This document defines how to retrieve multiple cost metrics in a single request for ALTO Filtered Cost Maps and Endpoint Cost Maps. To ensure compatibility with legacy ALTO Clients, only the Filtered Cost Map and Endpoint Cost Map services are extended to return Multi-Cost values.

Along with multi-cost values queries, the filtering capabilities need to be extended to allow constraints on multiple metrics. The base protocol allows an ALTO Client to provide optional constraint tests for a Filtered Cost Map or the Endpoint Cost Service, where the

constraint tests are limited to the AND-combination of comparison tests on the value of the (single) requested Cost Type. However, applications that are sensitive to several metrics and struggle with complicated network conditions may need to arbitrate between conflicting objectives such as routing cost and network performance. To this end, this document extends the base protocol with constraints that may test multiple metrics and may be combined with logical 'ORs' as well as logical 'ANDs'. This allows an application to make requests such as: "select solutions with either (moderate "hopcount" AND high "routingcost") OR (higher "hopcount" AND moderate "routingcost")".

This document is organized as follows: Section 2 defines terminology used in this document. Section 3 gives a non-normative overview of the multi-cost extensions, and Section 4 gives their formal definitions. Section 5 gives several complete examples. The remaining sections describe the IANA and privacy considerations.

1.1. 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].

When the words appear in lower case, their natural language meaning is used.

2. Terminology

- o ALTO transaction: A request/response exchange between an ALTO Client and an ALTO Server.
- o Client: This term refers to an ALTO client, when used with a capital "C".
- o Endpoint (EP): An endpoint is defined as in {2.1} of [RFC7285]. It can be for example a peer, a CDN storage location, a physical server involved in a virtual server-supported application, a party in a resource sharing swarm such as a computation grid or an online multi-party game.
- o Server: This term refers to an ALTO server, when used with a capital "S".

References with curly brackets such as '{1.2.3}' are to sections in the ALTO protocol specification [RFC7285], to avoid overloading the document with citations of [RFC7285].

3. Overview Of Approach

The following is a non-normative overview of the multi-cost extensions defined in this document. It assumes the reader is familiar with Cost Map resources in the ALTO Protocol ([RFC7285]).

3.1. Multi-Cost Data Format

Formally, the cost entries in an ALTO Cost Map can be any type of JSON value (see the DstCosts object in {11.2.3.6}). However, that section also says that an implementation may assume costs are JSON numbers, unless the implementation is using an extension which signals a different data type.

Therefore this document extends the definition of a Cost Map to allow a cost to be an array of costs, one per metric, instead of just one number. For example, here is a Cost Map with the "routingcost" and "hopcount" metrics. Note that this is identical to a regular ALTO Cost Map, except that the values are arrays instead of numbers. The multiple metrics are listed in member "multi-cost-types", indicating to the Client how to map values in the array to cost metrics.

```
{
  "meta" : {
    "dependent-vtags" : [ ... ],
    "cost-type" : {},
    "multi-cost-types" : [
      { "cost-mode": "numerical", "cost-metric": "routingcost" },
      { "cost-mode": "numerical", "cost-metric": "hopcount" }
    ]
  }
  "cost-map" : {
    "PID1": { "PID1": [1,0], "PID2": [5,23], "PID3": [10,5] },
    ...
  }
}
```

Note also the presence of member '"cost-type" : {}' to maintain backwards compatibility with [RFC7285].

3.2. Compatibility With Legacy ALTO Clients

This document does not define any new media types. Instead, as described below, it extends the specifications in the ALTO Server's Information Resource Directory (IRD) so that legacy Clients will not request array-valued Multi Cost Map resources. This relies on the requirement that ALTO Clients MUST ignore unknown fields ({8.3.7}).

3.3. Filtered Multi Cost Map Resources

This document extends the Filtered Cost Map service to allow the same resource to return either a single-valued Cost Map, as defined in [RFC7285], or an array-valued Multi Cost Map, as defined in this document. An extended Filtered Cost Map resource has a new capability, "max-cost-types". The value is the maximum number of cost types this resource can return for one request. The existence of this capability means the resource understands the extensions in this document.

For example, the following fragment from an IRD defines an extended Filtered Cost Map resource:

```
"filtered-multicost-map" : {
  "uri" : "http://alto.example.com/multi/costmap/filtered",
  "media-types" : [ "application/alto-costmap+json" ],
  "accepts" : [ "application/alto-costmapfilter+json" ],
  "uses" : [ "my-default-network-map" ],
  "capabilities" : {
    "max-cost-types" : 2,
    "cost-type-names" : [ "num-routingcost",
                          "num-hopcount" ],
    ...
  }
}
```

A legacy ALTO Client will ignore the "max-cost-types" capability, and will send a request with the input parameter "cost-type" describing the desired cost metric, as defined in [RFC7285]. The ALTO Server will return a single-valued legacy Cost Map.

However, a multi-cost-aware ALTO Client will realize that this resource supports the multi-cost extensions, and can send a POST request with the new input parameter "multi-cost-types", whose value is an array of cost types. Because the request has the "multi-cost-types" parameter (rather than the "cost-type" parameter defined in the base protocol), the Server realizes that the ALTO Client also supports the extensions in this document, and hence responds with a Multi Cost Map, with the costs in the order listed in "multi-cost-types".

3.4. Endpoint Cost Service Resources

Section {4.1.4} of [RFC7285] specifies that "The Endpoint Cost Service allows an ALTO Server to return costs directly amongst endpoints.", whereas the Filtered Cost Map Service returns costs amongst PIDs. This document uses the technique described in

Section 3.3 to extend the Endpoint Cost Service to return array-valued costs to ALTO Clients who also are aware of these extensions.

3.5. Full Cost Map Resources

Section {11.3.2.3} of [RFC7285] requires a Filtered Cost Map to return the entire Cost Map if the ALTO Client omits the source and destination PIDs. Hence a Multi-Cost aware ALTO Client can use an extended Filtered Cost Map resource to get a full Multi Cost Map.

Full Cost Map resources are GET-mode requests. The response for a Full Cost Map conveying multiple cost types would include a "meta" field that would itself include a "cost-type" field, that would list several values corresponding to the cost types of the cost map. A legacy ALTO Client would not be able to understand this list. Neither would it be able to interpret the cost values array provided by a Multi-Cost full maps.

3.6. Extended Constraint Tests

[RFC7285] defines a simple constraint test capability for Filtered Cost Maps and Endpoint Cost Services. If a resource supports constraints, the Server restricts the response to costs that satisfy a list of simple predicates provided by the ALTO Client. For example, if the ALTO Client gives the constraints

```
"constraints": ["ge 10", "le 20"]
```

Then the Server only returns costs in the range [10,20].

To be useful with multi-cost requests, the constraint tests require several extensions.

3.6.1. Extended constraint predicates

First, because a multi-cost request involves more than one cost metric, the simple predicates must be extended to specify the metric to test. Therefore we extend the predicate syntax to "[##] op value", where "##" is the index of a cost metric in this multi-cost request.

3.6.2. Extended logical combination of predicates

Second, once multiple cost metrics are involved, the "AND" of simple predicates is no longer sufficient. To be useful, Clients must be able to express "OR" tests. Hence we add a new field, "or-constraints", to the Client request. The value is an array of arrays

of simple predicates, and represents the OR of ANDs of those predicates.

Thus, the following request tells the Server to limit its response to cost points with "routingcost" <= 100 AND "hopcount" <= 2, OR else "routingcost" <= 10 AND "hopcount" <= 6:

```
{
  "multi-cost-types": [
    {"cost-metric": "routingcost", "cost-mode": "numerical"},
    {"cost-metric": "hopcount", "cost-mode": "numerical"}
  ],
  "or-constraints": [
    ["[0] le 100", "[1] le 2"],
    ["[0] le 10", "[1] le 6"]
  ],
  "pids": {...}
}
```

Note that a "constraints" parameter with the array of predicates [P1, P2, ...] is equivalent to an "or-constraints" parameter with one array of value [[P1, P2, ...]]. A Client is therefore allowed to express either "constraints" or "or-constraints" but not both.

3.6.3. Testable Cost Types in constraints

Finally, a Client may want to test a cost type whose actual value is irrelevant, as long as it satisfies the tests. For example, a Client may want the value of the cost metric "routingcost" for all PID pairs that satisfy constraints on the metric "hopcount", without needing the actual value of "hopcount".

To this end, we add a specific parameter named "testable-cost-types", that does not contain the same cost types as parameter "multi-cost-types". The Client can express constraints only on cost types listed in "testable-cost-types".

For example, the following request tells the Server to return just "routingcost" for those source and destination pairs for which "hopcount" is <= 6:

```
{
  "multi-cost-types": [
    {"cost-metric": "routingcost", "cost-mode": "numerical"},
  ],
  "testable-cost-types": [
    {"cost-metric": "hopcount", "cost-mode": "numerical"},
  ],
  "constraints": ["[0] le 6"],
  "pids": {...}
}
```

3.6.4. Testable Cost Type Names in IRD capabilities

In [RFC7285], when a resource's capability "constraints" is true, the Server accepts constraints on all the cost types listed in the "cost-type-names" capability. However, some ALTO Servers may not be willing to allow constraint tests on all available cost metrics. Therefore the Multi-Cost ALTO protocol extension defines the capability field "testable-cost-type-names". Like "cost-type-names", it is an array of cost type names. If present, that resource only allows constraint tests on the cost types in that list. "testable-cost-type-names" must be a subset of "cost-type-names".

3.6.5. Legacy ALTO Client issues

While a multi-cost-aware Client will recognize the "testable-cost-type-names" field, and will honor those restrictions, a legacy Client will not. Hence, when "constraints" has the value 'true', a legacy client may send a request with a constraint test on any of the cost types listed in "cost-type-names".

To avoid that problem, the "testable-cost-type-names" and "cost-constraints" fields are mutually exclusive: a resource may define one or the other capability, but MUST NOT define both. Thus a resource that does not allow constraint tests on all cost metrics will set "testable-cost-type-names" to the testable metrics, and will set "cost-constraints" to "false". A multi-cost-aware Client will recognize the "testable-cost-type-names" field, and will realize that its existence means the resource does allow (limited) constraint tests, while a legacy Client will think that resource does not allow constraint tests at all. To allow legacy Clients to use constraint tests, the ALTO Server can define an additional resource with "cost-constraints" set to "true" and "cost-type-names" set to the metrics which can be tested.

In the IRD example below, the resource "filtered-cost-map-extended" provides values for three metrics: "num-routingcost", "num-hopcount" and "num-bwscore". The capability "testable-cost-type-names"

indicates that the Server only allows constraints on "routingcost" and "hopcount". A multi-cost capable Client will see this capability, and will limit its constraint tests to those metrics. Because capability "cost-constraints" is false (by default), a legacy Client will not use constraint tests on this resource at all.

The second resource, "filtered-multicost-map", is similar to the first, except that all the metrics it returns are testable. Therefore it sets "cost-constraints" to "true", and does not set the "testable-cost-type-names" field. A legacy Client that needs a constraint test will use this resource rather than the first. A multi-cost-aware Client that does not need to retrieve the "num-bwscore" metric may use either resource.

Note that if a multi-cost Server specifies a "filtered-cost-map-extended", it will most likely not specify an "filtered-multicost-map" if the capabilities of the latter are covered by the capabilities of the former or unless the "filtered-multicost-map" resource is also intended for legacy Clients.

```
"filtered-cost-map-extended" : {
  "uri" : "http://alto.example.com/multi/extn/costmap/filtered",
  "media-types" : [ "application/alto-costmap+json" ],
  "accepts" : [ "application/alto-costmapfilter+json" ],
  "uses" : [ "my-default-network-map" ],
  "capabilities" : {
    "max-cost-types" : 3,
    "cost-type-names" : [ "num-routingcost",
                          "num-hopcount",
                          "num-bwscore" ],
    "testable-cost-type-names" : [ "num-routingcost",
                                   "num-hopcount" ]
  }
},

"filtered-multicost-map" : {
  "uri" : "http://alto.example.com/multi/costmap/filtered",
  "media-types" : [ "application/alto-costmap+json" ],
  "accepts" : [ "application/alto-costmapfilter+json" ],
  "uses" : [ "my-default-network-map" ],
  "capabilities" : {
    "cost-constraints" : true,
    "max-cost-types" : 2,
    "cost-type-names" : [ "num-routingcost",
                          "num-hopcount" ],
  }
}
```

4. Protocol Extensions for Multi-Cost ALTO Transactions

This section formally specifies the extensions to [RFC7285] to support Multi-Cost ALTO transactions.

This document uses the notation rules specified in {8.2}. In particular, an optional field is enclosed by []. In the definitions, the JSON names of the fields are case sensitive. An array is indicated by two numbers in angle brackets, <m..n>, where m indicates the minimal number of values and n is the maximum. When this document uses * for n, it means no upper bound.

4.1. Filtered Cost Map Extensions

This document extends Filtered Cost Maps, as defined in {11.3.2} of [RFC7285], by adding new input parameters and capabilities, and by returning JSONArrays instead of JSONNumbers as the cost values.

The media type (11.3.2.1}, HTTP method (11.3.2.2} and "uses" specifications (11.3.2.5} are unchanged.

4.1.1. Capabilities

The filtered cost map capabilities are extended with two new members:

- o max-cost-types,
- o testable-cost-type-names

The capability "max-cost-types" indicates whether this resource supports the Multi-Cost ALTO extensions, and the capability "testable-cost-type-names" allows the resource to restrict constraint tests to a subset of the available cost types. With these two additional members, the FilteredCostMapCapabilities object in {11.3.2.4} is structured as follows:

```
object {  
  JSONString cost-type-names<1..*>;  
  [JSONBool cost-constraints;]  
  [JSONNumber max-cost-types;]  
  [JSONString testable-cost-type-names<1..*>;]  
} FilteredCostMapCapabilities;
```

cost-type-names: As defined in {11.3.2.4} of [RFC7285].

cost-constraints: As defined in {11.3.2.4} of [RFC7285]. Thus if "cost-constraints" is true, the resource MUST accept constraint tests on any cost type in "cost-type-names". Note in addition

that if "cost-constraints" is "true", the "testable-cost-type-names" capability MUST NOT be present

max-cost-types: If present with value N greater than 0, this resource understands the multi-cost extensions in this document, and can return a Multi Cost Map with any combination of N or fewer cost types in the "cost-type-names" list. If omitted, the default value is 0.

testable-cost-type-names: If present, the resource allows constraint tests, but only on the cost type names in this array. Each name in "testable-cost-type-names" MUST also be in "cost-type-names". If "testable-cost-type-names" is present, the "cost-constraints" capability MUST NOT be true.

As discussed in Section 3.6.4, this capability is useful when a Server is unable or unwilling to implement constraint tests on all cost types. As discussed in Section 3.6.5, "testable-cost-type-names" and "cost-constraints" are mutually exclusive to prevent legacy Clients from issuing constraint tests on untestable cost types.

4.1.2. Accept Input Parameters

The ReqFilteredCostMap object in {11.3.2.3} of [RFC7285] is extended as follows:

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

cost-type: As defined in {11.3.2.3} of [RFC7285], with the additional requirement that the Client MUST specify either "cost-type" or "multi-cost-types", but MUST NOT specify both. Therefore this field is made optional. When placing a single cost request as specified in [RFC7285], a Client MUST use "cost-type".

multi-cost-types: If present, the ALTO Server MUST return array-valued costs for the cost types in this list. For each entry, the "cost-metric" and "cost-mode" fields MUST match one of the supported cost types indicated in member "cost-type-names" of this resource's "capabilities" field (Section 4.1.1). The Client MUST

NOT use this field unless this resource's "max-cost-types" capability exists and has a value greater than 0. This field MUST NOT have more than "max-cost-types" cost types. The Client MUST specify either "cost-type" or "multi-cost-types", but MUST NOT specify both.

Note that if "multi-cost-types" has one cost type, the values in the cost map will be arrays with one value.

testable-cost-types: A list of cost types used for extended constraint tests, as described for the "constraints" and "or-constraints" parameters. These cost types must either be a subset of the cost types in the resource's "testable-cost-type-names" capability (Section 4.1.1), or else, if the resource's capability "cost-constraints" is true, a subset of the cost types in the resource's "cost-type-names" capability.

If "testable-cost-types" is omitted, it is assumed to have the cost types in "multi-cost-types" or "cost-type".

This feature is useful when a Client wants to test a cost type whose actual value is irrelevant, as long as it satisfies the tests. For example, a Client may want the cost metric "routingcost" for those PID pairs whose "hopcount" is less than 10. The exact hopcount does not matter.

constraints: If this resource's "max-cost-types" capability (Section 4.1.1) has the value 0 (or is not defined), this parameter is as defined in {11.3.2.3} of [RFC7285]: an array of constraint tests related to each other by a logical AND. In this case it MUST NOT be specified unless the resource's "cost-constraints" capability is "true".

If this resource's "max-cost-types" capability has a value greater than 0, then this parameter is an array of extended constraint predicates as defined below and related to each other by a logical AND. In this case, it MAY be specified if the resource allows constraint tests (the resource's "cost-constraints" capability is "true" or its "testable-cost-type-names" capability is not empty).

This parameter MUST NOT be specified if the "or-constraints" parameter is specified.

An extended constraint predicate consists of two or three entities separated by white space: (1) an optional cost type index, of the form "[#]", with default value "[0]", (2) a required operator, and (3) a required target value. The operator and target value are as defined in {11.3.2.3} of [RFC7285]. The cost type index, i,

specifies the cost type to test. If the "testable-cost-type" parameter is present, the test applies to the i'th cost type in "testable-cost-types", starting with index 0. Otherwise if the "multi-cost-types" parameter is present, the test applies to the i'th cost type in that array. If neither parameters are present, the test applies to the cost type in the "cost-type" parameter, in which case the index MUST be 0. Regardless of how the tested cost type is selected, it MUST be in the resource's "testable-cost-type-names" capability, or, if not present, in the "cost-type-names" capability.

As an example, suppose "multi-cost-types" has the single element "routingcost", "testable-cost-types" has the single element "hopcount", and "constraints" has the single element "[0] le 5". This is equivalent to the database query "SELECT and provide routingcost WHERE hopcount <= 5".

Note that the index is optional, so a constraint test as defined in {11.3.2.3}, such as "le 10", is equivalent to "[0] le 10". Thus legacy constraint tests are also legal extended constraint tests.

Note that a "constraints" parameter with the array of extended predicates [P1, P2, ...] is equivalent to an "or-constraints" parameter as defined below, with the value [[P1, P2, ...]].

or-constraints: A JSONArray of JSONArrays of JSONStrings, where each string is an extended constraint predicate as defined above. The "or-constraint" tests are interpreted as the logical OR of ANDs of predicates. That is, the ALTO Server should return a cost point only if it satisfies all constraints in any one of the sub-arrays.

This parameter MAY be specified if this resource's "max-cost-types" capability is defined with a value greater than 0 (Section 4.1.1), and if the resource allows constraint tests (the resource's "cost-constraints" capability is "true" or its "testable-cost-type-names" capability is not empty). Otherwise this parameter MUST NOT be specified.

This parameter MUST NOT be specified if the "constraints" parameter is specified.

This parameter MUST NOT contain any empty array of AND predicates. An empty array would be equivalent to a constraint that is always "true". An OR combination including such a constraint would be always "true" and thus useless.

As an example, suppose "multi-cost-types" has the two elements "routingcost" and "bandwidthscore", and "testable-cost-types" has the two elements "routingcost" and "hopcount", and "or-constraints" has the two elements ["[0] le 100", "[1] le 2"] and ["[0] le 10", "[1] le 6"]. This is equivalent to the words: "SELECT and provide routingcost and bandwidthscore WHERE ("routingcost" <= 100 AND "hopcount" <= 2) OR ("routingcost" <= 10 AND "hopcount" <= 6)".

Note that if the "max-cost-types" capability has a value greater than 0, a Client MAY use the "or-constraints" parameter together with the "cost-type" parameter. That is, if the Client and Server are both aware of the extensions in this document, a Client MAY use an "OR" test for a single-valued cost request.

pids: As defined in {11.3.2.3} of [RFC7285].

4.1.3. Response

If the Client specifies the "cost-type" input parameter, the response is exactly as defined in {11.2.3.6} of [RFC7285]. If the Client provides the "multi-cost-types" instead, then the response is changed as follows:

- o In "meta", the value of field "cost-type" will be ignored by the receiver and set to {}. Instead, the field "multi-cost-types" is added with the same value as the "multi-cost-types" input parameter.
- o The costs are JSONArrays, instead of JSONNumbers. All arrays have the same cardinality as the "multi-cost-types" input parameter, and contain the cost type values in that order. If a cost type is not available for a particular source and destination, the ALTO Server MUST use the JSON "null" value for that array element. If none of the cost types are available for a particular source and destination, the ALTO Server MAY omit the entry for that source and destination.

4.2. Endpoint Cost Service Extensions

This document extends the Endpoint Cost Service, as defined in {11.5.1} of [RFC7285], by adding new input parameters and capabilities, and by returning JSONArrays instead of JSONNumbers as the cost values.

The media type {11.5.1.1}, HTTP method {11.5.1.2} and "uses" specifications {11.5.1.5} are unchanged.

4.2.1. Capabilities

The extensions to the Endpoint Cost Service capabilities are identical to the extensions to the Filtered Cost Map (see Section 4.1.1).

4.2.2. Accept Input Parameters

The ReqEndpointCostMap object in {11.5.1.3} of [RFC7285] is extended as follows:

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

cost-type: As defined in {11.5.1.3} of [RFC7285], with the additional requirement that the Client MUST specify either "cost-type" or "multi-cost-types", but MUST NOT specify both.

multi-cost-types: If present, the ALTO Server MUST return array-valued costs for the cost types in this list. For each entry, the "cost-metric" and "cost-mode" fields MUST match one of the supported cost types indicated in this resource's "capabilities" field (Section 4.2.1). The Client MUST NOT use this field unless this resource's "max-cost-types" capability exists and has a value greater than 0. This field MUST NOT have more than "max-cost-types" cost types. The Client MUST specify either "cost-type" or "multi-cost-types", but MUST NOT specify both.

Note that if "multi-cost-types" has one cost type, the values in the cost map will be arrays with one value.

testable-cost-types, constraints, or-constraints: Defined equivalently to the corresponding input parameters for an extended Filtered Cost Map (Section 4.1.2).

endpoints: As defined in {11.5.1.3} of [RFC7285].

4.2.3. Response

The extensions to the Endpoint Cost Service response are similar to the extensions to the Filtered Cost Map response (Section 4.1.3). Specifically, if the Client specifies the "cost-type" input parameter, the response is exactly as defined in {11.5.1.6} of [RFC7285]. If the Client provides the "multi-cost-types" instead, then the response is changed as follows:

- o In "meta", the value of field "cost-type" will be ignored by the receiver and set to {}. Instead, the field "multi-cost-types" is added with the same value as the "multi-cost-types" input parameter.
- o The costs are JSONArrays, instead of JSONNumbers. All arrays have the same cardinality as the "multi-cost-types" input parameter, and contain the cost type values in that order. If a cost type is not available for a particular source and destination, the ALTO Server MUST use the JSON "null" value for that array element. If none of the cost types are available for a particular source and destination, the ALTO Server MAY omit the entry for that source and destination.

5. Examples

This section provides examples of Multi-Cost ALTO transactions. It uses cost metrics, in addition to the mandatory legacy 'routingcost', that are deliberately irrelevant and not registered at the IANA.

5.1. Information Resource Directory

The following is an example of an ALTO Server's Information Resource Directory. In addition to Network and Cost Map resources, it defines two Filtered Cost Map and an Endpoint Cost Service, which all understand the multi-cost extensions.

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

```
HTTP/1.1 200 OK
Content-Length: 2704
Content-Type: application/alto-directory+json
```

```
{
  "meta" : {
    "default-alto-network-map" : "my-default-network-map",
```



```
"cost-types" : {
  "num-routing" : {
    "cost-mode" : "numerical",
    "cost-metric" : "routingcost"
  },
  "num-shoesize" : {
    "cost-mode" : "numerical",
    "cost-metric" : "shoesize"
  },
  "num-scenery" : {
    "cost-mode" : "numerical",
    "cost-metric" : "sceneryrate"
  }
},
"resources" : {
  "my-default-network-map" : {
    "uri" : "http://alto.example.com/networkmap",
    "media-type" : "application/alto-networkmap+json"
  },
  "numerical-routing-cost-map" : {
    "uri" : "http://alto.example.com/costmap/num-routing",
    "media-types" : [ "application/alto-costmap+json" ],
    "uses" : [ "my-default-network-map" ],
    "capabilities" : {
      "cost-type-names" : [ "num-routing" ]
    }
  },
  "numerical-shoesize-cost-map" : {
    "uri" : "http://alto.example.com/costmap/num-shoesize",
    "media-types" : [ "application/alto-costmap+json" ],
    "uses" : [ "my-default-network-map" ],
    "capabilities" : {
      "cost-type-names" : [ "num-shoesize" ]
    }
  },
  "filtered-multicost-map" : {
    "uri" : "http://alto.example.com/multi/costmap/filtered",
    "media-types" : [ "application/alto-costmap+json" ],
    "accepts" : [ "application/alto-costmapfilter+json" ],
    "uses" : [ "my-default-network-map" ],
    "capabilities" : {
      "cost-constraints" : true,
      "max-cost-types" : 2,
      "cost-type-names" : [ "num-routingcost",
                           "num-shoesize" ]
    }
  }
},
```

```

    "filtered-cost-map-extended" : {
      "uri" : "http://alto.example.com/multi/extn/costmap/filtered",
      "media-types" : [ "application/alto-costmap+json" ],
      "accepts" : [ "application/alto-costmapfilter+json" ],
      "uses" : [ "my-default-network-map" ],
      "capabilities" : {
        "max-cost-types" : 3,
        "cost-type-names" : [ "num-routingcost",
                              "num-shoesize",
                              "num-scenery" ],
        "testable-cost-type-names" : [ "num-routingcost",
                                       "num-shoesize" ]
      }
    },
    "endpoint-multicost-map" : {
      "uri" : "http://alto.example.com/multi/endpointcost/lookup",
      "media-types" : [ "application/alto-endpointcost+json" ],
      "accepts" : [ "application/alto-endpointcostparams+json" ],
      "uses" : [ "my-default-network-map" ],
      "capabilities" : {
        "cost-constraints" : true,
        "max-cost-types" : 2,
        "cost-type-names" : [ "num-routingcost",
                              "num-shoesize" ]
      }
    }
  }
}

```

5.2. Multi-Cost Filtered Cost Map: Example #1

This example illustrates a simple multi-cost ALTO transaction. The ALTO Server provides two Cost Types, "routingcost" and "shoesize", both in "numerical" mode. The Client wants the entire Multi-Cost Map. The Server does not know the value of "routingcost" between PID2 and PID3, and hence returns the value 'null' for "routingcost" between PID2 and PID3.

```
POST /multi/costmap/filtered" HTTP/1.1
Host: alto.example.com
Accept: application/alto-costmap+json,application/alto-error+json
Content-Type: application/alto-costmapfilter+json
Content-Length: 206
```

```
{
  "multi-cost-types": [
    {"cost-mode": "numerical", "cost-metric": "routingcost"},
    {"cost-mode": "numerical", "cost-metric": "shoesize"}
  ],
  "pids" : {
    "srcs" : [ ],
    "dsts" : [ ]
  }
}
```

```
HTTP/1.1 200 OK
Content-Type: application/alto-costmap+json
Content-Length: 549
```

```
{
  "meta" : {
    "dependent-vtags" : [
      {"resource-id": "my-default-network-map",
       "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"}
    ]
  },
  "cost-type" : {},
  "multi-cost-types" : [
    {"cost-mode": "numerical", "cost-metric": "routingcost"},
    {"cost-mode": "numerical", "cost-metric": "shoesize"}
  ]
}
"cost-map" : {
  "PID1": { "PID1":[1,0], "PID2":[4,3], "PID3":[10,2] },
  "PID2": { "PID1":[15,5], "PID2":[1,0], "PID3":[null,9] },
  "PID3": { "PID1":[20,12], "PID2":[null,1], "PID3":[1,0] }
}
}
```

5.3. Multi-Cost Filtered Cost Map: Example #2

This example uses constraints to restrict the returned source/destination PID pairs to those with "routingcost" between 5 and 10, or "shoesize" equal to 0.

```
POST /multi/costmap/filtered HTTP/1.1
Host: alto.example.com
Accept: application/alto-costmap+json,application/alto-error+json
Content-Type: application/alto-costmapfilter+json
Content-Length: 333
```

```
{
  "multi-cost-types" : [
    { "cost-mode": "numerical", "cost-metric": "routingcost" },
    { "cost-mode": "numerical", "cost-metric": "shoesize" }
  ],
  "or-constraints" : [ [ "[0] ge 5", "[0] le 10" ],
                       [ "[1] eq 0" ] ]
  "pids" : {
    "srcs" : [ "PID1", "PID2" ],
    "dsts" : [ "PID1", "PID2", "PID3" ]
  }
}
```

```
HTTP/1.1 200 OK
Content-Type: application/alto-costmap+json
Content-Length: 461
```

```
{
  "meta" : {
    "dependent-vtags" : [
      { "resource-id": "my-default-network-map",
        "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"
      }
    ],
    "cost-type" : {},
    "multi-cost-types" : [
      { "cost-mode": "numerical", "cost-metric": "routingcost" },
      { "cost-mode": "numerical", "cost-metric": "shoesize" }
    ]
  }
  "cost-map" : {
    "PID1": { "PID1": [1,0], "PID3": [10,5] },
    "PID2": { "PID2": [1,0] }
  }
}
```

5.4. Multi-Cost Filtered Cost Map: Example #3

This example uses extended constraints to limit the response to cost points with ("routingcost" <= 10 and "shoesize" <= 2), or else ("routingcost" <= 3 and "shoesize" <= 6). Unlike the previous example, the Client is only interested in the "routingcost" cost type, and uses the "cost-type" parameter instead of "multi-cost-types" to tell the Server to return scalar costs instead of array costs.

