# Draft HydroShare Access Control Policies

David Tarboton, Alva Couch

This document gives desirable specifics of the HydroShare access control policy intended to specify the rules for sharing and controlling who can access resources in HydroShare and what is enabled by different levels of access settings. This document is based upon an analysis of user desires, and not upon the capabilities available within the platform thatHydroShare is being implemented in. Thus, it should be considered a “wish list” rather than a prescriptive description of how HydroShare access control should work. In the Agile development model in which HydroShare is being developed, we plan to partly implement these policies as allowed by the platform, and to carefully consider which parts of this design are practical to implement. Thus, it is likely that this specification will change as agile development continues.

This is a subject that we know has had considerable study. This policy has been written drawing on experience with other systems (e.g. Windows, Unix, DropBox, Google, Facebook) and strives for a system that is simple and straightforward with the mindset of eliminating unnecessary mechanisms in favor of a set of behaviors specific to the tasks involved in data-centered research. We generally like the Google Model (Google Share Settings Overview, accessed 2013) and have followed it closely adapting it as necessary (e.g. we do not have the concept of visibility to anyone who has the link).

Definitions:

* **Resource** – A resource includes any object that is sharable in HydroShare. Each resource has independent access settings.
* **User** - An individual with an account that is authenticated to access HydroShare
* **Group** - A collection of users: a special kind of resource

A user is anyone who has a username and password for the HydroShare system. The System (system administrator) is a special user with the ability to change anything.

**Rule:** Any user can create a resource subject to space/quota limitations.

Each resource has a set of attributes. The following attributes are a subset of attributes common to all resources that pertain to access control policies

* **Creator** - the specific user who created the first version of the resource. The creator is independent of the owner of the resource and does not play a role in access control, though the creator will have been the first owner and often remains an owner.
* AKT: This is found via the uid on the node table
* **Last changed by** - the specific user who last changed the resource.
* AKT: This is found via the uid on the node\_revision table
* **Version** – a number indicating the number of revisions so far.
* AKT: This is the vid on the node\_revision table
* **Do not redistribute** – a Boolean flag that indicates whether entities who receive access to the resource can grant that access to others.
* AKT: would need to be an additional field added via CCK
* **Discoverable** – a Boolean flag that indicates whether metadata is discoverable independent of the ability to access data.
* AKT: would need to be an additional field added via CCK
* **Public** – a Boolean flag that indicates whether the resource is accessible to the public. Setting Public automatically sets Discoverable, but not the other way round.
* AKT: would need to be an additional field added via CCK
* **Share settings** - a logical list of sharing privileges, including assignment of various levels of privilege over the resource to specific users and/or groups.
* AKT: Could be stored as a new CCK field, and/or modeled via a combination of Rules ( Rules module ) + custom logic

Ownership of a resource is a share setting, as documented below.

\*\*AKT Note: We will probably need owner added as a CCK field and custom logic on to maintain the owner as within a workflow as described below.

**Rule:** “Owner”, “Creator”, and “Last changed by” must be users, and cannot be groups.

AKT: Enforced by owner maintenance UI + workflow

**Rule:** A resource may have more than one (co-)owner.

AKT: Allow the owner field to be a list of uid’s

**Rule:** An owner may revoke his or her ownership of a resource, in which case other owners retain ownership.

AKT: Enforced by owner maintenance UI + workflow

**Rule:** The last owner of a resource is not allowed to revoke ownership.

AKT: Enforced by owner maintenance UI – disable revoke ability if owner list has one uid

**Rule:** The initial owner of a new resource is the creator.

AKT: Add content creation ( node add ) callback/hook to assign owner field to logged in user

**Rule:** The site administrator (“system”) may create resources.

AKT: Admin will have privs to create resources

**Rule:** Making a change to a resource does not affect ownership or the original creator.

AKT: Changes will be tracked via node revision and will not change or impact the initial version

**Rule:** Ownership may only be modified by an owner or an administrator.

AKT: Achieved via the owner maintenance UI and workflow

**Rule:** An owner may add other owners, and may revoke his or her own privileges of ownership for a resource. An owner may also revoke the ownership of other owners. An owner is fully entrusted to do anything.

