ALTO WG
Internet-Draft
Intended status: Standards Track
Expires: 30 August 2022

W. Roome
S. Randriamasy
Nokia Bell Labs
Y. Yang
Yale University
J. Zhang
Tongji University
K. Gao
Sichuan University
26 February 2022

An ALTO Extension: Entity Property Maps
draft-ietf-alto-unified-props-new-23

Abstract

This document specifies an extension to the base Application-Layer Traffic Optimization (ALTO) protocol that generalizes the concept of "endpoint properties", which were so far tied to IP addresses, to entities defined by a wide set of objects. Further, these properties are presented as maps, similar to the network and cost maps in the base ALTO protocol. While supporting the endpoints and related endpoint property service defined in RFC7285, the ALTO protocol is extended in two major directions. First, from endpoints restricted to IP addresses to entities covering a wider and extensible set of objects; second, from properties on specific endpoints to entire entity property maps. These extensions introduce additional features allowing entities and property values to be specific to a given information resource. This is made possible by a generic and flexible design of entity and property types.

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 30 August 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 4
1.1. Terminology and notation 6
2. Requirements Language 6
3. Basic Features of the Entity Property Map Extension 7
3.1. Entity 7
3.2. Entity Domain 8
3.2.1. Entity Domain Type 8
3.2.2. Entity Domain Name 8
3.3. Entity Property Type 9
3.4. New Information Resource and Media Type: ALTO Property Map 10
4. Advanced Features of the Entity Property Map Extension . . . 10
4.1. Entity Identifier and Entity Domain Name 10
4.2. Resource-Specific Entity Domain Name 11
4.3. Resource-Specific Entity Property Value 12
4.4. Entity Hierarchy and Property Inheritance 13
4.4.1. Entity Hierarchy 13
4.4.2. Property Inheritance 14
4.4.3. Property Value Unicity 14
4.5. Supported Properties on Entity Domains in Property Map Capabilities 15
4.6. Defining Information Resource for Resource-Specific Entity Domains 15
4.6.1. Defining Information Resource and its Media Type . . 16
4.6.2. Examples of Defining Information Resources and Their Media Type 17
4.7. Defining Information Resource for Resource-Specific Property Values 18
5. Protocol Specification: Basic Data Types 19
5.1. Entity Domain 19
5.1.1. Entity Domain Type 19
5.1.2. Entity Domain Name 20

5.1.3. Entity Identifier 22
5.1.4. Hierarchy and Inheritance 23
5.2. Entity Property 23
5.2.1. Entity Property Type 23
5.2.2. Entity Property Name 24
5.2.3. Format for Entity Property Value 25
6. Entity Domain Types Defined in this Document 25
6.1. Internet Address Domain Types 25
6.1.1. Entity Domain Type: IPv4 25
6.1.2. Entity Domain Type: IPv6 26
6.1.3. Hierarchy and Inheritance of Internet Address Domains 26
6.1.4. Defining Information Resource Media Type for domain types IPv4 and IPv6 28
6.2. Entity Domain Type: PID 28
6.2.1. Entity Domain Type Identifier 28
6.2.2. Domain-Specific Entity Identifiers 28
6.2.3. Hierarchy and Inheritance 28
6.2.4. Defining Information Resource Media Type for Domain Type PID 28
6.2.5. Relationship To Internet Addresses Domains 29
6.3. Internet Address Properties vs. PID Properties 29
7. Property Map 29
7.1. Media Type 30
7.2. HTTP Method 30
7.3. Accept Input Parameters 30
7.4. Capabilities 30
7.5. Uses 30
7.6. Response 30
8. Filtered Property Map 32
8.1. Media Type 32
8.2. HTTP Method 32
8.3. Accept Input Parameters 32
8.4. Capabilities 34
8.5. Uses 34
8.6. Filtered Property Map Response 34
8.7. Entity Property Type Defined in This Document 36
8.7.1. Entity Property Type: pid 36
9. Impact on Legacy ALTO Servers and ALTO Clients 37
9.1. Impact on Endpoint Property Service 37
9.2. Impact on Resource-Specific Properties 37
9.3. Impact on Other Properties 37
10. Examples 38
10.1. Network Map 38
10.2. Property Definitions 38
10.3. Information Resource Directory (IRD) 39
10.4. Full Property Map Example 42
10.5. Filtered Property Map Example #1 43

10.6. Filtered Property Map Example #2 44
10.7. Filtered Property Map Example #3 45
10.8. Filtered Property Map Example #4 47
10.9. Filtered Property Map for ANEs Example #5 47
11. Security Considerations 48
12. IANA Considerations 50
12.1. application/alto-propmap+json Media Type 51
12.2. alto-propmapparams+json Media Type 52
12.3. ALTO Entity Domain Type Registry 53
12.3.1. Consistency Procedure between ALTO Address Type Registry and ALTO Entity Domain Type Registry 54
12.3.2. ALTO Entity Domain Type Registration Process 56
12.4. ALTO Entity Property Type Registry 57
13. Acknowledgments 58
14. References 59
14.1. Normative References 59
14.2. Informative References 60
Appendix A. Features introduced with the Entity Property Maps extension 61
Authors' Addresses 62

1. Introduction

The ALTO protocol [RFC7285] introduces the concept of "properties" attached to "endpoint addresses". It also defines the Endpoint Property Service (EPS) to allow ALTO clients to retrieve those properties. While useful, the EPS, as defined in [RFC7285], has at least three limitations that are further elaborated hereafter.

First, the EPS allows properties to be associated with only endpoints that are identified by individual communication addresses like IPv4 and IPv6 addresses. It is reasonable to think that collections of endpoints or Provider-Defined Identifiers (PIDs), may also have properties. Furthermore, recent ALTO use cases show that properties of entities such as abstracted network elements as defined in [I-D.ietf-alto-path-vector] are also useful. However, the current EPS is restricted to individual endpoints and cannot be applied to those entities.

Second, the EPS only allows endpoints identified by global communication addresses. However, an endpoint address may be a local IP address or an anycast IP address that may not be globally unique. Additionally, an entity such as a PID may have an identifier that is not globally unique. That is, a same PID may be used in multiple network maps, while in each network map, this PID points to a different set of addresses.

Third, in section 11.4 of [RFC7285], the EPS is only defined as a POST-mode service. ALTO clients must request the properties for an explicit set of endpoint addresses. By contrast, [RFC7285], in section 11.2.3, defines a GET-mode cost map resource which returns all available costs, so an ALTO Client can retrieve a full set of costs once, and then process cost lookups without querying the ALTO server. [RFC7285] does not define a similar service for endpoint properties. At first, a map of endpoint properties might seem impractical, because it could require enumerating the property value for every possible endpoint. In particular, the number of endpoint addresses involved by an ALTO server can be quite large. To avoid enumerating a large number of endpoint addresses inefficiently, the ALTO server might define properties for a sufficiently large subset of endpoints and uses an aggregation representation to reference endpoints to allow efficient enumeration. This is particularly true if blocks of endpoint addresses with a common prefix have the same value for a property. Entities in other domains may very well allow aggregated representation and hence be enumerable as well.

To address these three limitations, this document specifies an ALTO protocol extension for defining and retrieving ALTO properties:

* The first limitation is addressed by introducing a generic concept called ALTO Entity, which generalizes an endpoint and may represent a PID, a network element, a cell in a cellular network, an abstracted network element [I-D.ietf-alto-path-vector], or other physical or logical objects involved in a network topology. Each entity is included in a collection called an ALTO entity domain. Since each ALTO entity domain includes only one type of entities, each entity domain can be classified by the type of enclosed entities.

* The second limitation is addressed by using resource-specific entity domains. A resource-specific entity domain contains entities that are defined and identified with respect to a given ALTO information resource, which provides scoping. For example, an entity domain containing PIDs is identified with respect to the network map in which these PIDs are defined. Likewise, an entity domain containing local IP addresses may be defined with respect to a local network map.

* The third limitation is addressed by defining two new types of ALTO information resources: Property Map (Section 7) and Filtered Property Map (Section 8). The former is a resource that is requested using the HTTP GET method, returns the property values for all entities in one or more entity domains, and is analogous to a network map or a cost map in Section 11.2 of [RFC7285]. The latter is a resource that that is requested using the HTTP POST

method, returns the values for sets of properties and entities requested by the client, and is analogous to a filtered network map or a filtered cost map.

The Entity Property Maps extension described in this document introduces a number of features that are summarized in Appendix A, where Table 4 lists the features and references the sections in this document that give their high-level and their normative description.

The protocol extension defined in this document is augmentable. New entity domain types can be defined without revising the present specification. Similarly, new cost metrics and new endpoint properties can be defined in other documents without revising the protocol specification defined in [RFC7285].

1.1. Terminology and notation

This document uses the following terms and abbreviations, that will be further defined in the document. While this document introduces the feature "entity property map", it will use both the term "property map" and "entity property map" to refer to this feature.

* Transaction: A request/response exchange between an ALTO client and an ALTO server.

* Client: When used with a capital "C", this term refers to an ALTO client. Note that expressions "ALTO client", "ALTO Client" and "Client" are equivalent.

* Server: When used with a capital "S", this term refers to an ALTO server. Note that expressions "ALTO server", "ALTO Server" and "Server" are equivalent.

* EPS: An abbreviation for Endpoint Property Service.

This document uses the semi-formal notation defined in Section 8.2 of [RFC7285].

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. Basic Features of the Entity Property Map Extension

This section gives a high-level overview of the basic features involved in ALTO Entity Property Maps. It assumes the reader is familiar with the ALTO protocol [RFC7285]. The purpose of this extension is to convey properties on objects that extend ALTO Endpoints and are called ALTO Entities, or entities for short.

The features introduced in this section can be used as standalone. However, in some cases, these features may depend on particular information resources and need to be defined with respect to them. To this end, Section 4 introduces additional features that extend the ones presented in the present section.

3.1. Entity

The concept of an ALTO Entity generalizes the concept of an ALTO Endpoint defined in Section 2.1 of [RFC7285]. An entity is an object that can be an endpoint defined by its network address, but can also be an object that has a defined mapping to a set of one or more network addresses or an object that is not even related to any network address. Thus, whereas all endpoints are entities, not all entities are endpoints.

Examples of entities are:

* an ALTO endpoint that represents an application or a host identified by a communication address (e.g., an IPv4 or IPv6 address) in a network,

* a PID, defined in [RFC7285], that has a provider defined human-readable identifier specified by an ALTO network map, which maps a PID to a set of IPv4 and IPv6 addresses,

* an Autonomous System (AS), that has an AS number (ASN) as its identifier and maps to a set of IPv4 and IPv6 addresses, that is defined in [I-D.ietf-alto-cdni-request-routing-alto],

* a country with a code specified in [ISO3166-1], to which applications such as CDN providers associate properties and capabilities, that is defined in [I-D.ietf-alto-cdni-request-routing-alto],

* a TCP/UDP network flow, that is identified by a TCP/UDP 5-tuple specifying its source and destination addresses and port numbers, and the IP protocol,

* a routing element, that is specified in [RFC7921] and is associated with routing capabilities information,

* an abstract network element, that is specified in [I-D.ietf-alto-path-vector] and that represents an abstraction of a network part such as a router, one or more links, a network domain or their aggregation.