In this example, "[0]" means the constraint applies to "routingcost" because that is the first cost type in the "testable-cost-types" parameter. (If "testable-cost-types" is omitted, it is assumed to be the same as "multi-cost-types".) The choice of using an index to refer to cost types aims at minimizing the length of the expression of constraints, especially for those combining several OR and AND expressions. It was also the shortest path from the constraints design in [RFC7285].

```
POST /multi/multicostmap/filtered HTTP/1.1
Host: alto.example.com
Accept: application/alto-costmap+json,application/alto-error+json
Content-Type: application/alto-costmapfilter+json
Content-Length: 390
```

```
{
  "cost-type" : {
    "cost-mode": "numerical", "cost-metric": "routingcost"
  },
  "testable-cost-types" : [
    {"cost-mode": "numerical", "cost-metric": "routingcost"},
    {"cost-mode": "numerical", "cost-metric": "shoesize"}
  ],
  "or-constraints": [
    ["[0] le 10", "[1] le 2"],
    ["[0] le 3", "[1] le 6"]
  ],
  "pids" : {
    "srcs" : [ ],
    "dsts" : [ ]
  }
}
```

HTTP/1.1 200 OK

Content-Type: application/alto-costmap+json

Content-Length: 368

```
{
  "meta" : {
    "dependent-vtags" : [
      { "resource-id": "my-default-network-map",
        "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"
      }
    ],
    "cost-type" : {
      "cost-mode": "numerical", "cost-metric": "routingcost"
    }
  }
  "cost-map" : {
    "PID1": { "PID1": 1, "PID3": 10 },
    "PID2": { "PID2": 1 },
    "PID3": { "PID3": 1 }
  }
}
```

5.5. Multi-Cost Filtered Cost Map: Example #4

This example uses extended constraints to limit the response to cost points with ("routingcost" <= 10 and "shoesize" <= 2), or else ("routingcost" <= 3 and "shoesize" <= 6). In this example, the Client is interested in the "routingcost" and "sceneryrate" cost metrics, but not in the "shoesize" metric:

POST /multi/extn/costmap/filtered HTTP/1.1

Host: alto.example.com

Accept: application/alto-costmap+json,application/alto-error+json

Content-Type: application/alto-costmapfilter+json

Content-Length: 461

```
{
  "multi-cost-types" : [
    { "cost-mode": "numerical", "cost-metric": "routingcost" },
    { "cost-mode": "numerical", "cost-metric": "sceneryrate" }
  ],
  "testable-cost-types" : [
    { "cost-mode": "numerical", "cost-metric": "routingcost" },
    { "cost-mode": "numerical", "cost-metric": "shoesize" }
  ],
  "or-constraints": [
    "[0] le 10", "[1] le 2",
  ],
}
```

```

        ["[0] le 3", "[1] le 6"]
    ],
    "pids" : {
        "srcs" : [ ],
        "dsts" : [ ]
    }
}

```

HTTP/1.1 200 OK

Content-Type: application/alto-costmap+json

Content-Length: 481

```

{
  "meta" : {
    "dependent-vtags" : [
      { "resource-id": "my-default-network-map",
        "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"
      }
    ],
    "cost-type" : {},
    "multi-cost-types" : [
      { "cost-mode": "numerical", "cost-metric": "routingcost" },
      { "cost-mode": "numerical", "cost-metric": "sceneryrate" }
    ]
  }
  "cost-map" : {
    "PID1": { "PID1": [1,16] "PID3": [10,19] },
    "PID2": { "PID2": [1,8] },
    "PID3": { "PID3": [1,19] }
  }
}

```

5.6. Endpoint Cost Service

This example uses the Endpoint Cost Service to retrieve the "routingcost" and "shoesize" for selected endpoints, limiting the response to costs with either low shoesize and reasonable routingcost (shoesize <= 2 and routingcost <= 10), or else low routingcost and reasonable shoesize (routingcost <= 3 and shoesize <= 6).

POST /multi/endpointcost/lookup HTTP/1.1

Host: alto.example.com

Accept: application/alto-endpointcost+json,
application/alto-error+json

Content-Type: application/alto-endpointcostparams+json

Content-Length: 455

```
{
  "multi-cost-types" : [
    {"cost-mode": "numerical", "cost-metric": "routingcost"},
    {"cost-mode": "numerical", "cost-metric": "shoesize"}
  ],
  "or-constraints": [
    ["[0] le 10", "[1] le 2"],
    ["[0] le 3", "[1] le 6"]
  ],
  "endpoints" : {
    "srcs": [ "ipv4:192.0.2.2", "ipv6:2001:db8::1:0" ],
    "dsts": [
      "ipv4:192.0.2.89",
      "ipv4:198.51.100.34",
      "ipv4:203.0.113.45",
      "ipv6:2001:db8::10"
    ]
  }
}
```

HTTP/1.1 200 OK

Content-Length: 419

Content-Type: application/alto-endpointcost+json

```
{
  "meta" : {
    "multi-cost-types" : [
      {"cost-mode": "numerical", "cost-metric": "routingcost"},
      {"cost-mode": "numerical", "cost-metric": "shoesize"}
    ]
  }
  "endpoint-cost-map" : {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89": [15, 5],
      "ipv4:203.0.113.45": [4, 23]
    }
    "ipv6:2001:db8::1:0": {
      "ipv4:198.51.100.34": [16, 5],
      "ipv6:2001:db8::10": [10, 2]
    }
  }
}
```

6. IANA Considerations

This document does not define any new media types or introduce any new IANA considerations.

7. Privacy And Security Considerations

This document does not introduce any privacy or security issues not already present in the ALTO protocol.

The Multi-Cost optimization even tends to reduce the on the wire data exchange volume, compared to multiple single cost ALTO transactions. Likewise, the risk related to massive Multi-Cost requests is moderated by the fact that Multi-Cost constraints additionally filter ALTO Server responses and thus reduce their volume.

Note that, because queries for multiple metrics represent a stronger fingerprinting signal than queries for a single metric, implementations of this protocol may leak more information about the ALTO client than would occur with a succession of individual queries. Though, in many cases it would already be possible to link those queries by using the source IP address or other existing information.

8. Acknowledgements

The authors would like to thank Richard Alimi, Fred Baker, Dhruv Dhodi, Vijay Gurbani, Dave Mac Dusan, Young Lee, Richard Yang, for fruitful discussions and feedback on this document and previous versions. Gao Kai, Hans Seidel, Richard Yang, Qiao Xiang and Wang Xin provided substantial review feedback and suggestions to the protocol design.

9. References

9.1. 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>>.
- [RFC7285] Almi, R., Penno, R., Yang, Y., Kiesel, S., Previdi, S., Roome, W., Shalunov, S., and R. Woundy, "Application-Layer Traffic Optimization (ALTO) Protocol", RFC 7285, September 2014.

9.2. Informative References

- [RFC5693] "Application Layer Traffic Optimization (ALTO) Problem Statement", October 2009.
- [RFC6708] "Application-Layer Traffic Optimization (ALTO) Requirements", February 2012.

Authors' Addresses

Sabine Randriamasy
Nokia Bell Labs
Route de Villejust
NOZAY 91460
FRANCE

Email: Sabine.Randriamasy@nokia-bell-labs.com

Wendy Roome
Nokia Bell Labs
124 Burlington Rd
Murray Hill, NJ 07974
USA

Email: ietf@wdroome.com

Nico Schwan
Thales Deutschland
Lorenzstrasse 10
Stuttgart 70435
Germany

Email: nico.schwan@thalesgroup.com

ALTO Working Group
Internet-Draft
Intended status: Standards Track
Expires: 22 September 2022

Q. Wu
Huawei
Y. Yang
Yale University
Y. Lee
Samsung
D. Dhody
Huawei
S. Randriamasy
Nokia Bell Labs
L. Contreras
Telefonica
21 March 2022

ALTO Performance Cost Metrics
draft-ietf-alto-performance-metrics-28

Abstract

The cost metric is a basic concept in Application-Layer Traffic Optimization (ALTO), and different applications may use different types of cost metrics. Since the ALTO base protocol (RFC 7285) defines only a single cost metric (namely, the generic "routingcost" metric), if an application wants to issue a cost map or an endpoint cost request in order to identify a resource provider that offers better performance metrics (e.g., lower delay or loss rate), the base protocol does not define the cost metric to be used.

This document addresses this issue by extending the specification to provide a variety of network performance metrics, including network delay, delay variation (a.k.a, jitter), packet loss rate, hop count, and bandwidth.

There are multiple sources (e.g., estimation based on measurements or service-level agreement) to derive a performance metric. This document introduces an additional "cost-context" field to the ALTO "cost-type" field to convey the source of a performance metric.

Status of This Memo

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

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

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

This Internet-Draft will expire on 22 September 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	3
2. Requirements Language	6
3. Performance Metric Attributes	6
3.1. Performance Metric Context: "cost-context"	7
3.2. Performance Metric Statistics	9
4. Packet Performance Metrics	11
4.1. Cost Metric: One-Way Delay (delay-ow)	11
4.1.1. Base Identifier	11
4.1.2. Value Representation	12
4.1.3. Intended Semantics and Use	12
4.1.4. Cost-Context Specification Considerations	14
4.2. Cost Metric: Round-trip Delay (delay-rt)	16
4.2.1. Base Identifier	16
4.2.2. Value Representation	16
4.2.3. Intended Semantics and Use	16
4.2.4. Cost-Context Specification Considerations	17
4.3. Cost Metric: Delay Variation (delay-variation)	18
4.3.1. Base Identifier	18
4.3.2. Value Representation	18
4.3.3. Intended Semantics and Use	18
4.3.4. Cost-Context Specification Considerations	19
4.4. Cost Metric: Loss Rate (lossrate)	20
4.4.1. Base Identifier	20
4.4.2. Value Representation	20
4.4.3. Intended Semantics and Use	20

4.4.4.	Cost-Context Specification Considerations	21
4.5.	Cost Metric: Hop Count (hopcount)	22
4.5.1.	Base Identifier	22
4.5.2.	Value Representation	22
4.5.3.	Intended Semantics and Use	22
4.5.4.	Cost-Context Specification Considerations	23
5.	Throughput/Bandwidth Performance Metrics	24
5.1.	Cost Metric: TCP Throughput (tput)	24
5.1.1.	Base Identifier	24
5.1.2.	Value Representation	24
5.1.3.	Intended Semantics and Use	24
5.1.4.	Cost-Context Specification Considerations	25
5.2.	Cost Metric: Residual Bandwidth (bw-residual)	26
5.2.1.	Base Identifier	26
5.2.2.	Value Representation	26
5.2.3.	Intended Semantics and Use	26
5.2.4.	Cost-Context Specification Considerations	28
5.3.	Cost Metric: Available Bandwidth (bw-available)	28
5.3.1.	Base Identifier	28
5.3.2.	Value Representation	28
5.3.3.	Intended Semantics and Use	29
5.3.4.	Cost-Context Specification Considerations	30
6.	Operational Considerations	30
6.1.	Source Considerations	31
6.2.	Metric Timestamp Consideration	31
6.3.	Backward Compatibility Considerations	31
6.4.	Computation Considerations	32
6.4.1.	Configuration Parameters Considerations	32
6.4.2.	Aggregation Computation Considerations	32
7.	Security Considerations	32
8.	IANA Considerations	33
9.	Acknowledgments	35
10.	References	35
10.1.	Normative References	35
10.2.	Informative References	37
	Authors' Addresses	38

1. Introduction

Application-Layer Traffic Optimization (ALTO) provides a means for network applications to obtain network information so that the applications can identify efficient application-layer traffic patterns using the networks. Cost metrics are used in both the ALTO cost map service and the ALTO endpoint cost service in the ALTO base protocol [RFC7285].

Since different applications may use different cost metrics, the ALTO base protocol introduces an ALTO Cost Metric Registry (Section 14.2 of [RFC7285]) as a systematic mechanism to allow different metrics to be specified. For example, a delay-sensitive application may want to use latency related metrics, and a bandwidth-sensitive application may want to use bandwidth related metrics. However, the ALTO base protocol has registered only a single cost metric, i.e., the generic "routingcost" metric (Section 14.2 of [RFC7285]); no latency or bandwidth related metrics are defined in the base protocol.

This document registers a set of new cost metrics (Table 1) to allow applications to determine "where" to connect based on network performance criteria including delay and bandwidth related metrics.

Metric	Definition in this doc	Semantics Based On
One-way Delay	Section 4.1	Base: [RFC7471,8570,8571] sum Unidirectional Delay
Round-trip Delay	Section 4.2	Base: Sum of two directions from above
Delay Variation	Section 4.3	Base: [RFC7471,8570,8571] sum of Unidirectional Delay Variation
Loss Rate	Section 4.4	Base: [RFC7471,8570,8571] aggr Unidirectional Link Loss
Residual Bandwidth	Section 5.2	Base: [RFC7471,8570,8571] min Unidirectional Residual BW
Available Bandwidth	Section 5.3	Base: [RFC7471,8570,8571] min Unidirectional Avail. BW
TCP Throughput	Section 5.1	[I-D.ietf-tcpm-rfc8312bis]
Hop Count	Section 4.5	[RFC7285]

Table 1. Cost Metrics Defined in this Document.

The first 6 metrics listed in Table 1 (i.e., One-way Delay, Round-trip Delay, Delay Variation, Loss Rate, Residual Bandwidth, and Available Bandwidth) are derived from the set of traffic engineering performance metrics commonly defined in OSPF [RFC3630], [RFC7471]; IS-IS [RFC5305], [RFC8570]; and BGP-LS [RFC8571]. Deriving ALTO cost performance metrics from existing network-layer traffic engineering performance metrics, to expose to application-layer traffic optimization, can be a typical mechanism by network operators to deploy ALTO [RFC7971], [FlowDirector]. This document defines the base semantics of these metrics by extending them from link metrics

to end-to-end metrics for ALTO. The "Semantics Based On" column specifies at a high level how the end-to-end metric is computed from link metrics; the details will be specified in the following sections.

The common metrics Min/Max Unidirectional Delay defined in [RFC8570][RFC8571] and Max Link Bandwidth defined in [RFC3630,RFC5305] are not listed in Table 1 because they can be handled by applying the statistical operators defined in this document. The metrics related with utilized bandwidth and reservable bandwidth (i.e., Max Reservable BW and Unreserved BW defined in [RFC3630,RFC5305]) are outside the scope of this document.

The 7th metric (the estimated TCP-flow throughput metric) provides an estimation of the bandwidth of a TCP flow, using TCP throughput modeling, to support use cases of adaptive applications [Prophet], [G2]. Note that other transport-specific metrics can be defined in the future. For example, QUIC-related metrics [RFC9000] can be considered when the methodology to measure such metrics is more mature (e.g., [I-D.corre-quick-throughput-testing]).

The 8th metric (the hop count metric) in Table 1 is mentioned in the ALTO base protocol [RFC7285], but not defined, and this document defines it to be complete.

These 8 performance metrics can be classified into two categories: those derived from the performance of individual packets (i.e., One-way Delay, Round-trip Delay, Delay Variation, Loss Rate, and Hop Count), and those related to bandwidth/throughput (Residual bandwidth, and Available Bandwidth, and TCP throughput). These two categories are defined in Sections 4 and 5 respectively. Note that all metrics except Round-trip Delay are unidirectional. An ALTO client will need to query both directions if needed.

The purpose of this document is to ensure proper usage of these 8 performance metrics in the context of ALTO. This document follows the guideline defined in Section 14.2 of the ALTO base protocol [RFC7285] on registering ALTO cost metrics. Hence, it specifies the identifier, the intended semantics, and the security considerations of each one of the metrics specified in Table 1.

The definitions of the intended semantics of the metrics tend to be coarse-grained, for guidance only, and they may work well for ALTO. On the other hand, a performance measurement framework, such as the IP Performance Measurement (IPPM) framework, may provide more details in defining a performance metric. This document introduces a mechanism called "cost-context" to provide additional details, when they are available; see Section 3.

Following the ALTO base protocol, this document uses JSON to specify the value type of each defined metric. See [RFC8259] for JSON data type specification. In particular, [RFC7285] specifies that cost values should be assumed by default as JSONNumber. When defining the value representation of each metric in Table 1, this document conforms to [RFC7285], but specifies additional, generic constraints on valid JSONNumbers for each metric. For example, each new metric in Table 1 will be specified as non-negative (≥ 0); Hop Count is specified to be an integer.

An ALTO server may provide only a subset of the metrics described in this document. For example, those that are subject to privacy concerns should not be provided to unauthorized ALTO clients. Hence, all cost metrics defined in this document are optional; not all of them need to be exposed to a given application. When an ALTO server supports a cost metric defined in this document, it announces the metric in its information resource directory (IRD) as defined in Section 9.2 of [RFC7285].

An ALTO server introducing these metrics should consider related security issues. As a generic security consideration on the reliability and trust in the exposed metric values, applications SHOULD rapidly give up using ALTO-based guidance if they detect that the exposed information does not preserve their performance level or even degrades it. Section 7 discusses security considerations in more detail.

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. Performance Metric Attributes

The definitions of the metrics in this document are coarse-grained, based on network-layer traffic engineering performance metrics, for guidance only. A fine-grained framework specified in [RFC6390] requires that the fine-grained specification of a network performance metric include 6 components: (i) Metric Name, (ii) Metric Description, (iii) Method of Measurement or Calculation, (iv) Units of Measurement, (v) Measurement Points, and (vi) Measurement Timing. Requiring that an ALTO server provides precise, fine-grained values for all 6 components for each metric that it exposes may not be feasible or necessary for all ALTO use cases. For example, an ALTO server computing its metrics from network-layer traffic-engineering

performance metrics may not have information about the method of measurement or calculation (e.g., measured traffic patterns).

To address the issue and realize ALTO use cases, for metrics in Table 1, this document defines performance metric identifiers which can be used in the ALTO protocol with well-defined (i) Metric Name, (ii) Metric Description, (iv) Units of Measurement, and (v) Measurement Points, which are always specified by the specific ALTO services; for example, endpoint cost service is between the two endpoints. Hence, the ALTO performance metric identifiers provide basic metric attributes.

To allow the flexibility of allowing an ALTO server to provide fine-grained information such as Method of Measurement or Calculation, according to its policy and use cases, this document introduces context information so that the server can provide these additional details.

3.1. Performance Metric Context: "cost-context"

The core additional details of a performance metric specify "how" the metric is obtained. This is referred to as the source of the metric. Specifically, this document defines three types of coarse-grained metric information sources: "nominal", and "sla" (service level agreement), and "estimation".

For a given type of source, precise interpretation of a performance metric value can depend on specific measurement and computation parameters.

To make it possible to specify the source and the aforementioned parameters, this document introduces an optional "cost-context" field to the "cost-type" field defined by the ALTO base protocol (Section 10.7 of [RFC7285]) as the following:

```
object {
  CostMetric    cost-metric;
  CostMode      cost-mode;
  [CostContext  cost-context;]
  [JSONString   description;]
} CostType;

object {
  JSONString    cost-source;
  [JSONValue    parameters;]
} CostContext;
```

"cost-context" will not be used as a key to distinguish among performance metrics. Hence, an ALTO information resource MUST NOT announce multiple CostType with the same "cost-metric", "cost-mode" and "cost-context". They must be placed into different information resources.

The "cost-source" field of the "cost-context" field is defined as a string consisting of only US-ASCII alphanumeric characters (U+0030-U+0039, U+0041-U+005A, and U+0061-U+007A). The cost-source is used in this document to indicate a string of this format.

As mentioned above, this document defines three values for "cost-source": "nominal", "sla", and "estimation". The "cost-source" field of the "cost-context" field MUST be one registered in "ALTO Cost Source" registry (Section 8).

The "nominal" category indicates that the metric value is statically configured by the underlying devices. Not all metrics have reasonable "nominal" values. For example, throughput can have a nominal value, which indicates the configured transmission rate of the involved devices; latency typically does not have a nominal value.

The "sla" category indicates that the metric value is derived from some commitment which this document refers to as service-level agreement (SLA). Some operators also use terms such as "target" or "committed" values. For an "sla" metric, it is RECOMMENDED that the "parameters" field provide a link to the SLA definition.

The "estimation" category indicates that the metric value is computed through an estimation process. An ALTO server may compute "estimation" values by retrieving and/or aggregating information from routing protocols (e.g., [RFC7471], [RFC8570], [RFC8571]), traffic measurement management tools (e.g., TWAMP [RFC5357]), and measurement frameworks (e.g., IPPM), with corresponding operational issues. An illustration of potential information flows used for estimating these metrics is shown in Figure 1. Section 6 discusses in more detail the operational issues and how a network may address them.

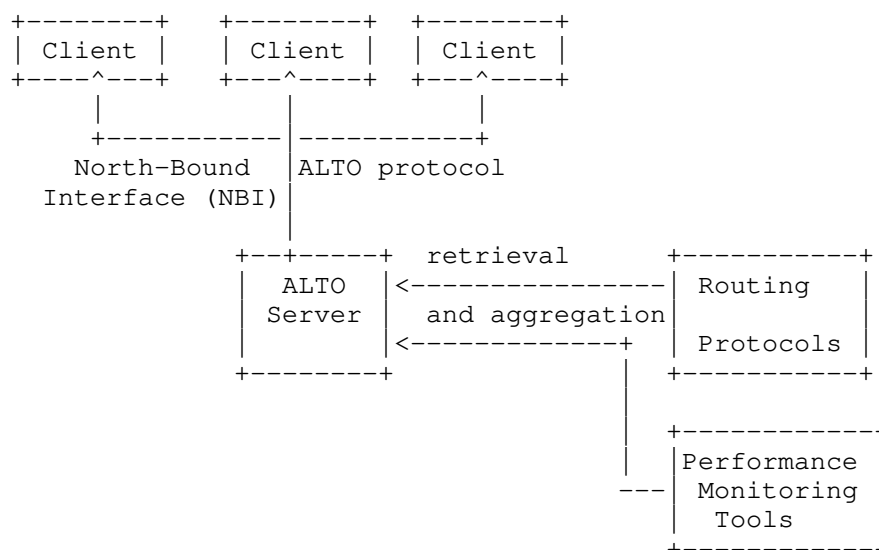


Figure 1. A framework to compute estimation to performance metrics

There can be multiple choices in deciding the cost-source category. It is the operator of an ALTO server who chooses the category. If a metric does not include a "cost-source" value, the application **MUST** assume that the value of "cost-source" is the most generic source, i.e., "estimation".

3.2. Performance Metric Statistics

The measurement of a performance metric often yields a set of samples from an observation distribution ([Prometheus]), instead of a single value. A statistical operator is applied to the samples to obtain a value to be reported to the client. Multiple statistical operators (e.g., min, median, and max) are commonly being used.

Hence, this document extends the general US-ASCII alphanumeric cost metric strings, formally specified as the CostMetric type defined in Section 10.6 of [RFC7285], as follows:

A cost metric string consists of a base metric identifier (or base identifier for short) string, followed by an optional statistical operator string, connected by the ASCII character colon (':', U+003A), if the statistical operator string exists. The total length of the cost metric string **MUST NOT** exceed 32, as required by [RFC7285].

The statistical operator string **MUST** be one of the following:

cur:

the instantaneous observation value of the metric from the most recent sample (i.e., the current value).

percentile, with letter 'p' followed by a number:

gives the percentile specified by the number following the letter 'p'. The number MUST be a non-negative JSON number in the range [0, 100] (i.e., greater than or equal to 0 and less than or equal to 100), followed by an optional decimal part, if a higher precision is needed. The decimal part should start with the '.' separator (U+002E), and followed by a sequence of one or more ASCII numbers between '0' and '9'. Assume this number is y and consider the samples coming from a random variable X. Then the metric returns x, such that the probability of X is less than or equal to x, i.e., $\text{Prob}(X \leq x) = y/100$. For example, delay-ow:p99 gives the 99% percentile of observed one-way delay; delay-ow:p99.9 gives the 99.9% percentile. Note that some systems use quantile, which is in the range [0, 1]. When there is a more common form for a given percentile, it is RECOMMENDED that the common form be used; that is, instead of p0, use min; instead of p50, use median; instead of p100, use max.

min:

the minimal value of the observations.

max:

the maximal value of the observations.

median:

the mid-point (i.e., p50) of the observations.

mean:

the arithmetic mean value of the observations.

stddev:

the standard deviation of the observations.

stdvar:

the standard variance of the observations.

Examples of cost metric strings then include "delay-ow", "delay-ow:min", "delay-ow:p99", where "delay-ow" is the base metric identifier string; "min" and "p99" are example statistical operator strings.

If a cost metric string does not have the optional statistical operator string, the statistical operator SHOULD be interpreted as the default statistical operator in the definition of the base metric. If the definition of the base metric does not provide a definition for the default statistical operator, the metric MUST be considered as the median value.

Note that RFC 7258 limits the overall cost metric identifier to 32 characters. The cost metric variants with statistical operator suffixes defined by this document are also subject to the same overall 32-character limit, so certain combinations of (long) base metric identifier and statistical operator will not be representable. If such a situation arises, it could be addressed by defining a new base metric identifier that is an "alias" of the desired base metric, with identical semantics and just a shorter name.

4. Packet Performance Metrics

This section introduces ALTO network performance metrics on one way delay, round-trip delay, delay variation, packet loss rate, and hop count. They measure the "quality of experience" of the stream of packets sent from a resource provider to a resource consumer. The measures of each individual packet (pkt) can include the delay from the time when the packet enters the network to the time when the packet leaves the network (pkt.delay); whether the packet is dropped before reaching the destination (pkt.dropped); the number of network hops that the packet traverses (pkt.hopcount). The semantics of the performance metrics defined in this section are that they are statistics computed from these measures; for example, the x-percentile of the one-way delay is the x-percentile of the set of delays {pkt.delay} for the packets in the stream.

4.1. Cost Metric: One-Way Delay (delay-ow)

4.1.1. Base Identifier

The base identifier for this performance metric is "delay-ow".

4.1.2. Value Representation

The metric value type is a single 'JSONNumber' type value conforming to the number specification of Section 6 of [RFC8259]. The unit is expressed in microseconds. Hence, the number can be a floating point number to express delay that is smaller than microseconds. The number MUST be non-negative.

4.1.3. Intended Semantics and Use

Intended Semantics: To specify the temporal and spatial aggregated delay of a stream of packets from the specified source to the specified destination. The base semantics of the metric is the Unidirectional Delay metric defined in [RFC8571,RFC8570,RFC7471], but instead of specifying the delay for a link, it is the (temporal) aggregation of the link delays from the source to the destination. A non-normative reference definition of end-to-end one-way delay is [RFC7679]. The spatial aggregation level is specified in the query context, e.g., provider-defined identifier (PID) to PID, or endpoint to endpoint, where PID is defined in Section 5.1 of [RFC7285].

Use: This metric could be used as a cost metric constraint attribute or as a returned cost metric in the response.

Example 1: Delay value on source-destination endpoint pairs

```
POST /endpointcost/lookup HTTP/1.1
Host: alto.example.com
Content-Length: 239
Content-Type: application/alto-endpointcostparams+json
Accept:
  application/alto-endpointcost+json,application/alto-error+json

{
  "cost-type": {
    "cost-mode": "numerical",
    "cost-metric": "delay-ow"
  },
  "endpoints": {
    "srcs": [
      "ipv4:192.0.2.2"
    ],
    "dsts": [
      "ipv4:192.0.2.89",
      "ipv4:198.51.100.34"
    ]
  }
}
```

HTTP/1.1 200 OK
Content-Length: 247
Content-Type: application/alto-endpointcost+json

```
{
  "meta": {
    "cost-type": {
      "cost-mode": "numerical",
      "cost-metric": "delay-ow"
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89": 10,
      "ipv4:198.51.100.34": 20
    }
  }
}
```

Note that since the "cost-type" does not include the "cost-source" field, the values are based on "estimation". Since the identifier does not include the statistical operator string component, the values will represent median values.

Example 1a below shows an example that is similar to Example 1, but for IPv6.

Example 1a: Delay value on source-destination endpoint pairs for IPv6

```
POST /endpointcost/lookup HTTP/1.1
Host: alto.example.com
Content-Length: 252
Content-Type: application/alto-endpointcostparams+json
Accept:
  application/alto-endpointcost+json,application/alto-error+json
```

```
{
  "cost-type": {
    "cost-mode": "numerical",
    "cost-metric": "delay-ow"
  },
  "endpoints": {
    "srcs": [
      "ipv6:2001:db8:100::1"
    ],
    "dsts": [
      "ipv6:2001:db8:100::2",
      "ipv6:2001:db8:100::3"
    ]
  }
}
```

```
HTTP/1.1 200 OK
Content-Length: 257
Content-Type: application/alto-endpointcost+json
```

```
{
  "meta": {
    "cost-type": {
      "cost-mode": "numerical",
      "cost-metric": "delay-ow"
    }
  },
  "endpoint-cost-map": {
    "ipv6:2001:db8:100::1": {
      "ipv6:2001:db8:100::2": 10,
      "ipv6:2001:db8:100::3": 20
    }
  }
}
```

4.1.4. Cost-Context Specification Considerations

"nominal": Typically network one-way delay does not have a nominal value.

"sla": Many networks provide delay-related parameters in their application-level SLAs. It is RECOMMENDED that the "parameters" field of an "sla" one-way delay metric include a link (i.e., a field named "link") providing an URI to the specification of SLA details, if available. Such a specification can be either free text for possible presentation to the user, or a formal specification. The format of the specification is out of the scope of this document.

"estimation": The exact estimation method is out of the scope of this document. There can be multiple sources to estimate one-way delay. For example, the ALTO server may estimate the end-to-end delay by aggregation of routing protocol link metrics; the server may also estimate the delay using active, end-to-end measurements, for example, using the IPPM framework [RFC2330].

If the estimation is computed by aggregation of routing protocol link metrics (e.g., OSPF [RFC7471], IS-IS [RFC8570], or BGP-LS [RFC8571]) Unidirectional Delay link metrics, it is RECOMMENDED that the "parameters" field of an "estimation" one-way delay metric include the following information: (1) the RFC defining the routing protocol metrics (e.g., <https://www.rfc-editor.org/info/rfc7471> for RFC7471 derived metrics); (2) configurations of the routing link metrics such as configured intervals; and (3) the aggregation method from link metrics to end-to-end metrics. During aggregation from link metrics to the end-to-end metric, the server should be cognizant of potential issues when computing an end-to-end summary statistic from link statistics. The default end-to-end average one-way delay is the sum of average link one-way delays. If an ALTO server provides the min and max statistical operators for the one-way delay metric, the values can be computed directly from the routing link metrics, as [RFC7471,RFC8570,RFC8571] provide Min/Max Unidirectional Link Delay.

If the estimation is from the IPPM measurement framework, it is RECOMMENDED that the "parameters" field of an "estimation" one-way delay metric includes the following information: the URI to the URI field of the IPPM metric defined in the IPPM performance metric [IANA-IPPM] registry (e.g., https://www.iana.org/assignments/performance-metrics/OWDelay_Active_IP-UDP-Poisson-Payload250B_RFC8912sec7_Seconds_95Percentile). The IPPM metric MUST be one-way delay (i.e., IPPM OWDelay* metrics). The statistical operator of the ALTO metric MUST be consistent with the IPPM statistical property (e.g., 95-th percentile).

4.2. Cost Metric: Round-trip Delay (delay-rt)

4.2.1. Base Identifier

The base identifier for this performance metric is "delay-rt".

4.2.2. Value Representation

The metric value type is a single 'JSONNumber' type value conforming to the number specification of Section 6 of [RFC8259]. The number MUST be non-negative. The unit is expressed in microseconds.

4.2.3. Intended Semantics and Use

Intended Semantics: To specify temporal and spatial aggregated round-trip delay between the specified source and specified destination. The base semantics is that it is the sum of one-way delay from the source to the destination and the one-way delay from the destination back to the source, where the one-way delay is defined in Section 4.1. A non-normative reference definition of end-to-end round-trip delay is [RFC2681]. The spatial aggregation level is specified in the query context (e.g., PID to PID, or endpoint to endpoint).

Note that it is possible for a client to query two one-way delays (delay-ow) and then compute the round-trip delay. The server should be cognizant of the consistency of values.

