Cornell Permit Server Grouper Comparison

Comparison of Cornell Permit Server Features and Capabilities with Grouper

May 22, 2006

Author: Joy Veronneau

In the following table we have tried to capture all the features of permits and cussplib commands that are used in the current permit server at Cornell. Where permit server features are not available in Grouper, we've made note of that in this chart and also added that issue to the requirements list.

Cornell PERMIT SERVER FEATURE	Grouper EQUIVALENT
Privileges associated with a permit (can be changed via the permit administration Utility - permit.cgi)	**Privileges associated with Groups (can be changed via the Grouper UI)**
Lookup - A netid or srvtab with lookup privilege can see who has that permit.	Read privilege - subject with read privilege for a group may see the membership list for this group. We have not yet worked out how servtabs will be represented in Grouper, since grouper will get its subject lists from the directory.
Update - A netid or srvtab with update privilege can see who has that permit, add someone to the permit, or deletesomeone.	Update privilege - subject with update privilege may modify the membership of this group - add, delete, read
Admin - A netid or srvtab with admin privilege has Update privileges. In addition they can revoke or reinstate a permit for someone. They can grant or remove lookup, update, or admin privileges for someone. They can also update permit characteristics: friendly name, public, visible, and largelist.	Admin - subject may modify the membership of this group, delete the group or assign privileges for the group, change the description of the group, assign privileges to group members (read, update, etc). See comments later in this doc regarding revoke/reinstate.
Owner - everything Admin can do, plus: remove all netids; remove all netids plus permit descriptor; add a permit off of this level.	The equivalent in Grouper is a person who has admin privileges for the group, plus they also have create and stem privileges for the stem. A Stem is a namespace in which groups exist.
Master - everything Owner can do, plus can issue the deleteAll command.	Wheel group - members of the Grouper wheel group have overall privileges
Download a list of netids with a permit.	This does not exist in the current UI but may be added.
Upload a list of netids to a permit.	This does not exist in the current UI but may be added.
public - viewed or verified by anyone	Any group can be set up such that everyone has Read privilege - meaning they can see the membership of the group
visible - show in permit lists	Any group can be set up such that everyone has View privilege - meaning they can see that the group exists
largelist - some sort of special indexing updated once/day? For permits with > 5000 members. Works like other permits, but the list of NetIDs assigned to a permit is stored in a text file, as opposed to a DBM record in the reverse index DBM file. Historical note: this text file is updated for permit assignment, but not updated on permit unassignment, until rebuild.sh is run. Fortunately, despite this, large list permit assign/unassign works right away, and does not depend on rebuild.sh, and so lookups are always correct. Bonus trivia note: the 5K NetID max is an approximation. According to the source code, actual max length is 30K bytes.	Since "largelist" was a workaround for limitations in the permit server db and since Grouper will use Oracle which handles small and large groups very well, no special handling of largelists will be necessary. We will likely have to add some protections in the UI so that when people choose something like "List group members" for lists above a certain size, the UI doesn't try to display them all. (There will be a download option though... possibly available only to those with update and admin privileges for the group.)
Helpdesk Utility (IDM permit tools - permitutil)	
(Permitutil is a Unix executable written in C by Ron DiNapoli, and accessed via PHP web pages on aads.cit.cornell.edu and also via some Perl scripts written by Todd Zino.)	As the security model for Phase 1 is developed, we will examine how to provide administrative functions. Grouper supports several functions.
List permits for a netid or srvtab	Available in the Grouper UI
Remove a permit issued to a specific netid	Available in the Grouper UI
Issue a permit for a netID	Available in the Grouper UI
Remove all permits associated with a netid	Need to add to the Grouper UI, will also need to be available possibly as a Web Service so that the cleanup script can do this.
Restore all permits for a netid	This is used during the restore process of a netid cleanup. We would probably have to modify the cleanup script to save a user's groups via xml and then have a restore process available from the Grouper UI.
Permit Server CUSSP commands	
NOOP: is the permit server responding to commands?	We need to examine whether this function needs to be available in the Grouper environment.
ckAuth: is authentication working properly? (mutual kerberos v4) - seems to be obsolete.	
getPermit: is netid x allowed to use permit y?	This function will need to be implemented. Possibilities include, but are not limited to, a Web Service or LDAP query.
getStatusP: send me the status of a particular permit. Returns: Authorization count (depracted - should now be 1 or 0); revoked (yes/no); date created; date modified; expiration date.	First check to see if this is ever used. If so, this function will need to be implemented. Possibilities include, but are not limited to, a Web Service or LDAP query.
addPermit: add permit y for netid x	This function will need to be implemented. Possibilities include, but are not limited to, a Web Service or LDAP query.
changePermit: change the user information for netid x and permit y	First check to see if this is ever used outside of permit.cgi or permitutil. This functionality is in the Grouper UI but if necessary, something similar could be implemented as a Web Service.