Some of the example entities listed above have already been documented as ALTO entities. The other examples are provided for illustration as potential entities.

3.2. Entity Domain

An entity domain defines a set of entities of the same semantic type. An entity domain is characterized by a type and identified by a name.

In this document, an entity is owned by exactly one entity domain name. An entity identifier points to exactly one entity. If two entities in two different entity domains refer to the same physical or logical object, they are treated as different entities. For example, if an end host has both an IPv4 and an IPv6 address, these two addresses will be treated as two entities, defined respectively in the "ipv4" and "ipv6" entity domains.

3.2.1. Entity Domain Type

The type of an entity domain type defines the semantics of a type of entity. Entity domain types can be defined in different documents. For example: the present document defines entity domain types "ipv4", "ipv6" and "pid" in Section 6.1 and Section 6.2. The entity domain type "ane", that defines Abstract Network Elements (ANEs), is introduced in [I-D.ietf-alto-path-vector]. The entity domain type that defines country codes is introduced in [I-D.ietf-alto-cdni-request-routing-alto]. An entity domain type MUST be registered at the IANA, as specified in Section 12.3.2 .

3.2.2. Entity Domain Name

In this document, the identifier of an entity domain is mostly called "entity domain name". The identifier of an entity domain is defined in the scope of an ALTO server. An entity domain identifier can sometimes be identical to the identifier of its relevant entity domain type. This is the case when the entities of a domain have an identifier that points to the same object throughout all the information resources of the Server that provide entity properties for this domain. For example, a domain of type "ipv4" containing entities that are identified by a public IPv4 address can be named

"ipv4" because its entities are uniquely identified by all the Server resources.

In some cases, the name of an entity domain cannot be simply its entity domain type. Indeed, for some domain types, entities are defined relative to a given information resource. This is the case for entities of domain type "pid". A PID is defined relative to a network map. For example, an entity "mypid10" of domain type "pid" may be defined in a given network map and be undefined in other network maps. Or "mypid10" may even be defined in two different network maps and map, in each of these network maps, to a different set of endpoint addresses. In this case, naming an entity domain only by its type "pid" does not guarantee that its set of entities is owned by exactly one entity domain.

Section 4.2 and Section 5.1.2 describe how a domain is uniquely identified, across the ALTO server, by a name that associates the domain type and the related information resource.

3.3. Entity Property Type

An entity property defines a property of an entity. This is similar to the endpoint property defined in Section 7.1 of [RFC7285]. An entity property can convey either network-aware or network-agnostic information. Similar to an entity domain, an entity property is characterized by a type and identified by a name. An entity property type MUST be registered at the IANA, as specified in Section 12.4.

Below are listed some examples with real and fictitious entity domain and property names:

* an entity in the "ipv4" domain type may have a property whose value is an Autonomous System (AS) number indicating the AS to which this IPv4 address belongs and another property named "countrycode" indicating a country code mapping to this address,

* an entity identified by its country code in the entity domain type "countrycode", defined in [I-D.ietf-alto-cdni-request-routing-alto] may have a property indicating what delivery protocol is used by a CDN,

* an entity in the "netmap1.pid" domain may have a property that indicates the central geographical location of the endpoints it includes.

It should be noted that some identifiers may be used for both an entity domain type and a property type. For example:

* the identifier "countrycode" may point to both the entity domain type "countrycode" and the fictitious property type "countrycode".

* the identifier "pid" may point to both the entity domain type "pid" and the property type "pid".

Likewise, the same identifier may point to both a domain name and a property name. For example: the identifier "netmap10.pid" may point to either the domain defined by the PIDs of network map "netmap10" or to a property that returns, for an entity defined by its IPv4 address, the PID of netmap10 that contains this entity. Such cases will be further explained in Section 4.

3.4. New Information Resource and Media Type: ALTO Property Map

This document introduces a new ALTO information resource named Property Map. An ALTO property map provides a set of properties on one or more sets of entities. A property may apply to different entity domain types and names. For example, an ALTO property map may define the "ASN" property for both "ipv4" and "ipv6" entity domains.

The present extension also introduces a new media type.

This document uses the same definition of an information resource as Section 9.1 of [RFC7285]. ALTO uses media types to uniquely indicate the data format used to encode the content to be transmitted between an ALTO server and an ALTO client in the HTTP entity body. In the present case, an ALTO property map resource is defined by the media type "application/alto-propmap+json".

A Property Map can be queried as a GET-mode resource, thus conveying all properties on all entities indicated in its capabilities. It can also be queried as a POST-mode resource, thus conveying a selection of properties on a selection of entities.

4. Advanced Features of the Entity Property Map Extension

This section gives a high-level overview of the advanced features involved in ALTO Entity Property Maps. Most of these features are defined to extend the ones defined in Section 3.

4.1. Entity Identifier and Entity Domain Name

In [RFC7285], an endpoint has an identifier that is explicitly associated with the "ipv4" or "ipv6" address domain. Examples are "ipv4:192.0.2.14" and "ipv6:2001:db8::12".

In this document, example IPv4 and IPv6 addresses and prefixes are taken from the address ranges reserved for documentation by [RFC5737] and [RFC3849].

In this document, an entity must be owned by exactly one entity domain name and an entity identifier must point to exactly one entity. To ensure this, an entity identifier is explicitly attached to the name of its entity domain and an entity domain type characterizes the semantics and identifier format of its entities.

The encoding format of an entity identifier is further specified in Section 5.1.3 of this document.

For instance:

* if an entity is an endpoint with IPv4 address "192.0.2.14", its identifier is associated with entity domain name "ipv4" and is "ipv4:192.0.2.14",

* if an entity is a PID named "mypid10" in network map resource "netmap2", its identifier is associated with entity domain name "netmap2.pid" and is "netmap2.pid:mypid10".

4.2. Resource-Specific Entity Domain Name

Some entities are defined and identified uniquely and globally in the context of an ALTO server. This is the case for instance when entities are endpoints that are identified by a reachable IPv4 or IPv6 address. The entity domain for such entities can be globally defined and named "ipv4" or "ipv6". Those entity domains are called resource-agnostic entity domains in this document, as they are not associated with any specific ALTO information resources.

Some other entities and entity types are only defined relative to a given information resource. This is the case for entities of domain type "pid", that can only be understood with respect to the network map where they are defined. For example, a PID named "mypid10" may be defined to represent a set S1 of IP addresses in a network map resource named "netmap1". Another network map "netmap2" may use the same name "mypid10" and define it to represent another set S2 of IP addresses. The identifier "pid:mypid10" may thus point to different objects because the information on the originating information resource is lost.

To solve this ambiguity, the present extension introduces the concept of resources-specific entity domain. This concept applies to domain types where entities are defined relative to a given information resource. It can also apply to entity domains that are defined locally, such as local networks of objects identified with a local IPv4 address.

In such cases, an entity domain type is explicitly associated with an identifier of the information resource where these entities are defined. Such an information resource is referred to as the "specific information resource". Using a resource-aware entity domain name, an ALTO property map can unambiguously identify distinct entity domains of the same type, on which entity properties may be queried. Examples of resource-specific entity domain names may look like: "netmap1.pid" or "netmap2.pid". Thus, a name association such as "netmap1.pid:mypid10" and "netmap2.pid:mypid10" allows to distinguish the two abovementioned PIDs that are both named "mypid10" but in two different resources, "netmap1" and "netmap2".

An information resource is defined in the scope of an ALTO Server and so is an entity domain name. The format of a resource-specific entity domain name is further specified in Section 5.1.2.

4.3. Resource-Specific Entity Property Value

Like entity domains, some types of properties are defined relative to an information resource. That is, an entity may have a property of a given type, whose values are associated to different information resources.

For example, suppose entity "192.0.2.34" defined in the "ipv4" domain has a property of type "pid", whose value is the PID to which address "192.0.2.34" is attached in a network map. The mapping of network addresses to PIDs is specific to a network map and probably different from one network map resource to another one. Thus, if a property "pid" is defined for entity "192.0.2.34" in two different network maps "netmap1" and "netmap2", the value for this property can be a different value in "netmap1" and "netmap2".

To support information resource dependent property values, this document uses the same approach as in Section 10.8.1 of [RFC7285] entitled "Resource-Specific Endpoint Properties". When a property value depends on a given information resource, the name of this property MUST be explicitly associated with the information resource that defines it.

For example, the property "pid" queried on entity "ipv4:192.0.2.34" and defined in both "netmap1" and "netmap2", can be named "netmap1.pid" and "netmap2.pid". This allows a Client to get a property of the same type but defined in different information resources with a single query. Specifications on the property name format are provided in Section 5.2.

4.4. Entity Hierarchy and Property Inheritance

For some domain types, there is an underlying structure that allows entities to efficiently be grouped into a set and be defined by the identifier of this set. This is the case for domain types "ipv4" and "ipv6", where individual Internet addresses can be grouped in blocks. When the same property value applies to a whole set, a Server can define a property for the identifier of this set instead of enumerating all the entities and their properties. This allows a substantial reduction of transmission payload both for the Server and the Client. For example, all the entities included in the set defined by the address block "ipv6:2001:db8::1/64" share the same properties and values defined for this block.

Additionally, entity sets sometimes are related by inclusion, hierarchy or other relations. This allows defining inheritance rules for entity properties that propagate properties among related entity sets. The Server and the Client can use these inheritance rules for further payload savings. Entity hierarchy and property inheritance rules are specified in the documents that define the applicable domain types. The present document defines these rules for the "ipv4" and "ipv6" domain types.

This document introduces, for applicable domain types, "Entity Property Inheritance rules", with the following concepts: Entity Hierarchy, Property Inheritance and Property Value Unicity. A detailed specification of entity hierarchy and property inheritance rules is provided in Section 5.1.4.

4.4.1. Entity Hierarchy

An entity domain may allow using a single identifier to identify a set of related individual entities. For example, a CIDR block can be used to identify a set of IPv4 or IPv6 entities. A CIDR block is called a hierarchical entity identifier, as it can reflect inclusion relations among entity sets. That is, in an entity hierarchy, "supersets" are defined at upper levels and include "subsets" defined at lower levels." For example, the CIDR "ipv4:192.0.1.0/24" includes all the individual IPv4 entities identified by the CIDR "ipv4:192.0.1.0/26". This document will sometimes use the term "hierarchical address" to refer to a hierarchical entity identifier.