AKT: Achieved via the owner maintenance UI and workflow

**Rule:** When an owner cedes ownership of a resource, that user immediately loses control of the resource, excluding other privileges that the user might have through other access entries. The “cede ownership” function should allow the user to set other access entries they wish to retain before ownership is ceded.

AKT: As soon as the owner field is updated and published, the UI’s and workflow will recognize the change in ownership. The new owner field will only be published based on the workflow described below

**Comment:** To permanently change the owner of a resource, one adds the other owner and then revokes one’s own ownership. The other owner must explicitly accept ownership.

AKT: Our workflow will “publish” the change of ownership when the other user accepts ownership

**Access control**

Share settings are associated with a resource and comprise a list of who the resource is visible to, what they can do with the resource and whether they may assign access privileges to others. Conceptually a table with two columns represents the share settings. For example:

|  |  |
| --- | --- |
| Entity (user or group name) | Access setting |
| User 1 | Owner |
| User 2 | Change |
| User 3 | View |
| Group 1 | Change |

The following table gives functionality associated with each access level

|  |  |  |  |
| --- | --- | --- | --- |
| Privilege | Owner | Change | View |
| Revoke access privileges, including one’s ownership of a resource; set and reset “Do not redistribute” and “Discoverable” flags. | Yes |  |  |
| Change Content and Metadata (excluding “Do not redistribute” and “Discoverable” flags) | Yes | Yes |  |
| Comment on or vote for a resource | Yes | Yes | Yes |

* **Owner** – An owner has full privileges over a resource including the ability to assign and revoke all access privileges and ownership. Can delete the resource.
* **Change** - The ability to change the resource and/or its metadata.
* **View** - The ability to access, view, comment upon, and vote for a resource

AKT: The above will be implemented via a custom module and set of DB tables. Our workflow and maintenance UI’s will reference these tables for permission and access checks.

**Redistribution, delegation, and discoverability**

**Rule:** The “Do not redistribute” flag determines whether privileges assigned to users can be granted to other users.

AKT: Do not distribute will be set using the maintenance UI for those users which have permission to do so

**Rule:** When “Do not distribute” is not checked,

1. A user with a level of privilege may grant others privileges up to that level:

* Owners may grant any privilege, including co-ownership.
* Changers may grant Change or View privilege.
* Viewers may only grant View privilege.

1. Any user may copy the resource in a way that preserves metadata.

AKT: Maintenance UI will reference custom access tables and node permissions and will display operations appropriate to the user

**Rule:** When “Do not redistribute” is checked,

1. No user other than the owner may grant privilege to the resource to another.
2. No user other than the owner may make an internal copy of the resource.

AKT: Maintenance UI will reference custom access tables and node permissions and will display operations appropriate to the user

**Rule:** Regardless of the setting of “Do not redistribute”,

1. A user with a level of privilege may remove privileges that he or she granted. Owners may in addition remove any privilege whatsoever.

AKT: Maintenance UI will reference custom access tables and node permissions and will display operations appropriate to the user

**Rule:** The setting for “Discoverable” determines whether the resource is listed in searches that match its metadata. This is independent of whether the resource is shared with Public. This discoverability applies to all searches by anyone including public non registered users. The metadata available through the search results will include the names and contact information of all owners to support the use case of them being contacted to request information about or access to the resource in the case that the resource is discoverable, but not publicly accessible.

AKT: Will need to create a custom search view which references the “Discoverable” field and returns the data described above

**Rule:** Privileges assigned to others are not revoked if the original grantor’s privileges are revoked.

AKT: Enforced by logic in the maint UI

**Rule:** A user with a privilege over a resource can downgrade that privilege to any lower level of privilege. The only exception to this rule is that the last owner cannot revoke ownership without transferring it.

AKT: Maint UI will query custom access tables and fields to determine appropriabe downgrade operations

**Rule:** When duplicating a resource inside HydroShare, privileges are not inherited for the duplicate.

AKT: Will we then create a default set of permissions, or take the user through a wizard to assign permissions?

**Rule:** The ability to comment upon or vote for a resource requires login, even for publicly viewable resources. There is no anonymous commenting or voting.

AKT: We use simple user permissions to achieve this

**Creating and managing groups**

Groups are similar to other resources. To Change a group is to add and/or delete members or to change the metadata about the group. To View a group is to be able to list its members and metadata. Only group members will be able to see and access resources that have been shared with the group.