Use: This metric could be used either as a cost metric constraint attribute or as a returned cost metric in the response.

Example 2: Round-trip Delay of source-destination endpoint pairs

```
POST /endpointcost/lookup HTTP/1.1
Host: alto.example.com
Content-Length: 238
Content-Type: application/alto-endpointcostparams+json
Accept:
  application/alto-endpointcost+json,application/alto-error+json
```

```
{
  "cost-type": {
    "cost-mode": "numerical",
    "cost-metric": "delay-rt"
  },
  "endpoints": {
    "srcs": [
      "ipv4:192.0.2.2"
    ],
    "dsts": [
      "ipv4:192.0.2.89",
      "ipv4:198.51.100.34"
    ]
  }
}
```

```
HTTP/1.1 200 OK
Content-Length: 245
Content-Type: application/alto-endpointcost+json
```

```
{
  "meta": {
    "cost-type": {
      "cost-mode": "numerical",
      "cost-metric": "delay-rt"
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89": 4,
      "ipv4:198.51.100.34": 3
    }
  }
}
```

4.2.4. Cost-Context Specification Considerations

"nominal": Typically network round-trip delay does not have a nominal value.

"sla": See the "sla" entry in Section 4.1.4.

"estimation": See the "estimation" entry in Section 4.1.4. For estimation by aggregation of routing protocol link metrics, the aggregation should include all links from the source to the destination and then back to the source; for estimation using IPPM, the IPPM metric MUST be round-trip delay (i.e., IPPM RTDelay* metrics). The statistical operator of the ALTO metric MUST be consistent with the IPPM statistical property (e.g., 95-th percentile).

4.3. Cost Metric: Delay Variation (delay-variation)

4.3.1. Base Identifier

The base identifier for this performance metric is "delay-variation".

4.3.2. Value Representation

The metric value type is a single 'JSONNumber' type value conforming to the number specification of Section 6 of [RFC8259]. The number MUST be non-negative. The unit is expressed in microseconds.

4.3.3. Intended Semantics and Use

Intended Semantics: To specify temporal and spatial aggregated delay variation (also called delay jitter) with respect to the minimum delay observed on the stream over the one-way delay from the specified source and destination, where the one-way delay is defined in Section 4.1. A non-normative reference definition of end-to-end one-way delay variation is [RFC3393]. Note that [RFC3393] allows the specification of a generic selection function *F* to unambiguously define the two packets selected to compute delay variations. This document defines the specific case that *F* selects as the "first" packet the one with the smallest one-way delay. The spatial aggregation level is specified in the query context (e.g., PID to PID, or endpoint to endpoint).

Note that in statistics, variations are typically evaluated by the distance from samples relative to the mean. In networking context, it is more commonly defined from samples relative to the min. This definition follows the networking convention.

Use: This metric could be used either as a cost metric constraint attribute or as a returned cost metric in the response.

Example 3: Delay variation value on source-destination endpoint pairs

```
POST /endpointcost/lookup HTTP/1.1
Host: alto.example.com
Content-Length: 245
Content-Type: application/alto-endpointcostparams+json
Accept:
    application/alto-endpointcost+json,application/alto-error+json
```

```
{
  "cost-type": {
    "cost-mode": "numerical",
    "cost-metric": "delay-variation"
  },
  "endpoints": {
    "srcs": [
      "ipv4:192.0.2.2"
    ],
    "dsts": [
      "ipv4:192.0.2.89",
      "ipv4:198.51.100.34"
    ]
  }
}
```

```
HTTP/1.1 200 OK
Content-Length: 252
Content-Type: application/alto-endpointcost+json
```

```
{
  "meta": {
    "cost-type": {
      "cost-mode": "numerical",
      "cost-metric": "delay-variation"
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89": 0,
      "ipv4:198.51.100.34": 1
    }
  }
}
```

4.3.4. Cost-Context Specification Considerations

"nominal": Typically network delay variation does not have a nominal value.

"sla": See the "sla" entry in Section 4.1.4.

"estimation": See the "estimation" entry in Section 4.1.4. For estimation by aggregation of routing protocol link metrics, the default aggregation of the average of delay variations is the sum of the link delay variations; for estimation using IPPM, the IPPM metric MUST be delay variation (i.e., IPPM OWPDV* metrics). The statistical operator of the ALTO metric MUST be consistent with the IPPM statistical property (e.g., 95-th percentile).

4.4. Cost Metric: Loss Rate (lossrate)

4.4.1. Base Identifier

The base identifier for this performance metric is "lossrate".

4.4.2. Value Representation

The metric value type is a single 'JSONNumber' type value conforming to the number specification of Section 6 of [RFC8259]. The number MUST be non-negative. The value represents the percentage of packet losses.

4.4.3. Intended Semantics and Use

Intended Semantics: To specify temporal and spatial aggregated one-way packet loss rate from the specified source and the specified destination. The base semantics of the metric is the Unidirectional Link Loss metric defined in [RFC8571,RFC8570,RFC7471], but instead of specifying the loss for a link, it is the aggregated loss of all links from the source to the destination. The spatial aggregation level is specified in the query context (e.g., PID to PID, or endpoint to endpoint).

Use: This metric could be used as a cost metric constraint attribute or as a returned cost metric in the response.

Example 5: Loss rate value on source-destination endpoint pairs

```
POST /endpointcost/lookup HTTP/1.1
Host: alto.example.com
Content-Length: 238
Content-Type: application/alto-endpointcostparams+json
Accept:
  application/alto-endpointcost+json,application/alto-error+json
```

```
{
  "cost-type": {
    "cost-mode": "numerical",
    "cost-metric": "lossrate"
  },
  "endpoints": {
    "srcs": [
      "ipv4:192.0.2.2"
    ],
    "dsts": [
      "ipv4:192.0.2.89",
      "ipv4:198.51.100.34"
    ]
  }
}
```

```
HTTP/1.1 200 OK
Content-Length: 248
Content-Type: application/alto-endpointcost+json
```

```
{
  "meta": {
    "cost-type": {
      "cost-mode": "numerical",
      "cost-metric": "lossrate"
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89": 0,
      "ipv4:198.51.100.34": 0.01
    }
  }
}
```

4.4.4. Cost-Context Specification Considerations

"nominal": Typically packet loss rate does not have a nominal value, although some networks may specify zero losses.

"sla": See the "sla" entry in Section 4.1.4..

"estimation": See the "estimation" entry in Section 4.1.4. For estimation by aggregation of routing protocol link metrics, the default aggregation of the average of loss rate is the sum of the link link loss rates. But this default aggregation is valid only if two conditions are met: (1) it is valid only when link loss rates are low, and (2) it assumes that each link's loss events are uncorrelated with every other link's loss events. When loss rates at the links are high but independent, the general formula for aggregating loss assuming each link is independent is to compute end-to-end loss as one minus the product of the success rate for each link. Aggregation when losses at links are correlated can be more complex and the ALTO server should be cognizant of correlated loss rates. For estimation using IPPM, the IPPM metric MUST be packet loss (i.e., IPPM OWLoss* metrics). The statistical operator of the ALTO metric MUST be consistent with the IPPM statistical property (e.g., 95-th percentile).

4.5. Cost Metric: Hop Count (hopcount)

The hopcount metric is mentioned in Section 9.2.3 of [RFC7285] as an example. This section further clarifies its properties.

4.5.1. Base Identifier

The base identifier for this performance metric is "hopcount".

4.5.2. Value Representation

The metric value type is a single 'JSONNumber' type value conforming to the number specification of Section 6 of [RFC8259]. The number MUST be a non-negative integer (greater than or equal to 0). The value represents the number of hops.

4.5.3. Intended Semantics and Use

Intended Semantics: To specify the number of hops in the path from the specified source to the specified destination. The hop count is a basic measurement of distance in a network and can be exposed as the number of router hops computed from the routing protocols originating this information. A hop, however, may represent other units. The spatial aggregation level is specified in the query context (e.g., PID to PID, or endpoint to endpoint).

Use: This metric could be used as a cost metric constraint attribute or as a returned cost metric in the response.

Example 4: hopcount value on source-destination endpoint pairs

```
POST /endpointcost/lookup HTTP/1.1
Host: alto.example.com
Content-Length: 238
Content-Type: application/alto-endpointcostparams+json
Accept:
  application/alto-endpointcost+json,application/alto-error+json
```

```
{
  "cost-type": {
    "cost-mode": "numerical",
    "cost-metric": "hopcount"
  },
  "endpoints": {
    "srcs": [
      "ipv4:192.0.2.2"
    ],
    "dsts": [
      "ipv4:192.0.2.89",
      "ipv4:198.51.100.34"
    ]
  }
}
```

```
HTTP/1.1 200 OK
Content-Length: 245
Content-Type: application/alto-endpointcost+json
```

```
{
  "meta": {
    "cost-type": {
      "cost-mode": "numerical",
      "cost-metric": "hopcount"
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89": 5,
      "ipv4:198.51.100.34": 3
    }
  }
}
```

4.5.4. Cost-Context Specification Considerations

"nominal": Typically hop count does not have a nominal value.

"sla": Typically hop count does not have an SLA value.

"estimation": The exact estimation method is out of the scope of this document. An example of estimating hopcounts is by importing from IGP routing protocols. It is RECOMMENDED that the "parameters" field of an "estimation" hop count define the meaning of a hop.

5. Throughput/Bandwidth Performance Metrics

This section introduces four throughput/bandwidth related metrics. Given a specified source to a specified destination, these metrics reflect the volume of traffic that the network can carry from the source to the destination.

5.1. Cost Metric: TCP Throughput (tput)

5.1.1. Base Identifier

The base identifier for this performance metric is "tput".

5.1.2. Value Representation

The metric value type is a single 'JSONNumber' type value conforming to the number specification of Section 6 of [RFC8259]. The number MUST be non-negative. The unit is bytes per second.

5.1.3. Intended Semantics and Use

Intended Semantics: To give the throughput of a TCP congestion-control conforming flow from the specified source to the specified destination. The throughput SHOULD be interpreted as only an estimation, and the estimation is designed only for bulk flows.

Use: This metric could be used as a cost metric constraint attribute or as a returned cost metric in the response.

Example 5: TCP throughput value on source-destination endpoint pairs

```
POST /endpointcost/lookup HTTP/1.1
Host: alto.example.com
Content-Length: 234
Content-Type: application/alto-endpointcostparams+json
Accept:
  application/alto-endpointcost+json,application/alto-error+json
```

```
{
  "cost-type": {
    "cost-mode": "numerical",
    "cost-metric": "tput"
  },
  "endpoints": {
    "srcs": [
      "ipv4:192.0.2.2"
    ],
    "dsts": [
      "ipv4:192.0.2.89",
      "ipv4:198.51.100.34"
    ]
  }
}
```

```
HTTP/1.1 200 OK
Content-Length: 251
Content-Type: application/alto-endpointcost+json
```

```
{
  "meta": {
    "cost-type": {
      "cost-mode": "numerical",
      "cost-metric": "tput"
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89": 256000,
      "ipv4:198.51.100.34": 128000
    }
  }
}
```

5.1.4. Cost-Context Specification Considerations

"nominal": Typically TCP throughput does not have a nominal value, and SHOULD NOT be generated.

"sla": Typically TCP throughput does not have an SLA value, and SHOULD NOT be generated.

"estimation": The exact estimation method is out of the scope of this document. It is RECOMMENDED that the "parameters" field of an "estimation" TCP throughput metric include the following information: (1) the congestion-control algorithm; and (2) the estimation methodology. To specify (1), it is RECOMMENDED that the "parameters" field (object) include a field named "congestion-control-algorithm", which provides a URI for the specification of the algorithm; for example, for an ALTO server to provide estimation to the throughput of a Cubic Congestion control flow, its "parameters" includes a field "congestion-control-algorithm", with value being set to [I-D.ietf-tcpm-rfc8312bis]; for an ongoing congestion control algorithm such as BBR, a link to its specification. To specify (2), the "parameters" includes as many details as possible; for example, for TCP Cubic throughput estimation, the "parameters" field specifies that the throughput is estimated by setting `_C_` to 0.4, and the Equation in Figure 8 of [I-D.ietf-tcpm-rfc8312bis] is applied; as an alternative, the methodology may be based on the NUM model [Prophet], or the G2 model [G2]. The exact specification of the parameters field is out of the scope of this document.

5.2. Cost Metric: Residual Bandwidth (bw-residual)

5.2.1. Base Identifier

The base identifier for this performance metric is "bw-residual".

5.2.2. Value Representation

The metric value type is a single 'JSONNumber' type value that is non-negative. The unit of measurement is bytes per second.

5.2.3. Intended Semantics and Use

Intended Semantics: To specify temporal and spatial residual bandwidth from the specified source and the specified destination. The base semantics of the metric is the Unidirectional Residual Bandwidth metric defined in [RFC8571,RFC8570,RFC7471], but instead of specifying the residual bandwidth for a link, it is the residual bandwidth of the path from the source to the destination. Hence, it is the minimal residual bandwidth among all links from the source to the destination. When the max statistical operator is defined for the metric, it typically provides the minimum of the link capacities along the path, as the default value of the residual bandwidth of a link is its link capacity [RFC8571,8570,7471]. The spatial aggregation unit is specified in the query context (e.g., PID to PID,

or endpoint to endpoint).

The default statistical operator for residual bandwidth is the current instantaneous sample; that is, the default is assumed to be "cur".

Use: This metric could be used either as a cost metric constraint attribute or as a returned cost metric in the response.

Example 7: bw-residual value on source-destination endpoint pairs

```
POST /endpointcost/lookup HTTP/1.1
```

```
Host: alto.example.com
```

```
Content-Length: 241
```

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

```
Accept:
```

```
  application/alto-endpointcost+json,application/alto-error+json
```

```
{
  "cost-type": {
    "cost-mode": "numerical",
    "cost-metric": "bw-residual"
  },
  "endpoints": {
    "srcs": [
      "ipv4:192.0.2.2"
    ],
    "dsts": [
      "ipv4:192.0.2.89",
      "ipv4:198.51.100.34"
    ]
  }
}
```

HTTP/1.1 200 OK
Content-Length: 255
Content-Type: application/alto-endpointcost+json

```
{
  "meta": {
    "cost-type": {
      "cost-mode": "numerical",
      "cost-metric": "bw-residual"
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89": 0,
      "ipv4:198.51.100.34": 2000
    }
  }
}
```

5.2.4. Cost-Context Specification Considerations

"nominal": Typically residual bandwidth does not have a nominal value.

"sla": Typically residual bandwidth does not have an "sla" value.

"estimation": See the "estimation" entry in Section 4.1.4 on aggregation of routing protocol link metrics. The current ("cur") residual bandwidth of a path is the minimal of the residual bandwidth of all links on the path.

5.3. Cost Metric: Available Bandwidth (bw-available)

5.3.1. Base Identifier

The base identifier for this performance metric is "bw-available".

5.3.2. Value Representation

The metric value type is a single 'JSONNumber' type value that is non-negative. The unit of measurement is bytes per second.

5.3.3. Intended Semantics and Use

Intended Semantics: To specify temporal and spatial available bandwidth from the specified source to the specified destination. The base semantics of the metric is the Unidirectional Available Bandwidth metric defined in [RFC8571,RFC8570,RFC7471], but instead of specifying the available bandwidth for a link, it is the available bandwidth of the path from the source to the destination. Hence, it is the minimal available bandwidth among all links from the source to the destination. The spatial aggregation unit is specified in the query context (e.g., PID to PID, or endpoint to endpoint).

The default statistical operator for available bandwidth is the current instantaneous sample; that is, the default is assumed to be "cur".

Use: This metric could be used either as a cost metric constraint attribute or as a returned cost metric in the response.

Example 8: bw-available value on source-destination endpoint pairs

```
POST /endpointcost/lookup HTTP/1.1
Host: alto.example.com
Content-Length: 244
Content-Type: application/alto-endpointcostparams+json
Accept:
  application/alto-endpointcost+json,application/alto-error+json

{
  "cost-type": {
    "cost-mode": "numerical",
    "cost-metric": "bw-available"
  },
  "endpoints": {
    "srcs": [
      "ipv4:192.0.2.2"
    ],
    "dsts": [
      "ipv4:192.0.2.89",
      "ipv4:198.51.100.34"
    ]
  }
}
```

HTTP/1.1 200 OK
Content-Length: 255
Content-Type: application/alto-endpointcost+json

```
{
  "meta": {
    "cost-type": {
      "cost-mode": "numerical",
      "cost-metric": "bw-available"
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89": 0,
      "ipv4:198.51.100.34": 2000
    }
  }
}
```

5.3.4. Cost-Context Specification Considerations

"nominal": Typically available bandwidth does not have a nominal value.

"sla": Typically available bandwidth does not have an "sla" value.

"estimation": See the "estimation" entry in Section 4.1.4 on aggregation of routing protocol link metrics. The current ("cur") available bandwidth of a path is the minimum of the available bandwidth of all links on the path.

6. Operational Considerations

The exact measurement infrastructure, measurement condition, and computation algorithms can vary from different networks, and are outside the scope of this document. Both the ALTO server and the ALTO clients, however, need to be cognizant of the operational issues discussed in the following sub-sections.

Also, the performance metrics specified in this document are similar, in that they may use similar data sources and have similar issues in their calculation. Hence, this document specifies common issues unless one metric has its unique challenges.

6.1. Source Considerations

The addition of the "cost-source" field is to solve a key issue: An ALTO server needs data sources to compute the cost metrics described in this document, and an ALTO client needs to know the data sources to better interpret the values.

To avoid too fine-grained information, this document introduces "cost-source" to indicate only the high-level type of data sources: "estimation", "nominal" or "lsa", where "estimation" is a type of measurement data source, "nominal" is a type of static configuration, and "lsa" is a type that is more based on policy.

For estimation, for example, the ALTO server may use log servers or the OAM system as its data source as recommended by [RFC7971]. In particular, the cost metrics defined in this document can be computed using routing systems as the data sources.

6.2. Metric Timestamp Consideration

Despite the introduction of the additional cost-context information, the metrics do not have a field to indicate the timestamps of the data used to compute the metrics. To indicate this attribute, the ALTO server SHOULD return HTTP "Last-Modified", to indicate the freshness of the data used to compute the performance metrics.

If the ALTO client obtains updates through an incremental update mechanism [RFC8895], the client SHOULD assume that the metric is computed using a snapshot at the time that is approximated by the receiving time.

6.3. Backward Compatibility Considerations

One potential issue introduced by the optional "cost-source" field is backward compatibility. Consider that an IRD which defines two cost-types with the same "cost-mode" and "cost-metric", but one with "cost-source" being "estimation" and the other being "lsa". Then an ALTO client that is not aware of the extension will not be able to distinguish between these two types. A similar issue can arise even with a single cost-type, whose "cost-source" is "lsa": an ALTO client that is not aware of this extension will ignore this field and consider the metric estimation.

To address the backward-compatibility issue, if a "cost-metric" is "routingcost" and the metric contains a "cost-context" field, then it MUST be "estimation"; if it is not, the client SHOULD reject the information as invalid.

6.4. Computation Considerations

The metric values exposed by an ALTO server may result from additional processing on measurements from data sources to compute exposed metrics. This may involve data processing tasks such as aggregating the results across multiple systems, removing outliers, and creating additional statistics. There are two challenges on the computation of ALTO performance metrics.

6.4.1. Configuration Parameters Considerations

Performance metrics often depend on configuration parameters, and exposing such configuration parameters can help an ALTO client to better understand the exposed metrics. In particular, an ALTO server may be configured to compute a TE metric (e.g., packet loss rate) in fixed intervals, say every T seconds. To expose this information, the ALTO server may provide the client with two pieces of additional information: (1) when the metrics are last computed, and (2) when the metrics will be updated (i.e., the validity period of the exposed metric values). The ALTO server can expose these two pieces of information by using the HTTP response headers Last-Modified and Expires.

6.4.2. Aggregation Computation Considerations

An ALTO server may not be able to measure the performance metrics to be exposed. The basic issue is that the "source" information can often be link level. For example, routing protocols often measure and report only per link loss, not end-to-end loss; similarly, routing protocols report link level available bandwidth, not end-to-end available bandwidth. The ALTO server then needs to aggregate these data to provide an abstract and unified view that can be more useful to applications. The server should consider that different metrics may use different aggregation computation. For example, the end-to-end latency of a path is the sum of the latency of the links on the path; the end-to-end available bandwidth of a path is the minimum of the available bandwidth of the links on the path; in contrast, aggregating loss values is complicated by the potential for correlated loss events on different links in the path

7. Security Considerations

The properties defined in this document present no security considerations beyond those in Section 15 of the base ALTO specification [RFC7285].

However, concerns addressed in Sections 15.1, 15.2, and 15.3 of [RFC7285] remain of utmost importance. Indeed, Traffic Engineering (TE) performance is highly sensitive ISP information; therefore, sharing TE metric values in numerical mode requires full mutual confidence between the entities managing the ALTO server and the ALTO client. ALTO servers will most likely distribute numerical TE performance to ALTO clients under strict and formal mutual trust agreements. On the other hand, ALTO clients must be cognizant on the risks attached to such information that they would have acquired outside formal conditions of mutual trust.

To mitigate confidentiality risks during information transport of TE performance metrics, the operator should address the risk of ALTO information being leaked to malicious Clients or third parties, through attacks such as the person-in-the-middle (PITM) attacks. As specified in "Protection Strategies" (Section 15.3.2 of [RFC7285]), the ALTO Server should authenticate ALTO Clients when transmitting an ALTO information resource containing sensitive TE performance metrics. "Authentication and Encryption" (Section 8.3.5 of [RFC7285]) specifies that "ALTO Server implementations as well as ALTO Client implementations MUST support the "https" URI scheme of [RFC7230] and Transport Layer Security (TLS) of [RFC8446]".

8. IANA Considerations

IANA has created and now maintains the "ALTO Cost Metric" registry, listed in Section 14.2, Table 3 of [RFC7285]. This registry is located at <<https://www.iana.org/assignments/alto-protocol/alto-protocol.xhtml#cost-metrics>>. This document requests to add the following entries to the "ALTO Cost Metric" registry.

Identifier	Intended Semantics
delay-ow	Section 4.1 of [RFCXXX]
delay-rt	Section 4.2 of [RFCXXX]
delay-variation	Section 4.3 of [RFCXXX]
lossrate	Section 4.4 of [RFCXXX]
hopcount	Section 4.5 of [RFCXXX]
tput	Section 5.1 of [RFCXXX]
bw-residual	Section 5.2 of [RFCXXX]
bw-available	Section 5.3 of [RFCXXX]

* [Note to the RFC Editor]: Please replace RFCXXX with the RFC number assigned to this document.

This document requests the creation of the "ALTO Cost Source" registry. This registry serves two purposes. First, it ensures uniqueness of identifiers referring to ALTO cost source types. Second, it provides references to particular semantics of allocated cost source types to be applied by both ALTO servers and applications utilizing ALTO clients.

A new ALTO cost source can be added after IETF Review [RFC8126], to ensure that proper documentation regarding the new ALTO cost source and its security considerations have been provided. The RFC(s) documenting the new cost source should be detailed enough to provide guidance to both ALTO service providers and applications utilizing ALTO clients as to how values of the registered ALTO cost source should be interpreted. Updates and deletions of ALTO cost source follow the same procedure.

Registered ALTO address type identifiers MUST conform to the syntactical requirements specified in Section 3.1. Identifiers are to be recorded and displayed as strings.

Requests to add a new value to the registry MUST include the following information:

- * Identifier: The name of the desired ALTO cost source type.
- * Intended Semantics: ALTO cost source type carry with them semantics to guide their usage by ALTO clients. Hence, a document defining a new type should provide guidance to both ALTO service providers and applications utilizing ALTO clients as to how values of the registered ALTO endpoint property should be interpreted.
- * Security Considerations: ALTO cost source types expose information to ALTO clients. ALTO service providers should be made aware of the security ramifications related to the exposure of a cost source type.

This specification requests registration of the identifiers "nominal", "sla", and "estimation" listed in the table below. Semantics for these are documented in Section 3.1, and security considerations are documented in Section 7.

Identifier	Intended Semantics	Security Considerations
nominal	Values in nominal cases; Section 3.1 of [RFCXXX]	Section 7 of [RFCXXX]
sla	Values reflecting service level agreement; Section 3.1 of [RFCXXXX]	Section 7 of [RFCXXX]
estimation	Values by estimation; Section 3.1 of [RFCXXX]	Section 7 of [RFCXXX]

9. Acknowledgments

The authors of this document would also like to thank Martin Duke for the highly informative, thorough AD reviews and comments. We thank Christian Amsuess, Elwyn Davies, Haizhou Du, Kai Gao, Geng Li, Lili Liu, Danny Alex Lachos Perez, and Brian Trammell for the reviews and comments. We thank Benjamin Kaduk, Eric Kline, Francesca Palombini, Lars Eggert, Martin Vigoureux, Murraray Kucherawy, Roman Danyliw, Zaheduzzaman Sarker, Eric Vyncke for discussions and comments that improve this document.

10. References

10.1. Normative References

- [I-D.ietf-tcpm-rfc8312bis]
 Xu, L., Ha, S., Rhee, I., Goel, V., and L. Eggert, "CUBIC for Fast and Long-Distance Networks", Work in Progress, Internet-Draft, draft-ietf-tcpm-rfc8312bis-07, 4 March 2022, <<https://www.ietf.org/archive/id/draft-ietf-tcpm-rfc8312bis-07.txt>>.
- [IANA-IPPM]
 IANA, "Performance Metrics Registry, <https://www.iana.org/assignments/performance-metrics/performance-metrics.xhtml>".
- [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>>.

- [RFC3630] Katz, D., Kompella, K., and D. Yeung, "Traffic Engineering (TE) Extensions to OSPF Version 2", RFC 3630, DOI 10.17487/RFC3630, September 2003, <<https://www.rfc-editor.org/info/rfc3630>>.
- [RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, DOI 10.17487/RFC5305, October 2008, <<https://www.rfc-editor.org/info/rfc5305>>.
- [RFC6390] Clark, A. and B. Claise, "Guidelines for Considering New Performance Metric Development", BCP 170, RFC 6390, DOI 10.17487/RFC6390, October 2011, <<https://www.rfc-editor.org/info/rfc6390>>.
- [RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <<https://www.rfc-editor.org/info/rfc7230>>.
- [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>>.
- [RFC7471] Giacalone, S., Ward, D., Drake, J., Atlas, A., and S. Previdi, "OSPF Traffic Engineering (TE) Metric Extensions", RFC 7471, DOI 10.17487/RFC7471, March 2015, <<https://www.rfc-editor.org/info/rfc7471>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [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>>.
- [RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <<https://www.rfc-editor.org/info/rfc8259>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.

- [RFC8570] Ginsberg, L., Ed., Previdi, S., Ed., Giacalone, S., Ward, D., Drake, J., and Q. Wu, "IS-IS Traffic Engineering (TE) Metric Extensions", RFC 8570, DOI 10.17487/RFC8570, March 2019, <<https://www.rfc-editor.org/info/rfc8570>>.
- [RFC8571] Ginsberg, L., Ed., Previdi, S., Wu, Q., Tantsura, J., and C. Filsfils, "BGP - Link State (BGP-LS) Advertisement of IGP Traffic Engineering Performance Metric Extensions", RFC 8571, DOI 10.17487/RFC8571, March 2019, <<https://www.rfc-editor.org/info/rfc8571>>.
- [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>>.

10.2. Informative References

- [FlowDirector] Pujol, E., Poese, I., Zerwas, J., Smaragdakis, G., and A. Feldmann, "Steering Hyper-Giants' Traffic at Scale", ACM CoNEXT 2020, 2020.
- [G2] Ros-Giralt, J., Bohara, A., Yellamraju, S., and et. al., "On the Bottleneck Structure of Congestion-Controlled Networks", ACM SIGMETRICS 2019, 2020.
- [I-D.corre-quick-throughput-testing] Corre, K., "Framework for QUIC Throughput Testing", Work in Progress, Internet-Draft, draft-corre-quick-throughput-testing-00, 17 September 2021, <<https://www.ietf.org/archive/id/draft-corre-quick-throughput-testing-00.txt>>.
- [Prometheus] Volz, J. and B. Rabenstein, "Prometheus: A Next-Generation Monitoring System", 2015.
- [Prophet] Gao, K., Zhang, J., and YR. Yang, "Prophet: Fast, Accurate Throughput Prediction with Reactive Flows", ACM/IEEE Transactions on Networking July, 2020.
- [RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, DOI 10.17487/RFC2330, May 1998, <<https://www.rfc-editor.org/info/rfc2330>>.

- [RFC2681] Almes, G., Kalidindi, S., and M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, DOI 10.17487/RFC2681, September 1999, <<https://www.rfc-editor.org/info/rfc2681>>.
- [RFC3393] Demichelis, C. and P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)", RFC 3393, DOI 10.17487/RFC3393, November 2002, <<https://www.rfc-editor.org/info/rfc3393>>.
- [RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <<https://www.rfc-editor.org/info/rfc5357>>.
- [RFC7679] Almes, G., Kalidindi, S., Zekauskas, M., and A. Morton, Ed., "A One-Way Delay Metric for IP Performance Metrics (IPPM)", STD 81, RFC 7679, DOI 10.17487/RFC7679, January 2016, <<https://www.rfc-editor.org/info/rfc7679>>.
- [RFC7971] Stiemerling, M., Kiesel, S., Scharf, M., Seidel, H., and S. Previdi, "Application-Layer Traffic Optimization (ALTO) Deployment Considerations", RFC 7971, DOI 10.17487/RFC7971, October 2016, <<https://www.rfc-editor.org/info/rfc7971>>.
- [RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <<https://www.rfc-editor.org/info/rfc9000>>.

Authors' Addresses

Qin Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing
Jiangsu, 210012
China
Email: bill.wu@huawei.com

Y. Richard Yang
Yale University
51 Prospect St
New Haven, CT 06520
United States of America
Email: yry@cs.yale.edu

Young Lee
Samsung
Email: young.lee@gmail.com

Dhruv Dhody
Huawei
Leela Palace
Bangalore 560008
Karnataka
India
Email: dhruv.ietf@gmail.com

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

Luis Miguel Contreras Murillo
Telefonica
Madrid
Spain
Email: luismiguel.contrerasmurillo@telefonica.com

ALTO
Internet-Draft
Intended status: Standards Track
Expires: September 10, 2020

S. Randriamasy
Nokia Bell Labs
March 9, 2020

ALTO Contextual Cost Values
draft-randriamasy-alto-cost-context-03

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 one JSON value for a requested metric.

This document introduces several protocol extensions to allow ALTO clients to support use cases such as context based connection selection in cellular networks and calendaring for unattended data. This document refers to other extension proposals posted in the ALTO WG that can support the present use cases as well. Likewise, some of the proposed extensions may serve other ALTO use cases.

