Subject Representations

This is a suggestion for a subject representation, and example API, that might be provided by a subject cache. It is not for a system of record for subjects.

Subject types and attributes

Subject types can vary widely. Our groups service, as an example, supports these subjects.

type	source
UW people	person registry
hosts (certificate CN)	DNS registry
groups	Groups service
AD computer names	Active Directory
external emails	groups service subject db

I'm sure there are others.

Subject cache attributes

A service that wants find or reference subjects, e.g., a people-finder or a group service, will generally not directly rely on the systems of record (sources) for the various subjects it can work with. Instead it will draw on one or more local subject caches. We need a uniform way to identify subjects both in the cache and at the system of record.

Subjects will have different attributes according to their type, cache needs, and user permissions.

For all subjects:

attribute	value
id	local cache uuid
source	uri of subject sor
source id	uuid of subject at the source
name	display name
acls	TBD

Other attributes, first and last name, email, department, .. will apply only to some subject types.

Subject representation

(less the standard cifer administrative attributes)

```
{
  "id": "some uuid",
  "type": "subject type",
  "url": "URL to the subject at the cache",
  "sourceUrl": "URL to the subject at the source",
  "displayName": "display name",
  "other attribute1": "attribute value",
  "...": "values"
}
```

Example: API for a subject finder

(A generalized people finder)

GET (root)/

```
{
   "subjectCacheId": "some-uuid",
   "subjectCacheUrl": "cache's home URL",
   "sources": [
      {"sourceId": "some-uuid",
       "sourceName": "text name",
       "sourceUrl": "source's home url",
       "subjectType": "inst. person|group|machine|email|...",
       "attributes": ["attr1", "attr2", ... ]
       ]
      },
      { more-sources }
   ]
 }
```

GET (root)/<source>/<id>

- returns the complete (for the cache) representation of the subject

PUT (root)/<source>/<id>

- replaces a subject

POST (root)/<source>/

- adds a subject

GET (root)/<source>/? <some query string>

- returns a list of matching subjects
- query string TBD

API for a subject finder using websockets

A user-friendly subject finder often wants to display a list of names that is progressively better filtered as the user types. That sort of repeated query and response might be better served by a websocket than by repeated ajax requests. In this situation the REST model still applies. The base URL establishes the socket. As the user types query string parts are sent to the server. What comes back is always standard subject representations.