AKT: already achieved via the Organic Groups Module – for additional granularity or visibility, we can create custom group views so users can view certain sets of metadata before deciding whether to join a group

**Rule:** The owner can delete the group and add and remove group members including other owners.

AKT: already achieved via the Organic Groups Module

**Rule:**  A member of a group may not be another group, to avoid the complexity of potential nesting and recursion of groups.

already enforced via the Organic Groups Module

**Rule:** If a group is not marked as “Do not redistribute”,

* View privileges allow one—in addition to View privilege – to authorize others to View the group.
* Change privilege allows one – in addition to Change privilege – to authorize others to Change the group (i.e. to add or remove group members).

Note that setting the “Do not redistribute” property on a group makes it so that only owners can add members since adding a member amounts to sharing it with that user. Changers will still be able to remove members.

AKT: to affect the above, it sounds like we’ll need to add the new CCK fields to the group node type as well and create a custom workflow for adding and removing members from groups

**Group privileges**

**Rule:** One may share a resource with any group, regardless of whether one is a member or not.

AKT: Assuming they have permission to share the resource

**Rule:** Assent is not required from the group in order to grant group privileges.

AKT: No workflow here. Just a maint UI to grant group privs

**Rule:** Sharing a resource with a whole group at a privilege above View requires that one own both the resource and the group.

AKT: We need to make a call here and implement the appropriate access check

**Comment:** This prevents the need for the whole group to assent to privileges.

**Assent to accept privilege**

Formal assent of the grantee is required in order to assign certain privileges, because these privileges imply some kind of responsibility for the resource or identification of the user as being associated with the resource.

**Rule:** Change or Owner privileges require the explicit assent of the grantee before being assigned.

AKT: Add custom workflow. Do not publish the change until the grantee assents

**Rule:** Granting a privilege to a group does not require assent from the group.

AKT: No workflow, automatic publish of change to privilege

**Rule:** One may remove or downgrade one’s own privileges whenever this is desirable.

AKT: No workflow, automatic publish of change to privilege

**Administrative privilege**

Note that the administrator is not listed in the share settings table, as implicitly the HydroShare system (via a system administrator account) has full control for each resource.

**Public resources**

A resource is made public by the Public property being set. Once a resource is made public, its behavior changes in several ways.

**Rule:** Only an Owner can make a resource Public.

**Rule:** Only an Owner can revoke the Public status of a resource.

**Rule:** Making a resource public (i.e. setting the Public property) automatically makes the resource metadata discoverable (sets the Discoverable property).

**Rule:** Making a resource Public does not add privilege to the access table. Thus, making an object private again revokes all privileges gained by making it Public.

AKT: All above rules achieved via workflow and updates to the access tables

**Behavior of persistent resources**

AKT: Agreed. All resources revisions exist in the DB and are therefore persistant. Published is the current viewable revision of a resource node. Probably the word we are looking here is Archived. And then we’ll need a separate workflow maintance controls for dealing with items that have been marked as “Archive”, including setting a DOI.

A persistent resource is one whose owner had decided to publicly publish it and that has been issued a Digital Object Identifier (DOI) for archival reasons. Once a resource is made persistent, there are several changes in access privileges.

**Rule:** A persistent resource is owned by “System”.

**Rule:** A persistent resource is immutable; no entity except system has rights above View.

**Rule:** A persistent resource has no access rights except to System and Public. These access rights are “Owner” and “View”, respectively, as documented above.

**Rule:** Sharing a persistent object does not add access rights to its access table, which is empty. Sharing simply sends a message to the grantee.

**Examples**

List of authenticated users used in examples that follow

* dtarb
* jeffh
* rayi
* dan
* larry

**Example 1. Shared resource**

Attributes:

|  |  |
| --- | --- |
| Name | 3amg |
| Description | Three Amigos |
| Do not redistribute | No |
| Discoverable | Yes |
| Public | No |

Share settings:

|  |  |
| --- | --- |
| Entity (user or group name) | Access setting |
| Dtarb | Owner |
| Rayi | Owner |
| Jeffh | View |
| Dan | View |
| Larry | Change |

For this resource, rayi and dtarb have full control. Larry may change the resource and grant others the ability to change it. Jeffh and Dan may view/access the resource and invite others to view/access the resource. All listed users may make copies of the resource. Information about this resource will be discoverable in searches that match its metadata and search results may display the contact information for Dtarb and Rayi.

