Emetsger__Masking

About

In DSpace **1.4.2**, browsing indexes and search indexes are built without regard for the logged in user. Generally this is useful: often browsing or searching takes place anonymously. If the user attempts to view an item that they don't have permissions to view, they'll be prompted to authenticate and view the item if the authorization system allows it.

We (Johns Hopkins) have a large collection (30 thousand plus items) which is not allowed to be accessed except for the DSpace Administrator. The collection is residing in DSpace for preservation purposes and not presentation (there is a separate presentation website for the collection). Currently when users search or browse our DSpace instance, items from this large collection dominate the results. The user cannot get to the items, even if they authenticate. So displaying the browse or search hits from this collection is reducing the usablility of our DSpace instance.

After thinking about this further, I think there are two different use cases:

Not indexing a Community/Collection/Item (masking - is there a better name? - described above)

- If a collection is masked, it isn't indexed for browse or search (e.g. not discoverable) ...
- ... therefore masked objects cannot be browsed or searched, even by administrators.
- The items in the masked collection can only be retrieved if the user knows the handle *a priori*.
- Masking is not a substitute or complement of the authorization manager: that is users can still get to an item if the handle is known.
- Masking could be configured by listing the masked handles in

  ```
  dspace.cfg
  ```

  :

```
#

    1. Masking
       #
       plugin.single.org.dspace.mask.MaskService = org.dspace.mask.PropertyMaskService
       plugin.reusable.org.dspace.mask.PropertyMaskService = true
       mask.browse = 1774.2/2085, 1774.2/1977, 1774.2/950, 1774.2/32403, 1774.2/55, 1774.2/932
       mask.index = 1774.2/2085, 1774.2/1977, 1774.2/950, 1774.2/32403, 1774.2/55, 1774.2/932
       #mask.harvest =
```

- Masking could also be configured by a UI widget, with a list of masked objects maintained in the database.
- There are different masks: one could mask a collection from search but not from browse, for example. Or one could mask the collection from being harvested while still allowing it to be indexed.

Filtering search and browse results from the user

- Filtering removes search hits and browse results that the user does not have access to
- Is done on-the-fly, probably will need some type of cache
- Uses the

  ```
  AuthorizeManager
  ```

  to filter the hits
- ...

Implementation

Masking

Masking impl is relatively straightforward. The search and browse implementations would be updated to obey the

```
mask.*
```

properties in

```
dspace.cfg
```

. If an object is masked, the search or browse index builders would skip it. Objects are masked at the time

```
bin/index-all
```

is called.

Alternate implementations could be designed to persist mutable configuration parameters in the database, allowing administrative toggles in the user interface.

An interface

```
MaskService
```

is responsible for determining whether or not an object is masked. An implementation can be retrieved via the DSpace Plugin Manager or can be injected by

```
[http://excalibur.apache.org/apidocs/org/apache/avalon/framework/service/Serviceable.html Serviceable]
```

sitemap components in Cocoon.

I have an implementation of this interface in local development. In my local implementation, an Item is masked when any of the following are true:

1. The Item itself is masked
2. The Item's owning collection is masked (this means that if an Item is mapped to multiple collections, it will be masked from mapped collections too).
3. All of the Items parent Communities are masked

Code

Hack for DSpace 1.4.2

Apply http://oldwiki.dspace.org/static_files/2/21/Mask-package.patch You'll need a fuzz of 2 for

```
DSIndexer
```

.

Add the following to dspace.cfg. Add handles of items, collections, or communities to mask. Comma delimited.

```
plugin.single.org.dspace.mask.MaskService = org.dspace.mask.PropertyMaskService
plugin.reusable.org.dspace.mask.PropertyMaskService = true
mask.browse = 1774.2/7, 1774.2/4
mask.index = 1774.2/7, 1774.2/4
```

You'll need to download and add Commons Logging 1.0.4 to your DSpace lib directory.

Stop DSpace, run

```
bin/index-all
```

.

Hack for Manakin 1.1a

Follow the masking instructions for DSpace.

Apply the Manakin patch: http://oldwiki.dspace.org/static_files/6/6b/Mask-manakin.patch.

Start DSpace. Your items, collections, and communities should be masked.

TODO