4.4.2. Property Inheritance

A property may be defined for a hierarchical entity identifier, while it may be undefined for individual entities covered by this identifier. In this case, these individual entities inherit the property value defined for the identifier that covers them. For example, suppose a property map defines a property P for which it assigns value V1 only for the hierarchical entity identifier "ipv4:192.0.1.0/24" but not for individual entities in this block. Suppose also that inheritance rules are specified for CIDR blocks in the "ipv4" domain type. When receiving this property map, a Client can infer that entity "ipv4:192.0.1.1" inherits the property value V1 of block "ipv4:192.0.1.0/24" because the address "ipv4:192.0.1.1" is included in the CIDR block "ipv4:192.0.1.0/24".

Property value inheritance rules also apply among entity sets. A property map may define values for an entity set belonging to a hierarchy but not for "subsets" that are covered by this set identifier. In this case, inheritance rules must specify how entities in "subsets" inherit property values from their "superset". For instance, suppose a property P is defined only for the entity set defined by address block "ipv4:192.0.1.0/24". We know that entity set "ipv4:192.0.1.0/30" is included in "ipv4:192.0.1.0/24". Therefore, the entities of "ipv4:192.0.1.0/30" may inherit the value of property P from set "ipv4:192.0.1.0/24", if an inheritance rule from "ipv4" CIDR blocks to included "ipv4" CIDR blocks, is specified.

4.4.3. Property Value Unicity

The inheritance rules must ensure that an entity belonging to a hierarchical set of entities inherits no more than one property value, for the sake of consistency. Indeed, a property map may define a property on a hierarchy of entity sets that inherit property values from one or more supersets (located at upper levels). On the other hand, a property value, defined on a subset (located at a lower level) may be different from the value defined on a superset. In such a case, subsets may potentially end up with different property values. This may be the case for address blocs with increasing prefix length, on which a property value gets increasingly accurate and thus may differ. For example, a fictitious property such as "geo-location" or "average transfer volume" may be defined at a progressively finer grain for lower level subsets of entities, defined with progressively longer CIDR prefixes. It seems more interesting to have property values of progressively higher accuracy. A unicity rule, applied to the entity domain type must specify an arbitration rule among the different property values for an entity. An example illustrating the need for such rules is provided in Section 6.1.3.

4.5. Supported Properties on Entity Domains in Property Map Capabilities

A property type is not necessarily applicable to any domain type, or an ALTO Server may choose not to provide a property on all applicable domains. For instance, a property type reflecting link bandwidth is likely not defined on entities of a domain of type "countrycode". Therefore, an ALTO server providing Property Maps needs to specify the properties that can be queried on the different entity domains it supports.

This document explains how the Information Resources Directory (IRD) capabilities of a Property Map resource unambiguously expose what properties a Client can query on a given entity domain:

* a field named "mappings" lists the names of the entity domains supported by the Property Map,

* for each listed entity domain, a list of the names of the applicable properties is provided.

An example is provided in Section 10.3. The "mappings" field associates entity domains and properties that can be resource-agnostic or resource-specific. This allows a Client to formulate compact and unambiguous entity property queries, possibly relating to one or more information resources. In particular:

* it prevents a Client from querying a property on entity domains on which it is not defined,

* it allows a Client to query, for an entity E, values for a property P that are defined in several information resources,

* it allows a Client to query a property P on entities that are defined in several information resources.

Further details are provided in Section 7.4.

4.6. Defining Information Resource for Resource-Specific Entity Domains

A Client willing to query properties on entities belonging to a domain needs to know how to retrieve these entities. To this end, the Client can look up the "mappings" field exposed in IRD capabilities of a property map, see Section 4.5. This field, in its keys, exposes all the entity domains supported by the property map. The syntax of the entity domain identifier specified in Section 5.1.2 allows the client to infer whether the entity domain is resource-specific or not. The Client can extract, if applicable, the

identifier of the specific resource, query the resource and retrieve the entities. For example:

* an entity domain named "netmap1.ipv4" includes the IPv4 addresses that appear in the "ipv4" field of the endpoint address group of each PID in the network map "netmap1", and that have no meaning outside "netmap1" because, for instance, these are local addresses not reachable outside some private network,

* an entity domain named "netmap1.pid" includes the PIDs listed in network map "netmap1".

* an entity domain named "ipv4" is resource-agnostic and covers all the reachable IPv4 addresses.

Besides, it is not possible to prevent a Server from mistakenly exposing inappropriate associations of information resources and entity domain types. To prevent failures due to invalid queries, it is necessary to inform the Client about which associations are allowed. An informed Client will just ignore inappropriate associations exposed by a Server and avoid error-prone transactions with the Server.

For example, the association "costmap3.pid" is not allowed for the following reason: although a cost map exposes PID identifiers, it does not define the set of addresses included in this PID. Neither does a cost map list all the PIDs on which properties can be queried, because a cost map only exposes PID pairs on which a queried cost type is defined. Therefore, the resource "costmap3" does not enable a Client to extract information on the existing PID entities or on the addresses they contain.

Instead, the cost map uses a network map, where all the PIDs used in a cost map are defined together with the addresses contained by the PIDs. This network map is qualified in this document as the Defining Information Resource for the entity domain of type "pid" and this concept is explained in Section 4.6.1.

4.6.1. Defining Information Resource and its Media Type

For the reasons explained in Section 4.6, this document introduces the concept of "Defining Information Resource and its Media Type".

A defining information resource for an entity domain D is the information resource where entities of D are defined. That is, all the information on the entities of D can be retrieved in this resource. A defining information resource is defined for resource-specific entity domains. It does not exist for entity domains that

are not resource-specific such as "ipv4" or "ipv6". Neither does it exist for entity domains that are covering entity identifiers already defined in other standardization documents, at it is the case for country code identifiers standardized in [ISO3166-1] or AS numbers allocated by the IANA. This is useful for entity domain types that are by essence domain-specific, such as the "pid" domain type. It is also useful for resource-specific entity domains constructed from resource-agnostic domain types, such as network map specific domains of local IPv4 addresses.

The defining information resource of a resource-specific entity domain D, when it exists, is unique and has the following specificities:

* it has an entry in the IRD,

* it defines the entities of D,

* it does not use another information resource that defines these entities,

* it defines and exposes entity identifiers that are all persistent,

* its media type is equal to the one that is specified for the defining information resource of an entity domain type.

A fundamental characteristic of a defining information resource is its media type. There is a unique association between an entity domain type and the media type of its defining information resource. When an entity domain type allows associations with defining information resources, the media type of the potential defining information resource MUST be specified:

* in the document that defines this entity domain type,

* in the IANA ALTO Entity Domain Type Registry and related information.

When the Client wants to use a resource-specific entity domain, it needs to be cognizant of the media-type of its defining information resource. If the Server exposes a resource-specific entity domain with a non-compliant media type for the defining resource, the Client MUST ignore the entities from that entity domain to avoid errors.

4.6.2. Examples of Defining Information Resources and Their Media Type

Here are examples of defining information resource types and their media types associated to different entity domain types:

* For entity domain type "pid": the media type of the specific resource is "application/alto-networkmap+json", because PIDs are defined in network map resources.

* For entity domain types "ipv4" and "ipv6": the media type of the specific resource is "application/alto-networkmap+json", because IPv4 and IPv6 addresses covered by the Server are defined in network map resources.

* For entities of domain type "ane": [I-D.ietf-alto-path-vector] defines entities named "ANE", where ANE stands for Abstracted Network Element, and the entity domain type "ane". An ANE may have a persistent identifier, say, "entity-4", that is provided by the Server as a value of the "persistent-entity-id" property of this ANE. Further properties may then be queried on an ANE by using its persistent entity ID. These properties are available from a persistent property map, that defines properties on a specific "ane" domain. Together with the persistent identifier, the Server also provides the property map resource identifier where the "ane" domain containing "entity-4" is defined. The definition of the "ane" entity domain containing "entity-4" is thus specific to the property map. Therefore, for entities of domain type "ane" that have a persistent identifier, the media type of the defining information resource is "application/alto-propmap+json".

* Last, the entity domain types "asn" and "countrycode" defined in [I-D.ietf-alto-cdni-request-routing-alto] do not have a defining information resource. Indeed, the entity identifiers in these two entity domain types are already standardized in documents that the Client can use.

4.7. Defining Information Resource for Resource-Specific Property Values

As explained in Section 4.3, a property type may take values that are resource-specific. This is the case for property type "pid", whose values are by essence defined relative to a specific network map. That is, the PID value returned for an IPv4 address is specific to the network map defining this PID and may differ from one network map to another one.

Another example is provided in [I-D.ietf-alto-cdni-request-routing-alto] that defines property type "cdni-capabilities". The value of this property is specific to a CDNI advertisement resource, that provides a list of CDNI capabilities. The property is provided for entity domain types "ipv4", "ipv6", "asn" and "countrycode". A CDNI Advertisement resource does however not define PID values for IPv4 addresses while a network map does not define CDNI capabilities for IPv4 addresses.

Similar to resource-specific entity domains, the Client needs to be cognizant of appropriate associations of information resource and property types. Therefore, when specifying and registering a property type whose values are resource-specific, the media type of its defining information resource needs to be specified. For example:

* The media type of the defining information resource for property type "pid" is "application/alto-networkmap+json".

* The media type of the defining information resource for property type "cdni-capabilities" defined in [I-D.ietf-alto-cdni-request-routing-alto] is "application/alto-cdni+json".

5. Protocol Specification: Basic Data Types

5.1. Entity Domain

5.1.1. Entity Domain Type

An entity domain has a type, which is uniquely identified by a string that MUST be no more than 64 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), the hyphen ('-', U+002D), the colon (':', U+003A), or the low line ('_', U+005F).

The usage of colon (':', U+003A) MUST obey the rules below:

* The colon (':', U+003A) character MUST NOT appear more than once,

* The colon character MUST NOT be used unless within the string "priv:",

* The string "priv:" MUST NOT be used unless it starts the string that identifies an entity domain type,

* For an entity domain type identifier with the "priv:" prefix , an additional string (e.g., company identifier or random string) MUST follow "priv:", to reduce potential collisions.

For example, the strings "ipv4", "ipv6", "pid" and "priv:example-test-edt", are valid entity domain types. "ipv4.anycast", "pid.local" and "priv:" are invalid.

Although "_", "-", "__--" are valid entity domain types, it is desirable to add characters such as alphanumeric ones, for better intelligibility.

The type EntityDomainType is used in this document to denote a JSON string meeting the preceding requirements.

An entity domain type defines the semantics of a type of entity, independently of any specifying resource. All entity domain types that are not prefixed with "priv:" MUST be registered with the IANA, in the "ALTO Entity Domain Type Registry", defined in Section 12.3, following the procedure specified in Section 12.3.2 of this document. The format of the entity identifiers (see Section 5.1.3) in that entity domain type, as well as any hierarchical or inheritance rules (see Section 5.1.4) for those entities, MUST be specified in the IANA registration.

Entity domain type identifiers prefixed with "priv:" are reserved for Private Use (see [RFC8126]) without a need to register with IANA. The definition of a private use entity domain type MUST apply the same way in all property maps of an IRD where it is present.

