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

W. Roome
S. Randriamasy
Nokia Bell Labs
Y. Yang
Yale University
J. Zhang
Tongji University
K. Gao
Sichuan University
March 9, 2020

Unified Properties for the ALTO Protocol
draft-ietf-alto-unified-props-new-11

Abstract

This document extends the Application-Layer Traffic Optimization (ALTO) Protocol [RFC7285] by generalizing the concept of "endpoint properties" to generic types of entities, and by presenting those properties as maps, similar to the network and cost maps in [RFC7285].

Requirements Language

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

Status of This Memo

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

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

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

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

Table of Contents

1. Introduction

The ALTO protocol [RFC7285] introduces the concept of "properties" attached to "endpoint addresses", and 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.

First, the EPS allows properties to be associated with only endpoints which are identified by individual communication addresses like IPv4 and IPv6 addresses. It is reasonable to think that collections of endpoints, as defined by CIDRs [RFC4632] or PIDs, may also have properties. Furthermore, recent ALTO use cases show that properties of network flows [RFC7011] and routing elements [RFC7921] are also very useful. Since the EPS cannot be extended to those generic entities, new services, with new request and response messages, would have to be defined for them.

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 which is also not globally unique. Additionally, a generic entity such as a PID may have an identifier that is not globally unique. For example, a PID identifier may be used in multiple network maps, where in each network map, this PID identifier points to a different set of addresses.

Third, the EPS is only defined as a POST-mode service. Clients must request the properties for an explicit set of endpoint addresses. By contrast, [RFC7285] defines a GET-mode cost map resource which

returns all available costs, so a client can get 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. However, in practice, it is highly unlikely that properties will be defined for every endpoint address. It is much more likely that properties may be defined for only a subset of endpoint addresses, and the specification of properties uses an aggregation representation to allow enumeration. This is particularly true if blocks of endpoint addresses with a common prefix (e.g., a CIDR) 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 the three limitations, this document specifies a 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 as defined in [REF path-vector], or other physical or logical objects used by ALTO. 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 entities in it.

- 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, detailed in Section 7 and Filtered Property Map, detailed in Section 8. The former is a GET-mode resource that 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 [RFC7285]. The latter is a POST-mode resource that returns the values for a set of properties and entities requested by the client, and is analogous to a filtered network map or a filtered cost map.

The protocol extension defined in this document is extensible. New entity domain types can be defined without revising the specification

defined in this document. Similarly, new cost metrics and new endpoint properties can be defined in other documents without revising the protocol specification defined in [RFC7285].

This document subsumes the Endpoint Property Service defined in [RFC7285], although that service may be retained for legacy clients (see Section 9).

2. Basic Features of the Unified Property Extension

The purpose of this extension is to convey properties on objects that extend ALTO Endpoints and are called ALTO Entities, entities for short. This section introduces the basic features involved in ALTO Property Maps.

2.1. 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 that is 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, where as all endpoints are entities, not all entities are endpoints.

Examples of entities are:

- an ALTO endpoint, defined in [RFC7285], 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 abstract 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;

- a country, identified by its country code defined by ISO 3166, to which applications such as CDN providers associate properties and capabilities;

- a TCP/IP network flow, that is identified by a TCP/IP 5-Tuple specifying its source and destination addresses and port numbers and the utilized protocol;

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

- an abstract network element, that represents an abstraction of a network part such as a routable network node, one or more link, a network domain or their aggregation.

2.2. Entity Domain

An entity domain defines a set of entities of the same type. This type, also called entity domain type, defines the semantics of a type of entity. Entity domain types could be defined in different documents. For example: the present document defines entity domain types "ipv4", "ipv6" and "pid" in sections Section 5.1 and Section 5.2; the entity domain type "ane", that defines Abstract Network Elements (ANEs), is introduced in [I-D.ietf-alto-path-vector].

An entity domain also has a name. The name and type of an entity domain can be the same. This is the case for the abovementionned types "ipv4", "ipv6" and "pid". The name of an entity domain may however be different from its type, in particular when the identifier of its entities cannot be recognized outside this domain. For example: an entity "mypid10" of domain type "pid" is only recognized with respect to a given network map resource and may be undefined in other network maps, or may even map to a different set of addresses. This document addresses this case in Section 3.2 and related.

2.3. Entity Property

An entity property defines a property of an entity. It is similar to the endpoint property defined by Section 7.1 of [RFC7285]. It can convey either network-aware or network-agnostic information.

For example:

- an entity in the "ipv4" domain may have a property whose value is an Autonomous System (AS) number indicating the AS that owns this IPv4 address,

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

It should be noted that some objects may be both entities and properties. For example, a PID may be both a property of an "ipv4" entity and an entity on which a Client may query properties such as geographical location.

2.4. 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 a set of entities. These entities may be of different types. For example, an ALTO property map may define the ASN property for both "ipv4" and "ipv6" type of entities.

The present extension also introduces a new media type.

This document uses the same definition of the information resource as defined by [RFC7285]. Each information resource usually has a JSON format representation following a specific schema defined by its media type. In the present case, an ALTO property map resource is represented by a JSON object of type InfoResourcePropertyMap and defined by the media type "application/alto-propmap+json".

A Property Map can be queried as a GET-mode resource, thus conveying values of 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.

3. Advanced Features of the Unified Property Extension

3.1. Entity Identifier and Entity Domain

In [RFC7285], an endpoint has an identifier explicitly associated with the "ipv4" or "ipv6" address domain. Examples are "ipv4:192.0.2.14" and "ipv6:2001:db8::12". In this extension, an entity domain characterizes the type semantics and identifier format of its entities. And the identifier of an entity is explicitly associated with its entity domain. For instance: an entity that is an endpoint with an IPv4 address will have an identifier associated with the "ipv4" domain, like "ipv4:192.0.2.14"; an entity which is a PID will have an identifier associated with a "pid" domain, like "pid:mypid10".