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 <https://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on September 10, 2020.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction	3
2. Use cases	4
2.1. Use Case 1: conditional RF costs in cellular networks . .	4
2.2. Use case 2: access-aware endpoint selection	5
3. Required ALTO extensions	6
4. Design options and examples	7
4.1. Overview of context features	7
4.1.1. Applicable ALTO services	8
4.2. Example IRD	8
4.3. Use case 1: Example scenario for the FCM Service	10
4.4. Design option: Network Map with cells as PIDs	11
4.4.1. Example: FCM Request for contextual 'ARFcost'	11
4.4.2. Example: FCM Response for contextual ARFcost	12
4.5. Use case 2: example ALTO transactions for the ECS	13
4.5.1. Use case 2: example with logical context parameter combinations	13
5. Deployment case: local ALTO Server cascaded with global ALTO Server	16
5.1. Cascaded ALTO Servers with one network map each	16
6. IANA Considerations	17
7. Security Considerations	17
8. Acknowledgements	17
9. References	17
9.1. Normative References	17
9.2. Informative References	17
Appendix A. An Appendix	18
Author's Address	18

1. Introduction

The IETF ALTO protocol specified in [RFC7285] provides guidance to over the top applications which have to select one or several hosts or endpoints from a set of candidates that are able to provide a desired data resource, or which need some provider-centric insight on the cost of application paths to these endhosts. The ALTO Service has defined network and cost maps to provide basic network information, where the cost maps allow only one JSON value for a requested metric.

This draft brings a use case where providing different values for the same cost metric can help in optimizing the application path selection. Typically, when an end host can connect to the network via multiple technologies or access points, the path performance for a metric may be accordingly impacted.

The present draft proposes to extend the cost information specified in [RFC7285] by providing, for the same cost metric, several possible cost values. Which value to provide depends on qualitative criteria as opposed to quantitative criteria such as time. The purpose is to allow a finer grained decision on which application endpoint or sub network to access.

Previous ALTO WG discussions have suggested introducing "the ability to "name" cost maps so that a single Information Resource Directory can link multiple cost maps with the same cost type to a single network map." The goal was to provide, for a given cost metric, multiple cost values depending on qualitative conditions named "circumstance", where a circumstance reflects a given policy.

For applications such as video download or streaming, a user equipment (UE) may use [RFC7285] to choose the best possible application resource location.

Currently, the insight of ALTO information on the path between a UE and a connection node (or say Endpoint) does not provide details below IP hops. However the major QoE challenges of wireless network users arise in the access network, that is, in the first hop between the UE and its one or more serving packet data network gateway (PGW). The path of a UE to its serving PGW(s) impacts the path to the content and thus the related QoE. Therefore, it is necessary to inform the UE, which could take the appropriate decisions w.r.t. the utilized access path. The access technology in current ALTO documents is accounted at the content location (last hop) side by distinguishing whether the requested content is located in a fixed or a wireless access network, as described in [draft-ietf-alto-deployment]

This document introduces several protocol extensions to allow ALTO clients to support use cases such as context-based connection selection in cellular networks and calendaring for unattended data. This document refers to other extension proposals posted in the ALTO WG that can support the present use cases as well. Likewise, some of the proposed extensions may serve other ALTO use cases.

2. Use cases

This section presents motivating use cases for contextual ALTO Costs with a focus on conditional RF costs in cellular networks. In these 2 use cases, a terminal UE is located in a LTE network and associated to a "local" ALTO Server(LAOS) that covers this access network, say up to the Packet Data Network (PDN) Gateway PGW and can itself connect to another ALTO Server having a more global view covering up to the whole ISP network. Such a deployment is proposed in section Section 5 of this draft.

2.1. Use Case 1: conditional RF costs in cellular networks

Let's assume a terminal UE located in a cellular network. An ALTO Client (LAOC) associated to the UE queries the local ALTO Server in order to know via which cell it should connect to the network. So in a first place, LAOC will query the connection cost associated to cells C1,... CK.

The present example assumes that the connection cost conveyed by the ALTO Server is a unitless value abstracting a non-real-time metric reflecting traffic conditions, aggregated over time and space. This metric aims at providing guidance to applications. It may integrate abstractions by the network provider, of actual costs impacted by other values such as congestion or available bandwidth are assumed to be not easily available to UEs or applications otherwise. The ALTO Server does not aim at reporting accurate radio conditions that indeed vary in time and space.

Let us call this metric: "ARF cost", standing for Abstracted RF cost.

Our example however includes 2 additional considerations:

- the ARF cost to a cell may be impacted by its load,
- a UE usually transmits a fair amount of "unattended data" (UD).

UD is considered in one of the key features for LTE enhancements in Release 13 and defined in 3GPP TS22.101 as follows: "Unattended Data Traffic : Data traffic of which the user is unaware he/she initiated, e.g. based on the screen/keypad lock being activated, length of time

since the UE last received any input from the user, known type of app (e.g. an application monitoring a user's health "mHealth" may need its data never treated as Unattended Data Traffic.)". UD traffic is often delay tolerant and it would be beneficial for the network if the UE can schedule its transmission. To this end, the UE can use an instant UD Indicator (UDI) sent by the LTE network. The UDI, accepted for LTE Release 13 is a single bit sent to the UE indicating whether UD in a cell is allowed (UDA) or not (UDNA). The status change of a UDI from UDA to UDNA is presumably triggered when the cell load exceeds a given threshold $T(udna)$. The value of $T(udna)$ may change across cells and in time but is not provided to UEs. If the UE had an ALTO calendar for either $T(udna)$ or for the abstracted cell load values, it could appropriately schedule the transmission of its UD, that will have to occur anyway. The UE could combine this calendar with the UDI it receives from the cellular network. The UDI state may change within sub-seconds and impact the data exchange. What is missing in the provided LTE information is:

- knowing whether the UDI threshold relates to downlink or uplink congestion.
- knowing the level of congestion that triggers a change in UDI and how it may evolve in time.

The UE thus can advantageously combine the non-real time ALTO information with the real-time UDI provided by the LTE network. The present draft illustrates how ALTO can fill these gaps with the support of:

- ALTO Cost Calendars,
- the proposed protocol extension providing context-dependent ALTO Cost values.

In this use case: ALTO calendars need to be requested via for the ALTO Filtered Cost Map (FCM) Service, the context parameters impacting the cost values are: "uda" (Unattended Data Allowed), "udna" (Unattended Data Not Allowed), "uplink", "downlink".

2.2. Use case 2: access-aware endpoint selection

In a second use case, an end-system called UEP is located in a LTE network and may connect via several access technologies, e.g. Cellular or WiFi. UEP may also benefit from a given Service Level Agreement SLA-m. Other parameters may characterize the UEP generated traffic.

Currently the insight of ALTO information in the path between a UE and a connection node (or say Endpoint) does not provide details below IP hops. However the major QoE challenges of wireless network users arise in the access network, that is, in the first hop between the UE and its one or more associated packet data network gateway (PGW). The path of a UE to its associated PGW(s) impacts the path to the content and thus the related QoE. Therefore, it is necessary to inform the UE, which could take the appropriate decisions w.r.t. the utilized access path. The access technology in current ALTO proposals is accounted at the content location (last hop) side by distinguishing whether the requested content is located in a fixed or a wireless access network, as described in [draft-ietf-alto-deployments] that states: "For ISPs with mobile network and fixed network, the traffic optimizing problems they focus will be optimizing the mobile traffic, except problems on last hop section."

For Mobile Network Operators (MNO) and their users, being connected via e.g. cellular or trusted Wifi can hugely impact the QoE and routing cost. Sometimes a 4G connection is preferable for users than a poor WiFi connection although potentially more expensive. Sometimes, MNOs have spare data resources or offer them for given SLAs. For both parties, access-aware Endpoint selection for Users is thus beneficial. One way to achieve this is that ALTO provides cost values depending on qualitative contextual parameters such as access technology and the access technology and SLA.

3. Required ALTO extensions

The aforementioned use cases can be supported with a few simple extensions to the ALTO protocol. A number of them have already been discussed in other WG drafts and use cases. The proposed extensions include:

- Cost value context parameters: a capability to allow exposing several possible context-dependent values for one metric, as proposed in the present document,
- Entities with associated domain and properties for cellular and wireless networks, that could be added to [draft-roome-alto-unified-props],
- Cost metrics for cellular and wireless networks: these features would extend current proposals in the WG, that could be added to [draft-ietf-alto-performance-metrics],
- Extended input for the Filtered Cost Map Service: to allow the input to comprise several(source-array, destination-array) pairs, as it has been proposed in [draft-yang-alto-path-vector].

4. Design options and examples

Similarly to Multi-Cost and Cost Calendar ([draft-ietf-alto-cost-calendar]), this proposal does not introduce new cost modes or new media-types. It ensures backwards compatibility with legacy ALTO Clients, that is: "A legacy ALTO Client must be able to send legacy requests to a Cost Context aware ALTO Server and get legacy responses as specified in RFC7285".

"A Cost Context aware ALTO Server must be able to receive and process requests sent by legacy ALTO Clients, as specified in RFC 7285".

Besides, the proposed extension is designed to be compatible with Multi-Cost ALTO and ALTO Cost Calendars ([draft-ietf-alto-cost-calendar]).

In the present draft version, the IRD indicates the supported context attributes as values encoded in JSON strings. This design simplifies the transactions, as it allows a limited number of context attributes or their combinations, say 1 to 5. Context attributes taking numerous or unpredictable values should be handled as values properties or metrics expressed in constraints.

- A cost context aware ALTO Server MUST provide metric values, as specified in RFC 7285, without any context consideration for all the Cost Types indicated in its "meta".

4.1. Overview of context features

Cost context attributes are strings with values such as "wifi", "cellular", "uda".

Cost context attributes are indicated in the IRD as capabilities of an information resource. They are associated to cost type names.

- A cost context aware ALTO may indicate in its IRD capabilities, whether and how context attributes may be combined in ALTO requests.

- A cost context aware ALTO Server MUST return metric values, without any context consideration, as specified in RFC 7285, if the value for a context attribute or a combination of attributes requested by the client is not available.

- A cost context aware ALTO may indicate a maximum number of context attributes or their combinations authorised in context-aware Client requests.

4.1.1. Applicable ALTO services

Draft [draft-bertz-alto-mobilitynets] proposes to identify network points of attachment (PoA) such as cells to PIDs, as PoAs are endpoint types not currently supported in ALTO. The current proposal is to represent cellular PIDs in an ALTO Network Map with no routes. PID properties as specified in [draft-roome-alto-unified-props] could be used to indicate the type of the PoA, together with other properties. ALTO properties are well suited for almost static attributes such as access type.

To abstract and convey connection properties with frequently changing values such as ARF Cost, load or congestion, the ALTO Filtered Cost Map service can be used. Connection properties may also be conveyed with the Endpoint property service or its extensions defined in [draft-roome-alto-unified-props].

Costs and properties with the extensions proposed in this document may be conveyed with different values depending on the context parameter. The present version of this draft focuses on context parameters associated to costs.

4.2. Example IRD

The purpose of ALTO is to guide the behavior of the end systems or applications without the need for networks to explicitly expose their performance values. In this example, the IRD does not expose the real load percentage of a cell to UE. Instead, it abstracts the cell congestion by a metric called 'ARFcost' represented by a number between 0 and 100, where the optimal value is 0. The values of 'ARFcost' are provided as an ALTO Calendar as specified in [draft-ietf-alto-cost-calendar-00] in shorter time intervals. In addition they differ, depending on the context values "uda" and "udna".

Besides, the IRD provides metric 'routingcost' as a MUST specified in [RFC7285], that may represent a more administrative or monetary access cost.

The IRD could publish the capability of a resource to provide context dependent 'routingcost' values as expressed for resource "filtered-cost-calendar-map".

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

```
{
  "meta" : {
    "cost-types": {
      "num-routingcost": {
        "cost-mode" : "numerical",
        "cost-metric" : "routingcost"
      },
      "num-ARFcost": {
        "cost-mode" : "numerical",
        "cost-metric": "ARFcost",
      }
    }
    ... other meta ...
  },
  "resources" : {
    "filtered-cost-calendar-map" : {
      "uri" : "http://alto.local.example.com/costmap/filtered/calendar/conte
xt",
      "media-types" : [ "application/alto-endpointcost+json" ],
      "accepts" : [ "application/alto-endpointcostparams+json" ],
      "capabilities" : {
        "cost-constraints" : true,
        "cost-type-names" : [ "num-routingcost",
                              "num-ARFcost"], // ++NEW
        "calendar-attributes" : [
          {"cost-type-names" : "num-routingcost",
           "time-interval-size" : "1 hour",
           "number-of-intervals" : 24}, // MAY ALSO BE SINGLE VALUE
          {"cost-type-names" : "num-ARFcost", // ++NEW
           "time-interval-size" : "5 minute",
           "number-of-intervals" : 12}
        ],
        "cost-context" : [ // ++NEW
          {"cost-type-names" : "num-ARFcost",
           "context-params" : [{"uda", "udna"},
                               ["uplink", "downlink"]}
          ]
        ], // ++NEW
        "max-context-attributes" : 10,
        "uses": [ "my-default-network-map" ]
      } // end FCM capab
      ... other resources ...
    } // end resources
  } // end IRD
```

4.3. Use case 1: Example scenario for the FCM Service

We assume an example scenario where a UE has the choice to connect to 2 cells C1 and C2.

As suggested in [draft-bertz-alto-mobilitynets], we may represent the cellular topology with an ALTO Network Map comprising PIDs representing the cells and named "Cell1", "Cell2", ... "Celln". A format for a cell identifier has been proposed in [draft-rauschenbach-alto-wireless-access] and is not being discussed here.

As a Network Map may cover a large number of cells, the Filtered Cost Map Service can be used to reduce data exchange and get information on a restricted number of cells.

We assume that the ALTO Client in the UE wants to get calendared values for ALTO metric "ARFcost" in order to appropriately schedule its unattended data transmission. The ALTO information resource 'ALTO Calendar' provides an array of time-dependent cost values and is being specified in [draft-ietf-alto-cost-calendar]. In addition, the ALTO Client wants these values for both the "uda" and "udna" context. Last, we assume that the UE needs the Cost values for both the uplink (UE to Cell-k) and downlink (Cell-k to UE) directions. We assume that the UE is located in the PID called "Cell1".

In this scenario, Cell1 is limited by its uplink capacity and Cell2 is limited by its downlink capacity. ALTO can be used to convey the following information:

At time interval T1 of the next Calendar:

- if Cell1 indicates "unattended data allowed" the downlink ARF cost is 20, and the uplink ARF cost is 70
- if Cell1 indicates "unattended data NOT allowed", the downlink ARF cost is 20, and the uplink ARF cost is 90
- if Cell2 indicates "unattended data allowed" the downlink ARF cost is 70, and the uplink ARF cost is 20
- if Cell2 indicates "unattended data NOT allowed", the downlink ARF cost is 90, in the uplink ARF cost is 20.

The ALTO Calendar provides values for the other 11 time intervals Ti.

4.4. Design option: Network Map with cells as PIDs

In this design, the cellular topology is represented with an ALTO Network Map comprising PIDs named "Cell1", "Cell2", ... "Celln". The UE is located in one of these PIDs. A Cost Map is associated to this Network Map and conveys metrics indicated in the IRD. The Cost Map can be to convey connection costs between firstly the UE to its serving cell (that is the PID to itself) and secondly the UE and neighboring cells (that is the PID to another one) and last, for both uplink and downlink directions.

The ALTO Server can regularly update the Cost Map and send filtered information to the ALTO Client. The proposed IRD design announces additional context attributes "uplink", "downlink". In this case and other potential cases, the context parameters need to be arranged w.r.t. their possible combinations (to be further specified in the IRD). For example, the IRD may announce that costs are provided for contexts "uda" and "udna" and this in both contexts "uplink" and "downlink". Or that costs are provided for contexts "uplink" and "downlink" and this in both contexts "udna" and "uda". In such a case, the IRD capability member may list the possible combinations in the capabilities as follows:

```
"cost-context" : [ // ++NEW
  { "cost-type-names" : "num-ARFcost",
    "context-params" : [["uda", "uplink"],
                        ["uda", "downlink"],
                        ["udna", "uplink"],
                        ["udna", "downlink"]] // ++NEW
  }
]
```

This arrangement indicates that for the metric named "num-ARFcost", the ALTO Server can provide 4 different values: v1 for ["uda" AND "uplink"], ... v4 for ["udna" AND "downlink"].

Further versions of this draft will specify more elaborated logical combinations of context attributes, to moderate the length of the ALTO request and support use cases as described in section 4.5.1.

4.4.1. Example: FCM Request for contextual 'ARFcost'

The ALTO Client can specify the desired cost value context parameters in the request input. In particular, it can select one or more combinations indicated in the IRD. Its input parameter "context-params" is an array of all the desired combinations. In this

example, the ALTO Client wants to know the ALTO connection costs within Cell1 and Cell2. For each cell, the Client wants the 4 values, corresponding to all the combinations indicated below.

```
POST /costmap/filtered/calendar/context HTTP/1.1
Host: alto.example.com
Accept: application/alto-costmap+json,application/alto-error+json
Content-Type: application/alto-costmapfilter+json
Content-Length: ###

{
  "cost-type" : { "cost-mode": "numerical", "cost-metric": "ARFcost"},
  "calendared" : true,
  "context-params" : [[ "uda", "uplink"], // ++NEW
                      [ "uda", "downlink"],
                      [ "udna", "uplink"],
                      [ "udna", "downlink"]],
  "pids" : [
    { "srcs" : [ "Cell1"], "dsts" : [ "Cell1"] },
    { "srcs" : [ "Cell2"], "dsts" : [ "Cell2"] }
  ]
}
```

4.4.2. Example: FCM Response for contextual ARFcost

The ALTO response provides, for each requested ("src", "dest") pair, a calendar of 12 JSON values, where each is an array of cost values arranged as specified in the "meta" of the ALTO response.

```

HTTP/1.1 200 OK
Content-Type: application/alto-costmap+json
Content-Length: ###

{
  "meta" : {
    "dependent-vtags" : [
      { "resource-id": "my-default-network-map",
        "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"
      }
    ],
    "cost-type" : { "cost-mode": "numerical", "cost-metric": "ARFcost" },
    "calendar-response-attributes" :
      { "calendar-start-time" : Tue, 1 Sept 2016 13:00:00 GMT,
        "time-interval-size" : "5 minute",
        "numb-intervals" : 12,
        "context-params" : [ ["uda", "uplink"], // ++NEW
                              ["uda", "downlink"],
                              ["udna", "uplink"],
                              ["udna", "downlink"] ]
      }
  } // end meta
  "cost-map" : {
    "Cell1": { "Cell1": [[70, 20, 90, 20], ... , [50, 20, 70, 20]],
    "Cell2": { "Cell2": [[20, 70, 20, 90], ... , [20, 50, 20, 70]]
  }
}

```

4.5. Use case 2: example ALTO transactions for the ECS

In this use case, the UE requests the ECS to a local ALTO server for the routingcost to the PGW and wants the metric values varying w.r.t. the "access-type" and "SLA-id". Note that the "context" related design feature can be easily transposed for the Cost Map Service.

4.5.1. Use case 2: example with logical context parameter combinations

This section proposes a design, allowing a Client to arrange input context parameters in logical combinations. The purpose is to show how such combinations of context parameters avoids specifying as many metrics and moderates the amount of exchanged data.

In this example the ALTO Server indicates in its IRD that it can provide endpoint cost maps for the example metrics "routingcost" and "bandwidthscore". Values for metric "routingcost" are provided w.r.t. 2 types of context parameters. The ALTO Client may query values for metric "routingcost" for either of these types of parameters or both or none.

For each type, the parameters are listed in an array. We have 2 arrays:

- ["cell", "wifi", "lan"]
- ["SLA-1", "SLA-2", "SLA-3"]

This indicates that in each array, the client can pick one or more parameters and combine them with one or more parameters in the second array. The ALTO Server will provide costs w.r.t. the AND combination across and within arrays.

In the present example, if the Client requests cost values for the combination:

```
[["cell", "wifi"], ["SLA-3"]]
```

The server will provide 2 values: one for ("cell" AND "SLA-3") and the second one for ("wifi" and "SLA-3").

4.5.1.1. Example IRD with logical context parameter combinations

The IRD below specifies the possibility to combine parameters from the two arrays of the example above.

```
"resources" : {
  "filtered-cost-calendar-map" : {
    "uri" : "http://alto.local.example.com/endpointcostmap/lookup/context"
  },
  "media-types" : [ "application/alto-endpointcost+json" ],
  "accepts" : [ "application/alto-endpointcostparams+json" ],
  "capabilities" : {
    "cost-constraints" : true,
    "cost-type-names" : [ "num-routingcost",
                          "num-bandwidthscore" ],
    "cost-context" : [// ++NEW
      { "cost-type-names" : "num-routingcost",
        "context-params" : [ ["cell", "wifi", "lan"],
                             ["SLA-1", "SLA-2", "SLA-3"] ]
      }
    ]
  } // end ECM capab
  ... other resources ...
} // end resources
```

4.5.1.2. Use case 2: example ECS request with logical context parameter combinations

The ALTO Client queries the ECS between 2 endpoints for the following combinations: ("cell" AND "SLA-3") and ("wifi" and "SLA-3") and thus arranges its input context parameters as follows:

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

{
  "cost-type" : {"cost-mode" : "numerical", "cost-metric" : "routingcost"},
  "context-params" : [{"cell", "wifi"}, ["SLA-3"]],
  "endpoints" : {
    "srcs": [ "ipv4:192.0.2.2" ],
    "dsts": [
      "ipv4:192.0.2.89",
      "ipv6:2000::1:2345:6789:abcd"
    ]
  }
}
```

4.5.1.3. Use case 2: example ECS response with logical context parameter combinations

Following the ALTO Client request of the above example, The ALTO Server provides a response as follows:


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

```
{
  "meta" : {
    "cost-type" : {"cost-mode" : "numerical", "cost-metric" : "routingcost"}
  },
  "context-params" : [{"cell", "wifi"}, ["SLA-3"]]
} // end meta

"endpoint-cost-map" : {
  "ipv4:192.0.2.2": {
    "ipv4:192.0.2.89" : [10, 4],
    "ipv6:2000::1:2345:6789:abcd" : [4, 6]
  }
}
```

5. Deployment case: local ALTO Server cascaded with global ALTO Server

To maintain scalability, the ALTO coverage network zone can be decomposed in one "local" ALTO Server part covering a restricted local network zone, for instance within the first IP hop range and another "global" part covering the rest of the ISP network, similarly to what is proposed in [draft-ietf-alto-deployments]. The local ALTO server may include the guidance given by the ISP ALTO server and compose it with the "global" guidance in its replies to its ALTO clients. Recent ALTO WG discussions open the possibility for one IRD to indicate multiple network maps having different levels of detail.

5.1. Cascaded ALTO Servers with one network map each

In the "cascaded" use case, the ALTO Service is preferably distributed among two ALTO Servers as follows:

The ALTO Client serving the UE is referred to as the LAOC and can be located either in the UE or in the network.

1. A Local ALTO Server (LAOS)

- * Hosts the information on the local EPS network, covering the paths between e.g. the UEs and the cells or the PGWs,
- * Hosts an ALTO Client that sends an ALTO request to a "global" ALTO Server, covering the zone beyond the PGW. It can possibly get the global Server updates using the extensions specified in [draft-ietf-alto-incr-update-sse].

- * receives the ALTO request issued by the ALTO Client associated to the UE.

2. a "core" ALTO Server covers the whole ISP network view, as it would if the "local ALTO Service" is not available or deactivated.

6. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

7. Security Considerations

8. Acknowledgements

Many thanks to Dawn Chan, Li Geng, Xin Wan, Yichen Qian for their feedback on this draft.

9. References

9.1. 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>>.
- [RFC7285] Alimi, R., Penno, R., Yang, Y., Kiesel, S., Previdi, S., Roome, W., Shalunov, S., and R. Woundy, "Application-Layer Traffic Optimization (ALTO) Protocol", RFC 7285, September 2014.

9.2. Informative References

- [draft-bertz-alto-mobilitynets] Bertz, L., "Mobility Network Models in ALTO", October 2015.
- [draft-ietf-alto-cost-calendar] Randriamasy, S., Yang, Y., Wu, Q., Deng, L., and N. Schwan, "ALTO Cost Calendar", February 2017.
- [draft-ietf-alto-deployment] Stiernerling, M., Kiesel, S., Scharf, M., Seidel, H., and S. Previdi, "draft-ietf-alto-deployments-16", July 2016.

[draft-ietf-alto-incr-update-sse]

Roome, W. and Y. Yang, "ALTO Incremental Updates Using Server-Sent Events (SSE)", Septembre 2016.

[draft-ietf-alto-performance-metrics]

Wu, Q., Yang, Y., Lee, Y., Dhody, D., and S. Randriamasy, "ALTO Performance Cost Metrics", March 2017.

[draft-rauschenbach-alto-wireless-access]

Rauschenbach, U., "ALTO in wireless access networks", October 2014.

[draft-roome-alto-unified-props]

Roome, W., "Extensible Property Maps for the ALTO Protocol", July 2016.

[draft-yang-alto-path-vector]

Bernstein, G., Gao, K., Lee, Y., Roome, W., Scharf, M., and Y. Yang, "ALTO Extension: Path Vector Cost Mode", July 2016.

Appendix A. An Appendix

Author's Address

Sabine Randriamasy
Nokia Bell Labs
Route de Villejust
Nozay 91460
FRANCE

Email: sabine.randriamasy@nokia-bell-labs.com

ALTO WG
Internet-Draft
Intended status: Standards Track
Expires: January 4, 2018

W. Roome
Nokia Bell Labs
R. Yang
Yale University
July 3, 2017

Extensible Property Maps for the ALTO Protocol
draft-roome-alto-unified-props-new-01

Abstract

This document extends the Application-Layer Traffic Optimization (ALTO) Protocol [RFC7285] by generalizing the concept of "endpoint properties" to other entity domains, and by presenting those properties as maps, similar to the network and cost maps in ALTO.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on January 4, 2018.

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	3
2. Definitions and Concepts	4
2.1. Entity	4
2.2. Domain	4
2.3. Entity Address	4
2.4. Domain Name	5
2.5. Property Name	5
2.6. Property Value	6
2.7. Hierarchy and Inheritance	6
2.8. Relationship to Network Maps	6
3. Entity Domains	7
3.1. Internet Address Domains	7
3.1.1. IPv4 Domain	7
3.1.2. IPv6 Domain	8
3.1.3. Hierarchy and Inheritance of ipv4/ipv6 Domains	8
3.1.4. Relationship to Network Maps	9
3.2. PID Domain	10
3.2.1. Domain Name	10
3.2.2. Domain-Specific Entity Addresses	10
3.2.3. Hierarchy and Inheritance	10
3.2.4. Relationship To Internet Addresses Domains	10
3.3. Internet Address Properties vs. PID Properties	10
3.4. ANE Domain	11
3.4.1. Domain Name	11
3.4.2. Domain-Specific Entity Addresses	11
3.4.3. Hierarchy and Inheritance	11
3.4.4. Relationship to Cost Map	11
4. Property Map Resource	11
4.1. Media Type	11
4.2. HTTP Method	12
4.3. Accept Input Parameters	12
4.4. Capabilities	12
4.5. Uses	12
4.6. Response	12
5. Filtered Property Map Resource	13
5.1. Media Type	14
5.2. HTTP Method	14
5.3. Accept Input Parameters	14
5.4. Capabilities	14

5.5. Uses	15
5.6. Response	15
6. Impact on Legacy ALTO Servers and ALTO Clients	15
6.1. Impact on Endpoint Property Service	15
6.2. Impact on Resource-Specific Properties	15
6.3. Impact on the "pid" Property	16
6.4. Impact on Other Properties	16
7. Examples	16
7.1. Network Map	16
7.2. Property Definitions	17
7.3. Information Resource Directory (IRD)	17
7.4. Property Map Example	19
7.5. Filtered Property Map Example #1	19
7.6. Filtered Property Map Example #2	20
7.7. Filtered Property Map Example #3	21
7.8. Filtered Property Map Example #4	22
8. Security Considerations	23
9. IANA Considerations	24
9.1. application/alto-* Media Types	24
9.2. ALTO Entity Domain Registry	25
9.3. ALTO Endpoint Property Type Registry	26
10. References	26
Authors' Addresses	27

1. Introduction

The ALTO protocol [RFC7285] introduced the concept of "properties" attached to "endpoint addresses," and defined the Endpoint Property Service (EPS) to allow clients to retrieve those properties. While useful, the EPS, as defined in RFC7285, has at least two limitations.

First, it only allows properties to be associated with a particular domain of entities, namely individual IP addresses. It is reasonable to think that collections of endpoints, as defined by CIDRs ([RFC4632]) or PIDs, may also have properties. Furthermore, recent proposal ([I-D.ietf-alto-path-vector]) have suggested new classes of entities (ANE) with properties. The EPS cannot be extended to new entity domains. Instead, new services, with new request and response messages, would have to be defined for each new entity domain.

Second, the EPS is only defined as a POST-mode service. Clients must request the properties for an explicit set of addresses. By contrast, [RFC7285] defines a GET-mode Cost Map resource which returns all available costs, so a client can get the full set of costs once, and then lookup costs without querying the ALTO server. RFC7285 does not define an equivalent service for endpoint properties. And it is unlikely a property will be defined for every possible address. It is very likely that properties will only be

defined for a subset of addresses, and that subset would be small enough to enumerate. This is particularly true if blocks of addresses with a common prefix (e.g., a CIDR) have the same value for a property. Furthermore, entities in other domains may very well be enumerable.

This document proposes a new approach to retrieve ALTO properties. Specifically, it defines two new resource types, namely Property Maps (see Section 4) and Filtered Property Maps (see Section 5). The former are GET-mode resources which return the property values for all entities in a domain, and are analogous to the ALTO's Network Maps and Cost Maps. The latter are POST-mode resources which return the values for a set of properties and entities requested by the client, and are analogous to ALTO's Filtered Network Maps and Filtered Cost Maps.

Entity domains and property names are extensible, so that new domains can be defined without revising the messages defined in this document, in the same way that new cost metrics and new endpoint properties can be defined without revising the messages defined by the ALTO protocol.

This proposal would subsume the Endpoint Property Service defined in RFC7285, although that service may be retained for legacy clients (see Section 6).

2. Definitions and Concepts

2.1. Entity

An entity is an object with a (possibly empty) set of properties. Every entity is in a domain, such as the IPv4 and IPv6 domains, and has a unique address.

2.2. Domain

A domain is a family of entities. Two examples are the Internet address and PID domain (see Section 3.1 and Section 3.2) that this document will define. An additional example is the proposed domain of Abstract Network Elements associated with topology and routing, as suggested by [I-D.ietf-alto-path-vector].

2.3. Entity Address

Each entity has a unique address of the format:

domain-name : domain-specific-entity-address

Examples from the IP domain include individual addresses such as "ipv4:192.0.2.14" and "ipv6:2001:db8::12", as well as address blocks such as "ipv4:192.0.2.0/26" and "ipv6:2001:db8::1/48".

The type `EntityAddr` denotes a JSON string with an entity address in this format.

The format of the second part of an entity address depends on the domain, and **MUST** be specified when registering a new domain. Addresses **MAY** be hierarchical, and properties **MAY** be inherited based on that hierarchy. Again, the rules defining any hierarchy or inheritance **MUST** be defined when the domain is registered.

Note that entity addresses **MAY** have different textual representations, for a given domain. For example, the strings "ipv6:2001:db8::1" and "ipv6:2001:db8:0:0:0:0:1" refer to the same entity.

2.4. Domain Name

Each domain has a unique name. A domain name **MUST** be no more than 32 characters, and **MUST NOT** contain characters other than US-ASCII alphanumeric characters (U+0030-U+0039, U+0041-U+005A, and U+0061-U+007A), hyphen ('-', U+002D), and low line ('_', U+005F). For example, the names "ipv4" and "ipv6" identify objects in the Internet address domain (Section 3.1).

The type `DomainName` denotes a JSON string with a domain name in this format.

Domain names **MUST** be registered with the IANA, and the format of the entity addresses in that domain, as well as any hierarchical or inheritance rules for those entities, **MUST** be specified at the same time.

2.5. Property Name

The space of property names associated with entities defined by this document is the same as, and is shared with, the endpoint property names defined by [RFC7285]. Thus entity property names are as defined in Section 10.8.2 of that document, and **MUST** be registered with the "ALTO Endpoint Property Type Registry" defined in Section 14.3 of that document.

The type `PropertyName` denotes a JSON string with a property name in this format.

A main design decision is that property names are defined in a single namespace, not specific to a domain, although some properties MAY only be applicable for particular domains. This design decision is to enforce a design that similar properties are named similarly.

The interpretation of the value of a property, however, MAY depend on the domain. For example, suppose the "geo-location" property is defined as the coordinates of a point, encoded as "latitude longitude [altitude]." When applied to an entity that represents a specific host computer, such as an Internet address, the property defines the host's location. When applied to an entity that represents a set of computers, such as a CIDR, the property would be the location of the center of that set. If it is necessary to represent the bounding box of a set of hosts, another property, such as "geo-region", SHOULD be defined.

2.6. Property Value

The property value MAY BE defined or undefined. If it is defined, it SHOULD BE a JSONString or a JSON "null" value. Otherwise, a protocol implementation SHOULD fail to parse the property value, unless the implementation is using an extension to this document that indicates when and how property values of other data types are signaled.

2.7. Hierarchy and Inheritance

Entities in a given domain MAY form hierarchy based on entity address. Each domain MAY define its own hierarchy and inheritance semantics. Given a property, the semantics MUST NOT allow an entity with a defined property value to inherit the property value of another entity. Given a property, the semantics MAY allow an entity with an undefined property value to inherit the property value of another entity. An entity may not be able to inherit a property value if no other entities satisfy the inheritance conditions defined by the semantics. The hierarchy and inheritance semantics SHOULD be defined carefully to avoid multiple inheritance.

2.8. Relationship to Network Maps

[RFC7285] recognizes that some properties MAY be specific to an ALTO resource, such as a network map. Accordingly [RFC7285] defines the concept of "resource-specific endpoint properties" (Section 10.8.1), and indicates that dependency by prefixing the property name with the ID of the resource on which it depends. That document defines one resource-specific property, namely the "pid" property, whose value is the name of the PID containing that endpoint in the associated network map.

This document takes a different approach. Instead of defining the dependency by qualifying the property name, this document attaches the dependency to the property map as a whole. Thus all properties in a given property map depend on the same resource. Furthermore, entity addresses MAY depend on a network map (for example, the Abstract Network Elements suggested by [I-D.ietf-alto-path-vector]). Associating the dependency with the property map handles any entity address dependencies as well.

The "uses" field in an IRD entry defines the dependencies of a property map resource, and the "dependent-vtags" field in a property map response defines the dependencies of that map. These fields are defined in Sections 9.1.5 and 11.1 of [RFC7285], respectively.

This is similar to how RFC7285 handles dependencies between cost maps and network maps. Recall that cost maps present the costs between PIDs, and PID names depend on a network map. If an ALTO server provides the "routingcost" metric for the network maps "net1" and "net2", then the server defines two separate cost maps, one for "net1" and the other for "net2".

According to [RFC7285], a legacy ALTO server with two network maps, with resource IDs "net1" and "net2", could offer a single Endpoint Property Service for the two properties "net1.pid" and "net2.pid". An ALTO server which supports the extensions defined in this document, would, instead, offer two different Property Maps for the "pid" property, one depending on "net1", the other on "net2".

3. Entity Domains

This document defines the following entity domains. For the definition of each domain, it includes the following template: domain name, domain-specific addresses, and hierarchy and inheritance semantics.

3.1. Internet Address Domains

The document defines two domains (IPv4 and IPv6) for Internet addresses. Both domains include individual addresses and blocks of addresses.

3.1.1. IPv4 Domain

3.1.1.1. Domain Name

ipv4

3.1.1.2. Domain-Specific Entity Addresses

Individual addresses are strings as specified by the IPv4Addresses rule of Section 3.2.2 of [RFC3986]. Blocks of addresses are prefix-match strings as specified in Section 3.1 of [RFC4632]. For the purpose of defining properties, an individual Internet address and the corresponding full-length prefix are considered aliases for the same entity. Thus "ipv4:192.0.2.0" and "ipv4:192.0.2.0/32" are equivalent.

3.1.2. IPv6 Domain

3.1.2.1. Domain Name

ipv6

3.1.2.2. Domain-Specific Entity Addresses

Individual addresses are strings as specified by Section 4 of [RFC5952]. Blocks of addresses are prefix-match strings as specified in Section 7 of [RFC5952]. For the purpose of defining properties, an individual Internet address and the corresponding 128-bit prefix are considered aliases for the same entity. That is, "ipv6:2001:db8::1" and "ipv6:2001:db8::1/128" are equivalent, and have the same set of properties.

3.1.3. Hierarchy and Inheritance of ipv4/ipv6 Domains

Both domains allow property values to be inherited. Specifically, if a property P is not defined for a specific Internet address IP, but P is defined for some block C which prefix-matches IP, then the address IP inherits the value of P defined for block C. If more than one such block defines a value for P, IP inherits the value of P in the block with the longest prefix. It is important to notice that this longest prefix rule will ensure no multiple inheritance, and hence no ambiguity.

Address blocks can also inherit properties: if property P is not defined for a block C, but is defined for some block C' prefix-matches C, and C' has a shorter mask than C, then block C inherits the property from C'. If there are several such blocks C', C inherits from the block with the longest prefix.