5.1.2. Entity Domain Name

As discussed in Section 3.2, an entity domain is characterized by a type and identified by a name.

This document distinguishes three categories of entity domains: resource-specific entity domains, resource-agnostic entity domains and self-defined entity domains. Their entity domain names are constructed as specified in the following sub-sections.

Each entity domain is identified by a unique entity domain name. Borrowing the symbol "::=" from the Backus-Naur Form notation [RFC5511], the format of an entity domain name is defined as follows:

EntityDomainName ::= [[ResourceID] '.']EntityDomainType

The presence and construction of the component

"[[ResourceID] '.']"

depends on the category of entity domain.

Note that the '.' separator is not allowed in EntityDomainType and hence there is no ambiguity on whether an entity domain name refers to a resource-agnostic entity domain or a resource-specific entity domain.

Note also that Section 10.1 of [RFC7285] specifies the format of the PID name which is the format of the resource ID including the following specification: "the '.' separator is reserved for future use and MUST NOT be used unless specifically indicated in this document, or an extension document". The present extension keeps the format specification of [RFC7285], hence the '.' separator MUST NOT be used in an information resource ID.

5.1.2.1. Resource-specific Entity Domain

A resource-specific entity domain is identified by an entity domain name constructed as follows. It MUST start with a resource ID using the ResourceID type defined in Section 10.2 of [RFC7285], followed by the '.' separator (U+002E), followed by a string of the type EntityDomainType specified in Section 5.1.1.

For example, if an ALTO server provides two network maps "netmap-1" and "netmap-2", these network maps can define two resource-specific domains of type "pid", respectively identified by "netmap-1.pid" and "netmap-2.pid".

5.1.2.2. Resource-agnostic Entity Domain

A resource-agnostic entity domain contains entities that are identified independently of any information resource. The identifier of a resource-agnostic entity domain is simply the identifier of its entity domain type. For example, "ipv4" and "ipv6" identify the two resource-agnostic Internet address entity domains defined in Section 6.1.

5.1.2.3. Self-defined Entity Domain

A property map can define properties on entities that are specific to a unique information resource, which is the property map itself. This may be the case when an ALTO Server provides properties on a set of entities that are defined only in this property map, are not relevant to another one and do not depend on another specific resource.

For example: a specialised property map may define a domain of type "ane", defined in [I-D.ietf-alto-path-vector], that contains a set of ANEs representing data centers, that each have a persistent identifier and are relevant only to this property map.

In this case, the entity domain is qualified as "self-defined". The identifier of a self-defined entity domain can be of the format:

```
EntityDomainName ::= '.' EntityDomainType
```

where '.' indicates that the entity domain only exists within the property map resource using it.

A self-defined entity domain can be viewed as a particular case of resource-specific entity domain, where the specific resource is the current resource that uses this entity domain. In that case, for the sake of simplification, the component "ResourceID" MUST be omitted in its entity domain name.

5.1.3. Entity Identifier

Entities in an entity domain are identified by entity identifiers (EntityID) of the following format:

```
EntityID ::= EntityDomainName ':' DomainTypeSpecificEntityID
```

Examples from the Internet address entity domains include individual IP addresses such as "net1.ipv4:192.0.2.14" and "net1.ipv6:2001:db8::12", as well as address blocks such as "net1.ipv4:192.0.2.0/26" and "net1.ipv6:2001:db8::/48".

The format of the second part of an entity identifier, DomainTypeSpecificEntityID, depends on the entity domain type, and MUST be specified when defining a new entity domain type and registering it with the IANA. Identifiers MAY be hierarchical, and properties MAY be inherited based on that hierarchy. The rules defining any hierarchy or inheritance MUST be defined when the entity domain type is registered.

The type EntityID is used in this document to denote a JSON string representing an entity identifier in this format.

Note that two entity identifiers with different valid textual representations may refer to the same entity, for a given entity domain. For example, the strings "net1.ipv6:2001:db8::1" and "net1.ipv6:2001:db8:0:0:0:0:0:1" refer to the same entity in the "ipv6" entity domain. Such equivalences should be established by the object represented by DomainTypeSpecificEntityID, for example, [RFC5952] establishes equivalence for IPv6 addresses, while [RFC4632] does so for IPv4 addresses.

5.1.4. Hierarchy and Inheritance

To simplify the representation, some types of entity domains allow the ALTO Client and Server to use a hierarchical entity identifier format to represent a block of individual entities. For instance, in an IPv4 domain "net1.ipv4", a CIDR "net1.ipv4:192.0.2.0/26" covers 64 individual IPv4 entities. In this case, the corresponding property inheritance rule MUST be defined for the entity domain type. The hierarchy and inheritance rule MUST have no ambiguity.

5.2. Entity Property

Each entity property has a type to indicate the encoding and the semantics of the value of this entity property, and has a name to identify it.

5.2.1. Entity Property Type

The type EntityPropertyType is used in this document to indicate a string denoting an entity property type. The string MUST be no more than 32 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), or the low line ('_', U+005F). Note that the '.' separator is not allowed because it is reserved to separate an entity property type and an information resource identifier when an entity property is resource-specific.

While, in Section 5.1.1, character ":" is allowed with restrictions on entity domain identifiers, it can be used without restrictions on entity property type identifiers. This relates to [RFC7285], where a Server can define properties on endpoints "ipv4" and "ipv6". In the present extension, there is a mapping of ALTO entity domain types "ipv4" and "ipv6", to ALTO address types "ipv4" and "ipv6". Properties defined on "ipv4" and "ipv6" endpoints should be re-usable on "ipv4" and "ipv6" entities. Forbidding the usage of ":" in a non-private entity property type identifier would not allow to use properties previously defined on "ipv4" and "ipv6" endpoints because their identifiers would be invalid.

Although ":" or "_::-" are valid entity domain types, it is desirable to add characters such as alphanumeric ones, for better intelligibility.

Identifiers prefixed with "priv:" are reserved for Private Use [RFC8126] without a need to register with IANA. All other identifiers for entity property types MUST be registered in the "ALTO Entity Property Type Registry", defined in Section 12.4. The intended semantics of the entity property type MUST be specified in the IANA registration.

For an entity property identifier with the "priv:" prefix, an additional string (e.g., company identifier or random string) MUST follow the prefix to reduce potential collisions, that is, the string "priv:" alone is not a valid entity property identifier. The definition of a private use entity property type must apply the same way in all property maps of an IRD where it is present.

To distinguish from the endpoint property type, the entity property type has the following characteristics:

* Some entity property types are applicable to entities in particular entity domain types only. For example, the property type "pid" is applicable to entities in the entity domain types "ipv4" or "ipv6" while is not applicable to entities in an entity domain of type "pid".

* The intended semantics of the value of an entity property may also depend on the entity domain type. For example, suppose that a property named "geo-location" is defined as the coordinates of a point, encoded as: "latitude longitude [altitude]." When applied to an entity that represents a specific host computer, identified by an address in an entity domain of type "ipv4" or "ipv6", the "geo-location" property would define the host's location. However, when applied to an entity in a "pid" domain type, the property would indicate a location representative of all hosts in this "pid" entity.

5.2.2. Entity Property Name

Each entity property is identified by an entity property name, which is a string of the following format:

```
EntityPropertyName ::= [[ResourceID]'.']EntityPropertyType
```

Similar to the endpoint property type defined in Section 10.8 of [RFC7285], each entity property may be defined by either the property map itself (self-defined) or some other specific information resource (resource-specific).

The entity property name of a resource-specific entity property starts with a string of the type ResourceID defined in [RFC7285], followed by the '.' separator (U+002E) and a EntityDomainType typed string. For example, the "pid" properties of an "ipv4" entity defined by two different maps "net-map-1" and "net-map-2" are identified by "net-map-1.pid" and "net-map-2.pid" respectively.

The specific information resource of an entity property may be the current information resource itself, that is, the property map defining the property. In that case, the ResourceID in the property name SHOULD be omitted. For example, the property name ".asn" applied to an entity identified by its IPv4 address, indicates the AS number of the AS that "owns" the entity, where the returned AS number is defined by the property map itself.

5.2.3. Format for Entity Property Value

Section 11.4.1.6 of [RFC7285] specifies that an implementation of the Endpoint Property Service specified in [RFC7285] SHOULD assume that the property value is a JSONString and fail to parse if it is not. This document extends the format of a property value by allowing it to be a JSONValue instead of just a JSONString.

6. Entity Domain Types Defined in this Document

The definition of each entity domain type MUST include (1) the entity domain type name and (2) domain-specific entity identifiers, and MAY include (3) hierarchy and inheritance semantics optionally. This document defines three initial entity domain types as follows.

6.1. Internet Address Domain Types

The document defines two entity domain types (IPv4 and IPv6) for Internet addresses. Both types are resource-agnostic entity domain types and hence define corresponding resource-agnostic entity domains as well. Since the two domains use the same hierarchy and inheritance semantics, we define the semantics together, instead of repeating for each.

6.1.1. Entity Domain Type: IPv4

6.1.1.1. Entity Domain Type Identifier

ipv4

6.1.1.2. Domain-Specific Entity Identifiers

Individual addresses are strings as specified by the IPv4address rule in Section 3.2.2 of [RFC3986]; Hierarchical addresses are strings as specified by the prefix notation in Section 3.1 of [RFC4632]. To define properties, an individual Internet address and the corresponding full-length prefix are considered aliases for the same entity on which to define properties. Thus, "ipv4:192.0.2.0" and "ipv4:192.0.2.0/32" are equivalent.

6.1.2. Entity Domain Type: IPv6

6.1.2.1. Entity Domain Type Identifier

ipv6

6.1.2.2. Domain-Specific Entity Identifiers

Individual addresses are strings as specified by Section 4 of [RFC5952]; Hierarchical addresses are strings as specified by IPv6 address prefixes notation in Section 2.3 of [RFC4291]. To define 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.

6.1.3. Hierarchy and Inheritance of Internet Address Domains

Both Internet address domains allow property values to be inherited. Specifically, if a property P is not defined for a specific Internet address I, but P is defined for a hierarchical Internet address C which represents a set of addresses containing I, then the address I inherits the value of P defined for the hierarchical address C. If more than one such hierarchical addresses define a value for P, I inherits the value of P in the hierarchical address with the longest prefix. Note that this longest prefix rule ensures no multiple value inheritances, and hence no ambiguity.

Hierarchical addresses can also inherit properties: if a property P is not defined for the hierarchical address C, but is defined for a set of hierarchical addresses, where each address C' in the set contains all IP addresses in C, and C' has a shorter prefix length than C, then C MUST inherit the property P from the C' having the longest prefix length.

