Security & Access Control in Islandora

Security in Islandora leverages both Drupal's Access Control infrastructure (Drupal Roles and Permissions) with Fedora's XACML security framework to create a highly flexible framework that can write restrictions to the datastream and IP level. Additional information about Fedora security is available at the FedoraCommons wiki (see our Selected Reading Section). Note that Fedora's restrictions are always enforced over Drupal's, but Fedora XACML policies cannot grant a user access that they do not have via Drupal permissions.

Drupal Security and Islandora

Namespace Restrictions

In the Islandora configuration pane. (admin/islandora/configure) under "namespaces," an administrator can set a Drupal site to only allow access to specific namespaces. This is particularly useful in multisite configurations.

Permissions and Roles

Drupal gives you the ability to divide your site users into different groups, by creating "Roles" for users. A "Role" defines who your user is, and what they should be able to access, update, delete, or create in a Drupal site. Roles are managed at admin/people/permissions/roles. Drupal 7 comes out-of-the-box with an administrative role, an anonymous user role (somebody without an account) and an authenticated user role (somebody with an account, that logs in to the site). Additional roles can be created and then assigned permissions at admin/people/permissions.

Any module installed will generally make additional types of permissions available. Islandora modules follow this model. For each Islandora module you install, there will usually be a permission to configure.

For example:

- If you allow a role to **add fedora datastreams,** users with that role will be able to add a datastream to an object in your repository (presuming the content model affiliated with that object has defined the datastream being added as part of the content model).
- If you allow a role to **create batch process**, users with that role will be able to upload tar files for ingest.
- If you allow a role to **delete entire collections**, users with that role will be able to purge entire collections (the collection object and all members) without iterating over all the member objects manually.
- If you allow a role to **edit fedora metadata**, users with that role will be able to edit the metadata record for any object.
- If you allow a role to **edit tags datastream**, this functionality appears incomplete.
- If you allow a role to **ingest new Fedora objects**, users with that role will be able to add items into the repository.
- If you allow a role to **manage collections**, users with that role will have access to some collection level utilities, such as changing the allowed content models.
- If you allow a role to **purge objects and datastreams**, users with that role will be able to purge objects, and replace and purge datastreams in an objects.
- If you allow a role to **view detailed list of content**, users with that role will be able to view the datastream details of a given object (available under the "detailed list of content" fieldset in any object view)
- If you allow a role to **view fedora collection**, users with that role will be able to view your collections

FedoraCommons Security and Islandora

Islandora uses the XACML framework in FedoraCommons. XACML (written in eXtensible Access Control Markup Language) as both an access control policy language implemented in XML and a processing model that describes how to interpret the policies. In order to use XACML, you need to have enforced *policies* in your Fedora configuration file (fedora.fcfg). XACML policies are first applied when your repository is set up, and these permissions would be managed by the person installing and maintaining Islandora.

These initial policies are always enforced. No object-specific XACML policy can ever override a repository-wide XACML policy. This means you cannot use a repository-wide policy to forbid users to see any objects, and then use XACML to grant viewing rights on particular objects. However, object-specific XACML **can** deny rights that are allowed at the Fedora-wide level. You can author object-level XACML policies (to the DSID and role level) by using the XACML Editor.

Islandora will parse XACML it finds in two places - either the datastream of the object (in the CHILD_SECURITY or POLICY datastreams) or global XACML policies found at

$FEDORA_HOME/data/fedora-xacml-policies/repository-policies/default

Collection-specific XACML policies

The CHILD_SECURITY Datastream of a collection object is an XACML policy file. Once this Datastream has been added, all objects that are ingested from that point onward will have a POLICY Datastream that enforces the CHILD_SECURITY policy on the members of a collection. XACML overrides Drupal security. So, for example, if you have a Drupal role that says you are allowed to add Datastreams, you will be allowed to add Datastreams to all objects except objects that have a XACML policy that denies it. To learn more about XACML policies at the Collection Object level, please go to the CHILD_SECURITY section of Chapter 7 - Customizing Islandora.

Global XACML Policies

XACML also provides default policies that restrict access to management functions in the Fedora repository (API-M) to the Fedora administrator, or permit any member of the public to access and view the Fedora Repository (API-A). XACML can establish other basic controls, such as allowing only localhost to access management functions in the repository.

For more information about writing custom global xacml policies, visit the Fedora XACML Policy Writing Guide: https://wiki.duraspace.org/display/FEDORA38/Fedora+XACML+Policy+Writing+Guide

For more information about how Fedora manages security, visit the Fedora Security documentation: https://wiki.duraspace.org/display/FEDORA38/Security

How does Islandora interpret XACML policies?

Islandora interprets XACML policies using the Drupal filter. The Drupal filter allows Fedora to authenticate against the Drupal database and it also gathers the roles belonging to users. These usernames and roles then become available for you to use in your xacml policies. A sample XACML policy is provided in the modules policy folder in the Drupal module, and it illustrates how XACML utlizes Drupal usernames and roles in order to provide granular security in an Islandora site. To discover more about how XACML is interpreted in Islandora, view the SecurityClass.inc file in the *Islandora Module.*