In this document, an entity must be owned by exactly one entity domain. And an entity identifier must point to exactly one entity. Even if two entities in two different entity domains refer to the same physical or logical object, they are treated as different entities. For example, an IPv4 and IPv6 address.

3.2. Resource-Specific Entity Domain Name

Some entities are defined and identified in a unique and global way. This is the case for instance for entities that are endpoints identified by a routable IPv4 or IPv6 address. The entity domain for

such entities can be globally defined and named "ipv4" or "ipv6". Those entity domains are also called resource-agnostic entity domains in this document, as they are not associated to any specific ALTO information resources.

Some other entities and entity types are only defined relatively to a given information resource. This is the case for entities of domain "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 an information resource of type Network Map and 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. The reason is that "pid" denotes an entity domain type rather than an unambiguous identifier.

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

In such cases, an entity domain name is explicitly associated with an identifier of the information resource where these entities are defined. Using a resource-specific entity domain name, an ALTO Property Map may unambiguously indicate entity domains of the same type, on which entity properties may be queried. Example resource-specific entity domain names may look like: "netmap1.pid" or "netmap2.pid". This allows to identify two distinct PID entities such as "netmap1.pid:mypid10" or "netmap2.pid:mypid10". Resource-specific entity domain name will be specified in Section 4.1.2.

3.3. Resource-Specific Entity Property

An entity may have properties of same type, whose values are associated to different information resources. For instance, entity "192.0.2.34" defined in the "ipv4" domain may have two "pid" properties defined in two different network maps "netmap1" and "netmap2". These properties will likely have different values in "netmap1" and "netmap2". To distinguish between them, this document uses the same approach proposed as in Section 10.8.1 of [RFC7285], which is called "Resource-Specific Entity Property". When a property value depends on a given information resource, the identifier of the 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", may 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 in a single query. Specifications are provided in Section 4.2.

3.4. Entity Hierarchy and Property Inheritance

Enumerating all individual entities is inefficient for both the Client and the Server.

- For the Client, even if it only wants to request properties for a "/24" ipv4 subnet, using the individual ipv4 entity, it has to enumerate all 256 entities.

- For the Server, in some cases, most of the entities may have the same properties. Simply enumerating all of them may introduce a lot of reduncency in the payload. For example, all the individual ipv4 entities in a "/24" ipv4 subnet is usually owned by the same AS. When a Client requests the ASN property for this ipv4 subnet, using the individual ipv4 address, the Server has to repeat the same ASN property for 256 times in the worst case.

To reduce the size of the property map request and response payloads, this document introduces, when appicable, an approach called "Property Inheritance". This approach consists of two parts: Entity Hierarchy and Property Inheritance.

- Entity Hierarchy: This approach allows an entity domain to support using a single identifier to identify a set of indivudial entities. For example, a CIDR can be used to identify a set of ipv4 or ipv6 entities. Such an identifier is called a hierarchical entity identifier, as the set identified by it can be hierarachical. For example, the CIDR "ipv4:192.0.1.0/24" covers all the individual ipv4 entities identified the CIDR "ipv4:192.0.1.0/26".

- Property Inheritance: This approach allows a property map to define a property for a hierarchical entity identifier. An undefined property of an entity may inherit the value of the property defined for a hierarchical entity identifier. For example, a property map only defines the ASN property for a hierarchical ipv4 entity identifier "ipv4:192.0.1.0/24". When receiving this property map, a Client could infer that the ipv4 entity "ipv4:192.0.1.1" inherits the same ASN property even if it does not appear in the property map.

The detailed specification will be specified in Section 4.1.4.

3.5. Applicable Entity Domains and Properties in the Property Map Capabilities

A property is not necessarily applicable to any domain, or an ALTO Server may just not support it for all applicable domains. For instance, a property reflecting link bandwidth is likely not defined on entities of a domain of type "country-code". Therefore an ALTO server supporting Property Maps specifies the properties that can be queried on the different domains defined in this server.

In Section 6 and related, this document explains how the IRD capabilities of a Property Map unambiguously expose what type of properties on what entity domains a Client can query. For short, a field called "mapping" enumerates the entity domains supported by the Property Map; For each entity domain, a list of applicable properties is provided. An example can be found in Section 10.4. Using resource-agnostic or resource-specific entity domains and properties allows to formulate compact and unambiguous entity property queries relating to one or more information resources, in particular:

- avoid a Client to query a property on entity domains on which P is not defined,
- query for en entity E property values defined in different information resources,
- query a property P on entities E defined in different information resources.

Specifications will be provided in Section 7.4.

3.6. Connection between Resource-Specific Entity Domain/Entity Property Mapping and Information Resources

Although the IRD capabilities of a Property Map can expose the supported mappings in this property map, it may still not be clear to a Client what a resource-specific entity domain is, and what an applicable resource-specific entity property means, as those concepts are not defined in other ALTO information resources. For example, a Client should understand that:

- a local IPv4 entity domain "netmap1.ipv4" includes the IPv4 addresses appearing in the "ipv4" field of the endpoint address group of each PID in the network map "netmap1";

- a "netmap1.pid" property of an IPv4 entity "ipv4:192.0.1.1" indicates the PID defined by the network map "netmap1" and including the IPv4 address "ipv4:192.0.1.1" in its endpoint address group.

