Unified Properties for ALTO

-05 updates and discussion

Wendy Roome

Dawn Chen

Sabine Randriamasy

Richard Yang

Jensen Zhang

ALTO Interim Meeting

Dec 11, 2018

Unified Properties in a Nutshell

- The goal of the revision -05:
  - To address the remaining issues discussed during IETF 102
  - Make the document complete
  - Clarify the terminologies and definitions
  - To address the problems reported by WG in the mailing list
- Two technical issues are reported in -04:
  - Issue 1: Address block filtering cannot be handled correctly based on the previous protocol specification
  - Issue 2: The interpretation of resource dependencies in "uses" are not clear for the client
- Section is missing in previous revisions:
  - The format and process of the ALTO Entity Property Type Registry is not specified

Overview of Updates

- Updates from -04 to -05:
    - Wording and terms updates:
        - "entity-domain-types" -> "entity-domains",
        - "prop-types" -> "properties"
        - Make definitions of terms more convincing
    - Changed the response semantics of Filtered Property Map to address the remaining issue 1
    - Changed the definition of "Property Name" and the specification of "Uses" attribute of Property Map to address the remaining issue 2
    - Added a specification in the "Section 9.3 ALTO Entity Property Type Registry"
        - as a starting point (To be discussed in WG)
    - Added a new column "Mapping to ALTO Address Type" in ALTO Entity Domain Registry
        - to indicate the correlation explicitly

Update: Attributes Names

- Rename attributes in "capabilities":
  - "entity-domain-types" -> "entity-domains"
  - "prop-types" -> "properties"
- **Reason**: Both "entity-domains" and "properties" already imply the meaning of "types". So we remove the redundancy to make the format simple.
- Changes applied to Capabilities of both property map and filtered property map. (Sec 4.4 and Sec 5.4)

Update: Address Block Filtering

Basic principle: The client SHOULD be able to get or derive correct properties for each address in the requested address block.

Goal: 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. (Only the requested entities should be included in the previous revision.)

Solution: Three rules to enforce this goal. (Already discussed at IETF 102)

Updated Sections: Sec 4.6, Sec 5.6 and Sec 6.3

Update: Dependent Resources Interpretation

- **Basic principle**: The client SHOULD be able to interpret the property map correctly.
  - Means interpreting both entities and the properties of entities correctly.
  - To guarantee this, the server SHOULD provide a pointer to the necessary (required) dependent resources in the corresponding property map.
- **Example**:

```
~~"uses": ["default-network-map"],~~
"capabilities": {
  "entity-domains": ["ipv4", "ipv6"],
  "properties": ["pid"]
}
```

- If no "uses" specified, the client cannot understand and use the "pid" property values.
- But how does the client know a provided dependent resource will be in the "application/alto-networkmap+json" type?
- There is no specification making the client and the server agree on this.

Update: Dependent Resources Interpretation

- **Key observation**: The types of dependent resources should be already implied by the semantics of the entities and its properties.
- **Example**:
  - Agreement between the server and the client: If a property map provides the "pid" property of the "ipv4" or "ipv6" entities, a dependent resource in "application/alto-networkmap+json" type MUST be provided for it. Otherwise, this property map is invalid.

```
"uses": ["default-network-map"],
"capabilities": {
  "entity-domains": ["ipv4", "ipv6"],
  "properties": ["pid"]
}
```

- How to specify such agreements?

Domain-independent design is not helpful. Change to domain-specific design.