```
ipv4:192.0.2.0/26: P=v1
ipv4:192.0.2.0/28: P=v2
ipv4:192.0.2.0/30: P=v3
ipv4:192.0.2.0:    P=v4
```

Figure 1: Defined Property Values.

Then the following entities have the indicated values:

```
ipv4:192.0.2.0:    P=v4
ipv4:192.0.2.1:    P=v3
ipv4:192.0.2.16:   P=v1
ipv4:192.0.2.32:   P=v1
ipv4:192.0.2.64:   (not defined)
ipv4:192.0.2.0/32: P=v4
ipv4:192.0.2.0/31: P=v3
ipv4:192.0.2.0/29: P=v2
ipv4:192.0.2.0/27: P=v1
ipv4:192.0.2.0/25: (not defined)
```

Figure 2: Inherited Property Values.

An ALTO Server MAY explicitly define a property as not having a value for a particular entity. That is, a server MAY say that a property is "defined to have no value", as opposed to the property being "undefined". If that entity would inherit a value for that property, then the ALTO server MUST return a "null" value for that property, and an ALTO client MUST recognize a "null" value means "do not apply the inheritance rules for this property." If the entity would not inherit a value, the ALTO server MAY return "null" or MAY just omit the property. TODO: Discuss more.

If the ALTO Server does not define any properties for an entity, then the server MAY omit that entity from the response.

3.1.4. Relationship to Network Maps

TODO: Need discussion. An Internet address domain MAY be associated with an ALTO network map resource. Logically, there is a map of Internet address entities to property values for each network map defined by the ALTO server, plus an additional property map for Internet address entities which are not associated with a network map. These maps are separate from each other. The prefixes in the property map do not have to correspond to the prefixes defining the network map's PIDs. For example, the property map for a network map MAY assign properties to "ipv4:192.0.2.0/24" even if that prefix is not associated with any PID in the network map.

3.2. PID Domain

The PID domain associates property values with the PIDs in a network map. Accordingly, this domain always depends on a network map.

3.2.1. Domain Name

pid

3.2.2. Domain-Specific Entity Addresses

The entity addresses are the PID names of the associated network map.

3.2.3. Hierarchy and Inheritance

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

3.2.4. Relationship To Internet Addresses Domains

The PID domain and the Internet address domains are completely independent; the properties associated with a PID have no relation to the properties associated with the prefixes or endpoint addresses in that PID. An ALTO server MAY choose to assign some or all properties of a PID to the prefixes in that PID.

For example, suppose "PID1" consists of the prefix "ipv4:192.0.2.0/24", and has the property "P" with value "v1". The Internet address entities "ipv4:192.0.2.0" and "ipv4:192.0.2.0/24", in the IPv4 domain MAY have a value for the property "P", and if they do, it is not necessarily "v1".

3.3. Internet Address Properties vs. PID Properties

Because the Internet address and PID domains are completely separate, the question may arise as to which domain is best for a property. In general, the Internet address domain is RECOMMENDED for properties that are closely related to the Internet address, or are associated with, and inherited through, blocks of addresses.

The PID domain is RECOMMENDED for properties that arise from the definition of the PID, rather than from the Internet address prefixes in that PID.

For example, because Internet addresses are allocated to service providers by blocks of prefixes, an "ISP" property would be best associated with the Internet address domain. On the other hand, a

property that explains why a PID was formed, or how it relates the a provider's network, would best be associated with the PID domain.

3.4. ANE Domain

3.4.1. Domain Name

ane

3.4.2. 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.

3.4.3. Hierarchy and Inheritance

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

3.4.4. Relationship to Cost Map

TBA

4. Property Map Resource

A Property Map returns the properties defined for all entities in one or more domains. Note that Property Map Resource is not applicable to ANE domain.

Section 7.4 gives an example of a property map request and its response.

4.1. Media Type

The media type of an ALTO Property Map resource is "application/alto-propmap+json".

4.2. HTTP Method

An ALTO Property Map resource is requested using the HTTP GET method.

4.3. Accept Input Parameters

None.

4.4. Capabilities

The capabilities are defined by an object of type `PropertyMapCapabilities`:

```
object {  
  DomainName domain-types<1..*>;  
  PropertyName prop-types<1..*>;  
} PropertyMapCapabilities;
```

where "domain-types" is an array with the domains of the entities in this property map, and "prop-types" is an array with the names of the properties returned for entities in those domains. TODO: discuss semantics and requirements of multiple domains.

4.5. Uses

An array with the resource ID(s) of resource(s) with which the domains in this map are associated. In most cases, this array will have at most one ID, for example, for a network map resource. TODO: discuss semantics and requirements of multiple resources.

4.6. Response

If the domains in this property map depend on other resources, the "dependent-vtags" field in the "meta" field of the response MUST be an array that includes the version tags of those resources. The data component of a Property Map response is named "property-map", which is a JSON object of type `PropertyMapData`, where:

```
object {  
  PropertyMapData property-map;  
} InfoResourceProperties : ResponseEntityBase;  
  
object-map {  
  EntityAddr -> EntityProps;  
} PropertyMapData;  
  
object {  
  PropertyName -> JSONValue;  
} EntityProps;
```

The ResponseEntityBase type is defined in Section 8.4 of [RFC7285].

Specifically, a PropertyMapData object has one member for each entity in the Property Map. The entity's properties are encoded in the corresponding EntityProps object. EntityProps encodes one name/value pair for each property, where the property names are encoded as strings of type PropertyName. A protocol implementation SHOULD assume that the property value is either a JSONString or a JSON "null" value, and fail to parse if it is not, unless the implementation is using an extension to this document that indicates when and how property values of other data types are signaled.

An ALTO Server MAY explicitly define a property as not having a value for a particular entity. That is, a server MAY say that a property is "defined to have no value", as opposed to the property being "undefined". If that entity would inherit a value for that property, then the ALTO server MUST return a "null" value for that property, and an ALTO client MUST recognize a "null" value means "do not apply the inheritance rules for this property." If the entity would not inherit a value, the ALTO server MAY return "null" or MAY just omit the property.

For each entity in the Property Map, the ALTO Server returns the value defined for each of the properties specified in this resource's "capabilities" list. For efficiency, the ALTO Server SHOULD omit property values that are inherited rather than explicitly defined; if a client needs inherited values, the client SHOULD use the domain's inheritance rules to deduce those values.

5. Filtered Property Map Resource

A Filtered Property Map returns the values of a set of properties for a set of entities selected by the client.

Section 7.5, Section 7.6 and Section 7.7 give examples of filtered property map requests and responses.

5.1. Media Type

The media type of an ALTO Property Map resource is "application/alto-propmap+json".

5.2. HTTP Method

An ALTO Filtered Property Map resource is requested using the HTTP POST method.

5.3. Accept Input Parameters

The input parameters for a Filtered Property Map request are supplied in the entity body of the POST request. This document specifies the input parameters with a data format indicated by the media type "application/alto-propmapparams+json", which is a JSON object of type ReqFilteredPropertyMap:

```
object {  
  EntityAddr      entities<1..*>;  
  PropertyName    properties<1..*>;  
} ReqFilteredPropertyMap;
```

with fields:

entities: List of entity addresses for which the specified properties are to be returned. The ALTO server MUST interpret entries appearing multiple times as if they appeared only once. The domain of each entity MUST be included in the list of domains in this resource's "capabilities" field (Section 5.4).

properties: List of properties to be returned for each entity. Each specified property MUST be included in the list of properties in this resource's "capabilities" field (Section 5.4). The ALTO server MUST interpret entries appearing multiple times as if they appeared only once.

Note that the "entities" and "properties" fields MUST have at least one entry each.

5.4. Capabilities

The capabilities are defined by an object of type PropertyMapCapabilities, as defined in Section 4.4.

5.5. Uses

An array with the resource ID(s) of resource(s) with which the domains in this map are associated. In most cases, this array will have at most one ID, and it will be for a network map resource.

5.6. Response

The response is the same as for the property map (Section 4.6), except that it only includes the entities and properties requested by the client.

Also, the Filtered Property Map response MUST include all inherited property values for the specified entities (unlike the Full Property Map, the Filtered Property Map response does not include enough information for the client to calculate the inherited values).

Discussion Needed: sometimes the client can compute some inherited property values. In this case, can the Filter Property Map response only contain the uncomputable inherited property values instead of all of them?

6. Impact on Legacy ALTO Servers and ALTO Clients

6.1. Impact on Endpoint Property Service

The Property Maps defined in this document provide the same functionality as the Endpoint Property Service (EPS) defined in Section 11.4 of [RFC7285]. Accordingly, it is RECOMMENDED that the EPS be deprecated in favor of Property Maps. However, ALTO servers MAY provide an EPS for the benefit of legacy clients.

6.2. Impact on Resource-Specific Properties

Section 10.8 of [RFC7285] defines two categories of endpoint properties: "resource-specific" and "global". Resource-specific property names are prefixed with the ID of the resource they depended upon, while global property names have no such prefix. The property map resources defined in this document do not distinguish between those two types of properties. Instead, if there is a dependency, it is indicated by the "uses" capability of a property map, and is shared by all properties and entity domains in that map. Accordingly, it is RECOMMENDED that resource-specific endpoint properties be deprecated, and no new resource-specific endpoint properties be defined.

6.3. Impact on the "pid" Property

Section 7.1.1 of [RFC7285] defines the resource-specific endpoint property "pid", whose value is the name of the PID containing that endpoint. For compatibility with legacy clients, an ALTO server which provides the "pid" property via the Endpoint Property Service MUST use that definition, and that syntax, in the EPS resource.

However, when used with Property Maps, this document amends the definition of the "pid" property as follows.

First, the name of the property is simply "pid"; the name is not prefixed with the resource ID of a network map. The "uses" capability of the property map resource indicates the associated network map. This implies that a property map can only return the "pid" property for one network map; if an ALTO server provides several network maps, it MUST provide a property map resource for each one.

Second, a client MAY request the "pid" property for a block of addresses. An ALTO server determines the value of "pid" for an address block C as follows. Let CS be the set of all address blocks in the network map. If C is in CS, then the value of "pid" is the name of the PID associated with C. Otherwise, find the longest block C' in CS such that C' prefix-matches C, but is shorter than C. If there is such a block C', the value of "pid" is the name of the PID associated with C'. If not, then "pid" has no value for block C.

Note that although an ALTO server MAY provide a GET-mode property map resource which returns the entire map for the "pid" property, there is no need to do so, because that map is simply the inverse of the network map.

6.4. Impact on Other Properties

In general, there should be little or no impact on other previously defined properties. The only consideration is that properties can now be defined on blocks of addresses, rather than just individual addresses, which might change the semantics of a property.

7. Examples

7.1. Network Map

The examples in this section use a very simple default network map:

```

defaultpid:  ipv4:0.0.0.0/0  ipv6:::0/0
pid1:         ipv4:192.0.2.0/25
pid2:         ipv4:192.0.2.0/28  ipv4:192.0.2.16/28

```

Figure 3: Example Network Map

7.2. Property Definitions

The examples in this section use four additional properties, "ISP", "ASN", "country" and "state", with the following values:

	ISP	ASN	country	state
ipv4:192.0.2.0/24:	BitsRus	-	us	-
ipv4:192.0.2.0/28:	-	12345	-	NJ
ipv4:192.0.2.16/28:	-	12345	-	CT
ipv4:192.0.2.0:	-	-	-	PA

Figure 4: Example Property Values

7.3. Information Resource Directory (IRD)

The following IRD defines the relevant resources of the ALTO server. It provides two Property Map resources, one for the "ISP" and "ASN" properties, and another for the "country" and "state" properties. The server could have provided a Property Map resource for all four properties, but did not, presumably because the organization that runs the ALTO server believes any given client is not interested in all four properties.

The server provides two Filtered Property Maps. The first returns all four properties, and the second just returns the "pid" property for the default network map.

The Filtered Property Maps for the "ISP", "ASN", "country" and "state" properties do not depend on the default network map (it does not have a "uses" capability), because the definitions of those properties do not depend on the default network map. The Filtered Property Map for the "pid" property does have a "uses" capability for the default network map, because that defines the values of the "pid" property.

Note that for legacy clients, the ALTO server provides an Endpoint Property Service for the "pid" property for the default network map.

```

"meta": { ... },
"resources" : {
  "default-network-map" : {
    "uri" : "http://alto.example.com/networkmap",

```

```
    "media-type" : "application/alto-networkmap+json"
  },
  .... property map resources ....
  "country-state-property-map" : {
    "uri" : "http://alto.example.com/propmap/full/inet-cs",
    "media-type" : "application/alto-propmap+json",
    "capabilities" : {
      "domain-types": [ "ipv4", "ipv6" ],
      "prop-types" : [ "country", "state" ]
    }
  },
  "isp-asn-property-map" : {
    "uri" : "http://alto.example.com/propmap/full/inet-ia",
    "media-type" : "application/alto-propmap+json",
    "capabilities" : {
      "domain-types": [ "ipv4", "ipv6" ],
      "prop-types" : [ "ISP", "ASN" ]
    }
  },
  "iacs-property-map" : {
    "uri" : "http://alto.example.com/propmap/lookup/inet-iacs",
    "media-type" : "application/alto-propmap+json",
    "accepts" : "application/alto-propmapparams+json",
    "capabilities" : {
      "domain-types": [ "ipv4", "ipv6" ],
      "prop-types" : [ "ISP", "ASN", "country", "state" ]
    }
  },
  "pid-property-map" : {
    "uri" : "http://alto.example.com/propmap/lookup/pid",
    "media-type" : "application/alto-propmap+json",
    "accepts" : "application/alto-propmapparams+json",
    "uses" : [ "default-network-map" ]
    "capabilities" : {
      "domain-types" : [ "ipv4", "ipv6" ],
      "prop-types" : [ "pid" ]
    }
  },
  "availbw-property-map" : {
    "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", "delay" ]
    }
  },
  "legacy-pid-property-map" : {
```

```
    "uri" : "http://alto.example.com/legacy/eps-pid",
    "media-type" : "application/alto-endpointprop+json",
    "accepts" : "application/alto-endpointpropparams+json",
    "capabilities" : {
      "prop-types" : [ "default-network-map.pid" ]
    }
  }
}
```

Figure 5: Example IRD

7.4. Property Map Example

The following example uses the properties and IRD defined above to retrieve a property map for entities with the "ISP" and "ASN" properties. Note that the response does not include the entity "ipv4:192.0.2.0", because it does not have a value for either of those properties. Also note that the entities "ipv4:192.0.2.0/28" and "ipv4:192.0.2.16/28" are refinements of "ipv4:192.0.2.0/24", and hence inherit its value for "ISP" property. But because that value is inherited, it is not

```
GET /propmap/full/inet-ia HTTP/1.1
Host: alto.example.com
Accept: application/alto-propmap+json,application/alto-error+json
```

```
HTTP/1.1 200 OK
Content-Length: ###
Content-Type: application/alto-propmap+json
```

```
{
  "property-map": {
    "ipv4:192.0.2.0/24": { "ISP": "BitsRus" },
    "ipv4:192.0.2.0/28": { "ASN": "12345" },
    "ipv4:192.0.2.16/28": { "ASN": "12345" }
  }
}
```

7.5. Filtered Property Map Example #1

The following example uses the Filtered Property Map resource to request the "ISP", "ASN" and "state" properties for several IPv4 addresses. Note that the value of "state" for "ipv4:192.0.2.0" is the only explicitly defined property; the other values are all derived by the inheritance rules for Internet address entities.

```
POST /propmap/lookup/inet-iacs HTTP/1.1
Host: alto.example.com
Accept: application/alto-propmap+json,application/alto-error+json
Content-Length: ###
Content-Type: application/alto-propmapparams+json
```

```
{
  "entities" : [ "ipv4:192.0.2.0",
                 "ipv4:192.0.2.1",
                 "ipv4:192.0.2.17" ],
  "properties" : [ "ISP", "ASN", "state" ]
}
```

```
HTTP/1.1 200 OK
Content-Length: ###
Content-Type: application/alto-propmap+json
```

```
{
  "property-map": {
    "ipv4:192.0.2.0":
      { "ISP": "BitsRus", "ASN": "12345", "state": "PA" },
    "ipv4:192.0.2.1":
      { "ISP": "BitsRus", "ASN": "12345", "state": "NJ" },
    "ipv4:192.0.2.17":
      { "ISP": "BitsRus", "ASN": "12345", "state": "CT" }
  }
}
```

7.6. Filtered Property Map Example #2

The following example uses the Filtered Property Map resource to request the "ASN", "country" and "state" properties for several IPv4 prefixes. Note that none of the returned property values were explicitly defined; all values are derived by the inheritance rules for Internet address entities.

Also note the "ASN" property has the value "12345" for both the blocks "ipv4:192.0.2.0/28" and "ipv4:192.0.2.16/28", so every address in the block "ipv4:192.0.2.0/27" has that property value. However the block "ipv4:192.0.2.0/27" itself does not have a value for "ASN": address blocks cannot inherit properties from blocks with longer prefixes, even if every such block has the same value.

```
POST /propmap/lookup/inet-iacs HTTP/1.1
Host: alto.example.com
Accept: application/alto-propmap+json,application/alto-error+json
Content-Length: ###
Content-Type: application/alto-propmapparams+json
```

```
{
  "entities" : [ "ipv4:192.0.2.0/26",
                 "ipv4:192.0.2.0/27",
                 "ipv4:192.0.2.0/28" ],
  "properties" : [ "ASN", "country", "state" ]
}
```

```
HTTP/1.1 200 OK
Content-Length: ###
Content-Type: application/alto-propmap+json
```

```
{
  "property-map": {
    "ipv4:192.0.2.0/26": { "country": "us" },
    "ipv4:192.0.2.0/27": { "country": "us" },
    "ipv4:192.0.2.0/28": { "ASN": "12345",
                           "country": "us",
                           "state": "NJ" }
  }
}
```

7.7. Filtered Property Map Example #3

The following example uses the Filtered Property Map resource to request the "pid" property for several IPv4 addresses and prefixes.

Note that the value of "pid" for the prefix "ipv4:192.0.2.0/26" is "pid1", even though all addresses in that block are in "pid2", because "ipv4:192.0.2.0/25" is the longest prefix in the network map which prefix-matches "ipv4:192.0.2.0/26", and that prefix is in "pid1".


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

```
{
  "entities" : [
    "ipv4:192.0.2.0",
    "ipv4:192.0.2.16",
    "ipv4:192.0.2.64",
    "ipv4:192.0.2.128",
    "ipv4:192.0.2.0/26",
    "ipv4:192.0.2.0/30" ],
  "properties" : [ "pid" ]
}
```

```
HTTP/1.1 200 OK
Content-Length: ###
Content-Type: application/alto-propmap+json
```

```
{
  "meta" : {
    "dependent-vtags" : [
      { "resource-id": "default-network-map",
        "tag": "7915dc0290c2705481c491a2b4ffbec482b3cf62" }
    ]
  },
  "property-map": {
    "ipv4:192.0.2.0":      { "pid": "pid2" },
    "ipv4:192.0.2.16":    { "pid": "pid2" },
    "ipv4:192.0.2.64":    { "pid": "pid1" },
    "ipv4:192.0.2.128":   { "pid": "defaultpid" },
    "ipv4:192.0.2.0/26":  { "pid": "pid1" },
    "ipv4:192.0.2.0/30":  { "pid": "pid2" }
  }
}
```

7.8. Filtered Property Map Example #4

The following example uses the Filtered Property Map resource to request the "availbw" property for several abstract network elements.

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

```
{
  "entities" : [
    "ane:L001",
    "ane:Lae0",
    "ane:L3eb" ],
  "properties" : [ "availbw" ]
}
```

```
HTTP/1.1 200 OK
Content-Length: ###
Content-Type: application/alto-propmap+json
```