To help the client understanding these connections, this document requests two new IANA registries for each information resource to define the connection to each supported resource-specific entity domain and entity property mapping respectively. Such a connection is called "Information Resource Export", to explain what is an resource-specific entity domain or an entity property mapping exported by an information resource. Examples of "Information Resource Exports" of existing ALTO information resources are provided in Section 6. Specifications are provided in Section 6.1. The details of these new IANA registries are provided in Section 12.4 and Section 12.5.

4. Protocol Specification: Basic Data Type

4.1. Entity Domain

4.1.1. Entity Domain Type

An entity domain has a type, which is defined 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), hyphen ("-", U+002D), and low line ("_", U+005F). For example, the strings "ipv4", "ipv6", and "pid" are valid entity domain types.

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

An entity domain type defines the semantics of a type of entity domains. Each entity domain type MUST be registered with the IANA. The format of the entity identifiers (see Section 4.1.3) in that type of entity domains, as well as any hierarchical or inheritance rules (see Section 4.1.4) for those entities, MUST be specified at the same time.

4.1.2. Entity Domain Name

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

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

This document distinguishes three types of entity domains: resource-specific entity domains, self-defined entity domains, and resource-agnostic entity domains. Their entity domain names are derived as follows.

Each ALTO information resource MAY define a resource-specific entity domain (which could be empty) in a given entity domain type. A resource-specific entity domain is identified by an entity domain name derived as follows. It MUST start with a resource ID using the ResourceID type defined in [RFC7285], followed by the "." separator (U+002E), followed by an EntityDomainType typed string. For example, if an ALTO server provides two network maps "netmap-1" and "netmap-2", they can define two different "pid" domains identified by "netmap-1.pid" and "netmap-2.pid" respectively. To be simplified, in the scope of a specific information resource, the resource-specific entity domain defined by itself can be identified by the "." EntityDomainTyep without the ResourceID.

When the associated information resource of a resource-specific entity domain is the current information resource itself, this resource-specific entity domain is a self-defined entity domain, and its ResourceID SHOULD be ignored from its entity domain name.

Given a set of ALTO information resources, there MAY be a resource-agnostic entity domain in a given entity domain type amongst them. A resource-agnostic entity domain is simply identified by its entity domain type. For example, given two network maps "net-map-1" and "net-map-2", "ipv4" and "ipv6" identify two resource-agnostic Internet address entity domains (see Section 5.1) between them.

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.

4.1.3. 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::1/48".

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

4.1.4. Hierarchy and Inheritance

To make the representation efficient, some types of entity domains MAY allow the ALTO client/server to use a hierarchical format entity identifier to represent a block of individual entities. e.g., In an IPv4 domain "net1.ipv4", a cidr "net1.ipv4:192.0.2.0/26" represents 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.

4.2. 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 be identified. One entity MAY have multiple properties in the same type.

4.2.1. 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).

Each entity property type MUST be registered with the IANA. The intended semantics of the entity property type MUST be specified at the same time.

To distinguish with the endpoint property type, the entity property type has the following features.

- Some entity property types may be applicable to entities in only particular types of entity domains, not all. For example, the "pid" property is not applicable to entities in a "pid" typed entity domain, but is applicable to entities in the "ipv4" or "ipv6" domains.

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

4.2.2. 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.

When the associated information resource of the entity property is the current information resource itself, the ResourceID in the property name SHOULD be ignored. For example, the ".asn" property of an "ipv4" entity indicates the AS number of the AS which this IPv4 address is owned by.

5. Entity Domain Types

This document requires 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.

5.1. 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.

5.1.1. IPv4 Domain

5.1.1.1. Entity Domain Type

ipv4

5.1.1.2. Domain-Specific Entity Identifiers

Individual addresses are strings as specified by the IPv4Addresses rule of Section 3.2.2 of [RFC3986]; Hierarchical addresses are prefix-match strings as specified 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. Thus "ipv4:192.0.2.0" and "ipv4:192.0.2.0/32" are equivalent.

5.1.2. IPv6 Domain

5.1.2.1. Entity Domain Type

ipv6

5.1.2.2. Domain-Specific Entity Identifiers

Individual addresses are strings as specified by Section 4 of [RFC5952]; Hierarchical addresses are prefix-match strings as specified in Section 7 of [RFC5952]. 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.

5.1.3. 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 a hierarchical Internet address C which prefix-matches 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 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 another hierarchical address C' which covers all IP addresses in C, and C' has a shorter prefix length than C, then C MAY inherits the property from C'. If there are multiple such hierarchical addresses like C', C MUST inherit from the hierarchical address 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 behaviours:

- If that entity would inherit a value for that property, then the ALTO server MUST return a "null" value for that property. In this case, the ALTO client MUST recognize a "null" value as "no value" and "do not apply the inheritance rules for this property."

- 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.

5.2. PID Domain

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

5.2.1. Entity Domain Type

pid

5.2.2. Domain-Specific Entity Identifiers

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

5.2.3. Hierarchy and Inheritance

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

5.2.4. Relationship To Internet Addresses Domains

The PID domain and the Internet address domains are completely independent; the properties associated with a PID have no relation to the properties associated with the prefixes or endpoint addresses in that PID. An ALTO server MAY choose to assign some or all properties of a PID to the prefixes in that PID.

For example, suppose "PID1" consists of the prefix "ipv4:192.0.2.0/24", and has the property "P" with value "v1". The Internet address entities "ipv4:192.0.2.0" and "ipv4:192.0.2.0/24" in

the IPv4 domain MAY have a value for the property "P", and if they do, it is not necessarily "v1".

5.3. Internet Address Properties vs. PID Properties

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

The PID domain is RECOMMENDED for properties that arise from the definition of the PID, rather than from the Internet address prefixes in that PID.