Update: ALTO Entity Property Type Registry

We initialize the process of the ALTO Entity Property Type Registry to provide the following guarantees:

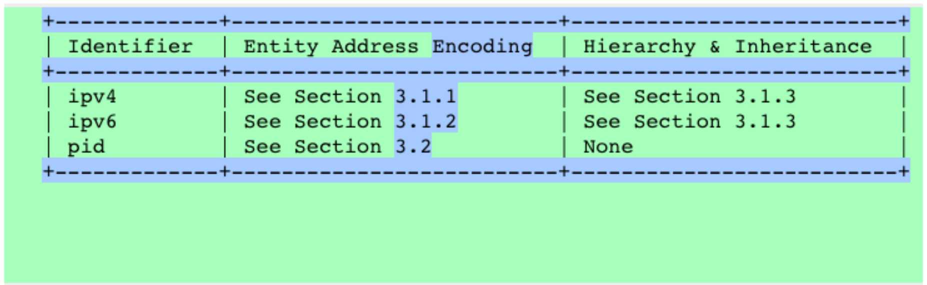
- **Consistency**: If an entity domain maps to an endpoint address type, all of their properties should have consistent semantics.
- **Domain-specific**: An entity property MUST be registered for some entity domains explicitly.
- **Dependencies**: An entity property MAY depend on a sequence of resources. The registry MUST specify how the client uses them in order.

Identifier	Property Name	Applied Entity Domain	Intended Semantics	Dependencies and Interpretation	Mapping to Endpoint Property Type
ipv4:pid	pid	ipv4	See Sec 3.1	application/alto-networkmap+json, See Sec 3.1	Yes

Update: ALTO Entity Domain Registry

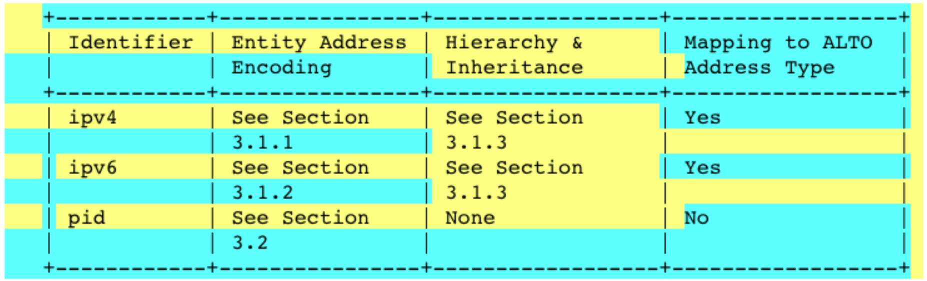
Kai's comment: only using the same identifier in Entity Domain Registry and Endpoint Address Type Registry cannot enforce the claimed consistency.

Introduce a new column to enforce the consistent semantics explicitly.



Identifier	Entity Address Encoding	Hierarchy & Inheritance
ipv4	See Section 3.1.1	See Section 3.1.3
ipv6	See Section 3.1.2	See Section 3.1.3
pid	See Section 3.2	None

Table 2: ALTO Entity Domains.



Identifier	Entity Address Encoding	Hierarchy & Inheritance	Mapping to ALTO Address Type
ipv4	See Section 3.1.1	See Section 3.1.3	Yes
ipv6	See Section 3.1.2	See Section 3.1.3	Yes
pid	See Section 3.2	None	No

Table 2: ALTO Entity Domains.

