# Grouper membership eligibility requirements

Wiki	Grouper Release	Grouper	Grouper Deployment	Community	Internal Developer
Home	Announcements	Guides	Guide	Contributions	Resources

This applies to Grouper v2.6.16+

To enforce membership eligibility, you can use a composite, rules, JEXL scripted groups, or you can use this new feature. You can link an attribute with an eligibility group so that immediate memberships (not effective, composite, loaded) from a (probably) manual group will be veto-ed or removed when users are no longer eligible because they are no longer effectively in the eligible population group.

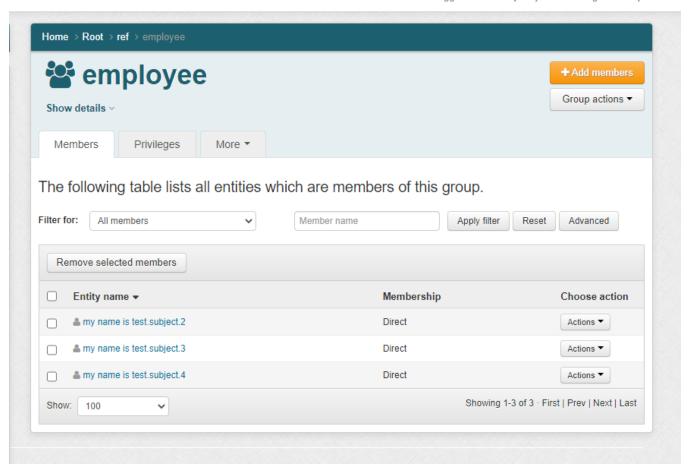
#### Steps to implement

These are explained below

- 1. Identify coarse grained populations that manual groups could be constrained to
- 2. Create attribute def(s) and attribute names.
  - a. Make as many attribute defs as you want (equal to or less than the number of populations)
  - b. The number of attribute names is 1-to-1 to the coarse grained populations.
- 3. Allow certain groups to be able to read and assign the attribute (e.g. power users)
- 4. Configure the attribute in grouper properties to be linked to an eligibility group (e.g. employees)
- 5. Configure the veto text in the externalized text file
- 6. Configure the attribute to be assigned in the group edit screen (optional)
  - a. We can allow attributes on stem edit screens in future
- 7. Assign the attribute to groups or folders
  - a. Note, loader groups are not affected
- 8. Membership hook will veto membership adds if the user is not eligible
- 9. Change log consumer will remove members when no longer eligible
- 10. Full sync daemon will make sure everything is correct
- 11. When members are removed a record is kept in the grouper mship reg change table
  - a. A simple GSH script could easily rollback changes made by this module

### Identify coarse grained populations

Note, these do not need to be direct members.



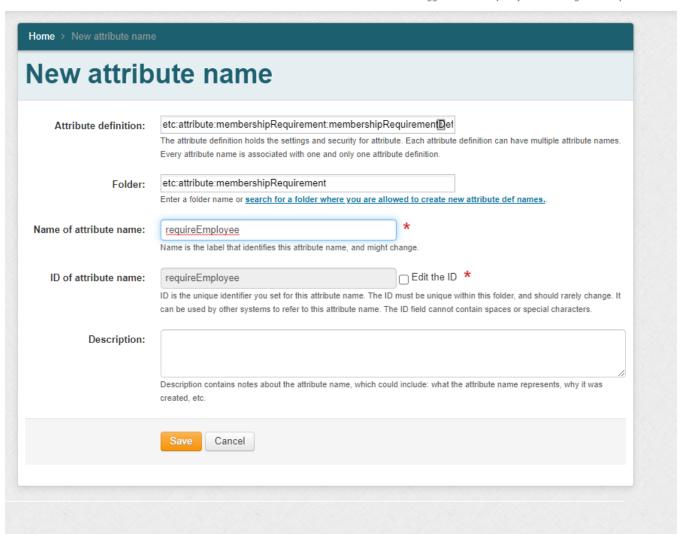
#### Create attributes

These settings need to be exactly like this. Needs to not be multi-assignable or have a value...