For example, because Internet addresses are allocated to service providers by blocks of prefixes, an "ISP" property would be best associated with the Internet address domain. On the other hand, a property that explains why a PID was formed, or how it relates to a provider's network, would best be associated with the PID domain.

6. Entity Domains and Property Mappings in Information Resources

6.1. Information Resource Export

Each information resource MUST export a set of entity domains and entity property mappings (which can be empty).

6.1.1. Resource-Specific Entity Domain Export

Each type of information resource MAY export different types of entity domains. For example, a network map resource MUST export a "pid" domain, an "ipv4" domain and an "ipv6" domain (which may be empty); if a facilitated endpoint type "ecgi" and its corresponding entity domain type defined for cellular network addresses are supported in a future ALTO extension, a network map supporting the "ecgi" endpoint type MUST also export an "ecgi" domain.

When a new ALTO information resource type is registered, if this type of information resource MAY export an existing type of entity domain, the corresponding document MUST define how to export such type of entity domain from such type of information resource.

When a new entity domain type is registered, if an existing type of information resource MAY export an entity domain in this entity domain type, the corresponding document MUST define how to export such type of entity domain from such type of information resource.

6.1.2. Entity Property Mapping Export

For each entity domain which MAY be exported by an information resource, this information resource MAY also export mappings from this entity domain to some entity property. For example, a network map resource MUST map an "ipv4" entity to its "pid" property; if a facilitated ALTO CDNI FCI information resource including "capabilities with footprint restrictions" [RFC8008] supporting ALTO PIDs as a new footprint type, this information ressource MUST map a "pid" entity to its corresponding "cdni-fci-capabilities" property.

When a new ALTO information resource type is registered, if this type of information resource MAY export an entity domain in an existing entity domain type, and map entities in this entity domain to an existing type of entity property, the corresponding document MUST define how to export such type of an entity property.

When a new ALTO entity domain type or a new entity property type is defined, if an existing type of resource MAY export an entity domain in this entity domain type, and map entities in this entity domain to this type of entity property, the corresponding document MUST define how to export such type of an entity property.

6.2. Network Map Resource

The ALTO network map resource defined by the media type "application/alto-networkmap+json" exports the following types of entity domains and entity property mappings.

6.2.1. Resource-Specific Entity Domain

An ALTO network map resource defines a "pid" domain, an "ipv4" domain and an "ipv6" domain by follows:

- The defined "pid" domain includes all PIDs in keys of the "network-map" object.

- The defined "ipv4" domain includes all IPv4 addresses appearing in the "ipv4" field of the endpoint address group of each PID.

- The defined "ipv6" domain includes all IPv6 addresses appearing in the "ipv6" field of the endpoint address group of each PID.

6.2.2. Entity Property Mapping

For each of the preceding entity domains, an ALTO network map resource provides the properties mapping as follows:

ipv4 -> pid: An "networkmap" typed resource can map an "ipv4" entity to a "pid" property whose value is a PID defined by this "networkmap" resource and including the IPv4 address of this entity.

ipv6 -> pid: An "networkmap" typed resource can map an "ipv6" entity to a "pid" property whose value is a PID defined by this "networkmap" resource and including the IPv6 address of this entity.

6.3. Endpoint Property Resource

The ALTO endpoint property resource defined by the media type "application/alto-endpointprop+json" exports the following types of entity domains and entity property mappings.

6.3.1. Resource-Specific Entity Domain

An ALTO endpoint property resource defined an "ipv4" domain and an "ipv6" domain by follows:

- The defined "ipv4" domain includes all IPv4 addresses appearing in keys of the "endpoint-properties" object.

- The defined "ipv6" domain includes all IPv6 addresses appearing in keys of the "endpoint-properties" object.

6.3.2. Entity Property Mapping

For each of the preceding entity domains, an ALTO endpoint property resource exports the properties mapping from it to each supported global endpoint property. The property value is the corresponding global endpoint property value in the "endpiont-properties" object.

6.4. Property Map Resource

To avoid the nested reference and its potential complexity, this document does not specify the export rule of resource-specific entity domain and entity property mapping for the ALTO property map resource defined by the media type "application/alto-propmap+json" (see Section 7.1).

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 10.5 gives an example of a property map request and its response.

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 dependent resources of this property map. It is an array of the resource ID(s) of the resource(s).

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 SHOULD only return self-defined properties and resource-specific properties which depend on the same resource as the entity does. The ALTO client SHOULD ignore the resource-specific property in this entity if their mapping is not registered in the ALTO Resource Entity Property Transfer Registry of the type of the corresponding resource.

- If the entity is in a shared entity domain, the ALTO server SHOULD return self-defined properties and all resource-specific properties defined for all resource-specific entities which have the same domain-specific entity identifier as this entity does.

For efficiency, the ALTO server SHOULD omit property values that are inherited rather than explicitly defined; if a client needs inherited values, the client SHOULD use the 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 10.6, Section 10.7, Section 10.8 and Section 10.9 give examples of filtered property map requests and responses.

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:

```
  object {
    EntityID             entities<1..*>;
    EntityPropertyName   properties<1..*>;
  } ReqFilteredPropertyMap;
```

with fields:

entities: List of entity identifiers for which the specified properties are to be returned. The ALTO server MUST interpret entries appearing multiple times as if they appeared only once. The domain of each entity MUST be included in the list of entity domains in this resource's "capabilities" field (see Section 8.4).

properties: List of properties to be returned for each entity. Each specified property MUST be included in the list of properties in this resource's "capabilities" field (see Section 8.4). The ALTO server MUST interpret entries appearing multiple times as if they appeared only once.

Note that the "entities" and "properties" fields MUST have at least one entry each.