```
{
  "property-map": {
    "ane:L001": { "availbw": "55" },
    "ane:Lae0": { "availbw": "70" },
    "ane:L3eb": { "availbw": "40" }
  }
}
```

8. Security Considerations

As discussed in Section 15 of [RFC7285], properties MAY have sensitive customer-specific information. If this is the case, an ALTO Server MAY limit access to those properties by providing several different Property Maps. For non-sensitive properties, the ALTO Server would provide a URI which accepts requests from any client. Sensitive properties, on the other hand, would only be available via a secure URI which would require client authentication.

Also, while technically this document does not introduce any security risks not inherent in the Endpoint Property Service defined by [RFC7285], the GET-mode property map resource defined in this document does make it easier for a client to download large numbers of property values. Accordingly, an ALTO Server SHOULD limit GET-mode Property Maps to properties which do not contain sensitive data.

9. IANA Considerations

This document defines additional application/alto-* media types, and extends the ALTO endpoint property registry.

9.1. application/alto-* Media Types

This document registers two additional ALTO media types, listed in Table 1.

Type	Subtype	Specification
application	alto-propmap+json	Section 4.1
application	alto-propmapparams+json	Section 5.3

Table 1: Additional ALTO Media Types.

Type name: application

Subtype name: This document registers multiple subtypes, as listed in Table 1.

Required parameters: n/a

Optional parameters: n/a

Encoding considerations: Encoding considerations are identical to those specified for the "application/json" media type. See [RFC7159].

Security considerations: Security considerations related to the generation and consumption of ALTO Protocol messages are discussed in Section 15 of [RFC7285].

Interoperability considerations: This document specifies formats of conforming messages and the interpretation thereof.

Published specification: This document is the specification for these media types; see Table 1 for the section documenting each media type.

Applications that use this media type: ALTO servers and ALTO clients either stand alone or are embedded within other applications.

Additional information: ~ Magic number(s): ~ n/a File extension(s): ~ This document uses the mime type to refer to protocol messages and

thus does not require a file extension. Macintosh file type code(s):
~ n/a

Person & email address to contact for further information: See
Authors' Addresses section.

Intended usage: COMMON

Restrictions on usage: n/a

Author: See Authors' Addresses section.

Change controller: Internet Engineering Task Force
(mailto:iesg@ietf.org).

9.2. ALTO Entity Domain Registry

This document requests IANA to create and maintain the "ALTO Entity Domain Registry", listed in Table 2.

Identifier	Entity Address Encoding	Hierarchy & Inheritance
ipv4 ipv6 pid ane	See Section 3.1.1 See Section 3.1.2 See Section 3.2 See Section 3.4	See Section 3.1.3 See Section 3.1.3 None None

Table 2: ALTO Entity Domain Names.

This registry serves two purposes. First, it ensures uniqueness of identifiers referring to ALTO entity domains. Second, it states the requirements for allocated domain names.

New ALTO entity domains are assigned after IETF Review [RFC5226] to ensure that proper documentation regarding the new ALTO entity domains and their security considerations has been provided. RFCs defining new entity domains SHOULD indicate how an entity in a registered domain is encoded as an EntityName, and, if applicable, the rules defining the entity hierarchy and property inheritance. Updates and deletions of ALTO entity domains follow the same procedure.

Registered ALTO entity domain identifiers MUST conform to the syntactical requirements specified in Section 2.4. Identifiers are to be recorded and displayed as strings.

Requests to add a new value to the registry MUST include the following information:

- o Identifier: The name of the desired ALTO entity domain.
- o Entity Address Encoding: The procedure for encoding the address of an entity of the registered type as an EntityAddr (see Section 2.3).
- o Hierarchy: If the entities form a hierarchy, the procedure for determining that hierarchy.
- o Inheritance: If entities can inherit property values from other entities, the procedure for determining that inheritance.
- o Security Considerations: In some usage scenarios, entity addresses carried in ALTO Protocol messages MAY reveal information about an ALTO client or an ALTO service provider. Applications and ALTO service providers using addresses of the registered type SHOULD be made aware of how (or if) the addressing scheme relates to private information and network proximity.

This specification requests registration of the identifiers "ipv4", "ipv6" and "pid", as shown in Table 2.

9.3. ALTO Endpoint Property Type Registry

The ALTO Endpoint Property Type Registry was created by [RFC7285]. If possible, the name of that registry SHOULD be changed to "ALTO Entity Property Type Registry", to indicate that it is not restricted to Endpoint Properties. If it is not feasible to change the name, the description MUST be amended to indicate that it registers properties in all domains, rather than just the Internet address domain.

10. References

- [I-D.ietf-alto-path-vector]
Bernstein, G., Chen, S., Gao, K., Lee, Y., Roome, W., Scharf, M., Yang, Y., and J. Zhang, "ALTO Extension: Path Vector Cost Mode", draft-ietf-alto-path-vector-00 (work in progress), May 2017.
- [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>>.

- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <<http://www.rfc-editor.org/info/rfc3986>>.
- [RFC4632] Fuller, V. and T. Li, "Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan", BCP 122, RFC 4632, DOI 10.17487/RFC4632, August 2006, <<http://www.rfc-editor.org/info/rfc4632>>.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <<http://www.rfc-editor.org/info/rfc5226>>.
- [RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, DOI 10.17487/RFC5952, August 2010, <<http://www.rfc-editor.org/info/rfc5952>>.
- [RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <<http://www.rfc-editor.org/info/rfc7159>>.
- [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

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

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

Y. Yang
Yale University
51 Prospect Street
New Haven, CT 06511
USA

Phone: +1-203-432-6400
Email: yry@cs.yale.edu

CDNI
Internet-Draft
Intended status: Informational
Expires: September 6, 2015

J. Seedorf
NEC
Y. Yang
Yale
J. Peterson
Neustar
March 5, 2015

CDNI Footprint and Capabilities Advertisement using ALTO
draft-seedorf-cdni-request-routing-alto-08

Abstract

Network Service Providers (NSPs) are currently considering to deploy Content Delivery Networks (CDNs) within their networks. As a consequence of this development, there is a need for interconnecting these local CDNs. The necessary interfaces for inter-connecting CDNs are currently being defined in the Content Delivery Networks Interconnection (CDNI) WG. This document focuses on the CDNI Footprint & Capabilities Advertisement interface (FCI). Specifically, this document specifies a new Application Layer Traffic Optimization (ALTO) service to facilitate Footprint & Capabilities Advertisement in a CDNI context.

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 September 6, 2015.

Copyright Notice

Copyright (c) 2015 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	2
2. ALTO within CDNI Request Routing	3
3. Assumptions and High-Level Design Considerations	4
3.1. General Assumptions and Considerations	4
3.2. Semantics for Footprint/Capabilities Advertisement	5
3.3. Advantages of using ALTO as the CDNI FCI protocol	7
3.4. Selection of a Downstream CDN with ALTO	7
4. CDNI FCI ALTO Service	8
4.1. Server Response Encoding	8
4.1.1. CDNI FCI Map	8
4.1.2. Meta Information	8
4.1.3. Data Information	8
4.2. Protocol Errors	9
4.3. Example	9
5. Useful ALTO extensions for CDNI Request Routing	10
6. Security Considerations	12
7. Acknowledgements	12
8. References	12
8.1. Normative References	12
8.2. Informative References	13
Authors' Addresses	14

1. Introduction

Many Network Service Providers (NSPs) are currently considering or have already started to deploy Content Delivery Networks (CDNs) within their networks. As a consequence of this development, there is a need for interconnecting these local CDNs. Content Delivery Networks Interconnection (CDNI) has the goal of standardizing protocols to enable such interconnection of CDNs [RFC6707].

The CDNI problem statement [RFC6707] envisions four interfaces to be standardized within the IETF for CDN interconnection:

- o CDNI Request Routing Interface

- o CDNI Metadata Interface
- o CDNI Logging Interface
- o CDNI Control Interface

This document focuses solely on the CDNI Request Routing Interface, which can be further divided into two interfaces (see [RFC6707] for a detailed description): the CDNI Request Routing Redirection interface (RI), and the CDNI Footprint & Capabilities Advertisement interface (FCI). This document specifies a new Application Layer Traffic Optimization (ALTO) [RFC7285] service called 'CDNI Footprint & Capabilities Advertisement Service'. This service is used to transport CDNI FCI JSON objects, which are defined in a separate document [I-D.ma-cdni-capabilities]. An abstraction for managing individual CDNI capabilities in an opaque manner is defined as 'FCIBase' object in [I-D.ietf-cdni-footprint-capabilities-antics].

Throughout this document, we use the terminology for CDNI defined in [RFC6707].

2. ALTO within CDNI Request Routing

The main purpose of the CDNI Request Routing Interface is described in [RFC6707] as follows: "The CDNI Request Routing interface enables a Request Routing function in an Upstream CDN to query a Request Routing function in a Downstream CDN to determine if the Downstream CDN is able (and willing) to accept the delegated Content Request. It also allows the Downstream CDN to control what should be returned to the User Agent in the redirection message by the upstream Request Routing function." On a high level, the scope of the CDNI Request Routing Interface therefore contains two main tasks:

- o A) Determining if the downstream CDN is willing to accept a delegated content request
- o B) Redirecting the content request coming from an upstream CDN to the proper entry point or entity in the downstream CDN

More precisely, in [RFC7336] the request routing interface is broadly divided into two functionalities:

- o 1) the asynchronous advertisement of footprint and capabilities by a dCDN that allows a uCDN to decide whether to redirect particular user requests to that dCDN (the CDNI FCI)
- o 2) the synchronous operation of actually redirecting a user request (the CDNI RI)

Application Layer Traffic Optimization (ALTO) [RFC7285] is an approach for guiding the resource provider selection process in distributed applications that can choose among several candidate resources providers to retrieve a given resource. By conveying network layer (topology) information, an ALTO server can provide important information to "guide" the resource provider selection process in distributed applications. Usually, it is assumed that an ALTO server conveys information these applications cannot measure themselves [RFC5693].

Originally, ALTO was motivated by the huge amount of cross-ISP traffic generated by P2P applications [RFC5693]. Recently, however, ALTO is also being considered for improving the request routing in CDNs [I-D.jenkins-alto-cdn-use-cases]. In this context, it has also been proposed to use ALTO for selecting an entry-point in a downstream NSP's network (see section 3.4 "CDN delivering Over-The-Top of a NSP's network" in [I-D.jenkins-alto-cdn-use-cases]). Also, the CDNI problem statement explicitly mentions ALTO as a candidate protocol for "algorithms for selection of CDN or Surrogate by Request-Routing systems" [RFC6707].

3. Assumptions and High-Level Design Considerations

In this section we list some assumptions and design issues to be considered when using ALTO for the CDNI Footprint and Capabilities Advertisement interface.

3.1. General Assumptions and Considerations

Below we list some general assumptions and considerations:

- o As explicitly being out-of-scope for CDNI [RFC6707], it is assumed that ingestion of content or acquiring content across CDNs is not part of request routing as considered within CDNI standardization work. The focus of using ALTO (as considered in this document) is hence on request routing only, assuming that the content (desired by the end user) is available in the downstream CDN (or can be acquired by the downstream CDN by some means).
- o Federation Model: "Footprint and Capabilities Advertisement" and in general CDN request routing depends on the federation model among the CDN providers. Designing a suitable solution thus depends on whether a solution is needed for different settings, where CDNs consist of both NSP CDNs (serving individual ASes) and general, traditional CDNs (such as Akamai). We assume that CDNI is not designed for a setting where only NSP CDNs each serve a single AS only.

- o In this document, it is assumed that the upstream CDN (uCDN) makes the decision on selecting a downstream CDN, based on information that each downstream CDN has made available to the upstream CDN. Further, we assume that in principle more than one dCDN may be suitable for a given end-user request (i.e. different dCDNs may claim "overlapping" footprints). The uCDN hence potentially has to select among several candidate downstream CDNs for a given end user request.
- o It is not clear what kind(s) of business, contract, and operational relationships two peering CDNs may form. For the Internet, we see provider-customer and peering as two main relations; providers may use different charging models (e.g., 95-percentile, total volume) and may provide different SLAs. Given such unknown characteristics of CDN peering business agreements, we should design the protocol to support as much diverse potential business and operational models as possible.

3.2. Semantics for Footprint/Capabilities Advertisement

The CDNI document on "Footprint and Capabilities Semantics" [I-D.ietf-cdni-footprint-capabilities-semantics] defines the semantics for the CDNI FCI. It thus provides guidance on what Footprint and Capabilities mean in a CDNI context and how a protocol solution should in principle look like. Here we briefly summarize the key points of the semantics of Footprint and Capabilities (for a detailed discussion, the reader is referred to [I-D.ietf-cdni-footprint-capabilities-semantics]):

- o Often, footprint and capabilities are tied together and cannot be interpreted independently from each other. In such cases, i.e. where capabilities must be expressed on a per footprint basis, it may be beneficial to combine footprint and capabilities advertisement.
- o Given that a large part of Footprint and Capabilities Advertisement will actually happen in contractual agreements, the semantics of CDNI Footprint and Capabilities advertisement refer to answering the following question: what exactly still needs to be advertised by the CDNI FCI? For instance, updates about temporal failures of part of a footprint can be useful information to convey via the CDNI request routing interface. Such information would provide updates on information previously agreed in contracts between the participating CDNs. In other words, the CDNI FCI is a means for a dCDN to provide changes/updates regarding a footprint and/or capabilities it has prior agreed to serve in a contract with a uCDN.

- o It seems clear that "coverage/reachability" types of footprint must be supported within CDNI. The following such types of footprint are mandatory and must be supported by the CDNI FCI:

- * List of ISO Country Codes
- * List of AS numbers
- * Set of IP-prefixes

A 'set of IP-prefixes' must be able to contain full IP addresses, i.e., a /32 for IPv4 and a /128 for IPv6, and also IP prefixes with an arbitrary prefix length. There must also be support for multiple IP address versions, i.e., IPv4 and IPv6, in such a footprint.

- o For all of these mandatory-to-implement footprint types, footprints can be viewed as constraints for delegating requests to a dCDN: A dCDN footprint advertisement tells the uCDN the limitations for delegating a request to the dCDN. For IP prefixes or ASN(s), the footprint signals to the uCDN that it should consider the dCDN a candidate only if the IP address of the request routing source falls within the prefix set (or ASN, respectively). The CDNI specifications do not define how a given uCDN determines what address ranges are in a particular ASN. Similarly, for country codes a uCDN should only consider the dCDN a candidate if it covers the country of the request routing source. The CDNI specifications do not define how a given uCDN determines the country of the request routing source. Multiple footprint constraints are additive, i.e. the advertisement of different types of footprint narrows the dCDN candidacy cumulatively.
- o The following capabilities seem useful as 'base' capabilities, i.e. ones that are needed in any case and therefore constitute mandatory capabilities to be supported by the CDNI FCI:

- * Delivery Protocol (e.g., HTTP vs. RTMP)
- * Acquisition Protocol (for acquiring content from a uCDN)
- * Redirection Mode (e.g., DNS Redirection vs. HTTP Redirection as discussed in [RFC7336])
- * Capabilities related to CDNI Logging (e.g., supported logging mechanisms)

- * Capabilities related to CDNI Metadata (e.g., authorization algorithms or support for proprietary vendor metadata)

3.3. Advantages of using ALTO as the CDNI FCI protocol

The following reasons make ALTO a suitable candidate protocol for downstream CDN selection as part of CDNI request routing and in particular for an FCI protocol:

- o CDN request routing is done at the application layer. ALTO is a protocol specifically designed to improve application layer traffic (and application layer connections among hosts on the Internet) by providing additional information to applications that these applications could not easily retrieve themselves. For CDNI, this is exactly the case: a uCDN wants to improve application layer CDN request routing by using dedicated information (provided by a dCDN) that the uCDN could not easily obtain otherwise.
- o The semantics of an ALTO network map are an exact match for the needed information to convey a footprint by a downstream CDN, in particular if such a footprint is being expressed by IP-prefix ranges.
- o Security: ALTO maps can be signed and hence provide inherent integrity protection (see Section 6)
- o RESTful-Design: The ALTO protocol has undergone extensive revisions in order to provide a RESTful design regarding the client-server interaction specified by the protocol. A CDNI FCI interface based on ALTO would inherit this RESTful design.
- o Error-handling: The ALTO protocol has undergone extensive revisions in order to provide sophisticated error-handling, in particular regarding unexpected cases. A CDNI FCI interface based on ALTO would inherit this thought-through and mature error-handling.
- o Filtered network map: The ALTO Map Filtering Service (see [RFC7285] for details) would allow a uCDN to query only for parts of an ALTO map.

3.4. Selection of a Downstream CDN with ALTO

Under the considerations stated in Section 3, ALTO can help the upstream CDN provider to select a proper downstream CDN provider for a given end user request as follows: Each downstream CDN provider

hosts an ALTO server which provides ALTO services which convey CDNI FCI information to an ALTO client at the upstream CDN provider.

4. CDNI FCI ALTO Service

The ALTO protocol is based on an ALTO Information Service Framework which consists of several services, where all ALTO services are 'provided through a common transport protocol, messaging structure and encoding, and transaction model' [RFC7285]. The ALTO protocol specification [RFC7285] defines several such services, e.g. the ALTO map service.

This document defines a new ALTO Service called 'CDNI Footprint & Capabilities Advertisement Service' which conveys JSON objects of media type 'application/alto-fcimap+json'. This media type and JSON object format is defined in [I-D.ma-cdni-capabilities]; this document specifies how to transport such JSON objects via the ALTO protocol with the ALTO 'CDNI Footprint & Capabilities Advertisement Service'. An abstraction for managing individual CDNI capabilities in an opaque manner is defined as 'FCIBase' object in [I-D.ietf-cdni-footprint-capabilities-antics].

4.1. Server Response Encoding

4.1.1. CDNI FCI Map

The media type of the CDNI FCI Map is 'application/alto-cdni-fcimap+json'. The HTTP Method, Accept Input Parameters, Capabilities, Uses, and Response of the CDNI FCI Map are specified in [I-D.ma-cdni-capabilities].

4.1.2. Meta Information

The 'meta' field of a FCIMapData response MUST include 'vtag', which is an ALTO Version Tag of the retrieved FCIMapData according to [RFC7285] (Section 10.3.). It thus contains a 'resource-id' attribute, and a 'tag' is an identifier string.

4.1.3. Data Information

The data component of a CDNI FCI Map resource is named 'fcimap' which is a JSON object of type FCIMapData. This JSON object of type FCIMapData is derived from ResponseEntityBase as specified in the ALTO protocol [RFC7285] (Section 8.4.) and specified in [I-D.ma-cdni-capabilities].

4.2. Protocol Errors

Protocol errors are handled as specified in the ALTO protocol [RFC7285] (Section 8.5.).

4.3. Example

The following example shows an CDNI FCI Map as in [I-D.ma-cdni-capabilities], however with meta-information as defined in Section 4.1.2 of this document.


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

HTTP/1.1 200 OK
Content-Length: 439
Content-Type: application/alto-fcimap+json
{
  "meta" : {
    "vtag": {
      "resource-id": "my-default-fcimap",
      "tag": "da65eca2eb7a10ce8b059740b0b2e3f8eb1d4785"
    }
  },
  "fcimap": [
    { "name": "delivery_protocol",
      "values": [
        "HTTP",
        "RTSP",
        "MMS"
      ]
    },
    { "name": "delivery_protocol",
      "values": [
        "RTMP",
        "HTTPS"
      ],
      "footprint": [
        { "type": "IPv4CIDR",
          "values": [
            "10.1.0.0/16",
            "10.10.10.0/24"
          ]
        }
      ]
    }
  ]
}
```

5. Useful ALTO extensions for CDNI Request Routing

It is envisioned that yet-to-be-defined ALTO extensions will be standardized that make the ALTO protocol more suitable and useful for applications other than the originally considered P2P use case [I-D.marocco-alto-next]. Some of these extensions to the ALTO protocol would be useful for ALTO to be used as a protocol within CDNI request routing, and in particular within the "Footprint and

Capabilities Advertisement" part of the CDNI request routing interface.

The following proposed extensions to ALTO would be beneficial to facilitate CDNI request routing with ALTO as outlined in Section 3.4:

- o **Server-initiated Notifications and Incremental Updates:** In case the footprint or the capabilities of a downstream CDN change abruptly (i.e. unexpectedly from the perspective of an upstream CDN), server initiated notifications would enable a dCDN to directly inform an upstream CDN about such changes. Consider the case where - due to failure - part of the footprint of the dCDN is not functioning, i.e. the CDN cannot serve content to such clients with reasonable QoS. Without server-initiated notifications, the uCDN might still use a very recent network and cost map from dCDN, and therefore redirect request to dCDN which it cannot serve. Similarly, the possibility for incremental updates would enable efficient conveyance of the aforementioned (or similar) status changes by the dCDN to the uCDN. A proposal for server-initiated ALTO updates can be found in [I-D.marocco-alto-ws]. A discussion of incremental ALTO updates can be found in [I-D.schwan-alto-incr-updates].
- o **Content Availability on Hosts:** A dCDN might want to express CDN capabilities in terms of certain content types (e.g. codecs/formats, or content from certain content providers). A new endpoint property for ALTO that would be able to express such "content availability" would enable a dCDN to make available such information to an upstream CDN. This would enable a uCDN to determine if a given dCDN actually has the capabilities for a given request with respect to the type of content requested.
- o **Resource Availability on Hosts or Links:** The capabilities on links (e.g. maximum bandwidth) or caches (e.g. average load) might be useful information for an upstream CDN for optimized downstream CDN selection. For instance, if a uCDN receives a streaming request for content with a certain bitrate, it needs to know if it is likely that a dCDN can fulfill such stringent application-level requirements (i.e. can be expected to have enough consistent bandwidth) before it redirects the request. In general, if ALTO could convey such information via new endpoint properties, it would enable more sophisticated means for downstream CDN selection with ALTO.

6. Security Considerations

One important security consideration is the proper authentication of advertisement information provided by a downstream CDN. The ALTO protocol provides a specification for a signature of ALTO information (see 8.2.2. of [RFC7285]). ALTO thus provides a proper means for protecting the integrity of FCI information.

More Security Considerations will be discussed in a future version of this document.

7. Acknowledgements

The authors would like to thank Kevin Ma, Daryl Malas, and Matt Caulfield for their timely reviews and invaluable comments.

Jan Seedorf is partially supported by the GreenICN project (GreenICN: Architecture and Applications of Green Information Centric Networking), a research project supported jointly by the European Commission under its 7th Framework Program (contract no. 608518) and the National Institute of Information and Communications Technology (NICT) in Japan (contract no. 167). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the GreenICN project, the European Commission, or NICT.

8. References

8.1. Normative References

- [RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement", RFC 5693, October 2009.
- [RFC6707] Niven-Jenkins, B., Le Faucheur, F., and N. Bitar, "Content Distribution Network Interconnection (CDNI) Problem Statement", RFC 6707, September 2012.
- [RFC6770] Bertrand, G., Stephan, E., Burbridge, T., Eardley, P., Ma, K., and G. Watson, "Use Cases for Content Delivery Network Interconnection", RFC 6770, November 2012.
- [RFC7285] Alimi, R., Penno, R., Yang, Y., Kiesel, S., Previdi, S., Roome, W., Shalunov, S., and R. Woundy, "Application-Layer Traffic Optimization (ALTO) Protocol", RFC 7285, September 2014.

- [RFC7336] Peterson, L., Davie, B., and R. van Brandenburg, "Framework for Content Distribution Network Interconnection (CDNI)", RFC 7336, August 2014.
- [RFC7337] Leung, K. and Y. Lee, "Content Distribution Network Interconnection (CDNI) Requirements", RFC 7337, August 2014.

8.2. Informative References

- [I-D.peterson-cdni-strawman]
Peterson, L. and J. Hartman, "Content Distribution Network Interconnection (CDNI) Problem Statement", draft-peterson-cdni-strawman-01 (work in progress), May 2011.
- [I-D.marocco-alto-next]
Marocco, E. and V. Gurbani, "Extending the Application-Layer Traffic Optimization (ALTO) Protocol", draft-marocco-alto-next-00 (work in progress), January 2012.
- [I-D.marocco-alto-ws]
Marocco, E. and J. Seedorf, "WebSocket-based server-to-client notifications for the Application-Layer Traffic Optimization (ALTO) Protocol", draft-marocco-alto-ws-02 (work in progress), February 2014.
- [I-D.schwan-alto-incr-updates]
Schwan, N. and B. Roome, "ALTO Incremental Updates", draft-schwan-alto-incr-updates-02 (work in progress), July 2012.
- [I-D.jenkins-alto-cdn-use-cases]
Niven-Jenkins, B., Watson, G., Bitar, N., Medved, J., and S. Previdi, "Use Cases for ALTO within CDNs", draft-jenkins-alto-cdn-use-cases-03 (work in progress), June 2012.
- [I-D.ma-cdni-capabilities]
Ma, K. and J. Seedorf, "CDNI Footprint & Capabilities Advertisement Interface", draft-ma-cdni-capabilities-06 (work in progress), June 2014.
- [I-D.liu-cdni-cost]
Liu, H., "A Cost Perspective on Using Multiple CDNs", draft-liu-cdni-cost-00 (work in progress), October 2011.

[I-D.ietf-cdni-metadata]

Niven-Jenkins, B., Murray, R., Caulfield, M., and K. Ma,
"CDN Interconnection Metadata", draft-ietf-cdni-
metadata-09 (work in progress), March 2015.

[I-D.ietf-cdni-logging]

Faucheur, F., Bertrand, G., Oprescu, I., and R.
Peterkofsky, "CDNI Logging Interface", draft-ietf-cdni-
logging-15 (work in progress), February 2015.

[I-D.ietf-cdni-footprint-capabilities-semantic]

Seedorf, J., Peterson, J., Previdi, S., Brandenburg, R.,
and K. Ma, "CDNI Request Routing: Footprint and
Capabilities Semantics", draft-ietf-cdni-footprint-
capabilities-semantic-05 (work in progress), March 2015.

Authors' Addresses

Jan Seedorf
NEC Laboratories Europe, NEC Europe Ltd.
Kurfuersten-Anlage 36
Heidelberg 69115
Germany

Phone: +49 (0) 6221 4342 221
Email: jan.seedorf@neclab.eu
URI: <http://www.neclab.eu>

Y.R. Yang
Yale University
51 Prospect Street
New Haven 06511
USA

Email: yry@cs.yale.edu
URI: <http://www.cs.yale.edu/~yry/>

Jon Peterson
NeuStar
1800 Sutter St Suite 570
Concord CA 94520
USA

Email: jon.peterson@neustar.biz

ALTO WG
Internet-Draft
Intended status: Informational
Expires: January 4, 2018

Q. Xiang
Tongji/Yale University
H. Newman
California Institute of Technology
G. Bernstein
Grotto Networking
H. Du
Tongji/Yale University
K. Gao
Tsinghua University
A. Mughal
J. Balcas
California Institute of Technology
J. Zhang
Tongji University
Y. Yang
Tongji/Yale University
July 3, 2017

Resource Orchestration for Multi-Domain Data Analytics
draft-xiang-alto-exascale-network-optimization-03.txt

Abstract

Data-intensive analytics is entering the era of multi-domain, geographically-distributed, collaborative computing, where different organizations contribute various resources to collaboratively collect, share and analyze extremely large amounts of data. Examples of this paradigm include the Compact Muon Solenoid (CMS) and A Toroidal LHC ApparatuS (ATLAS) experiments of the Large Hadron Collider (LHC) program. Massive datasets continue to be acquired, simulated, processed and analyzed by globally distributed science networks in these collaborations. Applications that manage and analyze such massive data volumes can benefit substantially from the information about networking, computing and storage resources from each member's site, and more directly from network-resident services that optimize and load balance resource usage among multiple data transfers and analytics requests, and achieve a better utilization of multiple resources in clusters.

The Application-Layer Traffic Optimization (ALTO) protocol can provide via extensions the network information about different clusters/sites, to both users and proactive network management services where applicable, with the goal of improving both application performance and network resource utilization. In this document, we propose that it is feasible to use existing ALTO services to provides not only network information, but also

information about computation and storage resources in data analytics networks. We introduce a uniform resource orchestration framework (Unicorn), which achieves an efficient multi-resource allocation to support low-latency dataset transfer and data intensive analytics for collaborative computing. It collects cluster information from multiple ALTO services utilizing topology extensions and leverages emerging SDN control capabilities to orchestrate the resource allocation for dataset transfers and analytics tasks, leading to improved transfer and analytics latency as well as more efficient utilization of multi-resources in sites.

Status of This Memo

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

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

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

This Internet-Draft will expire on January 4, 2018.

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	3
2. Requirements Language	5
3. Changes Since Version -02	5
4. Problem Settings	6
4.1. Motivation	6
4.2. Challenges	6
5. Basic Idea	7
5.1. Use ALTO services to provide multi-resource information .	7
5.2. Example 1: network information impacts data analytics	
scheduling	8
5.3. Example 2: encode multi-resources in abstract network	
elements	9
6. Key Issues	10
7. Unified Resource Orchestration Framework	11
7.1. Architecture	11
7.2. Workflow converter	14
7.2.1. User API	14
7.3. Resource Demand Estimator	15
7.4. Entity Locator	15
7.5. ALTO Client	15
7.5.1. Query Mode	16
7.6. ALTO Server	16
7.7. Resource View Extractor	16
7.8. Execution Agents	18
7.9. Multi-Resource Orchestrator	18
7.9.1. Orchestration Algorithms	18
7.9.2. Online, Dynamic Orchestration	18
7.9.3. Example: A Max-Min Fairness Resource Allocation	
Algorithm	19
8. Discussion	20
8.1. Deployment	20
8.2. Benefiting From ALTO Extension Services	21
8.3. Limitations of the MFRA Algorithm	21
9. Security Considerations	22
10. IANA Considerations	22
11. Acknowledgments	22
12. References	22
12.1. Normative References	22
12.2. Informative References	22
Authors' Addresses	24

1. Introduction

As the data volume increases exponentially over time, data intensive analytics is transiting from single-domain computing to multi-organizational, geographically-distributed, collaborative computing,

where different organizations contribute various resources, e.g., computation, storage and networking resources, to collaboratively collect, share and analyze extremely large amounts of data. One leading example is the Large Hadron Collider (LHC) high energy physics (HEP) program, which aims to find new particles and interactions in a previously inaccessible range of energies. The scientific collaborations that have built and operate large HEP experimental facilities at the LHC, such as the Compact Muon Solenoid (CMS) and A Toroidal LHC ApparatuS (ATLAS), currently have more than 300 petabytes of data under management at hundreds of sites around the world, and this volume is expected to grow to one exabyte by approximately 2018.

With such an increasing data volume, how to manage the storage and analytics of these data in a globally distributed infrastructure has become an increasingly challenging issue. Applications such as the Production AND Distributed Analysis system (PanDA) in ATLAS and the Physics Experiment Data Export system (PhEDEx) in CMS have been developed to manage the data transfers among different cluster sites. Given a data transfer request, these applications make data transfer decisions based on the availability of dataset replicas at different sites and initiate retransmission from a different replica if the original transmission fails or is excessively delayed. And HTCondor [HTCondor] is deployed to achieve coarse-grained data analytics parallelization across these sites. When a data analytics task is submitted, HTCondor adopts a match-making process to assign the task to a certain set of servers in one site, based on the coarse-grained description of resource availability, such as the number of cores, the size of memory, the size of hard disk, etc. However, neither dataset transfers nor data analytics task parallelization takes fine-grained information of cluster resources, such as data locality, memory speed, network delay, network bandwidth, etc., into account, leading to high data transfer and analytics latency and underutilization of cluster resources.

The Application-Layer Traffic Optimization (ALTO) services defined in [RFC7285] provide network information with the goal of improving the network resource utilization while maintaining or improving application performance. Though ALTO is not designed to provide information about other resources, such as computing and storage resources in cluster networks, in this document we propose that exascale science networks can leverage existing ALTO services defined in [RFC7285] and ALTO topology extension services defined in network graph [DRAFT-NETGRAPH], path vector [DRAFT-PV], routing state abstraction [DRAFT-RSA], multi-cost [DRAFT-MC] and cost-calendar [DRAFT-CC] and etc. to encode information about multiple types of resources in science networks, such as memory I/O speed, CPU utilization, network bandwidth, and provide such information to

orchestration applications to improve the performance of dataset transfer and data analytics tasks, including throughput, latency, etc.

This document introduces a unified resource orchestration framework (Unicorn), which provides efficient multi-resource allocation to support low-latency, multi-domain, geo-distributed data analytics. Unicorn provides a set of simple APIs for authorized users to submit, update and delete dataset transfer requests and data intensive analytics requests. One important proposal we make in this document is that it is feasible to use ALTO services to provide not only network information, but also information on other resources in multi-domain, geo-distributed analytics networks including computing and storage.

A prototype of Unicorn with the dataset transfer scheduling component has been implemented on a single-domain Caltech SDN development testbed, where the ALTO OpenDaylight controller is used to collect topology information. We are currently designing the resource orchestration components to achieve low-latency data-intensive analytics.

This document is organized as follows: Section 3 summarizes the change of this document since version -01. Section 4 elaborates on the motivation and challenges for coordinating storage, computing and network resources in a globally distributed science network infrastructure. Section 5 discusses the basic idea of encoding multi-resource information into ALTO path vector and abstraction services and gives an example. Section 6 lists several key issues to address in order to realize the proposal of providing multi-resource information by ALTO topology services. Section 7 gives the details of Unicorn architecture for multi-domain, geo-distributed data analytics. Section 8 discusses current development progress of Unicorn and next steps.

2. 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 [RFC2119].

3. Changes Since Version -02

- o Add an example in Section 7 to show the importance of network information in resource allocation for data analytics.

- o Update the architecture of Unicorn in Section 7, i.e., add the entity locator and rename ANE aggregator to resource view extractor.
- o Add detailed description of how the entity locator and the resource view extractor work.
- o Minor changes in abstract and discussion sections.

4. Problem Settings

4.1. Motivation

Multi-domain, geo-distributed data analytics usually involves the participation of countries and sites all over the world. Science programs such as the CMS experiment and the ATLAS experiment at CERN are typical examples. The site located at the LHC laboratory is called a Tier-0 site, which processes the data selected and stored locally by the online systems that select and record the data in real-time as it comes off the particle detector, archives it and transfers it to over 10 Tier-1 sites around the globe. Raw datasets and processed datasets from Tier-1 sites are then transferred to over 160 Tier-2 sites around the world based on users' requests. Different sites have different resources and belong to different administration domains. With the exponentially increasing data volume in the CMS experiment, the management of large data transfers and data intensive analytics in such a global multi-domain science network has become an increasingly challenging issue. Allocating resources in different clusters to fulfill different users' dataset transfer requests and data analytics requests require careful orchestrating as different requests are competing for limited storage, computation and network resources.

4.2. Challenges

Orchestrating exascale dataset transfers and analytics in a globally distributed science network is non-trivial as it needs to cope with two challenges.

- o Different sites in this network belong to different administration domains. Sharing raw site/cluster information would violate sites' privacy constraints. Orchestrating data transfers and analytics requests based on highly abstracted, non-real-time network information may lead to suboptimal scheduling decisions. Hence the orchestrating framework must be able to collect sufficient resource information about different clusters/sites in real-time as well as over the longer term, to allow reasonably

optimized network resource utilization without violating sites' privacy requirements.

- o Different science programs tend to adopt different software infrastructures for managing dataset transfers and analytics, and may place different requirements. Hence the orchestrating framework must be modular so that it can support different dataset management systems and different orchestrating algorithms.

The orchestrating framework must support the interaction between the multi-resource orchestration module, the dataset transfer module, and the data analytics execution module. The key information to be exchanged between modules includes dataset information, the resource state of different clusters and sites, the transfer and analytic requests in progress, as well as trends and network-segment and site performance from the network point of view. Such interaction ensures that (1) the various programs can adopt their own data transfer and analytics systems to be multi-resource-aware, and more efficient in achieving their goals; and (2) the various orchestrating algorithms can achieve a reasonably optimized utilization on not only the network resource but also the computing and storage resources.