**Example 2. Shared resource with limited distribution**

Attributes:

|  |  |
| --- | --- |
| Do not redistribute | Yes |
| Discoverable | No |

Share settings:

|  |  |
| --- | --- |
| Entity (user or group name) | Access setting |
| Dtarb | Owner |
| Rayi | Owner |
| Jeffh | View |
| Dan | View |
| Larry | Change |

Privileges of users are as before, except that no non-owner may grant privileges to others. Also, no non-owner may make an internal copy of the resource.

**Rule:**  Each time a resource is changed its version is incremented. Because versioning is being done by the HydroShare system, the version numbers can be simple integers.

**Example 3. Publicly shared resource**

Attributes:

|  |  |
| --- | --- |
| Do not redistribute | No |
| Discoverable | Yes |
| Public | Yes |

Share settings

|  |  |
| --- | --- |
| Entity (user or group name) | Access setting |
| Dtarb | Owner |
| Dan | Owner |
| Larry | Change |

For this resource, owner dtarb has full control as does Dan. Larry may change the resource. The resource is shared with the public, which means that anyone can access and view the resource. Because Do Not Redistribute is set to “No” anyone can also make a copy of the resource.

**Example 4. System owned resource**

Attributes:

|  |  |
| --- | --- |
| Do not redistribute | No |
| Discoverable | Yes |
| Public | Yes |

Share settings

|  |  |
| --- | --- |
| Entity (user or group name) | Access setting |
| System | Owner |

This is a persistent resource published in the system and available to everyone. In general system resources should be accessible to the public, as their purpose is to provide a general resource, but access may be restricted while the resource is still under development or testing.

**Example 5. Group**

Attributes

|  |  |
| --- | --- |
| Name | 3amg |
| Description | Three Amigos |
| Do not redistribute | No |
| Discoverable | Yes |

Share settings

|  |  |
| --- | --- |
| Entity | Access setting |
| Dtarb | Owner |
| Jeffh | Change |
| Rayi | View |
| Dan | Change |

Membership in a group is equivalent to being provided access to the resource that is a group. Change privilege for a group allows adding and removal of group members and editing the group’s descriptive metadata. Thus, those with Owner and Change privileges can add group members. Since “Do not redistribute” is not checked, all users may grant rights to other users at their level of delegation.

In this case dtarb, jeffh, and dan can add and remove members. Only dtarb can delete the whole group. Rayi has View access and cannot add or remove members.

**Rule:** Resources can be shared with a group but are owned by individual users. Deleting a group has no effect on the existence of resources within the HydroShare system.

**Example 6. Resource shared with group**

Attributes:

|  |  |
| --- | --- |
| Do not redistribute | Yes |
| Discoverable | Yes |
| Public | Yes |

Share settings:

|  |  |
| --- | --- |
| Entity (user or group name) | Access setting |
| Dtarb | Owner |
| 3amg | Change |
| Public | View |

For this resource dtarb, the owner has full control. Members of the group 3amg may change the resource. The resource is Public, which means that anyone can access and view the resource.

Note that the resource access setting applies to all group members. So in this example rayi who is a member of 3amg with view setting still has the ability to change a resource shared with the group. He is not able to change the group membership.

Note that when group membership changes the set of users who have access to resources shared to that group changes automatically (and as quickly as the change propagates through the system). Also, a side effect is that the owner of a resource shared with a group has relinquished who has control over the access to the resource to the users who control the group membership.

Implementation suggestion. When building functionality to share resources with a group, HydroShare should provide the option to “share with the group” which has the effect of adding the group as a sharing entity, and the option to “share with all current group members” which has the effect of adding individual group members. In the second case the sharing is separate from the group. The set of group members at the time the share is created may be notified that the resource is available to them (for them to attach to if they want depending on their notification settings). Users who later become group members or leave the group will not automatically have access to the resource changed.

**References**

Google Share Settings Overview <http://support.google.com/drive/bin/answer.py?hl=en&answer=2494886&p=visibility_options>