As an example, suppose that a server defines a property P for the following entities:

```
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 indicate a property as not having a value for a particular entity. That is, a server MAY say that property P of entity X is "defined to have no value", instead of "undefined". To indicate "no value", a server MAY perform different behaviors:

* If entity X would inherit a value for property P, and if the ALTO server decides to say that "X has no value for P", then the ALTO server MUST return a "null" value for that property on X. In this case, the ALTO client MUST recognize the JSON "null" value as "no value" and interpret it as "do not apply the inheritance rules for this property on X".

* If the entity would not inherit a value, then the ALTO server MAY return "null" or just omit the property. In this case, the ALTO client cannot infer the value for this property of this entity from the Inheritance rules. So, the client MUST interpret that this property has no value.

If the ALTO server does not define any properties for an entity, then the server MAY omit that entity from the response.

6.1.4. Defining Information Resource Media Type for domain types IPv4 and IPv6

Entity domain types "ipv4" and "ipv6" both allow to define resource specific entity domains. When resource specific domains are defined with entities of domain type "ipv4" or "ipv6", the defining information resource for an entity domain of type "ipv4" or "ipv6" MUST be a Network Map. The media type of a defining information resource is therefore:

application/alto-networkmap+json

6.2. Entity Domain Type: PID

The PID entity domain associates property values with the PIDs in a network map. Accordingly, this entity domain always depends on a network map.

6.2.1. Entity Domain Type Identifier

pid

6.2.2. Domain-Specific Entity Identifiers

The entity identifiers are the PID names of the associated network map.

6.2.3. Hierarchy and Inheritance

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

6.2.4. Defining Information Resource Media Type for Domain Type PID

The entity domain type "pid" allows to define resource specific entity domains. When resource specific domains are defined with entities of domain type "pid", the defining information resource for entity domain type "pid" MUST be a Network Map. The media type of a defining information resource is therefore:

application/alto-networkmap+json

6.2.5. 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 all the properties of a PID to the prefixes in that PID or only some of these properties.

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".

6.3. Internet Address Properties vs. PID Properties

Because the Internet address and PID domains relate to completely distinct domain types, the question may arise as to which entity domain type is the best for a property. In general, the Internet address domain types are RECOMMENDED for properties that are closely related to the Internet address, or are associated with, and inherited through, hierarchical addresses.

The PID domain type 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 Internet address domain types. On the other hand, a property that explains why a PID was formed, or how it relates to a provider's network, would best be associated with the PID domain type.

7. Property Map

A property map returns the properties defined for all entities in one or more domains, e.g., the "location" property of entities in "pid" domain, and the "ASN" property of entities in "ipv4" and "ipv6" domains. Section 10.4 gives an example of a property map request and its response.

Downloading the whole property map is a way for the Client to obtain the Entity IDs that can be used as input for a Filtered Property Map request. However, a whole property map may be too voluminous for a Client that only wants the list of applicable Entity IDs. How to obtain the list of entities of a filtered property map in a simplified response is specified in Section 8.

7.1. Media Type

The media type of a property map is "application/alto-propmap+json".

7.2. HTTP Method

The property map is requested using the HTTP GET method.

7.3. Accept Input Parameters

None.

7.4. Capabilities

The capabilities are defined by an object of type PropertyMapCapabilities:

```
object {
  EntityPropertyMapping mappings;
} PropertyMapCapabilities;

object-map {
  EntityDomainName -> EntityPropertyName<1..*>;
} EntityPropertyMapping
```

with fields:

mappings: A JSON object whose keys are names of entity domains and values are the supported entity properties of the corresponding entity domains.

7.5. Uses

The "uses" field of a property map resource in an IRD entry specifies the resources in this same IRD on which this property map directly depends. It is an array of resource ID(s). This array identifies the defining information resources associated with the resource-specific entity domains and properties that are indicated in this resource.

7.6. Response

If the entity 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, and the order MUST be consistent with the "uses" field of this property map resource. 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 {
  EntityID -> EntityProps;
} PropertyMapData;

object {
  EntityPropertyName -> 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.

For each entity in the property map:

* If the entity is in a resource-specific entity domain, the ALTO server MUST only return self-defined properties and resource-specific properties which depend on the same resource as the entity does. The ALTO client MUST ignore any resource-specific property for this entity if the mapping between this resource-specific property and this entity is not indicated, in the IRD, in the "mappings" capability of the property map resource.

* If the entity identifier is resource-agnostic, the ALTO server SHOULD return the self-defined properties and all the resource-specific properties that are defined in the property defining information resources indicated, in the IRD, in the "mappings" capability of the property map resource, unless property values can be omitted upon some inheritance rules.

The ALTO server MAY omit property values that are inherited rather than explicitly defined, in order to achieve more compact encoding. As a consequence, the ALTO Client MUST NOT assume inherited property values will all be present. If the Client needs inherited values, it MUST use the entity domain's inheritance rules to deduce those values.

8. Filtered Property Map

A filtered property map returns the values of a set of properties for a set of entities selected by the client.

Section 10.5, Section 10.6, Section 10.7 and Section 10.8 give examples of filtered property map requests and responses.

While the IRD lists all the names of the supported properties, it only lists the names of the supported entity domains and not the entity IDs. A client, sometimes, may only want to know what entity IDs it can provide as input to a filtered property map request but wants to avoid the burden of downloading the full property map. Or it may want to check whether some given entity IDs are eligible for a query. To support such a case, the filtered property map supports a light weight response, with empty property values.

8.1. Media Type

The media type of a property map resource is "application/alto-propmap+json".

8.2. HTTP Method

The filtered property map is requested using the HTTP POST method.

8.3. 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. ReqFilteredPropertyMap is designed to support the following cases of client requests:

* The client wants the value of a selected set of properties on a selected set of entities,

* The client wants all properties values on all the entities,

* The client wants all entities on which a property is defined but is not interested in their property values,

* The Client wants to cross-check whether some entity IDs are present in the Filtered Property Map but is not interested in their property values.

The third case is equivalent to querying the whole unfiltered property map, which can also be achieved with a GET request. Some Clients however, may prefer to systematically make filtered property map queries, where filtering parameters may sometimes be empty.

The JSON object ReqFilteredPropertyMap is specified as follows:

```
object {
  EntityID             entities<0..*>;
  [EntityPropertyName   properties<0..*>;]
} ReqFilteredPropertyMap;
```

with fields:

entities: List of entity identifiers for which the specified properties are to be returned. If the list is empty, the ALTO Server MUST interpret the list as if it contained a list of all entities currently defined in the filtered property map. The domain of each entity MUST be included in the list of entity domains in this resource's "capabilities" field (see Section 8.4). The ALTO server MUST interpret entries appearing multiple times as if they appeared only once.

properties: List of properties to be returned for each entity. If the list is empty, the ALTO Sever MUST interpret the list as if it contained a list of all properties currently defined in the filtered property map. Each specified property MUST be included in the list of properties in this resource's "capabilities" field (see Section 8.4). The ALTO server MUST interpret entries appearing multiple times as if they appeared only once. This field is optional. If it is absent, the Server returns a property value equal to the literal string "{}" for all the entity IDs of the "entities" field on which at least one property is defined.

Note that the field "properties" is optional. When in addition, the "entities" field is an empty list, it corresponds to a query for all applicable entity IDs of the filtered property map, with no current interest on any particular property. When the "entities" field is not empty, it allows the Client to check whether the listed entity IDs can be used as input to a filtered property map query.

8.4. Capabilities

The capabilities are defined by an object of type PropertyMapCapabilities, as defined in Section 7.4.

8.5. Uses

Same to the "uses" field of the Property Map resource (see Section 7.5).

8.6. Filtered Property Map Response

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

Specifically, a filtered property map request can be invalid in the following cases:

* The input field "entities" is absent from the Client request. In this case, the Server MUST return an "E_MISSING_FIELD" error as defined in Section 8.5.2 of [RFC7285].

* An entity identifier in the "entities" field of the request is invalid. This occurs when:

  - The domain of this entity is not defined in the "entity-domains" capability of this resource in the IRD,

  - The entity identifier is not valid for the entity domain.

  A valid entity identifier does never generate an error, even if the filtered property map resource does not define any properties for it.

  If an entity identifier in the "entities" field of the request is invalid, the ALTO server MUST return an "E_INVALID_FIELD_VALUE" error defined in Section 8.5.2 of [RFC7285], and the "value" field of the error message SHOULD indicate the provided invalid entity identifier.

* A property name in the "properties" field of the request is invalid. This occurs when this property name is not defined in the "properties" capability of this resource in the IRD.

When a filtered property map resource does not define a value for a property requested on a particular entity, it is not an error. In this case, the ALTO server MUST omit that property from the response for that endpoint.

If a property name in "properties" in the request is invalid, the ALTO server MUST return an "E_INVALID_FIELD_VALUE" error defined in Section 8.5.2 of [RFC7285]. The "value" field of the error message SHOULD indicate the property name.

Some identifiers can be interpreted as both an entity name and a property name, as it is the case for "pid" if it would be erroneously used alone. In such a case, the Server SHOULD follow Section 8.5.2 of [RFC7285], that says: "For an E_INVALID_FIELD_VALUE error, the server may include an optional field named "field" in the "meta" field of the response, to indicate the field that contains the wrong value."

The response to a valid request is the same as for the Property Map (see Section 7.6), except that:

* If the requested entities include entities with a resource-agnostic identifier, the "dependent-vtags" field in its "meta" field MUST include version tags of all dependent resources appearing in the "uses" field.

* If the requested entities only include entities in resource-specific entity domains, the "dependent-vtags" field in its "meta" field MUST include the version tags of the resources on which the requested resource-specific entity domains and the requested resource-specific properties are dependent on.

* The response only includes the entities and properties requested by the client. If an entity in the request is identified by a hierarchical identifier (e.g., a "ipv4" or "ipv6" prefix), the response MUST return all properties that are present on any address covered by the prefix, even though some of those properties may not be present on all addresses covered by the prefix.

* When the input member "properties" is absent from the client request, the Server returns a property map containing all the requested entity identifiers on which one or more properties are defined. For all the entities of the returned map, the returned property value is equal to '{}'.

The filtered property map response MUST include all the inherited property values for the requested entities and all the entities which are able to inherit property values from the requested entities. To achieve this goal, the ALTO server MAY follow two rules:

* If a property for a requested entity is inherited from another entity not included in the request, the response MUST include this property for the requested entity. For example, A full property map may skip a property P for an entity A (e.g., ipv4:192.0.2.0/31) if P can be derived using inheritance from another entity B (e.g., ipv4:192.0.2.0/30). A filtered property map request may include only A but not B. In such a case, the property P MUST be included in the response for A.

* If there are entities covered by a requested entity but having different values for the requested properties, the response MUST include all those entities and the different property values for them. For example, considering a request for property P of entity A (e.g., ipv4:192.0.2.0/31), if P has value v1 for A1=ipv4:192.0.2.0/32 and v2 for A2=ipv4:192.0.2.1/32, then, the response SHOULD include A1 and A2.

For the sake of response compactness, the ALTO server SHOULD obey the following rule:

* If an entity identifier in the response is already covered by other entities identifiers in the same response, it SHOULD be removed from the response. In the previous example, the entity A = ipv4:192.0.2.0/31 SHOULD be removed because A1 and A2 cover all the addresses in A.

An ALTO client should be aware that the entities in the response may be different from the entities in its request.

8.7. Entity Property Type Defined in This Document

This document defines the entity property type "pid". This property type extends the ALTO Endpoint Property Type "pid" defined in section 7.1.1 of [RFC7285] as follows: the property has the same semantics and applies to IPv4 and IPv6 addresses; the difference is that the IPv4 and IPv6 addresses have evolved from the status of endpoints to the status of entities.

The defining information resource for property type MUST be a network map. This document requests a IANA registration for this property

8.7.1. Entity Property Type: pid

1. Identifier: pid

2. Semantics: the intended semantics are the same as in [RFC7285] for the ALTO Endpoint Property Type "pid"

3. Media type of defining information resource: application/alto-networkmap+json

4. Security considerations: for entity property type "pid" are the same as documented in [RFC7285] for the ALTO Endpoint Property Type "pid".

9. Impact on Legacy ALTO Servers and ALTO Clients

9.1. Impact on Endpoint Property Service

Since the Property Map and the Filtered Property Map defined in this document provide a functionality that covers the EPS defined in Section 11.4 of [RFC7285], ALTO servers may prefer to provide Property Map and Filtered Property Map in place of EPS. However, for the legacy endpoint properties, it is recommended that ALTO servers also provide EPS so that legacy clients can still be supported.

9.2. 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 depend on, while global property names have no such prefix. The property map and the filtered property map defined in this document define similar categories of entity properties. The difference is that entity property maps do not define "global" entity properties. Instead, they define "self-defined" entity properties as a special case of "resource-specific" entity properties, where the specific resource is the property map itself. This means that "self-defined" properties are defined within the scope of the property map.

9.3. Impact on Other Properties

In the present extension, properties can be defined on sets of entity addresses, rather than just individual endpoint addresses as initially defined in [RFC7285]. This might change the semantics of a property. These sets can be for example hierarchical IP address blocks. For instance, a property such as fictitious "geo-location", defined on a set of IP addresses would have a value corresponding to a location representative of all the addresses in this set.

10. Examples

In this document, the HTTP message bodies of all the examples use Unix-style line-ending character (%x0A) as the line separator.

10.1. Network Map

The examples in this section use a very simple default network map:

```
defaultpid:  ipv4:0.0.0.0/0  ipv6:::/0
pid1:        ipv4:192.0.2.0/25
pid2:        ipv4:192.0.2.0/27
pid3:        ipv4:192.0.3.0/28
pid4:        ipv4:192.0.3.16/28
```

Figure 3: Example Default Network Map

And another simple alternative network map:

```
defaultpid:  ipv4:0.0.0.0/0  ipv6:::/0
pid1:        ipv4:192.0.2.0/27
pid2:        ipv4:192.0.3.0/27
```

Figure 4: Example Alternative Network Map

10.2. Property Definitions

Beyond "pid", the examples in this section use four additional fictitious property types for entities of domain type "ipv4": "countrycode", "ASN", "ISP", and "state". These properties are assumed to be resource-agnostic so their name is identical to their type. The entities have the following values:

	ISP	ASN	countrycode	state
ipv4:192.0.2.0/23:	BitsRus	-	us	-
ipv4:192.0.2.0/28:	-	65543	-	NJ
ipv4:192.0.2.16/28:	-	65543	-	CT
ipv4:192.0.2.1:	-	-	-	PA
ipv4:192.0.3.0/28:	-	65544	-	TX
ipv4:192.0.3.16/28:	-	65544	-	MN

Figure 5: Example Property Values for Internet Address Domains

And the examples in this section use the property "region" for the PID domain of the default network map with the following values:

	region
pid:defaultpid:	-
pid:pid1:	us-west
pid:pid2:	us-east
pid:pid3:	us-south
pid:pid4:	us-north

Figure 6: Example Property Values for Default Network Map's PID Domain

Note that "-" means the value of the property for the entity is "undefined". So the entity would inherit a value for this property by the inheritance rule if possible. For example, the value of the "ISP" property for "ipv4:192.0.2.1" is "BitsRus" because of "ipv4:192.0.2.0/24". But the "region" property for "pid:defaultpid" has no value because no entity from which it can inherit.

Similar to the PID domain of the default network map, the examples in this section use the property "ASN" for the PID domain of the alternative network map with the following values:

	ASN
pid:defaultpid:	-
pid:pid1:	65543
pid:pid2:	65544

Figure 7: Example Property Values for Alternative Network Map's PID Domain

10.3. Information Resource Directory (IRD)

The following IRD defines ALTO Server information resources that are relevant to the Entity Property Service. It provides a property map for the "ISP" and "ASN" properties. The server could have provided a single property map for all four properties, but does not, presumably because the organization that runs the ALTO server believes that a client is not necessarily interested in getting all four properties.

The server provides several filtered property maps. The first returns all four properties, and the second returns only the "pid" property for the default network map and the "alt-network-map".

The filtered property maps for the "ISP", "ASN", "countrycode" 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 providing the "pid" property does have a "uses" capability for the default network map because the default network map defines the values of the "pid" property.

Note that for legacy clients, the ALTO server provides an Endpoint Property Service for the "pid" property defined on the endpoints of the default network map and the "alt-network-map".

The server provides another filtered Property map resource, named "ane-dc-property-map", that returns fictitious properties named "storage-capacity", "ram" and "cpu" for ANEs that have a persistent identifier. The entity domain to which the ANEs belong is "self-defined" and valid only within the property map.

The other property maps in the returned IRD are here for purposes of illustration.

```
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: 2713
Content-Type: application/alto-directory+json

{
  "meta" : {
    "default-alto-network-map" : "default-network-map"
  },
  "resources" : {
    "default-network-map" : {
      "uri" : "http://alto.example.com/networkmap/default",
      "media-type" : "application/alto-networkmap+json"
    },
    "alt-network-map" : {
      "uri" : "http://alto.example.com/networkmap/alt",
      "media-type" : "application/alto-networkmap+json"
    },
    "ia-property-map" : {
      "uri" : "http://alto.example.com/propmap/full/inet-ia",
      "media-type" : "application/alto-propmap+json",
      "capabilities" : {
        "mappings": {
          "ipv4": [ ".ISP", ".ASN" ],
```

```
            "ipv6": [ ".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" : {
          "mappings": {
            "ipv4": [ ".ISP", ".ASN", ".countrycode", ".state" ],
            "ipv6": [ ".ISP", ".ASN", ".countrycode", ".state" ]
          }
        }
      },
      "region-property-map": {
        "uri": "http://alto.example.com/propmap/lookup/region",
        "media-type": "application/alto-propmap+json",
        "accepts": "application/alto-propmapparams+json",
        "uses" : [ "default-network-map", "alt-network-map" ],
        "capabilities": {
          "mappings": {
            "default-network-map.pid": [ ".region" ],
            "alt-network-map.pid": [ ".ASN" ]
          }
        }
      },
      "ip-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", "alt-network-map" ],
        "capabilities" : {
          "mappings": {
            "ipv4": [ "default-network-map.pid",
                      "alt-network-map.pid" ],
            "ipv6": [ "default-network-map.pid",
                      "alt-network-map.pid" ]
          }
        }
      },
      "legacy-endpoint-property" : {
        "uri" : "http://alto.example.com/legacy/eps-pid",
        "media-type" : "application/alto-endpointprop+json",
        "accepts" : "application/alto-endpointpropparams+json",
        "capabilities" : {
          "properties" : [ "default-network-map.pid",
                           "alt-network-map.pid" ]
```

```
        }
      },
      "ane-dc-property-map": {
        "uri" : "http://alto.example.com/propmap/lookup/ane-dc",
        "media-type" : "application/alto-propmap+json",
        "accepts": "application/alto-propmapparams+json",
        "capabilities": {
          "mappings": {
            ".ane" : [ "storage-capacity", "ram", "cpu" ]
          }
        }
      }
    }
  }
```

Figure 8: Example IRD

10.4. Full Property Map Example

The following example uses the properties and IRD defined in Section 10.3 to retrieve a Property Map for entities with the "ISP" and "ASN" properties.

Note that, to be compact, the response does not include the entity "ipv4:192.0.2.1" because values of all those properties for this entity are inherited from other entities.

Also note that the entities "ipv4:192.0.2.0/28" and "ipv4:192.0.2.16/28" are merged into "ipv4:192.0.2.0/27", because they have the same value of the "ASN" property. The same rule applies to the entities "ipv4:192.0.3.0/28" and "ipv4:192.0.3.16/28". Both of "ipv4:192.0.2.0/27" and "ipv4:192.0.3.0/27" omit the value for the "ISP" property, because it is inherited from "ipv4:192.0.2.0/23".

```
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: 418
Content-Type: application/alto-propmap+json

{
  "meta": {
    "dependent-vtags": [
      {"resource-id": "default-network-map",
       "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"},
      {"resource-id": "alt-network-map",
       "tag": "c0ce023b8678a7b9ec00324673b98e54656d1f6d"}
    ]
  },
  "property-map": {
    "ipv4:192.0.2.0/23":   {".ISP": "BitsRus"},
    "ipv4:192.0.2.0/27":   {".ASN": "65543"},
    "ipv4:192.0.3.0/27":   {".ASN": "65544"}
  }
}
```

10.5. 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 "ipv4:192.0.2.1" 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: 158
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: 540
Content-Type: application/alto-propmap+json

{
  "meta": {
    "dependent-vtags": [
      {"resource-id": "default-network-map",
       "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"},
      {"resource-id": "alt-network-map",
       "tag": "c0ce023b8678a7b9ec00324673b98e54656d1f6d"}
    ]
  },
  "property-map": {
    "ipv4:192.0.2.0":
          {".ISP": "BitsRus", ".ASN": "65543", ".state": "NJ"},
    "ipv4:192.0.2.1":
          {".ISP": "BitsRus", ".ASN": "65543", ".state": "PA"},
    "ipv4:192.0.2.17":
          {".ISP": "BitsRus", ".ASN": "65543", ".state": "CT"}
  }
}
```

10.6. Filtered Property Map Example #2

The following example uses the filtered property map resource to request the "ASN", "countrycode" and "state" properties for several IPv4 prefixes.

Note that the property values for both entities "ipv4:192.0.2.0/26" and "ipv4:192.0.3.0/26" are not explicitly defined. They are inherited from the entity "ipv4:192.0.2.0/23".

Also note that some entities like "ipv4:192.0.2.0/28" and "ipv4:192.0.2.16/28" in the response are not explicitly listed in the request. The response includes them because they are refinements of the requested entities and have different values for the requested properties.