5. Basic Idea

5.1. Use ALTO services to provide multi-resource information

The ALTO protocol is designed to provide network information to applications so that applications can achieve better performance and the network more efficient use of resources. Different ALTO topology services including path vector, routing state abstraction, multi-cost, cost calendar, etc., have been proposed to provide fine-grained network information to applications. In this document, we propose that not only can ALTO provide network information of different cluster sites, it can also provide information of multiple resources, including computing and storage resources. To this end, the basic "one-big-switch" abstraction provided by the base ALTO protocol is not sufficient. Several examples have already been given in [DRAFT-PV] and [DRAFT-RSA] to demonstrate that. There has been a similar proposal before about using ALTO to provide resource information for data centers [DRAFT-DC]. However, that proposal requires a new information model for clusters or data centers, which may affect the compatibility of ALTO. The solution of this proposal is simpler. Its basic idea is that each computer node and storage node can be seen as an "abstract network element" defined in ALTO-path-vector [DRAFT-PV]. In this way, Unicorn can fully reuse all existing ALTO services by introducing only one cost-mode (pv) and two cost-metrics (ne and ane), instead of introducing a new information model.

5.2. Example 1: network information impacts data analytics scheduling

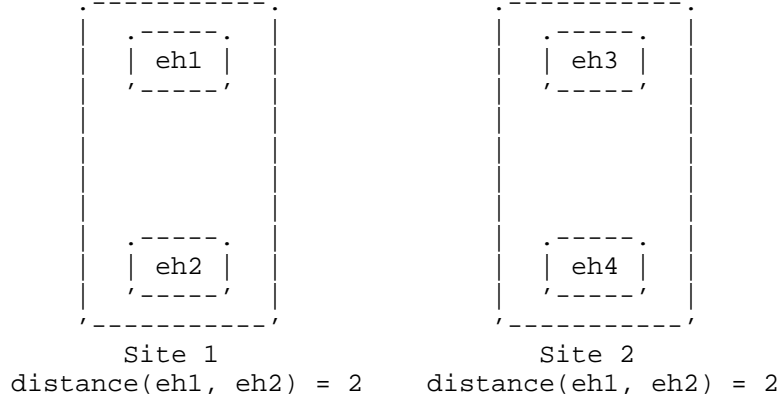


Figure 1: An Example for Data Locality Information.

We first use the example in Figure 1 to show that only network information itself has a huge impact on resource allocation for data analytics. In this scenario, a MapReduce task needs to be executed. The input data has two copies at end hosts eh1 and eh3, respectively. And the end hosts eh2 and eh4 will be the computation nodes with the same computation power, correspondingly. Using the common data analytics resource management framework such as Hadoop it can be revealed that both allocation options, i.e., eh1->eh2 and eh3->eh4, have a storage-computation distance of 2, i.e., they have the same data locality from Hadoop's view. As a result, it appears that both options would provide the same performance for this task.

However, assume that the bandwidth between eh1 and eh2 is 100Mb/s while that between eh3 and eh4 is 1Gb/s. These significant different data accessing speeds decide that these options have very different performances for the same task and the only optimal allocation option is to allocate this task to eh3->eh4. This example demonstrates the imperativeness of network information in making efficient resource allocation decisions. Such information is not provided in Hadoop or other similar or related projects such as Mesos. On the contrary, if ALTO servers are deployed at these sites, applications can retrieve such information via ALTO queries.

5.3. Example 2: encode multi-resources in abstract network elements

We use the same dumbbell topology in [DRAFT-RSA] as an example to show the feasibility of using ALTO topology service to provide multi-resource information. In this topology, we assume the bandwidth of each network cable is 1Gbps, including the cables connecting end hosts to switches. Consider a dataset transfer request which needs to schedule the traffic among a set of end host source-destination pairs, say eh1 → eh2, and eh3 → eh4. Assume that the transfer application receives information from the ALTO Cost Map service that both eh1 → eh2 and eh3 → eh4 have bandwidth 1Gbps. In [DRAFT-RSA], it is shown that whether each of the two traffic flows can receive 1Gbps bandwidth depends on whether the routes of two flows share a bottleneck link. Path vector and routing state abstraction services provide additional information about network state encoded in abstract network elements. If the returned state is $ane1 + ane2 \leq 1\text{Gbps}$, it means two flows cannot each get 1Gbps bandwidth at the same time. If the returned state is $ane1 \leq 1\text{Gbps}$ and $ane2 \leq 1\text{Gbps}$, it means two flows each can get 1Gbps bandwidth.

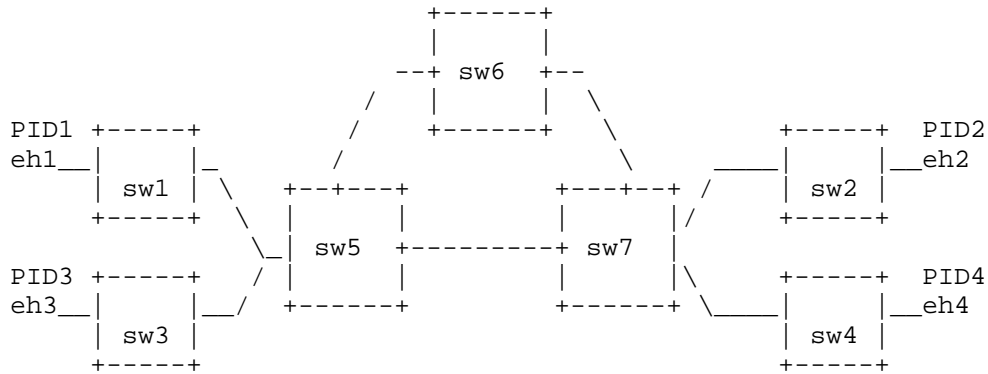


Figure 2: A Dumbbell Network Topology

Other than network resource, assume in this topology eh1 and eh3 are equipped with commodity hard disk drives (HDD) while eh2 and eh4 are equipped with SSDs. Because the bandwidth of an HDD is typically 0.8Gbps and that of SSD is typically 3Gbps. Even if the returned routing state is $ane1 \leq 1\text{Gbps}$ and $ane2 \leq 1\text{Gbps}$, the actual bottleneck of each traffic flow is the storage I/O bandwidth at source host. As a result, the total bandwidth of both traffic flows can only reach 1.6Gbps.

It has been verified in the CMS experiment, and also several studies on commercial data centers that network resource are not always the bottleneck of large dataset transfers and data analytics. Many have reported that storage resources and computing resources become the bottleneck in a fairly large percentage of dataset transfers and data analytics tasks in science networks and commercial data centers.

In this example, if we look at the end hosts as abstract network elements, the storage I/O bandwidth of each host can also be encoded as an abstract element into the path-vector. And under the storage and route settings above, the returned cluster state would be $ane1 \leq 0.8\text{Gbps}$ and $ane2 \leq 0.8\text{Gbps}$, which provides a more accurate capacity region for the requested traffic flows.

6. Key Issues

The last section described the basic idea of using ALTO topology services to provide multi-resource information and gives an example to demonstrate its feasibility. Next we list and discuss several key issues to address in this proposal.

- o Can ALTO topology services provide data locality information? Existing ALTO topology services do not provide such information. Many studies have pointed out that such information plays a vital role in reducing the latency of data-intensive analytics. If ALTO topology services can encode such information together with information of other resources together, data-intensive applications can benefit a great deal in terms of information aggregation and communication overhead.
- o How to quickly map applications' resource allocation decision on abstract multi-resource view back to the physical multi-resource view of clusters/sites? Fine-grained resource information can be encoded into abstract network elements to reduce overhead and provide certain privacy protection of clusters. Such information can be highly compressed (see the dumbbell example used in this document as well as in [DRAFT-PV] and [DRAFT-RSA]). In preliminary evaluations on RSA, the network element compression ratio can be as high as 80 percent. This ratio is expected to be even higher in large-scale data center or cluster setting, e.g. a fat-tree topology with $k=48$. Therefore a fast mapping from the resource orchestration decisions based on the abstract view back to the physical view is needed to satisfy the stringent latency requirement of large dataset transfers and data-intensive analytics.
- o How much privacy, including key resource configurations, raw topology, intra-cluster scheduling policy, etc., will be exposed?

Compared with the "one-big-switch" abstraction, other ALTO topology services such as path vector [DRAFT-PV] and routing state abstraction [DRAFT-RSA] provide fine-grained resource information to applications. Even if such information can be encoded into abstract network elements, it still risks exposing private information of different clusters/sites. Current internet drafts of these services did not provide any formal privacy analysis or performance measurement. This would be one of the key issues this document plans to investigate in the future.

- o How does current ALTO services such as path-vector and RSA scale when they are used to provide abstract information concerning multiple resources in clusters? Another issue along this line is how to balance the liveness of fine-grained resource information and the corresponding information delivery overhead? Although encoding information of network elements into abstract network elements can achieve a very competitive information compression ratio, large dataset transfers and analytics applications always involve many network elements in multiple clusters/sites and the absolute number of involved network elements keep increasing as the scale of clusters increase. In addition, when resource information in a cluster changes, the ALTO services need to inform all related applications. In either cases, delivering fine-grained resource information would cause high communication overhead. There still lacks of an analytics or experimental understanding on the scalability of path-vector and RSA services.

7. Unified Resource Orchestration Framework

7.1. Architecture

This section describes the design details of key components of the Unicorn framework: the workflow converter, the resource demand estimator, the ALTO clients, the ALTO servers, the resource view extractor, the multi-resource orchestrator and the execution agents. Figure 3 shows the architecture of Unicorn. The overall process is as follows.

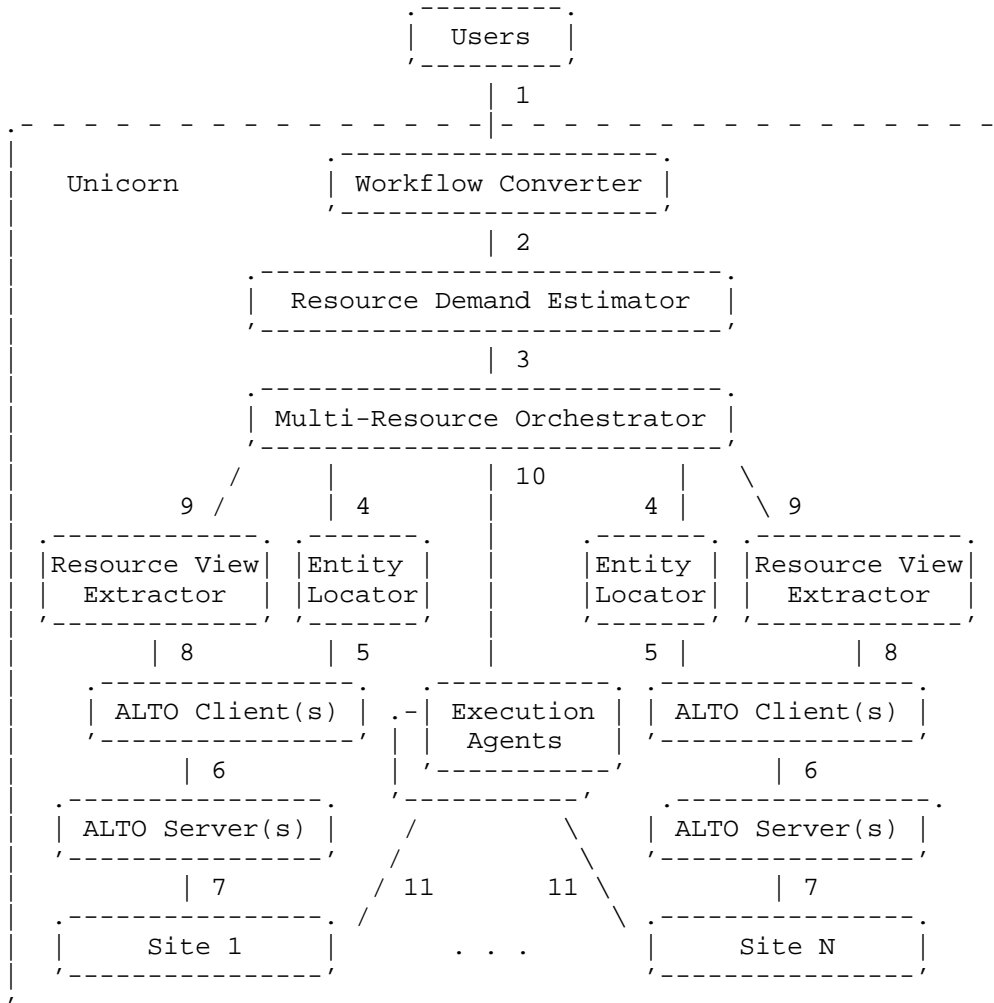


Figure 3: Architecture of Unicorn.

- o STEP 1 Authorized users submit high-level data analytics workflows to Unicorn through a set of simple APIs.
- o STEP 2 The workflow converter transforms the high-level data analytics workflows into low-level task workflows, i.e., a set of analytics tasks with precedence encoded in a directed acyclic graph (DAG).

- o STEP 3 The resource demand estimator automatically finds the optimal configuration (resource demand) of each task, i.e., the number of CPUs, the size of memory and disk, I/O bandwidth, etc.
- o STEP 4 The multi-resource orchestrator receives the resource demand of a set of tasks and sends them to the entity locator at each site.
- o STEP 5 The entity locator at each site retrieves the entity addresses of the end hosts that would be allocated for the tasks to be scheduled, and passes these addresses to the ALTO clients, and asks the ALTO clients to collect information about these end hosts, i.e., properties of corresponding computing, storage and networking resources.
- o STEP 6 The ALTO clients issue ALTO queries defined in the base ALTO protocol [RFC7285], e.g., EPS, ECS, Network Map, etc. and ALTO extension services, e.g., routing state abstraction (RSA) [DRAFT-RSA], path vector [DRAFT-PV], network graph [DRAFT-NETGRAPH], multi-cost [DRAFT-MC], cost-calendar [DRAFT-CC] and property map [DRAFT-PM], to collect resource information.
- o STEP 7 The ALTO servers at each site accept the queries from the ALTO client, collect resource information from the residing site and send back to the ALTO clients.
- o STEP 8 The ALTO clients send the response from ALTO servers to the resource view extractor.
- o STEP 9 The extractor uses a lightweight, optimal algorithm to compress the raw resource information provided by ALTO servers into a minimal, equivalent view of resources and sends back to the multi-resource orchestrator.
- o STEP 10 The orchestrator makes resource allocation decisions, e.g., dataset transfer scheduling and analytics task placement, based on the resource demand of analytics tasks and the resource supply sent back from the resource view extractor. Decisions are then sent to the execution agents deployed in corresponding sites.
- o STEP 11 The execution agents receive and execute instructions from the multi-resource orchestrator. They also monitor the status of different tasks and send the updated status to the multi-resource orchestrator.

Unicorn provides a unified, automatic solution for multi-domain, geo-distributed data analytics. In particular, its benefits include:

- o On the resource demand side, it provides a set of simple APIs for authorized users to submit and manage data analytics requests and enables real-time requests' status monitoring. And it automatically converts high-level analytics workflow into low-level task workflows and finds the optimal configuration for each task.
- o On the resource supply side, it collects the resource information of different sites through a common, REST based interface specified by the ALTO protocol, encodes such information into different entity abstractions and computes a minimal, yet accurate view on resource supply dynamic.
- o It provides a scalable multi-resource orchestrator that makes efficient resource allocation decisions to achieve high resource utilization and low-latency data analytics.
- o The architecture of Unicorn is modular to support different resource orchestration algorithms and the deployment of different ALTO servers.

7.2. Workflow converter

The converter is the front end of Unicorn. It is responsible for collecting high-level data analytics workflows from users and transforming them into low-level task workflows, e.g., HTCondor ClassAds. It provides a set of simple APIs for users to submit and manage requests, and to track the status of requests in real-time.

7.2.1. User API

- o `submitReq(request, [options])`

This API allows users to submit a request and specify corresponding options. The request can be a data transfer request or a data analytics request. Request options include priority, delay, etc. It returns a request identifier `reqID` that allows users to update, delete this request or track its status. The additional options may or may not be approved, and the relative priorities may be modified by the resource orchestrator depending on the role of users (regular users or administrators at different levels), the resource availability and the status of other ongoing requests.

- o `updateReq(requestID, [options])`

This API allows users to update the options of requests. It will return a SUCCESS if the new options are received by the request

parser. But these new options may or may not be approved, and may be modified by the resource orchestrator depending on the role of users (regular users or administrators), the resource availability and the status of other ongoing requests.

- o deleteReq(requestID)

This API allows users to delete a request by passing the corresponding requestID. A completed request cannot be deleted. An ongoing request will be stopped and the output data will be deleted.

- o getReqStatus(requestID)

This API allows users to query the status of a request by specifying the corresponding requestID. The returned status information includes whether the request has started, the assigned priority, the percentage of finished sub-requests, transmission statistics, the expected remaining time to finish, etc.

7.3. Resource Demand Estimator

The estimator leverages the fact that low-level tasks are typically repetitive or have very high similarities. It uses reinforcement learning to predict the optimal configuration for each task and passes the resource demand to the multi-resource orchestrator for further processing.

7.4. Entity Locator

The task configurations computed by the demand estimator has no knowledge on the underlying structure of topology of each site, i.e., the addresses of end hosts and network devices. Hence they cannot be directly used by the ALTO clients for querying resource information. The entity locator retrieves the entity addresses of the end hosts that would be allocated for the tasks to be scheduled, and passes these addresses to the ALTO clients, and asks the ALTO clients to collect information about these end hosts, i.e., properties of corresponding computing, storage and networking resources.

7.5. ALTO Client

The ALTO client is in the back end of Unicorn and is responsible for retrieving resource information through querying ALTO servers deployed at different sites. The resource information needed in Unicorn includes the topology, link bandwidth, computing node memory I/O speed, computing node CPU utilization, etc. The base ALTO protocol [RFC7285] provides an extreme single-node abstraction for

this information, which only allows the multi-resource orchestrator to make coarse-grained resource allocation decisions. To enable fine-grained multi-resource orchestration for dataset transfer and data analytics in cluster networks, ALTO topology extension services such as routing state abstraction (RSA) [DRAFT-RSA], path vector [DRAFT-PV], network graph [DRAFT-NETGRAPH], multi-cost [DRAFT-MC] and cost-calendar [DRAFT-CC] are needed to provide fine-grained information about different types of resources in clusters.

7.5.1. Query Mode

The ALTO client should operate in different query modes depending on the implementation of ALTO servers. If an ALTO server does not support incremental updates using server-sent events (SSE) [DRAFT-SSE], the ALTO client sends queries to this server periodically to get the latest resource information. If the cluster state changes after one query, the ALTO client will not be aware of the change until next query. If an ALTO server supports SSE, the ALTO client only sends one query to the ALTO server to get the initial cluster information. When the resource state changes, the ALTO client will be notified by the ALTO server through SSE.

7.6. ALTO Server

ALTO servers are deployed at different sites around the world, and at strategic locations in the network itself, to provide information about different types of resources in the cluster networks in response to queries from the ALTO client. Such information include topology, link bandwidth, memory I/O speed and CPU utilization at computing nodes, storage constraints in storage nodes and etc. Each ALTO server must provide basic information services as specified in [RFC7285] such as network map, cost map, endpoint cost service (ECS), etc. To support the fine-grained multi-resource allocation in Unicorn, each ALTO server should also provide more fine-grained information about different resources in clusters through ALTO extension services such as the routing state abstraction [DRAFT-RSA], path vector [DRAFT-PV], network graph [DRAFT-NETGRAPH], multi-cost [DRAFT-MC] and cost-calendar [DRAFT-CC] services.

7.7. Resource View Extractor

In each site, the resource information collected by the ALTO clients is not directly sent back to the orchestrator. Instead, we design a resource view extractor to compress the resource information provided by the ALTO servers into a minimal, equivalent view of all the resources, i.e., computing, storage and networking resources, that would be allocated to a set of tasks. The extractor works in the following steps.

- o Resource-Task Matrix

Depending on specific services provided by the ALTO servers, the responses collected by ALTO clients may include information of different entities, e.g., endpoints, PIDs, etc. Each entity possesses a set of resources available for tasks to be scheduled. For each entity property p in the responses, such as bandwidth, delay, etc., the extractor composes an entity-task matrix $M(p)$. Each row of this $M(p)$ represents an entity in the responses provides information about property p and each column represents a task to be scheduled. The element $m(i, j)$ of $M(p)$ is a variable representing the amount of property p of entity i that task j can get.

- o Resource-Task Constraints

For each property p , the extractor composes a series of resource-task constraints. The first set of constraints is $\sum m(i, j) = r(p, j)$ for each task j . These constraints calculate $r(p, j)$, the total amount of property p provided to j by all the entities. The exact rule of summation depends on the property p . For instance, if p is latency, the summation is through common addition operations, but if p is bandwidth, the summation is a minimum function.

The second set of constraints is $\sum m(i, j) \leq r(p, i)$ for each entity i . These constraints represent the usage of resource i on property p for all the tasks cannot exceed the capacity of resource i on this property. The exact rule of summation also depends on the property p . For instance, if p is bandwidth, the summation is through common addition operations, but if p is latency, the constraints become $m(i, j) = r(p, i)$ for each entity i and each task j . This means that entity i provides the same delay property for each task.

- o Resource View Compression

The whole set of resource-task constraints are linear, and express the view of resources that are available for the tasks to be scheduled. The extractor uses a lightweight, optimal algorithm to compress them into a minimal, equivalent view of resources, i.e., a minimal set of linear constraints that represent the same feasible region as the original constraints. The basic idea of this algorithm is described in [DRAFT-RSA].

7.8. Execution Agents

Execution agents are deployed at each site and are responsible for the following functions:

- o Receive and process instructions from the multi-resource orchestrator, e.g. dataset transfer scheduling, data analytics task placement and execution, task update and abortion, etc.
- o Monitor the status of data analytics tasks and send the updated status to the multi-resource orchestrator.

Depending on the supporting data analytics frameworks, different request execution agents may be deployed in each site. For instance, in the CMS experiment at CERN, both MPI and Hadoop execution agents are deployed.

7.9. Multi-Resource Orchestrator

The multi-resource orchestrator receives the resource demand information, i.e., a set of low-level task workflows and their configurations, from the resource demand estimator. It then asks the entity locator in each site to get the addresses of end hosts that would be allocated for the tasks to be scheduled and ALTO clients to query properties related to these end hosts. When the resource view extractor sends the response back, the orchestrator makes resource allocation decisions, e.g., dataset transfer scheduling and analytics task execution, based on both resource demand dynamic and resource supply dynamic. The dataset transfer scheduling decisions include dataset replica selection, path selection, and bandwidth allocation, etc. The analytics task execution decisions include which cluster should allocate how much resources to execute which tasks. These decisions are sent to the execution agents at different sites for execution.

7.9.1. Orchestration Algorithms

The modular design of Unicorn allows the adoption of different orchestration algorithms and methodologies, depending on the specific performance requirements. In Section 7.8.3, a max-min fairness resource allocation algorithm for dataset transfer is described as an example.

7.9.2. Online, Dynamic Orchestration

The multi-resource orchestrator should adjust the resource allocation decisions based on the progress of ongoing requests, the utilization and dynamics of cluster resources. In normal cases, the multi-

resource orchestrator periodically collects such information and executes the orchestration algorithm. When it is notified of events such as request status update, cluster state update and etc., the orchestrator will also execute the orchestration algorithm to adjust resource allocations.

7.9.3. Example: A Max-Min Fairness Resource Allocation Algorithm

In this section, we describe a max-min fair resource allocation (MFRA) scheduling algorithm which aims to minimize the maximal time to complete a dataset transfer subject to a set of constraints. To make resource allocation decisions, MFRA requires sufficient network information including topology, link bandwidth and recent historical information in some cases. In a small-scale single-domain network, an SDN controller can provide the raw complete topology information for the MFRA algorithm. However, in a large-scale multi-domain science network such as CMS, providing the raw network topology is infeasible because (1) it would incur significant communication overhead; and (2) it would violate the privacy constraints of some sites. Several ALTO extension topology services including Abstract Path Vector [DRAFT-PV], Network Graphs [DRAFT-NETGRAPH] and RSA [DRAFT-RSA] can provide the fine-grained yet aggregated/abstract topology information for MFRA to efficiently utilize bandwidth resources in the network.

Ongoing pre-production deployment efforts of Unicorn in the CMS network involve the implementation of the RSA service. Other than topology information, the additional input of the MFRA algorithm is the priority of each class of flows, expressed in terms of upper and lower limits on the allocated bandwidth between the source and the destination for each data transfer requests.

The basic idea of the MFRA algorithm is to iteratively maximize the volume of data that can be transferred subject to the constraints. It works in quantized time intervals such that it schedules network paths and data volumes to be transferred in each time slot. When the DTR scheduler is notified of events such as the cancellation of a DTR, the completion of a DTR or network state changes, the MFRA algorithm will also be invoked to make updated network path and bandwidth allocation decisions.

In each execution cycle, MFRA first marks all transfers as unsaturated. Then it solves a linear programming model to find the common minimum transfer satisfaction rate (i.e., the ratio of transferred data volume in a time interval over the whole data volume of this request) that is satisfied by all transfer requests. With this common rate found, MFRA then randomly selects an unsaturated request in each iteration, increases its transfer rate as much as

possible by finding residual paths available in the network, or by increasing the allocated bandwidth along an existing path, until it reaches its upper limit or can otherwise not be increased further, so it is saturated. At each iteration, newly saturated requests are removed from the subsequent process by fixing their corresponding rate value, and completed transfers are removed from further consideration. After all the data transfer rates are saturated in the given time slot, then a feasible set of data transfer volumes scheduled to be transferred in the slot across each link in the network can be derived.

The MFRA algorithm yields a full utilization of limited network resources such as bandwidth so that all DTR can be completed in a timely manner. It allocates network resources fairly so that no DTR suffers starvation. It also achieves load balance among the sites and the network paths crossing a complex network topology so that no site and no network link is oversubscribed. Moreover, MFRA can handle the case where particular routing constraints are specified, e.g., where all routes are fixed ahead of time, or where each transfer request only uses one single path in each time slot, by introducing an additional set of linear constraints.

8. Discussion

8.1. Deployment

The Unicorn framework is the first step towards a new class of intelligent, SDN-driven global systems for multi-domain, geo-distributed data analytics involving a worldwide ensemble of sites and networks, such as CMS and ATLAS. Unicorn relies heavily on the ALTO services for collecting and expressing abstract, real-time resource information from different sites, and the SDN centralized control capability to orchestrate data analytics workflows. It aims to provide a new operational paradigm in which science programs can use complex network and computing infrastructures with high throughput, while allowing for coexistence with other network traffic.

A prototype case study implementation of Unicorn has been demonstrated on the Caltech/StarLight/Michigan/Fermilab SDN development testbed. Because this testbed is a single-domain network, the current Unicorn prototype leverages the ALTO OpenDaylight controller, to collect topology information. The CMS experiment is currently exploring pre-production deployments of Unicorn, looking towards future widespread production use. To achieve this goal, it is imperative to collect sufficient resource information from the various sites in the multi-domain CMS network, without causing any privacy leak. To this end, the ALTO RSA service

[DRAFT-RSA] is under development. Furthermore, as will be discussed next, other ALTO topology extension services can also substantially improve the performance of Unicorn.

8.2. Benefiting From ALTO Extension Services

The current ALTO base protocol [RFC7285] exposes network topology and endpoint properties using the extreme "my-Internet-view" representation, which abstracts a whole network as a single node that has a set of access ports, with each port connects to a set of end hosts called endpoints. Such an extreme abstraction leads to significant information loss on network topology [DRAFT-PV], which is the key information for Unicorn to make dynamic scheduling and resource allocation decisions. Though Unicorn can still allocate resource for data transfer and analytics requests on this abstract view, the resource allocation decisions are suboptimal. Alternatively, feeding the raw, complete network topology of each site to Unicorn is not desirable, either. First, this would violate privacy constraints of different sites. Secondly, a raw network topology would significantly increase the problem space and the solution space of the orchestrating algorithm, leading to a long computation time. Hence, Unicorn desires an ALTO topology service that is able to provide only enough fine-grained topology information.

Several ALTO topology extension services including Path Vector [DRAFT-PV], Network Graphs [DRAFT-NETGRAPH] and RSA [DRAFT-RSA], [DRAFT-PM] are potential candidates for providing fine-grained abstract network formation to Unicorn. In addition, we propose that these services can also be used to provide information about computing and storage resources of different cluster/sites by viewing each computing node and storage node as a network element or abstract network element. For instance, the path vector service supports the capacity region query, which accepts multiple concurrent data flows as the input and returns the information of bottleneck resources, which could be a set of links, computing devices or storage devices, for the given set of concurrent flows. This information can be interpreted as a set of linear constraints for the multi-resource orchestrator, which can help data transfer and analytics requests better utilize multiple types of resources in different clusters.

8.3. Limitations of the MFRA Algorithm

The first limitation of the MFRA algorithm is computation overhead. The execution of MFRA involves solving linear programming problems repeatedly at every time slot. The overhead of computation time is acceptable for small sets of dataset transfer requests, but may increase significantly when handling large sets of requests, e.g.,

hundreds of transfer requests. Current efforts towards addressing this issue include exploring the feasibility of incremental computation of scheduling policies, and reducing the problem scale by finding the minimal equivalent set of constraints of the linear programming model. The latter approach can benefit substantially from the ALTO RSA service [DRAFT-RSA].

The second limitation is that the current version of MFRA does not involve dataset replica selection. Simply denoting the replica selection as a set of binary constraint will significantly increase the computation complexity of the scheduling process. Current efforts focus on finding efficient algorithms to make dataset replica selection.

9. Security Considerations

This document does not introduce any privacy or security issue not already present in the ALTO protocol.

10. IANA Considerations

This document does not define any new media type or introduce any new IANA consideration.

11. Acknowledgments

The authors thank discussions with Shenshen Chen, Linghe Kong, Xiao Lin and Xin Wang.

12. References

12.1. 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>>.

12.2. Informative References

[DRAFT-CC] Randriamasy, S., Yang, R., Wu, Q., Deng, L., and N. Schwan, "ALTO Cost Calendar", 2017, <<https://datatracker.ietf.org/doc/draft-ietf-alto-cost-calendar>>.

[DRAFT-DC]

Lee, Y., Bernstein, G., Dhody, D., and T. Choi, "ALTO Extensions for Collecting Data Center Resource Information", 2014, <<https://datatracker.ietf.org/doc/draft-lee-alto-ext-dc-resource/>>.

[DRAFT-MC]

Randriamasy, S., Roome, W., and N. Schwan, "Multi-Cost ALTO", 2017, <<https://datatracker.ietf.org/doc/draft-ietf-alto-multi-cost/>>.

[DRAFT-NETGRAPH]

Bernstein, G., Lee, Y., Roome, W., Scharf, M., and Y. Yang, "ALTO Topology Extensions: Node-Link Graphs", 2015, <<https://tools.ietf.org/html/draft-yang-alto-topology-06>>.

[DRAFT-PM]

Roome, W. and Y. Yang, "Extensible Property Maps for the ALTO Protocol", 2015, <<https://datatracker.ietf.org/doc/draft-roome-alto-unified-props-new/>>.