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. 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 as follows:

- An entity identifier in "entities" in the request is invalid if:
  - The domain of this entity is not defined in the "entity-domains" capability of this resource in the IRD;
  - The entity identifier is an invalid identifier in the entity domain.

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

  If an entity identifier in "entities" in 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 this entity identifier.

- A property name in "properties" in the request is invalid if this property name is not defined in the "properties" capability of this resource in the IRD.

  It is not an error that a filtered property map resource does not define a requested property's value for a particular entity. 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.

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 in the shared entity domain, 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 version tags of resources which requested resource-specific entity domains and 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 cover properties for all identifiers in this hierarchical identifier.

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

- If a property for a requested entity is inherited from another entity not included in the request, the response SHOULD 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 SHOULD 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 SHOULD 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.

- If an entity in the response is already covered by some other entities in the same response, it SHOULD be removed from the response for compactness. For example, 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.

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 the functionality of the Endpoint Property Service (EPS) defined in Section 11.4 of [RFC7285], it is RECOMMENDED that the EPS be deprecated in favor of Property Map and Filtered Property Map. However, ALTO servers MAY provide an EPS for the benefit of legacy clients.

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 upon, while global property names have no such prefix. The property map and the filtered property map defined in this document defines the similar categories for entity properties. The difference is that there is no "global" entity properties but the "self-defined" entity properties as the special case of the "resource-specific" entity properties instead.

9.3. Impact on Other Properties

In general, there should be little or no impact on other previously defined properties. The only consideration is that properties can now be defined on hierarchical entity identifiers, rather than just individual entity identifiers, which might change the semantics of a property.

10. Examples

10.1. Network Map

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

```
defaultpid:  ipv4:0.0.0.0/0  ipv6:::0/0
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:::0/0
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 properties for Internet address domains, "ISP", "ASN", "country" and "state", with the following values:

	ISP	ASN	country	state
ipv4:192.0.2.0/23:	BitsRus	-	us	-
ipv4:192.0.2.0/28:	-	12345	-	NJ
ipv4:192.0.2.16/28:	-	12345	-	CT
ipv4:192.0.2.1:	-	-	-	PA
ipv4:192.0.3.0/28:	-	12346	-	TX
ipv4:192.0.3.16/28:	-	12346	-	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:       12345
                pid:pid2:       12346
```

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

10.3. Properties for Abstract Network Elements

Additionally, the examples in this section consider a facilitated entity domain: "ane" (Abstract Network Element). Abstract network elements allow ALTO clients to discover information beyond the end-to-end routing costs. Examples of abstract network elements include:

Forwarding elements: Forwarding elements include optical wires, physical layer links, IP tunnels, etc. Forwarding elements share the common property "maxresbw".

Value-added services: Value-added services include HTTP caches, 5G UPF nodes, mobile edge computing, etc. Value-added services share the common property "persistent-entities", which contains information that points to the entry point of the service. Different value-added services may have specific properties, e.g., an abstract network element of a mobile edge may provide a list of flavors to the client.

```
           maxresbw    persistent-entities     mec-flavors
ane:L001   100 Mbps
ane:L002   100 Mbps
ane:CACHE1             http-proxy:192.0.2.1
ane:MEC01              mec:192.0.2.1         {gpu:2G, ssd:128G}
ane:MEC02              mec:192.0.2.2         {gpu:1G, ssd:128G}
```

The "ane" entities are usually not used alone, but associated with other ALTO resources, e.g., cost maps. It means that the ALTO server may not define a property map resource to provide properties of "ane" entities. The property map payload for "ane" entities may be provided in the response of other ALTO resources in some way.

10.4. Information Resource Directory (IRD)

The following IRD defines the relevant resources of the ALTO server. It provides two property maps, one for the "ISP" and "ASN" properties, and another for the "country" and "state" properties. The server could have provided a single property map for all four properties, but did not, presumably because the organization that

runs the ALTO server believes any given client is not interested in all four properties.

The server provides two filtered property maps. The first returns all four properties, and the second just returns the "pid" property for the default network map.

The filtered property maps for the "ISP", "ASN", "country" and "state" properties do not depend on the default network map (it does not have a "uses" capability), because the definitions of those properties do not depend on the default network map. The Filtered Property Map for the "pid" property does have a "uses" capability for the default network map, because that defines the values of the "pid" property.

Note that for legacy clients, the ALTO server provides an Endpoint Property Service for the "pid" property for the default network map.

The server also provides a facilitated ALTO resource which accepts the filtered cost map request but returns a multipart message including a cost map and an associated property map for "ane" entities.

```
"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"
  },
  .... property map resources ....
  "ia-property-map" : {
    "uri" : "http://alto.example.com/propmap/full/inet-ia",
    "media-type" : "application/alto-propmap+json",
    "uses": [ "default-network-map", "alt-network-map" ],
    "capabilities" : {
      "mappings": {
        "ipv4": [ ".ISP", ".ASN" ],
        "ipv6": [ ".ISP", ".ASN" ]
      }
    }
  },
```

```
          "iacs-property-map" : {
            "uri" : "http://alto.example.com/propmap/full/inet-iacs",
            "media-type" : "application/alto-propmap+json",
            "accepts": "application/alto-propmapparams+json",
            "uses": [ "default-network-map", "alt-network-map" ],
            "capabilities" : {
              "mappings": {
                "ipv4": [ ".ISP", ".ASN", ".country", ".state" ],
                "ipv6": [ ".ISP", ".ASN", ".country", ".state" ]
              }
            }
          },
          "region-property-map": {
            "uri": "http://alto.exmaple.com/propmap/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" ]
            }
          },
          "path-vector-map": {
```

```
          "uri": "http://alto.example.com/costmap/pv",
          "media-type":
            "multipart/related;type=applicatoin/alto-costmap+json",
          "accepts": "applicatoin/alto-costmapfilter+json",
          "capabilities": {
            "cost-type-names": ["path-vector"],
            "ane-properties": ["maxresbw", "persistent-entities",
                               "mec-flavors"]
          },
          "uses": [ "default-network-map" ]
        }
      }
```