Next Steps

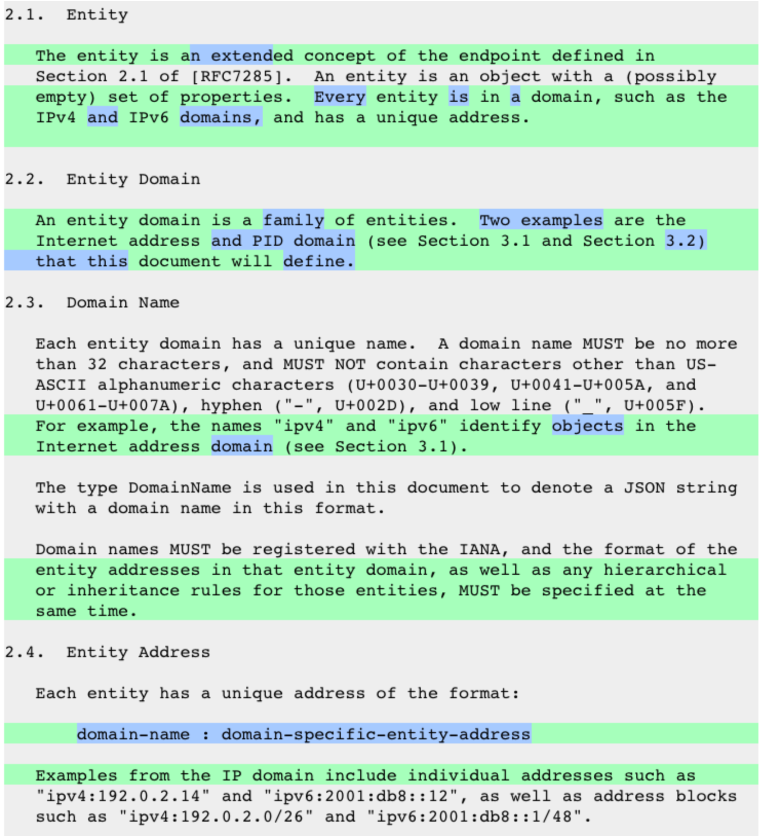
- Waiting for feedbacks and reviews from WG
- Finish all the proposed updates
- Request WGLC

Backup Slides

Skip this slide to save time?

Update: Terms

Revision of the terms based on Richard's and Kai's comments



2.1. Entity

The entity is an extended concept of the endpoint defined in Section 2.1 of [RFC7285]. An entity is an object with a (possibly empty) set of properties. Every entity is in a domain, such as the IPv4 and IPv6 domains, and has a unique address.

2.2. Entity Domain

An entity domain is a family of entities. Two examples are the Internet address and PID domain (see Section 3.1 and Section 3.2) that this document will define.

2.3. Domain Name

Each entity domain has a unique name. A domain name MUST be no more than 32 characters, and MUST NOT contain characters other than US-ASCII alphanumeric characters (U+0030-U+0039, U+0041-U+005A, and U+0061-U+007A), hyphen ("-", U+002D), and low line ("_", U+005F). For example, the names "ipv4" and "ipv6" identify objects in the Internet address domain (see Section 3.1).

The type DomainName is used in this document to denote a JSON string with a domain name in this format.

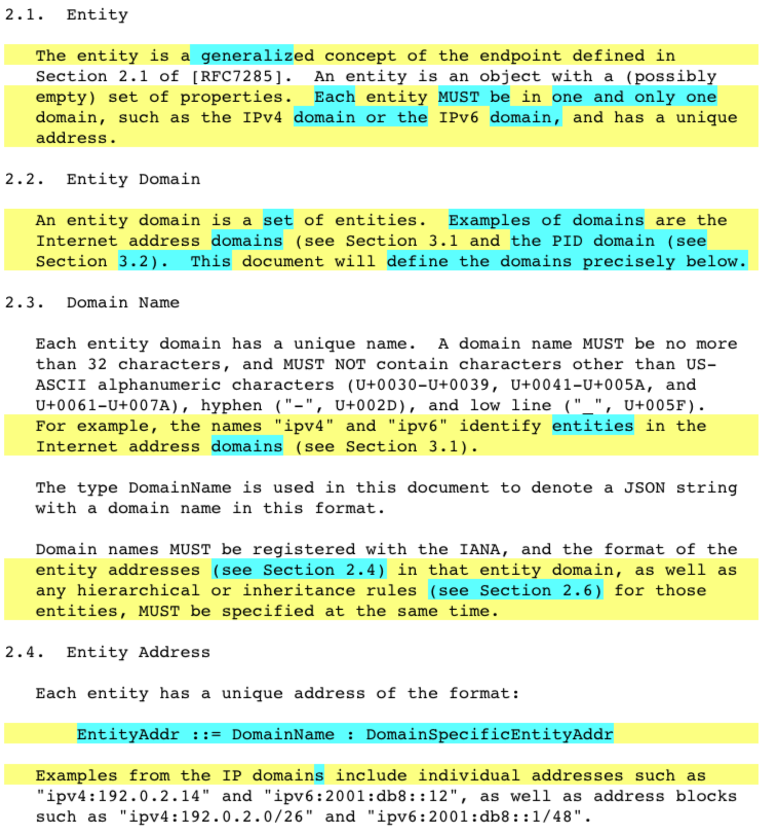
Domain names MUST be registered with the IANA, and the format of the entity addresses in that entity domain, as well as any hierarchical or inheritance rules for those entities, MUST be specified at the same time.