deletePermit: delete permit y for netid x	This function will need to be implemented. Possibilities include, but are not limited to, a Web Service or LDAP query.
revokePermit: turn off permit y for netid x. Used for bad behavior, requires explicit reinstate.	Currently there is no concept of "revoke" in grouper. We probably need to look at how often revoke and reinstate are used.
reinstatePermit: reinstate permit y for user x	Currently there is no concept of "reinstate" in grouper. We probably need to look at how often revoke and reinstate are used.
listPermits: list permits for netid x or list netids for permit y	This function will need to be implemented. Possibilities include, but are not limited to, a Web Service, an LDAP lookup, or java library.
showPermit: show a detailed description of permit y	First check to see if this is ever used outside of permit.cgi or permitutil. This functionality is in the Grouper UI but if necessary, something similar could be implemented as a Web Service or via some other mechanism.
createPermit: add permit y	First check to see if this is ever used outside of permit.cgi or permitutil. This functionality is in the Grouper UI but if necessary, something similar could be implemented as a Web Service or via some other mechanism.
changeOwner: change owner of permit y	First check to see if this is ever used outside of permit.cgi or permitutil. This functionality is in the Grouper UI but if necessary, something similar could be implemented as a Web Service or via some other mechanism.
addAcl: add netid x to access list of the permit	First check to see if this is ever used outside of permit.cgi or permitutil. This functionality is in the Grouper UI but if necessary, something similar could be implemented as a Web Service or via some other mechanism.
deleteAcl: delete netid x from the access list of permit y	First check to see if this is ever used outside of permit.cgi or permitutil. This functionality is in the Grouper UI but if necessary, something similar could be implemented as a Web Service or via some other mechanism.
setMode: set particular characteristics of permit y	First check to see if this is ever used outside of permit.cgi or permitutil. This functionality is in the Grouper UI but if necessary, something similar could be implemented as a Web Service or via some other mechanism.
setDelay: turn on a major deletion of permits to happen overnight.	probably don't need this?
deleteAll: permit db administrator can delete all references to a particular netid, or can replace a netid name if it has changed. Must be issued twice within 30 seconds before it is accepted.	This function will need to be implemented. Possibilities include, but are not limited to, as Web Service or java library.
largeListPermits: lists large lists (Due to implementation, only works on "large lists" less than 32K in size.)	Probably we won't need to differentiate between large and small lists since they are handled the same in Grouper
showPD: show a detailed description of a particular permit descriptor (the permit descriptor is the characteristics of a permit - ACL's etc.)	First check to see if this is ever used outside of permit.cgi or permitutil. This functionality is in the Grouper UI but if necessary, something similar could be implemented as a Web Service or via some other mechanism.
createPD: add a new permit descriptor	First check to see if this is ever used outside of permit.cgi or permitutil. This functionality is in the Grouper UI but if necessary, something similar could be implemented as a Web Service or via some other mechanism.
changeFriendly: change the friendly name of an existing permit	Changing a display extension or description is available in the Grouper UI
removeNetID: permit db admin can delete all references to a particular netid or can replace a netid name if it has changed	Need to add to the Grouper UI, will also need to be available possibly as a Web Service so that the cleanup script can do this.
removePermits: permit db admin can remove all permits and permit descriptor from db.	Group deletion is available in the Grouper UI.
bulkReplace:anybody with update access or above can replace a file full of netids for a permit.	Can be part of the Grouper UI file upload command.
bulkAdd: anybody with update access or above can add a file full of netids to a permit.	Will be added to the Grouper UI.
bulkDelete: anybody with update access or above can delete a file full of netids from a permit.	Will be added to the Grouper UI.
GETINFO: general server info.	Not sure we need this?
AUTHSVC: what is the kerberos id of the permit server?	Not sure we need this?
AUTHCHK: returns the kerberos principal that issued the AUTHCHK command.	Not sure we need this?
Logging:	
A typical log entry is ... Date/time stamp IP address and port of incoming request Kerberos principal of process making request CUSSP Permit Server command Permit name NetID Response sent by Permit Server	Grouper logging is currently being expanded for version 1.0 and 1.1. This section will be filled in when the next version is available.
Logs are rotated monthly, stats are run, and are then archived for 7 years.	We can do the same with Grouper logs.
Permitclient admin utility	
Used only by Identity Management Permit server admins. Permitclient is a Unix executable written C, but usually invoked via script, to alleviate typing many command line options. Does many handy Permit Server CUSSP commands, but not all.	The Grouper UI will provide some if not all of this functionality.
Other:	
The owner of a permit can create new permits in the next level, and the owner can be changed to give ownership of that permit to someone else. (doesn't give away the entire hierarchy.)	A subject with correct privileges for a stem can create new groups or stems and assign owners to them. Inheritance of privileges is handled differently in Grouper, but we can do some user education on that point.
To start a new hierarchy at the top level requires the permit db administrator.	Grouper requirements are similar - you must be a member of the Wheel group. The Wheel group defines users who are overall administrators for Grouper.

For each netid for each permit they own we keep track of: numpermits (how many ways a person has been authorized) any unassignment will immediately put that reference count to 0, regardless of its current value; revoked (yes or no); created (time/date created); modified (time/date modified); time scheduled to expire - never used; userinfo (service provider space) - appears not to be used.	Numpermits is now a deprecated function so we don't care about that. The revoke function needs to be assessed as mentioned in items above. The developers of Grouper are aware of the need for more detailed logging. Created/Modified date may be available in the future.