The entity "ipv4:192.0.4.0/26" is not included in the response, because there are neither entities which it is inherited from, nor entities inherited from it.

```
POST /propmap/lookup/inet-iacs HTTP/1.1
Host: alto.example.com
Accept: application/alto-propmap+json,application/alto-error+json
Content-Length: 174
Content-Type: application/alto-propmapparams+json

{
  "entities" : [ "ipv4:192.0.2.0/26",
                 "ipv4:192.0.3.0/26",
                 "ipv4:192.0.4.0/26" ],
  "properties" : [ ".ASN", ".countrycode", ".state" ]
}

HTTP/1.1 200 OK
Content-Length: 774
Content-Type: application/alto-propmap+json

{
  "meta": {
    "dependent-vtags": [
      {"resource-id": "default-network-map",
       "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"},
      {"resource-id": "alt-network-map",
       "tag": "c0ce023b8678a7b9ec00324673b98e54656d1f6d"}
    ]
  },
  "property-map": {
    "ipv4:192.0.2.0/26":  {".countrycode": "us"},
    "ipv4:192.0.2.0/28":  {".ASN": "65543",
                           ".state": "NJ"},
    "ipv4:192.0.2.16/28": {".ASN": "65543",
                           ".state": "CT"},
    "ipv4:192.0.2.1":     {".state": "PA"},
    "ipv4:192.0.3.0/26":  {".countrycode": "us"},
    "ipv4:192.0.3.0/28":  {".ASN": "65544",
                           ".state": "TX"},
    "ipv4:192.0.3.16/28": {".ASN": "65544",
                           ".state": "MN"}
  }
}
```

10.7. Filtered Property Map Example #3

The following example uses the filtered property map resource to request the "default-network-map.pid" property and the "alt-network-map.pid" property for a set of IPv4 addresses and prefixes.

Note that the entity "ipv4:192.0.3.0/27" is decomposed into two entities "ipv4:192.0.3.0/28" and "ipv4:192.0.3.16/28", as they have different "default-network-map.pid" property values.

```
POST /propmap/lookup/pid HTTP/1.1
Host: alto.example.com
Accept: application/alto-propmap+json,application/alto-error+json
Content-Length: 222
Content-Type: application/alto-propmapparams+json

{
  "entities" : [
                "ipv4:192.0.2.128",
                "ipv4:192.0.2.0/27",
                "ipv4:192.0.3.0/27" ],
  "properties" : [ "default-network-map.pid",
                   "alt-network-map.pid" ]
}

HTTP/1.1 200 OK
Content-Length: 774
Content-Type: application/alto-propmap+json

{
  "meta": {
    "dependent-vtags": [
      {"resource-id": "default-network-map",
       "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"},
      {"resource-id": "alt-network-map",
       "tag": "c0ce023b8678a7b9ec00324673b98e54656d1f6d"}
    ]
  },
  "property-map": {
    "ipv4:192.0.2.128":   {"default-network-map.pid": "defaultpid",
                           "alt-network-map.pid": "defaultpid"},
    "ipv4:192.0.2.0/27":  {"default-network-map.pid": "pid2",
                           "alt-network-map.pid": "pid1"},
    "ipv4:192.0.3.0/28":  {"default-network-map.pid": "pid3",
                           "alt-network-map.pid": "pid2"},
    "ipv4:192.0.3.16/28": {"default-network-map.pid": "pid4",
                           "alt-network-map.pid": "pid2"}
  }
}
```

10.8. Filtered Property Map Example #4

Here is an example of using the filtered property map to query the regions for several PIDs in "default-network-map". The "region" property is specified as a "self-defined" property, i.e., the values of this property are defined by this property map resource.

```
POST /propmap/lookup/region HTTP/1.1
Host: alto.example.com
Accept: application/alto-propmap+json,application/alto-error+json
Content-Length: 132
Content-Type: application/alto-propmapparams+json

{
  "entities" : ["default-network-map.pid:pid1",
                "default-network-map.pid:pid2"],
  "properties" : [ ".region" ]
}

HTTP/1.1 200 OK
Content-Length: 326
Content-Type: application/alto-propmap+json

{
  "meta" : {
    "dependent-vtags" : [
       {"resource-id": "default-network-map",
        "tag": "7915dc0290c2705481c491a2b4ffbec482b3cf62"}
    ]
  },
  "property-map": {
    "default-network-map.pid:pid1": {
      ".region": "us-west"
    },
    "default-network-map.pid:pid2": {
      ".region": "us-east"
    }
  }
}
```

10.9. Filtered Property Map for ANEs Example #5

The following example uses the filtered property map resource "ane-dc-property-map" to request properties "storage-capacity" and "cpu" on several ANEs defined in this property map.

```
POST /propmap/lookup/ane-dc HTTP/1.1
Host: alto.example.com
Accept: application/alto-propmap+json,application/alto-error+json
Content-Length: 155
Content-Type: application/alto-propmapparams+json

{
  "entities" : [".ane:dc21",
                ".ane:dc45.srv9",
                ".ane:dc6.srv-cluster8"],
  "properties" : [ "storage-capacity", "cpu"]
}

HTTP/1.1 200 OK
Content-Length: 295
Content-Type: application/alto-propmap+json

{
  "meta" : {
  },
  "property-map": {
    ".ane:dc21":
      {"storage-capacity" : 40000, "cpu" : 500},
    ".ane:dc45.srv9":
      {"storage-capacity" : 100, "cpu" : 20},
    ".ane:dc6.srv-cluster8":
      {"storage-capacity" : 6000, "cpu" : 100}
  }
}
```

11. Security Considerations

Both Property Map and Filtered Property Map defined in this document fit into the architecture of the ALTO base protocol, and hence the Security Considerations (Section 15 of [RFC7285]) of the base protocol fully apply: authenticity and integrity of ALTO information (i.e., authenticity and integrity of Property Maps), potential undesirable guidance from authenticated ALTO information (e.g., potentially imprecise or even wrong value of a property such as geo-location), confidentiality of ALTO information (e.g., exposure of a potentially sensitive entity property such as geo-location), privacy for ALTO users, and availability of ALTO services should all be considered.

ALTO clients using this extension should in addition be aware that the entity properties they require may convey more details than the endpoint properties conveyed by using [RFC7285]. Client requests may reveal details on their activity or plans thereof, that a malicious

Server, that is in a position to do so, may monetize or use for attacks or undesired surveillance. Likewise, ALTO Servers expose entities and properties related to specific parts of the infrastructure that reveal details on capabilities, locations, or resource availability. These details may be maliciously used for competition purposes, or to cause resource shortage or undesired publication.

To address these concerns, the Property Maps provided by this extension require additional attention on two security considerations discussed in [RFC7285]: "potential undesirable guidance from authenticated ALTO information" (Section 15.2 of [RFC7285]) and "confidentiality of ALTO information" (Section 15.3 of [RFC7285]). Threats to the availability of the ALTO Service caused by highly demanding queries should be addressed as specified in Section 15.5 of [RFC7285].

* Potential undesirable guidance from authenticated ALTO information: it can be caused by Property values that change over time and thus lead to performance degradation or system rejection of application requests.

  To avoid these consequences, a more robust ALTO client should adopt 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 implement "Application-Layer Traffic Optimization (ALTO) Incremental Updates Using Server-Sent Events (SSE)" [RFC8895].

* Confidentiality of ALTO information: 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. Another way is to expose highly abstracted coarse-grained property values to all Clients while restricting access to URIs exposing more fine-grained values to authorized Clients. Restricted access URIs may be gathered in delegate IRDs as specified in Section 9.2.4 of [RFC7285].

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 that do not contain sensitive data.

Section 12 of this document specifies that the ALTO service provider MUST be aware of the potential sensitivity of exposed entity domains and properties. Section 12.2.2. (ALTO Entity Domain Type Registration Process) of this document specifies that when the registration of an entity domain type is requested at the IANA, the request MUST include security considerations that show awareness of how the exposed entity addresses may be related to private information about an ALTO client or an infrastructure service provider. Likewise, Section 12.3. (ALTO Entity Property Type Registry) of this document specifies that when the registration of a property type is requested at the IANA, the request MUST include security considerations that explain why this property type is required for ALTO-based operations.

The risk of ALTO information being leaked to malicious Clients or third parties is addressed similarly to Section 7 of [RFC8896]. ALTO clients and servers SHOULD support TLS 1.3 [RFC8446].

12. IANA Considerations

This document defines additional application/alto-* media types, that are listed in Table 1. It defines an ALTO Entity Domain Type Registry that extends the ALTO Address Type Registry defined in [RFC7285]. It also defines an ALTO Entity Property Type Registry that extends the ALTO endpoint property registry defined in [RFC7285].

Type	Subtype	Specification
application	alto-propmap+json	Section 7.1
application	alto-propmapparams+json	Section 8.3

Table 1: Additional ALTO Media Types.

12.1. application/alto-propmap+json Media Type

Type name:
application

Subtype name:
alto-propmap+json

Required parameters:
n/a

Optional parameters:
n/a

Encoding considerations:
Encoding considerations are identical to those specified for the "application/json" media type. See [RFC8259].

Security considerations:
Security considerations related to the generation and consumption of ALTO Protocol messages are discussed in Section 15 of [RFC7285] and Section 11 of this document.

Interoperability considerations:
n/a

Published specification:
This document is the specification for this media type. See Section 7.1.

Applications that use this media type:
ALTO servers and ALTO clients [RFC7285], either stand alone or embedded within other applications, when the queried resource is a property map, whether filtered or not.

Fragment identifier considerations:
n/a

Additional information:
Magic number(s): n/a

File extension(s): n/a

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).

12.2. alto-propmapparams+json Media Type

Type name:
application

Subtype name:
alto-propmapparams+json

Required parameters:
n/a

Optional parameters:
n/a

Encoding considerations:
Encoding considerations are identical to those specified for the "application/json" media type. See [RFC8259].

Security considerations:
Security considerations related to the generation and consumption of ALTO Protocol messages are discussed in Section 15 of [RFC7285] and Section 11 of this document.

Interoperability considerations:
n/a

Published specification:
This document is the specification for this media type. See Section 8.3.

Applications that use this media type:
ALTO servers and ALTO clients [RFC7285], either stand alone or embedded within other applications, when the queried resource is a filtered property map. This media-type indicates the data format used by the ALTO client to supply the property map filtering parameters.

Fragment identifier considerations:
n/a

Additional information:
Magic number(s): n/a

File extension(s): n/a

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).

12.3. ALTO Entity Domain Type Registry

This document requests IANA to create and maintain the "ALTO Entity Domain Type Registry", listed in Table 2. The first line lists information items that must be provided with each registered entity domain type. Section 12.3.2 specifies how to document these items and provides guidance on the security considerations item that must be documented in addition.

Identifier	Entity Identifier Encoding	Hierarchy & Inheritance	Media Type of Defining Resource	Mapping to ALTO Address Type
ipv4	See Section 6.1.1	See Section 6.1.3	application/alto-networkmap+json	true
ipv6	See Section 6.1.2	See Section 6.1.3	application/alto-networkmap+json	true
pid	See Section 6.2	None	application/alto-networkmap+json	false

Table 2: ALTO Entity Domain Types

This registry serves two purposes. First, it ensures uniqueness of identifiers referring to ALTO entity domain types. Second, it states the requirements for allocated entity domain types.