Figure 8: Example IRD

10.5. Property Map Example

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

Note that, to be compact, the response does not include the entity "ipv4:192.0.2.0", 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 "ipv4:192.0.3.0/28". 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: ###
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": "12345"},
    "ipv4:192.0.3.0/27":   {".ASN": "12346"}
  }
}
```

10.6. Filtered Property Map Example #1

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

Note that the value of "state" for "ipv4:192.0.2.0" is the only explicitly defined property; the other values are all derived by the inheritance rules for Internet address entities.

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

{
  "entities" : [ "ipv4:192.0.2.0",
                 "ipv4:192.0.2.1",
                 "ipv4:192.0.2.17" ],
  "properties" : [ ".ISP", ".ASN", ".state" ]
}
```

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

{
  "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": "12345", ".state": "PA"},
    "ipv4:192.0.2.1":
          {".ISP": "BitsRus", ".ASN": "12345", ".state": "NJ"},
    "ipv4:192.0.2.17":
          {".ISP": "BitsRus", ".ASN": "12345", ".state": "CT"}
  }
}
```

10.7. Filtered Property Map Example #2

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

Note that 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 listed in the request explicitly. 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: ###
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", ".country", ".state" ]
}

HTTP/1.1 200 OK
Content-Length: ###
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":  {".country": "us"},
    "ipv4:192.0.2.0/28":  {".ASN": "12345",
                           ".state": "NJ"},
    "ipv4:192.0.2.16/28": {".ASN": "12345",
                           ".state": "CT"},
    "ipv4:192.0.2.0":     {".state": "PA"},
    "ipv4:192.0.3.0/26":  {".country": "us"},
    "ipv4:192.0.3.0/28":  {".ASN": "12345",
                           ".state": "TX"},
    "ipv4:192.0.3.16/28": {".ASN": "12345",
                           ".state": "MN"}
  }
}
```

10.8. 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: ###
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: ###
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"}
  }
}
```

10.9. Filtered Property Map Example #4

The following example uses the filtered property map resource to request the "region" property for several PIDs defined in "default-network-map". The value of the "region" property for each PID is not defined by "default-network-map", but the reason why the PID is defined by the network operator.

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

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

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

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

10.10. Property Map in Path Vector Example #1

The following example requests the "maxresbw", "persistent-entities" and "mec-flavors" properties for abstract network elements between "pid1" and "pid3" in "default-network-map".

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

  {
    "cost-type": {
      "cost-mode": "array",
      "cost-metric": "ane-path"
    },
    "pids": {
      "srcs": [ "pid1" ],
      "dsts": [ "pid3" ]
    },
    "ane-properties": ["maxresbw", "persistent-entities", "mec-flavors"]

  }

   HTTP/1.1 200 OK
   Content-Length: [TBD]
   Content-Type: multipart/related; boundary=example-1;
                 type=application/alto-costmap+json

   --example-1
   Content-Id: costmap
   Content-Type: application/alto-costmap+json

   {
     "meta": {
       "vtag": {
         "resource-id": "cost-map-pv.costmap",
         "tag": "d827f484cb66ce6df6b5077cb8562b0a"
       },
       "dependent-vtags": [
         {
           "resource-id": "my-default-networkmap",
           "tag": "75ed013b3cb58f896e839582504f6228"
         }
       ],
       "cost-type": {
         "cost-mode": "array",
         "cost-metric": "ane-path"
       }
     },
     "cost-map": {
       "pid1": {
```

```
        "pid3": [ "ane:L001", "ane:L002", "ane:MEC01", "ane:MEC02" ],
      }
    }
  }
  --example-1
  Content-Id: propmap
  Content-Type: application/alto-propmap+json

  {
    "meta": {
      "dependent-vtags": [
        {
          "resource-id": "cost-map-pv.costmap",
          "tag": "d827f484cb66ce6df6b5077cb8562b0a"
        }
      ]
    },
    "property-map": {
      "ane:L001": { "maxresbw": 100000000 },
      "ane:L002": { "maxresbw": 100000000 },
      "ane:MEC01": { "persistent-entities": "mec:192.0.2.1",
                     "mec-flavors": [ {"gpu": "2G", "ssd": "128G"}]},
      "ane:MEC02": { "persistent-entities": "mec:192.0.2.2",
                     "mec-flavors": [ {"gpu": "1G", "ssd": "128G"}]}
    }
  }
```

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.

A particular fundamental security consideration when an ALTO server provides a Property Map is to define precisely the policies on who can access what properties for which entities. Security mechanisms such as authentication and confidentiality mechanisms then should be applied to enforce the policy. For example, a policy can be that a property P can be accessed only by its owner (e.g., the customer who is allocated a given IP address). Then, the ALTO server will need to

deploy corresponding mechanisms to realize the policy. The policy may allow non-owners to access a coarse-grained value of the property P. In such a case, the ALTO server may provide a different URI to provide the information.

12. IANA Considerations

This document defines additional application/alto-* media types, and extends the ALTO endpoint property registry.

12.1. application/alto-* Media Types

This document registers two additional ALTO media types, listed in Table 1.

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

Table 1: Additional ALTO Media Types.

Type name: application

Subtype name: This document registers multiple subtypes, as listed in Table 1.