[DRAFT-PV]

Bernstein, G., Lee, Y., Roome, W., Scharf, M., and Y. Yang, "ALTO Extension: Abstract Path Vector as a Cost Mode", 2015, <<https://tools.ietf.org/html/draft-yang-alto-path-vector-01>>.

[DRAFT-RSA]

Gao, K., Wang, X., Yang, Y., and G. Chen, "ALTO Extension: A Routing State Abstraction Service Using Declarative Equivalence", 2015, <<https://datatracker.ietf.org/doc/draft-gao-alto-routing-state-abstraction/>>.

[DRAFT-SSE]

Roome, W. and Y. Yang, "ALTO Incremental Updates Using Server-Sent Events (SSE)", 2015, <<https://datatracker.ietf.org/doc/draft-ietf-alto-incr-update-sse/>>.

[HTCondor]

Thain, D., Tannenbaum, T., and M. Livny, "Distributed computing in practice: the Condor experience", 2005, <<http://dl.acm.org/citation.cfm?id=1064336>>.

[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

Qiao Xiang
Tongji/Yale University
51 Prospect Street
New Haven, CT
USA

Email: qiao.xiang@cs.yale.edu

Harvey Newman
California Institute of Technology
1200 California Blvd.
Pasadena, CA
USA

Email: newman@hep.caltech.edu

Greg Bernstein
Grotto Networking
Fremont, CA
USA

Email: gregb@grotto-networking.com

Haizhou Du
Tongji/Yale University
51 Prospect Street
New Haven, CT
USA

Email: duhaizhou@gmail.com

Kai Gao
Tsinghua University
30 Shuangqinglu Street
Beijing
China

Email: gaok12@mails.tsinghua.edu.cn

Azher Mughal
California Institute of Technology
1200 California Blvd.
Pasadena, CA
USA

Email: azher@hep.caltech.edu

Justas Balcas
California Institute of Technology
1200 California Blvd.
Pasadena, CA
USA

Email: justas.balcas@cern.ch

Jingxuan Jensen Zhang
Tongji University
4800 Cao'an Hwy
Shanghai 201804
China

Email: jingxuan.n.zhang@gmail.com

Y. Richard Yang
Tongji/Yale University
51 Prospect Street
New Haven, CT
USA

Email: yry@cs.yale.edu

ALTO WG
Internet-Draft
Intended status: Standards Track
Expires: September 14, 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
March 13, 2017

ALTO Extension: Path Vector Cost Mode
draft-yang-alto-path-vector-04.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 September 14, 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	4
2. Overview of Approach	5
2.1. Graph Model and Path Vector Data Format	5
2.2. Network Element Property Map Data Format	6
2.3. Flow-based Query Data Format	6
2.4. Routing State Abstraction Service	6
3. Changes Since Version -03	6
4. Use Case: Capacity Region for Multi-Flow Scheduling	6
5. Protocol Extensions	8
5.1. Cost Type	8
5.1.1. Cost Metric	8
5.1.2. Cost Mode: Path Vector	9
5.2. Network Element Name	9
5.3. Entity Domains	9
5.3.1. NE Domain	9
5.3.2. ANE Domain	10
5.4. (Filtered) Network Element Property Map Resource	10
5.4.1. Media Type	10
5.4.2. HTTP Method	10
5.4.3. Accept Input Parameters	10
5.4.4. Capabilities	11

5.4.5.	Uses	11
5.4.6.	Response	11
5.5.	Cost Map Extensions	11
5.5.1.	Media Types	11
5.5.2.	HTTP Method	11
5.5.3.	Accept Input Parameters	11
5.5.4.	Capabilities	11
5.5.5.	Uses	12
5.5.6.	Response	12
5.6.	Filtered Cost Map Extensions	12
5.6.1.	Media Type	12
5.6.2.	HTTP Method	12
5.6.3.	Accept Input Parameters	12
5.6.4.	Capabilities	13
5.6.5.	Uses	14
5.6.6.	Response	14
5.7.	Endpoint Cost Service Extensions	14
5.7.1.	Media Type	14
5.7.2.	HTTP Method	15
5.7.3.	Accept Input Parameters	15
5.7.4.	Capabilities	15
5.7.5.	Uses	16
5.7.6.	Response	16
5.8.	Optional: RSA Extension	16
5.8.1.	Media Type	16
5.8.2.	HTTP Method	16
5.8.3.	Accept Input Parameters	17
5.8.4.	Capabilities	17
5.8.5.	Response	17
6.	Examples	17
6.1.	Information Resource Directory Example	17
6.2.	Network Element Property Map Example	19
6.3.	Filtered Cost Map Example #1	19
6.4.	Filtered Cost Map Example #2	21
6.5.	Endpoint Cost Map Example #1	22
6.6.	Endpoint Cost Map Example #2	23
7.	Compatibility	24
7.1.	Compatibility with Legacy ALTO Clients/Servers	24
7.2.	Compatibility with Multi-Cost Extensions	25
7.3.	Compatibility with Incremental Update	25
8.	Design Decisions and Discussions	25
8.1.	Path Vector or Path Graph?	25
8.2.	Provide More General Calendar Extension?	26
9.	Security Considerations	26
10.	IANA Considerations	27
10.1.	ALTO Cost Mode Registry	27
10.2.	ALTO Cost Metric Registry	27
10.3.	ALTO Entity Domain Registry	27

11. Acknowledgments	28
12. References	28
12.1. Normative References	28
12.2. Informative References	28
Authors' Addresses	29

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 a 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 metrics 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. Third, flow scheduling can be conducted with several independent sets of flows. Using the cross product of "src" and "dst" lists will introduce a lot of redundancies.

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 family of cost types, which uses "path-vector" as cost mode and "ne" (network element) or "ane" (abstract network element) as cost metric. It also extends "Unified 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, and declares "pid-flows" and

"endpoint-flows" to support queries for multiple independent flow sets. This document also registers new domains, entity specification and properties in the ALTO Entity Domain Registry.

The document is organized as follows. Section 2 gives an overview of the path vector extension. Section 4 gives an example of flow scheduling to illustrate the need to introduce cost mode "path-vector" and new cost metrics. Section 5 specifies the new cost mode and cost metrics, new domain types and entity properties, new resource Network Element Property Map, and protocol extensions for (Filtered) Cost Maps and Endpoint Cost Services. Section 6 presents examples of Information Resources, requests and responses. Section 7 discusses the compatibility issues with some other proposed ALTO extensions. Section 8 lists several to-be-determined design decisions. Section 9 discusses about security and Section 10 discusses about IANA Considerations.

2. 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 Cost Map and Endpoint Cost Service defined in [RFC7285], and Unified Property Map defined in [I-D.roome-alto-unified-props].

2.1. Graph Model and Path Vector Data Format

In this document, the graph model presented to ALTO clients is represented by path vectors. A path vector between two entities (either PIDs or endpoints) is an array of Network Element Names, where each Network Element Name represents a network element (usually a link) in the path between the two entities.

A specific Network Element Name MUST represent the same network element in the same ALTO Network Element Property Map resource. Thus, ALTO clients can find the flows that share this specific network element by finding the source-destination pairs whose corresponding path vectors contain the Network Element Name.

The cost entries contained in an ALTO Cost Map or Endpoint Cost Map are formally defined in Section 11.2.3.6 [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.

An example can be found in Section 6.3.

2.2. Network Element Property Map Data Format

An ALTO client may need not know all attributes associated with all network elements. Thus, Network Element Property Map is provided so after an ALTO client obtains the path vectors, it can use Network Element Names to selectively query the associated attributes in the corresponding Network Element Property Map.

2.3. Flow-based Query Data Format

Flow scheduling may involve multiple sets of flows which have different source-destination combinations. Using source and destination lists can lead to a lot of redundancies. To allow more flexible specification of path vector requests, two new filter types for ReqFilteredCostMap and ReqEndpointCostMap are specified in this document.

2.4. Routing State Abstraction Service

For security and scalability considerations, this document specifies an optional feature called Routing State Abstraction (RSA). Routing State Abstraction feature compresses the original path vector information without loss of equivalence. A Routing State Abstraction compression feature MUST be applied only when a (Filtered) Cost Map or Endpoint Cost Service provides the path vector cost type with cost metric being "ane".

3. Changes Since Version -03

- o Define "ne" and "ane" as the only cost metrics of the cost mode "path-vector".
- o Define new entity domains for network property map and add the "availbw", "delay" property to these domains.
- o Use Unified Property service to query properties of network elements.
- o Augment the request schema of the (Filtered) Cost Map and Endpoint Cost Service.

4. 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 $F = \{f_1, f_2, \dots, f_{|F|}\}$. If routing is given, what the application can control is x_1, x_2, \dots ,

$x_{|F|}$, where x_i is the amount of traffic for flow i . Let $x = [x_1, \dots, x_{|F|}]$ 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.

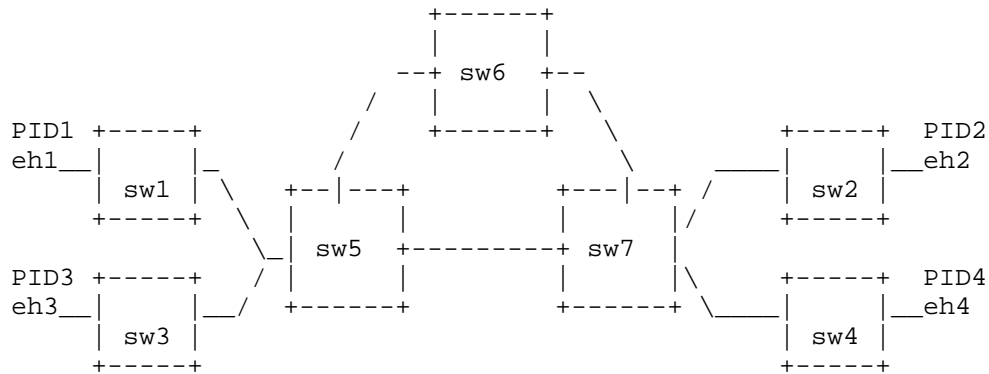


Figure 1: Raw Network Topology.

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 $eh1 \rightarrow eh2$, and $eh3 \rightarrow eh4$. 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:

- o 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.
- o 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. The path vector extension defined in this document resolves this issue.

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

5. Protocol Extensions

This section formally specifies the path vector extension.

5.1. Cost Type

The path vector extension defined in this document extends the Cost Types as specified in Section 6.1 of [RFC7285].

5.1.1. Cost Metric

This document specifies two new cost metrics: "ne" and "ane". Both cost metrics are of type CostMetric as defined in Section 10.6 of [RFC7285]. These cost metrics MUST NOT be used when the cost mode is not "path-vector" unless it is explicitly specified in a future extension. An ALTO server with path vector extension MUST support at least one of these two cost metrics.

Cost metric "ne": This cost metric MUST be encoded as the JSONString "ne". When cost metric is "ne", Network Element Names contained in the path vectors MUST be resource-specific. In this case, different path vector queries to the same (Filtered) Cost Map or

Endpoint Cost Service MUST have the same Network Element Property Map in the responses.

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 vector MUST be query-specific. In this case, different path vector queries to the same (Filtered) Cost Map or Endpoint Cost Service MAY have different Network Element Property Maps.

5.1.2. Cost Mode: Path Vector

This document extends the cost mode as defined in Section 10.5 of [RFC7285] by allowing an ALTO server to provide a new cost mode other than "numerical" and "ordinal". The path vector cost mode is of type CostMode 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 costs are JSONArrays of type NetworkElementName as specified in Section 5.2.

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

5.2. Network Element Name

This document also extends [RFC7285] with a new basic data type: Network Element Name. A Network Element Name is of type EntityAddr as defined in Section 2.3 of [I-D.roome-alto-unified-props] and is encoded as a JSONString. A Network Element Name MUST be an EntityAddr either of the NE domain (Section 5.3.1) or of the ANE domain (Section 5.3.2).

5.3. Entity Domains

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

5.3.1. NE Domain

5.3.1.1. Domain Name

ne

5.3.1.2. Domain-Specific Entity Addresses

Entity address of NE domain MUST be encoded as a JSONString with the same format as PID name defined in Section 10.1 of [RFC7285].

5.3.2. ANE Domain

5.3.2.1. Domain Name

ane

5.3.2.2. Domain-Specific Entity Addresses

The same as Section 5.3.1.2.

5.4. (Filtered) Network Element Property Map Resource

This document extends the base ALTO protocol with a new (filtered) resource: (Filtered) Network Element Property Map.

A Network Element Property Map MUST be a Property Map as defined in Section 4 of [I-D.roome-alto-unified-props] and a Filtered Network Element Property Map MUST be a Filtered Property Map as defined in Section 5 of [I-D.roome-alto-unified-props].

5.4.1. Media Type

The media type of a (Filtered) Network Element Property Map resource is "application/alto-propmap+json".

5.4.2. HTTP Method

An ALTO Network Element Property Map is requested using the HTTP GET method.

An ALTO Filtered Network Element Property Map is requested using the HTTP POST method.

5.4.3. Accept Input Parameters

Network Element Property Map does not accept any input parameters.

The input parameters of a Filtered Network Element Property Map MUST conform to the format in Section 5.3 of [I-D.roome-alto-unified-props]. The EntityAddr in the request MUST have the same format as Network Element Name specified in Section 5.2.

5.4.4. Capabilities

A (Filtered) 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 either "ne" or "ane" but not both at the same time and the "prop-types" capability MUST contain property type "availbw".

5.4.5. Uses

None.

5.4.6. Response

The "vtag" field MUST be included in the "meta" field of a response to a (Filtered) Network Element Map, with the same format as defined in Section 11.2.1.6 of [RFC7285].

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 6.2.

5.5. Cost Map Extensions

5.5.1. Media Types

The same as Section 11.2.3.1 of [RFC7285].

5.5.2. HTTP Method

The same as Section 11.2.3.2 of [RFC7285].

5.5.3. Accept Input Parameters

The same as Section 11.2.3.3 of [RFC7285].

5.5.4. Capabilities

If a Cost Map resource supports the path vector extension defined in this document, its "cost-type-names" capability MUST have exactly one single cost type with the cost mode being "path-vector" and the cost metric being either "ne" or "ane", unless it is explicitly specified by a future extension.

5.5.5. Uses

The same as Section 11.2.3.5 of [RFC7285].

5.5.6. Response

The response has the same format as in Section 4.1.3 of [I-D.ietf-alto-multi-cost], except the follows.

- o If cost mode is "path-vector", the costs MUST be JSONArrays of Network Element Names.
- o If cost mode is "path-vector", the "meta" field MUST include the "nep-map" field, whose value is of type IRDResourceEntry as defined in Section 9.2.2 of [RFC7285] and represents the Filtered Network Element Property Map associated with this Cost Map as defined in Section 5.4. An ALTO server providing this resource MUST verify the following conditions are met:
 - o If cost metric of this Cost Map resource is "ne", the "nep-map" entry MUST have "domain-types" capability which includes domain name "ne".
 - o If cost metric of this Cost Map resource is "ane", the "nep-map" entry MUST have "domain-types" capability which includes domain name "ane".

ALTO servers MUST also verify the conditions in Section 5.1.1 are also met.

5.6. Filtered Cost Map Extensions

5.6.1. Media Type

The same as Section 11.3.2.1 of [RFC7285].

5.6.2. HTTP Method

The same as Section 11.3.2.2 of [RFC7285].

5.6.3. Accept Input Parameters

This document extends the ReqFilteredCostMap as follows:

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

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

pids: As defined in Section 11.3.2.3 of [RFC7285].

cost-type, multi-cost-types, testable-cost-types, constraints, or-constraints:

As defined in Section 4.1.2 of [I-D.ietf-alto-multi-cost]. A valid query MUST be considered a path vector query, either when "cost-type" of this query exists with "path-vector" cost mode, or when "multi-cost-types" exists and exact one cost type uses "path-vector" cost mode. For a path vector query, the path vector cost type MUST follow the definition in Section 5.1, otherwise the ALTO server SHOULD return an error with error code "E_INVALID_FIELD_VALUE". If "multi-cost-types" contains multiple path vector cost types, ALTO servers SHOULD return an error with the error code "E_INVALID_FIELD_VALUE". If a query is a path vector query and its "constraints" or "or-constraints" field is present, the "testable-cost-types" field MUST be explicitly specified and MUST NOT include any path vector cost type. If a path vector cost type is included, an ALTO server SHOULD return an error with the error code "E_INVALID_FIELD_VALUE".

An ALTO client MUST specify either "pids" or "pid-flows", but MUST NOT specify both in a single query.

5.6.4. Capabilities

A Filtered Cost Map with the path vector extension MAY have the "flow-based-filter" in its IRD capabilities.

```
object {  
  JSONString cost-type-names<1..*>;  
  [JSONBool cost-constraints;]  
  [JSONNumber max-cost-types;]  
  [JSONString testable-cost-type-names<1..*>;]  
  [JSONBool flow-based-filter;]  
} FilteredCostMapCapabilities;
```

flow-based-filter: If the value is true, the ALTO server allows ALTO clients to use "pid-flows" to query the Filtered Cost Map. If false or not present, ALTO clients MUST NOT include this field in the queries and the ALTO server SHOULD return an error with the error code "E_INVALID_FIELD_TYPE" for the queries including this field.

cost-type-names and cost-constraints: As defined in Section 11.3.2.4 of [RFC7285].

max-cost-types and testable-cost-type-names: As defined in Section 4.1.1 of [I-D.ietf-alto-multi-cost]. If a cost type with "path-vector" mode is included in "cost-type-names", either the "testable-cost-type-names" field is explicitly specified which MUST NOT include any path vector cost type, or the "testable-cost-type-names" field is empty where ALTO clients MUST interpret this as specified in Section 4.1.1 of [I-D.ietf-alto-multi-cost] except that the resource MUST NOT accept tests on any path vector cost type.

5.6.5. Uses

The same as Section 5.5.5.

5.6.6. Response

The response MUST have the same format as defined in Section 5.5.6 with the supplement that if a query uses the field "pid-flows", the response MUST still conform to the CostMapData format defined in Section 11.2.3.6 of [RFC7285]. Examples can be found in Section 6.3.

5.7. Endpoint Cost Service Extensions

5.7.1. Media Type

The same as Section 11.5.1.1 of [RFC7285].

5.7.2. HTTP Method

The same as Section 11.5.1.2 of [RFC7285].

5.7.3. Accept Input Parameters

This document extends the input parameters of Endpoint Cost Service as follows:

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

endpoint-flows: A list of source-destination endpoint pairs for which path costs are to be returned.

endpoints: As defined in Section 11.5.1.3 of [RFC7285].

cost-type, multi-cost-types, testable-cost-types, constraints, or-constraints:

As defined in Section 5.6.3.

ALTO clients MUST specify either "endpoints" or "endpoint-flows", but MUST NOT specify both in the same query.

5.7.4. Capabilities

This document defines EndpointCostMapCapabilities the same as FilteredCostMapCapabilities in Section 5.6.4.

If the value of capability flow-based-filter is true, the ALTO server MUST be able to process "endpoint-flows" in a query. If the value is false or not present, ALTO clients MUST assume the "endpoint-flows" is not supported and ALTO servers SHOULD return an error with the error code "E_INVALID_FIELD_TYPE" if "endpoint-flows" is included in the query.

5.7.5. Uses

The same as Section 11.5.1.5 of [RFC7285].

5.7.6. Response

The response has the same format as in Section 4.2.3 of [I-D.ietf-alto-multi-cost] for compatibility, except the follows.

- o If cost mode is "path-vector", the costs MUST be JSONArrays of Network Element Names.
- o If cost mode is "path-vector", the "meta" field MUST include the "nep-map" field, whose value is of type IRDResourceEntry as defined in Section 9.2.2 of [RFC7285] and represents the Filtered Network Element Property Map associated with this Endpoint Cost Service as defined in Section 5.4. An ALTO server providing this resource MUST verify the following conditions are met:
 - o If cost metric of this Endpoint Cost Service is "ne", the "nep-map" entry MUST have "domain-types" capability which includes domain name "ne".
 - o If cost metric of this Endpoint Cost Service is "ane", the "nep-map" entry MUST have "domain-types" capability which includes domain name "ane".

ALTO servers MUST also verify the conditions in Section 5.1.1 are also met.

Examples can be found in Section 6.5.

5.8. Optional: RSA Extension

5.8.1. Media Type

RSA extension MUST NOT change media types of (Filtered) Cost Map resource or Endpoint Cost Service.

5.8.2. HTTP Method

RSA extension MUST NOT change HTTP method for (Filtered) Cost Map or Endpoint Cost Service.

5.8.3. Accept Input Parameters

RSA extension SHOULD NOT change the input parameters of a Filtered Cost Map or an Endpoint Cost Service unless it is explicitly specified by a future extension.

5.8.4. Capabilities

If a (Filtered) Cost Map or an Endpoint Cost Service supports RSA extension, the "cost-type-names" MUST have one cost type with "path-vector" cost mode and "ane" cost metric, and ALTO clients MUST ignore this field when no such path vector cost type exists. The resource/service MUST also have the field "rsa" explicitly specified to "true" in the "capabilities" field. If the "rsa" field has a value of "false" or is not present, ALTO clients MUST assume this resource/service does not provide RSA compression.

An example can be found in Section 6.1.

5.8.5. Response

RSA extension SHOULD NOT change the output of a (Filtered) Cost Map or an Endpoint Cost Service unless it is explicitly specified in a future extension.

6. Examples

6.1. Information Resource Directory Example

Here is an example of an ALTO server's Information Resource Directory.

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

```

        "cost-metric": "routingcost"
    },
    "num-delay": {
        "cost-mode": "numerical",
        "cost-metric": "delay"
    },
    "num-availbw": {
        "cost-mode": "numerical",
        "cost-metric": "availbw"
    }
}
},
"resource": {
    "default-network-map": {
        "uri": "http://alto.example.com/networkmap",
        "media-type": "application/alto-networkmap+json"
    },
    ... Filtered Cost Map Resource ...

    "filtered-multi-cost-map1": {
        "uri": "http://alto.example.com/costmap/multi/filtered1",
        "media-type": "application/alto-costmap+json",
        "accepts": "application/alto-costmapfilter+json",
        "uses": ["default-network-map"],
        "capabilities": {
            "max-cost-types": 3,
            "cost-type-names": ["pv-ne", "num-availbw", "num-hopcount"],
            "testable-cost-types-names": ["num-availbw", "num-hopcount"]
        }
    },

    "filtered-multi-cost-map2": {
        "uri": "http://alto.example.com/costmap/multi/filtered2",
        "media-type": "application/alto-costmap",
        "accepts": "application/alto-costmapfilter+json",
        "uses": ["default-network-map"],
        "capabilities": {
            "max-cost-types": 2,
            "cost-type-names": ["pv-ne", "num-routingcost", "num-delay"],
            "cost-constraints": true
        }
    }
}
... Endpoint Cost Map Resource ...

"default-endpoint-cost-map": {
    "uri": "http://alto.example.com/endpointcost/lookup/ne-ane",
    "media-type": "application/alto-endpointcostmap+json",

```



```

    "accepts": "application/alto-endpointcostparams+json",
    "capabilities": {
      "rsa": true,
      "max-cost-types": 4,
      "cost-type-names": ["pv-ne", "pv-ane", "num-routingcost", "num-hopcount"]
    },
    "testable-cost-types-names": ["num-hopcount", "num-routingcost"]
  }
}

```

6.2. Network Element Property Map Example

```

POST /propmap/lookup/nep-map 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

```

```

{
  "entities" : [ "ne:ne12",
                 "ne:ne23",
                 "ne:ne34" ],
  "properties" : [ "availbw", "delay" ]
}

```

```

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

```

```

{
  "meta": {
    "vtag": {
      "resource-id": "default-network-element-prop-map",
      "tag": "babbc14521772381472bffff627813909875dd"
    }
  }
  "property-map": {
    "ne:ne12": { "availbw": "90", "delay": "30" },
    "ne:ne23": { "availbw": "80", "delay": "15" },
    "ne:ne34": { "availbw": "70", "delay": "25" }
  }
}

```

6.3. Filtered Cost Map Example #1

Assume that the available bandwidth between PID1 and PID3 is less than 50.

```
POST /costmap/multi/filtered1 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
```

```
{
  "cost-type": {
    "cost-mode": "path-vector",
    "cost-metric": "ne"
  },
  "testable-cost-types": [
    { "cost-mode": "numerical", "cost-metric": "availbw" },
    { "cost-mode": "numerical", "cost-metric": "hopcount" }
  ],
  "or-constraints": [
    ["[0] ge 50", "[1] le 100"]
  ],
  "pid-flows": [
    { "src": "PID1", "dst": "PID2" },
    { "src": "PID1", "dst": "PID3" },
    { "src": "PID3", "dst": "PID4" }
  ]
}
```

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

```
{
  "meta": {
    "dependent-vtags": [
      {
        "resource-id": "my-default-network-map",
        "tag": "75ed013b3cb58f896e839582504f622838ce670f"
      }
    ],
    "cost-type": {
      "cost-mode": "path-vector",
      "cost-metric": "ne"
    },
    "nep-map": {
      "uri": "http://alto.example.com/propmap/lookup/nep-map",
      "media-type": "application/alto-promppmap+json",
      "accepts": "application/alto-propmapparams+json",
      "capabilities": {
        "domain-types": ["ne"],
        "prop-types": ["availbw", "delay"]
      }
    }
  }
}
```

```

    }
  },
  "cost-map": {
    "PID1": { "PID2": [ "ne:ne12", "ne:ne23" ] },
    "PID3": { "PID4": [ "ne:ne23", "ne:ne34" ] }
  }
}

```

6.4. Filtered Cost Map Example #2

Assume that the delay between PID1 and PID2 is greater than 80.

```

POST /costmap/multi/filtered2 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": "ne" },
    { "cost-mode": "numerical", "cost-metric": "delay" }
  ],
  "testable-cost-types": [
    { "cost-mode": "numerical", "cost-metric": "delay" }
  ],
  "constraints": [ "[0] le 80" ],
  "pid-flows": [
    { "src": "PID1", "dst": "PID2" },
    { "src": "PID1", "dst": "PID3" },
    { "src": "PID3", "dst": "PID4" }
  ]
}

```

```

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

```

```

{
  "meta": {
    "dependent-vtags": [
      {
        "resource-id": "my-default-network-map",
        "tag": "75ed013b3cb58f896e839582504f622838ce670f"
      }
    ],
    "cost-type": {},
  }
}

```

```

    "multi-cost-types": [
      { "cost-mode": "path-vector", "cost-metric": "ne" },
      { "cost-mode": "numerical", "cost-metric": "delay" }
    ],
    "nep-map": {
      "uri": "http://alto.example.com/propmap/lookup/nep-map",
      "media-type": "application/alto-prompmap+json",
      "accepts": "application/alto-propmapparams+json",
      "capabilities": {
        "domain-types": [ "ne" ],
        "prop-types": [ "availbw", "delay" ]
      }
    },
    "cost-map": {
      "PID1": { "PID3": [ [ "ne:ne23", "ne:ne45" ], 60 ] },
      "PID3": { "PID4": [ [ "ne:ne23", "ne:ne34" ], 40 ] }
    }
  }
}

```

6.5. Endpoint Cost Map Example #1

Assume that the delay between ipv4:203.0.113.45 and ipv4:198.51.100.34 is greater than 100.

```

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

```

```

{
  "cost-type": {
    "cost-mode": "path-vector",
    "cost-metric": "ne"
  },
  "testable-cost-types": [
    {
      "cost-mode": "numerical",
      "cost-metric": "delay"
    }
  ],
  "constraints": [
    "[0] le 100"
  ],
  "endpoint-flows": [
    { "src": "ipv4:192.0.2.2", "dst": "ipv4:192.0.2.89" },
    { "src": "ipv4:192.0.2.2", "dst": "ipv4:198.51.100.34" },
  ]
}

```

```

    { "src": "ipv4:203.0.113.45", "dst": "ipv4:198.51.100.34" }
  ]
}

```

HTTP/1.1 200 OK

Content-Length: [TBD]

Content-Type: application/alto-endpointcost+json

```

{
  "meta": {
    "cost-type": {
      "cost-mode": "path-vector",
      "cost-metric": "ne"
    },
    "nep-map": {
      "uri": "http://alto.example.com/propmap/lookup/nep-map",
      "media-type": "application/alto-promppmap+json",
      "accepts": "application/alto-propmapparams+json",
      "capabilities": {
        "domain-types": ["ne"],
        "prop-types": ["availbw", "delay"]
      }
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89": [ "ne:ne12", "ne:ne23" ],
      "ipv4:198.51.100.34": [ "ne:ne12", "ne:ne24", "ne:ne45" ]
    }
  }
}

```

6.6. Endpoint Cost Map Example #2

POST /endpointcost/lookup/ne-ane 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": "delay"
  }],
  "endpoint-flows": [
    { "src": "ipv4:192.0.2.2", "dst": "ipv4:192.0.2.89" },
    { "src": "ipv4:192.0.2.2", "dst": "ipv4:198.51.100.34" },
    { "src": "ipv4:203.0.113.45", "dst": "ipv4:198.51.100.34" }
  ]
}

HTTP/1.1 200 OK
Content-Length: [TBD]
Content-Type: application/alto-endpointcost+json
{
  "meta": {
    "cost-type": {},
    "multi-cost-types": [
      { "cost-mode": "path-vector", "cost-metric": "ane" },
      { "cost-mode": "numerical", "cost-metric": "delay" }
    ],
    "nep-map": {
      "uri": "http://alto.example.com/propmap/lookup/nep-map/31415926",
      "media-type": "application/alto-promppmap+json",
      "accepts": "application/alto-propmapparams+json",
      "capabilities": {
        "domain-types": ["ane"],
        "prop-types": ["availbw", "delay"]
      }
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89": [ ["ane:ane12", "ane:ane23"], 45 ],
      "ipv4:198.51.100.34": [ ["ane:ane12", "ane:ane24"], 90 ]
    },
    "ipv4:203.0.113.45": {
      "ipv4:198.51.100.34": [ ["ane:ane34"], 120 ]
    }
  }
}

```

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 extension SHOULD be fully compatible with Multi-Cost extensions.

7.3. Compatibility with Incremental Update

There is no compatibility issue with incremental update extension.

8. Design Decisions and Discussions

8.1. 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.

8.2. 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 a new ALTO service, 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. 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.

10. IANA Considerations

10.1. 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.

10.2. ALTO Cost Metric Registry

Two new cost metrics need to be registered in the "ALTO Cost Metric Registry", listed in Table 1.

Identifier	Intended Semantics
ne	See Section 5.1.1
ane	See Section 5.1.1

Table 1: ALTO Cost Metrics

10.3. ALTO Entity Domain Registry

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

Identifier	Entity Address Encoding	Hierarchy & Inheritance
ne	See Section 5.3.1.2	None
ane	See Section 5.3.2.2	None

Table 2: ALTO Entity Domain

11. Acknowledgments

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

12. References

12.1. 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>>.

12.2. 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, W., 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-05 (work in progress), February 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