2.4. Entity Address

Each entity has a unique address of the format:

```
    domain-name : domain-specific-entity-address
```

Examples from the IP domain include individual addresses such as "ipv4:192.0.2.14" and "ipv6:2001:db8::12", as well as address blocks such as "ipv4:192.0.2.0/26" and "ipv6:2001:db8::1/48".



2.1. Entity

The entity is a generalized concept of the endpoint defined in Section 2.1 of [RFC7285]. An entity is an object with a (possibly empty) set of properties. Each entity MUST be in one and only one domain, such as the IPv4 domain or the IPv6 domain, and has a unique address.

2.2. Entity Domain

An entity domain is a set of entities. Examples of domains are the Internet address domains (see Section 3.1 and the PID domain (see Section 3.2). This document will define the domains precisely below.

2.3. Domain Name

Each entity domain has a unique name. A domain name MUST be no more than 32 characters, and MUST NOT contain characters other than US-ASCII alphanumeric characters (U+0030-U+0039, U+0041-U+005A, and U+0061-U+007A), hyphen ("-", U+002D), and low line ("_", U+005F). For example, the names "ipv4" and "ipv6" identify entities in the Internet address domains (see Section 3.1).

The type DomainName is used in this document to denote a JSON string with a domain name in this format.

Domain names MUST be registered with the IANA, and the format of the entity addresses (see Section 2.4) in that entity domain, as well as any hierarchical or inheritance rules (see Section 2.6) for those entities, MUST be specified at the same time.

2.4. Entity Address

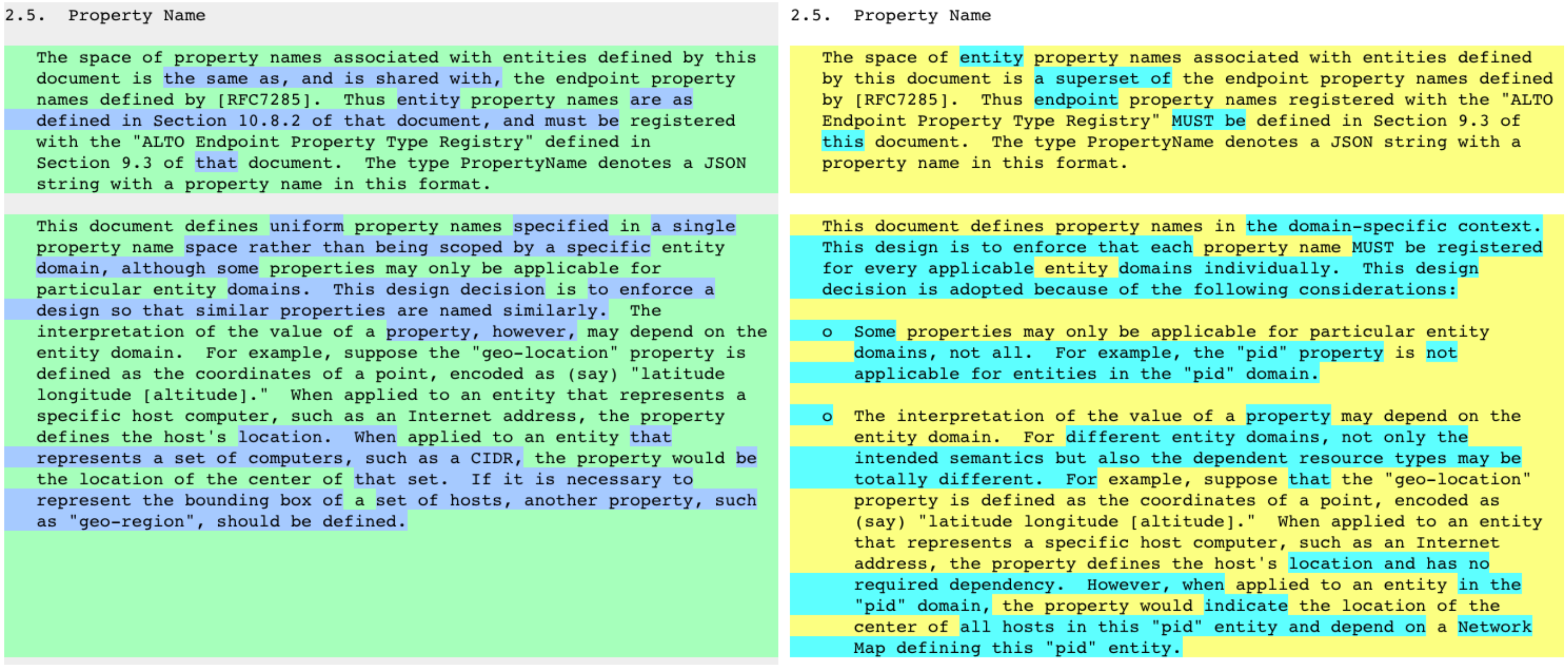
Each entity has a unique address of the format:

```
    EntityAddr ::= DomainName : DomainSpecificEntityAddr
```