As specified in Section 5.1.1, identifiers prefixed with "priv:" are reserved for Private Use without a need to register with IANA

12.3.1. Consistency Procedure between ALTO Address Type Registry and ALTO Entity Domain Type Registry

One potential issue of introducing the "ALTO Entity Domain Type Registry" is its relationship with the "ALTO Address Types Registry" already defined in Section 14.4 of [RFC7285]. In particular, the entity identifier of a type of an entity domain registered in the "ALTO Entity Domain Type Registry" MAY match an address type defined in "ALTO Address Type Registry". It is necessary to precisely define and guarantee the consistency between "ALTO Address Type Registry" and "ALTO Entity Domain Registry".

We define that the ALTO Entity Domain Type Registry is consistent with ALTO Address Type Registry if two conditions are satisfied:

* When an address type is already or able to be registered in the ALTO Address Type Registry [RFC7285], the same identifier MUST be used when a corresponding entity domain type is registered in the ALTO Entity Domain Type Registry.

* If an ALTO entity domain type has the same identifier as an ALTO address type, their addresses encoding MUST be compatible.

To achieve this consistency, the following items MUST be checked before registering a new ALTO entity domain type in a future document:

* Whether the ALTO Address Type Registry contains an address type that can be used as an identifier for the candidate entity domain type identifier. This has been done for the identifiers "ipv4" and "ipv6" of Table 2.

* Whether the candidate entity domain type identifier can potentially be an endpoint address type, as defined in Sections 2.1 and 2.2 of [RFC7285].

When a new ALTO entity domain type is registered, the consistency with the ALTO Address Type Registry MUST be ensured by the following procedure:

* Test: Do corresponding entity domain type identifiers match a known "network" address type?

  - If yes (e.g., cell, MAC or socket addresses):

    o Test: Is such an address type present in the ALTO Address Type Registry?

      + If yes: Set the new ALTO entity domain type identifier to be the found ALTO address type identifier.

      + If no: Define a new ALTO entity domain type identifier and use it to register a new address type in the ALTO Address Type Registry following Section 14.4 of [RFC7285].

    o Use the new ALTO entity domain type identifier to register a new ALTO entity domain type in the ALTO Entity Domain Type Registry following Section 12.3.2 of this document.

  - If no (e.g., pid name, ane name or country code): Proceed with the ALTO Entity Domain Type registration as described in Section 12.3.2.

12.3.2. ALTO Entity Domain Type Registration Process

New ALTO entity domain types are assigned after IETF Review [RFC8126] to ensure that proper documentation regarding the new ALTO entity domain types and their security considerations has been provided. RFCs defining new entity domain types SHOULD indicate how an entity in a registered type of domain is encoded as an EntityID, and, if applicable, the rules defining the entity hierarchy and property inheritance. Updates and deletions of ALTO entity domains types follow the same procedure.

Registered ALTO entity domain type identifiers MUST conform to the syntactical requirements specified in Section 5.1.2. Identifiers are to be recorded and displayed as strings.

Requests to the IANA to add a new value to the Entity Domain Type registry MUST include the following information:

* Identifier: The name of the desired ALTO entity domain type.

* Entity Identifier Encoding: The procedure for encoding the identifier of an entity of the registered domain type as an EntityID (see Section 5.1.3). If corresponding entity identifiers of an entity domain type match a known "network" address type, the Entity Identifier Encoding of this domain identifier MUST include both Address Encoding and Prefix Encoding of the same identifier registered in the ALTO Address Type Registry [RFC7285]. To define properties, an individual entity identifier and the corresponding full-length prefix MUST be considered aliases for the same entity.

* Hierarchy: If the entities form a hierarchy, the procedure for determining that hierarchy.

* Inheritance: If entities can inherit property values from other entities, the procedure for determining that inheritance.

* Media type of defining information resource: Some entity domain types allow an entity domain name to be combined with an information resource name to define a resource-specific entity domain. Such an information resource is called "defining information resource", defined in Section 4.6. For each entity domain type, the potential defining information resources have one common media type. This unique common media type is specific to the entity domain type and MUST be specified.

* Mapping to ALTO Address Type: A boolean value to indicate if the entity domain type can be mapped to the ALTO address type with the same identifier.

* Security Considerations: In some usage scenarios, entity identifiers 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 cognizant 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.

12.4. ALTO Entity Property Type Registry

This document requests IANA to create and maintain the "ALTO Entity Property Type Registry", listed in Table 3.

This registry extends the "ALTO Endpoint Property Type Registry", defined in [RFC7285], in that a property type is defined on one or more entity domains, rather than just on IPv4 and IPv6 Internet address domains. An entry in this registry is an ALTO entity property type defined in Section 5.2.1. Thus, a registered ALTO entity property type identifier MUST conform to the syntactical requirements specified in that section.

As specified in Section 5.2.1, identifiers prefixed with "priv:" are reserved for Private Use without a need to register with IANA.

The first line of Table 3 lists information items that must be provided with each registered entity property type.

Identifier	Intended Semantics	Media Type of Defining Resource
pid	See Section 7.1.1 of [RFC7285]	application/alto-networkmap+json

Table 3: ALTO Entity Property Types.

New ALTO entity property types are assigned after IETF Review [RFC8126] to ensure that proper documentation regarding the new ALTO entity property types and their security considerations has been provided. RFCs defining new entity property types SHOULD indicate how a property of a registered type is encoded as a property name. Updates and deletions of ALTO entity property types follow the same procedure.

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

* Identifier: The identifier for the desired ALTO entity property type. The format MUST be as defined in Section 5.2.1 of this document. It

* Intended Semantics: ALTO entity properties 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 entity property should be interpreted.

* Media type of defining information resource: when the property type allows values to be defined relative to a given information resource, the latter is referred to as the "defining information resource", see description in Section 4.7. For each property type, the potential defining information resources have one common media type. This unique common media type is specific to the property type and MUST be specified.

* Security Considerations: ALTO entity properties expose information to ALTO clients. ALTO service providers should be cognizant of the security ramifications related to the exposure of an entity property.

In security considerations, the request should also discuss the sensitivity of the information, and why it is required for ALTO-based operations. Regarding this discussion, the request SHOULD follow the recommendations of Section 14.3. ALTO Endpoint Property Type Registry in [RFC7285].

This document requests registration of the identifier "pid", listed in Table 3. Semantics for this property are documented in Section 7.1.1 of [RFC7285]. No security issues related to the exposure of a "pid" identifier are considered, as it is exposed with the Network Map Service defined and mandated in [RFC7285].

13. Acknowledgments

The authors would like to thank Dawn Chen, and Shenshen Chen for their contributions to earlier drafts. Thank you also to Qiao Xiang, Shawn Lin, Xin Wang and Vijay Gurbani for fruitful discussions. Last, big thanks to Danny Perez and Luis Contreras for their substantial Working Group review feedback and suggestions to improve this document, to Vijay Gurbani, ALTO WG Chair and Martin Duke, Transport Area Director, for their thorough review, discussions, guidance and shepherding, that further helped to enrich this

document.

14. References

14.1. Normative References

[ISO3166-1] ISO (International Organization for Standardization), ., "ISO 3166-1: Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes", 2020.

[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>.

[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, <https://www.rfc-editor.org/info/rfc3986>.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", February 2006.

[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, <https://www.rfc-editor.org/info/rfc4632>.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, DOI 10.17487/RFC5952, August 2010, <https://www.rfc-editor.org/info/rfc5952>.

[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>.

[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>.

[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>.

14.2. Informative References

[I-D.ietf-alto-cdni-request-routing-alto]
Seedorf, J., Yang, Y., Ma, K., Peterson, J., and J. Zhang, "Content Delivery Network Interconnection (CDNI) Request Routing: CDNI Footprint and Capabilities Advertisement using ALTO", Work in Progress, Internet-Draft, draft-ietf-alto-cdni-request-routing-alto-16, 12 January 2021, <http://www.ietf.org/internet-drafts/draft-ietf-alto-cdni-request-routing-alto-16.txt>.

[I-D.ietf-alto-path-vector]
Gao, K., Lee, Y., Randriamasy, S., Yang, Y., and J. Zhang, "ALTO Extension: Path Vector", Work in Progress, Internet-Draft, draft-ietf-alto-path-vector-13, 20 November 2020, <http://www.ietf.org/internet-drafts/draft-ietf-alto-path-vector-13.txt>.

[RFC3849] Huston, G., Lord, A., and P. Smith, "IPv6 Address Prefix Reserved for Documentation", July 2004.

[RFC5511] Farrel, A., "Routing Backus-Naur Form (RBNF): A Syntax Used to Form Encoding Rules in Various Routing Protocol Specifications", April 2009.

[RFC5737] Arkko, J., Cotton, M., and L. Vegoda, "IPv4 Address Blocks Reserved for Documentation", January 2010.

[RFC7921] Atlas, A., Halpern, J., Hares, S., Ward, D., and T. Nadeau, "An Architecture for the Interface to the Routing System", RFC 7921, DOI 10.17487/RFC7921, June 2016, <https://www.rfc-editor.org/info/rfc7921>.

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

Appendix A. Features introduced with the Entity Property Maps extension

The Entity Property Maps extension described in this document introduces a number of features that are summarized in table below. The first column provides the name of the feature. The second column provides the section number of this document that gives a high level description of the feature. The third column provides the section number of this document that gives a normative description relating to the feature, when applicable.

Feature	High-level description	Related normative description
Entity	Section 3.1	Section 5.1.3
Entity domain (ED)	Section 3.2	
Entity domain type	Section 3.2.1	Section 5.1.1
Entity domain name	Section 3.2.2	Section 5.1.2
Entity property (EP) type	Section 3.3	Section 5.2, Section 5.2.1, Section 5.2.2, Section 5.2.3
Entity property map	Section 3.4	Section 7, Section 8
Resource-specific ED name	Section 4.2	Section 5.1.2, Section 5.1.2.1
Resource-specific EP value	Section 4.3	Section 5.2.3
Entity Hierarchy and property inheritance	Section 4.4	Section 5.1.4
Defining information resource	Section 4.6, Section 4.7	Section 12.3.2, Section 12.4

Table 4: Features introduced with ALTO Entity Property Maps

Authors' Addresses

Wendy Roome
Nokia Bell Labs (Retired)
124 Burlington Rd
Murray Hill, NJ 07974
United States of America
Phone: +1-908-464-6975
Email: wendy@wdroome.com

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

Y. Richard Yang
Yale University
51 Prospect Street
New Haven, CT 06511
United States of America
Phone: +1-203-432-6400
Email: yry@cs.yale.edu

Jingxuan Jensen Zhang
Tongji University
4800 Cao'An Hwy
Shanghai
201804
China
Email: jingxuan.n.zhang@gmail.com

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