Required parameters: n/a

Optional parameters: n/a

Encoding considerations: Encoding considerations are identical to those specified for the "application/json" media type. See [RFC7159].

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

Interoperability considerations: This document specifies formats of conforming messages and the interpretation thereof.

Published specification: This document is the specification for these media types; see Table 1 for the section documenting each media type.

Applications that use this media type: ALTO servers and ALTO clients either stand alone or are embedded within other applications.

Additional information:

Magic number(s): n/a

File extension(s): This document uses the mime type to refer to protocol messages and thus does not require a file extension.

Macintosh file type code(s): n/a

Person & email address to contact for further information: See Authors' Addresses section.

Intended usage: COMMON

Restrictions on usage: n/a

Author: See Authors' Addresses section.

Change controller: Internet Engineering Task Force (mailto:iesg@ietf.org).

12.2. ALTO Entity Domain Type Registry

This document requests IANA to create and maintain the "ALTO Entity Domain Type Registry", listed in Table 2.

Identifier	Entity Identifier Encoding	Hierarchy & Inheritance
ipv4	See Section 5.1.1	See Section 5.1.3
ipv6	See Section 5.1.2	See Section 5.1.3
pid	See Section 5.2	None

Table 2: ALTO Entity Domains.

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

12.2.1. 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 entity identifier for the candidate domain identifier. This has been done for the identifiers "ipv4" and "ipv6" in Table 2.

- Whether the candidate entity identifier of the type of the entity domain is able to be an endpoint address, 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 identifiers match a known "network" address type?

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

+ 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].

+ 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 12.2.2 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 12.2.2.

12.2.2. ALTO Entity Domain Type Registration Process

New ALTO entity domain types are assigned after IETF Review [RFC5226] 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 follow the same procedure.

Registered ALTO entity domain type identifiers MUST conform to the syntactical requirements specified in Section 4.1.2. Identifiers are to be recorded and displayed as strings.

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

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

o Entity Identifier Encoding: The procedure for encoding the identifier of an entity of the registered type as an EntityID (see Section 4.1.3). If corresponding entity identifiers of an entity domain 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]. For the purpose of defining 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.

- 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 made aware of how (or if) the addressing scheme relates to private information and network proximity.

This specification requests registration of the identifiers "ipv4", "ipv6" and "pid", as shown in Table 2.

12.3. ALTO Entity Property Type Registry

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

To distinguish with the "ALTO Endpoint Property Type Registry", each entry in this registry is an ALTO entity property type defined in Section 4.2.1. Thus, registered ALTO entity property type identifier MUST conform to the syntactical requirements specified in that section.

The initial registered ALTO entity property types are listed in Table 3.

Identifier	Intended Semantics
pid	See Section 7.1.1 of [RFC7285]

Table 3: ALTO Entity Property Types.

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

- Identifier: The unique id for the desired ALTO entity property type. The format MUST be as defined in Section 4.2.1 of this document. It includes the information of the applied ALTO entity domain and the property name.

- 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.

This document requests registration of the identifier "pid", as shown in Table 3.

12.4. ALTO Resource-Specific Entity Domain Registries

12.4.1. Network Map

Media-type: application/alto-networkmap+json

Entity Domain Type	Intended Semantics
ipv4	See Section 6.2.1
ipv6	See Section 6.2.1
pid	See Section 6.2.1

Table 4: ALTO Network Map Resource-Specific Entity Domain.

12.4.2. Endpoint Property

Media-type: application/alto-endpointprop+json

Entity Domain Type	Intended Semantics
ipv4	See Section 6.3.1
ipv6	See Section 6.3.1

Table 5: ALTO Endpoint Property Resource-Specific Entity Domain.

12.5. ALTO Resource Entity Property Mapping Registries

12.5.1. Network Map

Media-type: application/alto-networkmap+json

Mapping Descriptor	Entity Domain Type	Property Type	Intended Semantics
ipv4 -> pid	ipv4	pid	See Section 6.2.2
ipv6 -> pid	ipv6	pid	See Section 6.2.2

Table 6: ALTO Network Map Entity Property Mapping.

13. Acknowledgments

The authors would like to thank discussions with Kai Gao, Qiao Xiang, Shawn Lin, Xin Wang, Danny Perez, and Vijay Gurbani. The authors thank Dawn Chen (Tongji University), and Shenshen Chen (Tongji/Yale University) for their contributions to earlier drafts.

14. References

14.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[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>.

[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>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <https://www.rfc-editor.org/info/rfc5226>.

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

[RFC7011] Claise, B., Ed., Trammell, B., Ed., and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information", STD 77, RFC 7011, DOI 10.17487/RFC7011, September 2013, <https://www.rfc-editor.org/info/rfc7011>.

[RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <https://www.rfc-editor.org/info/rfc7159>.

[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>.

[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>.

[RFC8008] Seedorf, J., Peterson, J., Previdi, S., van Brandenburg, R., and K. Ma, "Content Delivery Network Interconnection (CDNI) Request Routing: Footprint and Capabilities Semantics", RFC 8008, DOI 10.17487/RFC8008, December 2016, <https://www.rfc-editor.org/info/rfc8008>.