Examples from the IP domains include individual addresses such as "ipv4:192.0.2.14" and "ipv6:2001:db8::12", as well as address blocks such as "ipv4:192.0.2.0/26" and "ipv6:2001:db8::1/48".

Update: Dependent Resources Interpretation

Domain-independent design is not helpful. Change to domain-specific design.



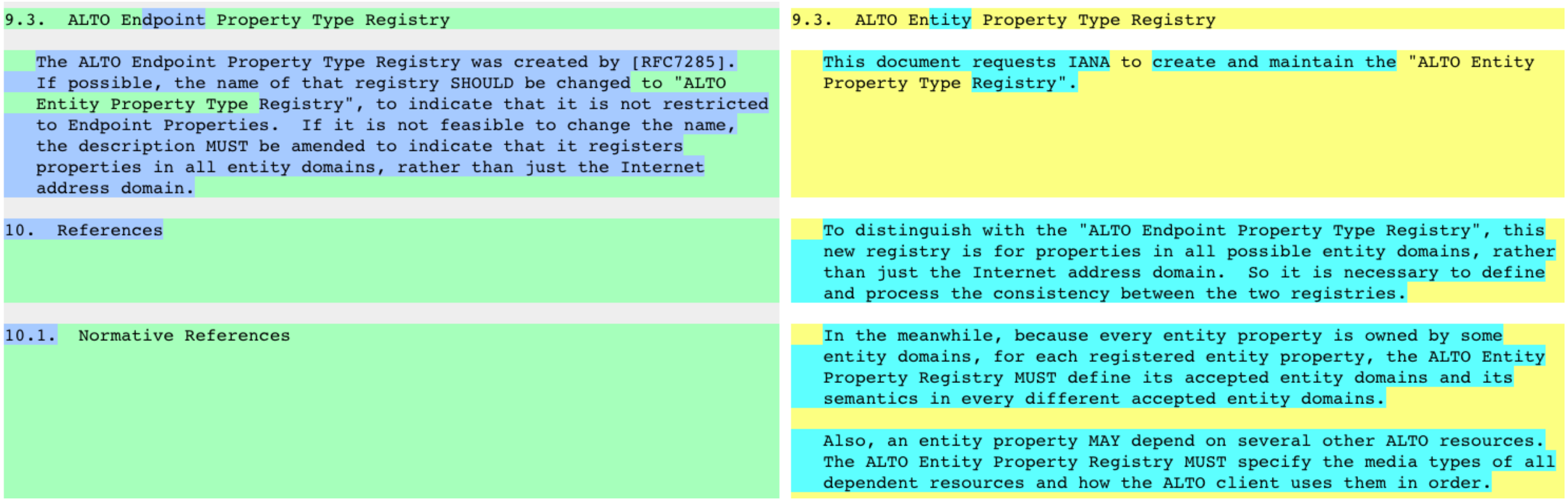
```
2.5.  Property Name

   The space of property names associated with entities defined by this
   document is the same as, and is shared with, the endpoint property
   names defined by [RFC7285].  Thus entity property names are as
   defined in Section 10.8.2 of that document, and must be registered
   with the "ALTO Endpoint Property Type Registry" defined in
   Section 9.3 of that document.  The type PropertyName denotes a JSON
   string with a property name in this format.

   This document defines uniform property names specified in a single
   property name space rather than being scoped by a specific entity
   domain, although some properties may only be applicable for
   particular entity domains.  This design decision is to enforce a
   design so that similar properties are named similarly.  The
   interpretation of the value of a property, however, may depend on the
   entity domain.  For example, suppose 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, such as an Internet address, the property
   defines the host's location.  When applied to an entity that
   represents a set of computers, such as a CIDR, the property would be
   the location of the center of that set.  If it is necessary to
   represent the bounding box of a set of hosts, another property, such
   as "geo-region", should be defined.
```

```
2.5.  Property Name

   The space of entity property names associated with entities defined
   by this document is a superset of the endpoint property names defined
   by [RFC7285].  Thus endpoint property names registered with the "ALTO
   Endpoint Property Type Registry" MUST be defined in Section 9.3 of
   this document.  The type PropertyName denotes a JSON string with a
   property name in this format.

   This document defines property names in the domain-specific context.
   This design is to enforce that each property name MUST be registered
   for every applicable entity domains individually.  This design
   decision is adopted because of the following considerations:

   o  Some properties may only be applicable for particular entity
      domains, not all.  For example, the "pid" property is not
      applicable for entities in the "pid" domain.

   o  The interpretation of the value of a property may depend on the
      entity domain.  For different entity domains, not only the
      intended semantics but also the dependent resource types may be
      totally different.  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, such as an Internet
      address, the property defines the host's location and has no
      required dependency.  However, when applied to an entity in the
      "pid" domain, the property would indicate the location of the
      center of all hosts in this "pid" entity and depend on a Network
      Map defining this "pid" entity.
```

Update: ALTO Entity Property Type R

Hard so see. Skip this slide and expand the former with examples instead

The initial write-up of the ALTO Entity Property Type Registry.



```
9.3.  ALTO Endpoint Property Type Registry

   The ALTO Endpoint Property Type Registry was created by [RFC7285].
   If possible, the name of that registry SHOULD be changed to "ALTO
   Entity Property Type Registry", to indicate that it is not restricted
   to Endpoint Properties.  If it is not feasible to change the name,
   the description MUST be amended to indicate that it registers
   properties in all entity domains, rather than just the Internet
   address domain.

10.  References

10.1.  Normative References
```

```
9.3.  ALTO Entity Property Type Registry

   This document requests IANA to create and maintain the "ALTO Entity
   Property Type Registry".

   To distinguish with the "ALTO Endpoint Property Type Registry", this
   new registry is for properties in all possible entity domains, rather
   than just the Internet address domain.  So it is necessary to define
   and process the consistency between the two registries.

   In the meanwhile, because every entity property is owned by some
   entity domains, for each registered entity property, the ALTO Entity
   Property Registry MUST define its accepted entity domains and its
   semantics in every different accepted entity domains.

   Also, an entity property MAY depend on several other ALTO resources.
   The ALTO Entity Property Registry MUST specify the media types of all
   dependent resources and how the ALTO client uses them in order.
```