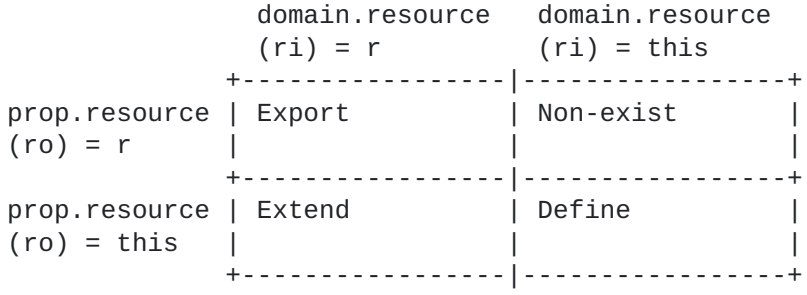
14.2. Informative References

[I-D.ietf-alto-path-vector]
Gao, K., Lee, Y., Randriamasy, S., Yang, Y., and J. Zhang, "ALTO Extension: Path Vector", draft-ietf-alto-path-vector-09 (work in progress), November 2019, <http://www.ietf.org/internet-drafts/draft-ietf-alto-path-vector-09.txt>.

Appendix A. Scope of Property Map

Using entity domains to organize entities, an ALTO property map resource can be regarded as given sets of properties for given entity domains. If we ignore the resource-agnostic entity domains, we can regard an ALTO property map resource as a set of (ri, di) => (ro, po) mappings, where (ri, di) means a resource-specific entity domain of type di defined by the information resource ri, and (ro, po) means a resource-specific entity property po defined by the information resource ro.

For each (ri, di) => (ro, po) mapping, the scope of an ALTO property map resource must be one of the cases in the following diagram:



where "this" represents the resulting property map resource, and "r" represents an existing ALTO information resource other the resulting property map resource.

- ri = ro = r ("export" mode): the property map resource just transforms the property mapping di => po defined by r into the unified representation format and exports it. For example: r = "netmap1", di = "ipv4", po = "pid". The property map resource exports the "ipv4 => pid" mapping defined by "netmap1".

- ri = r, ro = this ("extend" mode): the property map extends properties of entities in the entity domain (r, di) and defines a new property po on them. For example: the property map resource ("this") defines a "geolocation" property on domain "netmap1.pid".

- ri = ro = this ("define" mode): the property map defines a new intrinsic entity domain and defines property po for each entity in this domain. For example: the property map resource ("this") defines a new entity domain "asn" and defines a property "ipprefixes" on this domain.

- ri = this, ro = r: in the scope of a property map resource, it does not make sense that another existing ALTO information resource defines a property for this property map resource.

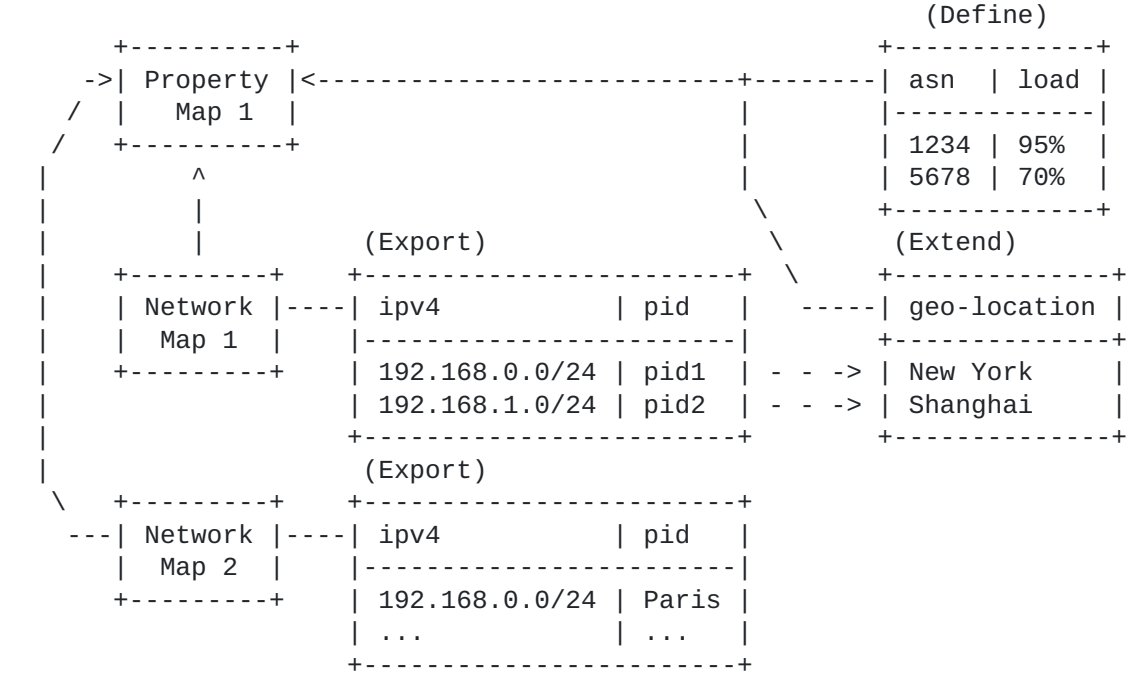
A.1. Example Property Map

The following figure shows an example property map called Property Map 1, which depends on two network maps and provides three sets of mappings by

- exporting a mapping from ipv4 entities to PIDs defined by two different network maps,

- extending geo-location properties to ipv4 entities defined by Network Map 1,

- and defining a new mapping from ASNs to traffic load properties.



More detailed examples are shown in Section 10.

Authors' Addresses

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

Phone: +1-908-464-6975
Email: wendy@wdroome.com

Sabine Randriamasy
Nokia Bell Labs
Route de Villejust
NOZAY 91460
FRANCE

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

Y. Richard Yang
Yale University
51 Prospect Street
New Haven, CT 06511
USA

Phone: +1-203-432-6400
Email: yry@cs.yale.edu

Jingxuan Jensen Zhang
Tongji University
4800 Caoan Road
Shanghai 201804
China

Email: jingxuan.n.zhang@gmail.com

Kai Gao
Sichuan University
Chengdu 610000
China

Email: kaigao@scu.edu.cn