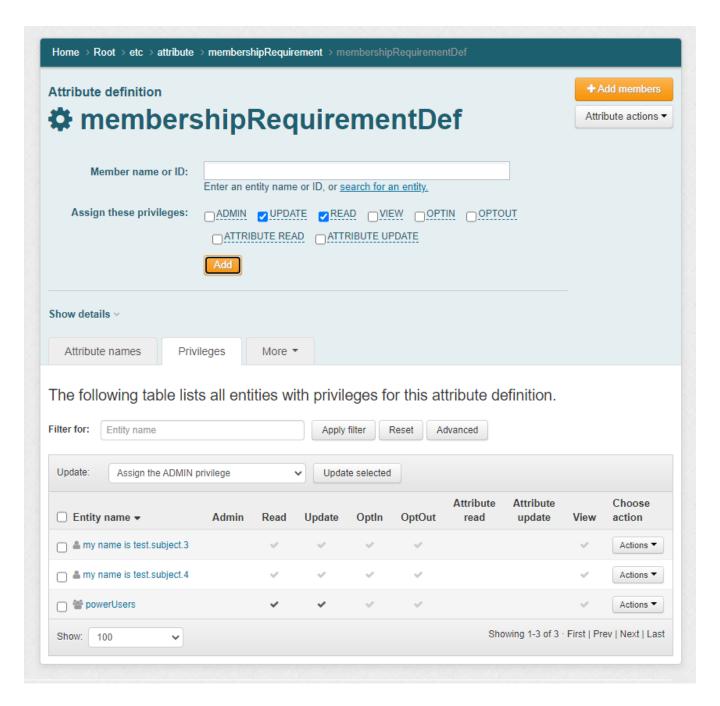
# membershipRequirementDef

# Edit attribute definition

Attribute definition ID:					
	•	e definition. It should be short and simple, and might have character restrictions. The			
	ID should rarely change, if ever.				
Description:					
		//			
	Description contains notes about the attribute definition, which could include: what the attribute definition represents, why it was created, etc.				
Type:	Attribute	•			
	Attribute definition type describes the att	tribute definition. Generally it will be attribute or permission. Type is used for templates,			
	limit describes a permission, and service identifies which application the object refers to.				
Assign to:	Attribute definition	Attribute definition attribute assignment			
Assign to.	✓ Folder	Folder attribute assignment			
	Group / Role / Local entity	Group / Role / Local entity attribute assignment			
	Member	Member attribute assignment			
	Membership	Membership attribute assignment			
		Membership - immediate only - attribute assignment			
	Membership - Infinediate only	Membership - Inimediate only - attribute assignment			
	Designate which types of objects that this definition can be assigned to. There are six base object types, or you can assign				
	_	to those base object types. Membership can be assigned to an immediate or an			
	• • • • • • • • • • • • • • • • • • • •	as an orphan if the membership is unassigned until the membership is reassigned.  ments are only assignable to immediate memberships and are automatically deleted			
	once the membership is unassigned.	monto are only accognizate to miniculate memberships and are automatically account			
Multi-assignable:	_				
	If this attribute can be assigned to the sa Rule attached to it, so the Rule attribute	ame owner object more than once. For instance, a Group can have more than one			
	rate ditacted to it, 30 the rate ditribute	is muni-assignatio			
Value type:	No value	~			
	If this attribute assignment holds one or	more values, this is the type. If there are no allowed values, select No value.			
Multi-valued:					
Widia-Valued.	If this attribute has values, if it can have more than one value assigned at once.				
Show advanced properties >					



Allow certain people to be able to read and assign the attribute (e.g. power users)



#### Configure the attribute in grouper.properties to be linked to an eligibility group (e.g. employees)

```
# ui key to externalize text (error message)
grouper.membershipRequirement.requireEmployee.uiKey = vetoRequireEmployee

# attribute name that signifies this requirement
grouper.membershipRequirement.requireEmployee.attributeName = etc:attribute:membershipRequirement:
requireEmployee

# group name which is the population group
grouper.membershipRequirement.requireEmployee.requireGroupName = ref:employee

# if the overall hook is enabled, is the hook for this specific config enabled? defaults to true.
grouper.membershipRequirement.requireEmployee.hookEnable = true
```

```
## Custom veto composites membership requirement
## This feature allows users to auto-veto ineligible members or remove them when they become ineligible.
## Note that each custom composite also needs to be defined in the Grouper UI text properties in order
## to provide a friendly description in the UI. customCompositeMinusEmployees and customCompositeIntersectIt
are also defined as examples there.
# how long should logs of membership requirement logs be stored in database?
grouper.membershipRequirement.keepLogsForDays = 90
# should hook for membership veto be enabled
grouper.membershipRequirement.hookEnable = true
# should changeLog for membership veto change log be enabled in general
grouper.membershipRequirement.changeLogEnable = true
# ui key to externalize text
#grouper.membershipRequirement.someConfigId.uiKey = customVetoCompositeRequireEmployee
# attribute name that signifies this requirement
#grouper.membershipRequirement.someConfigId.attributeName = etc:attribute:customComposite:requireEmployee
# group name which is the population group
#grouper.membershipRequirement.someConfigId.requireGroupName = org:centralIt:staff:itStaff
# if the overall hook is enabled, is the hook for this specific config enabled? defaults to true.
#grouper.membershipRequirement.someConfigId.hookEnable = true
```

### Configure the veto text in the externalized text file

"grouper.text.en.us.properties" (veto.membershipVeto.customComposite.%uiKey% = %text you want displayed as error%)

veto.membershipVeto.customComposite.vetoRequireEmployee = Only employees can be members of this group

### Configure the attribute to be assigned in the group edit screen (optional)

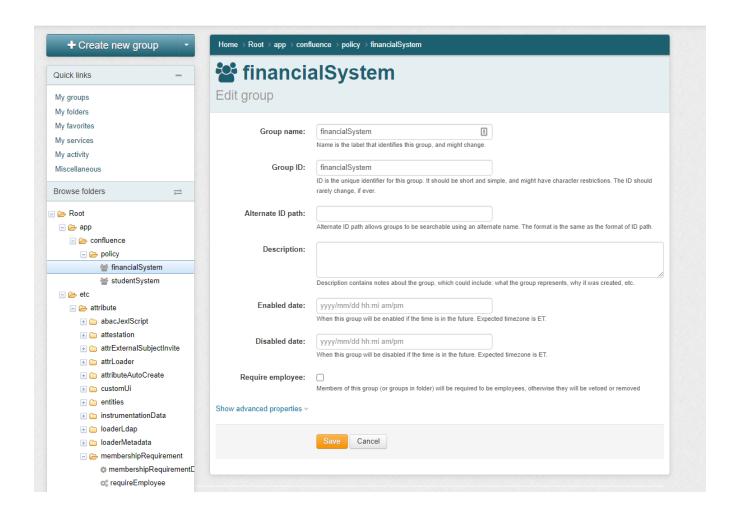
grouper.properties

```
groupScreen.attribute.requireEmployee.attributeName = etc:attribute:membershipRequirement:requireEmployee groupScreen.attribute.requireEmployee.label = Require employee groupScreen.attribute.requireEmployee.description = Members of this group (or groups in folder) will be required to be employees, otherwise they will be vetoed or removed groupScreen.attribute.requireEmployee.index = 1
```

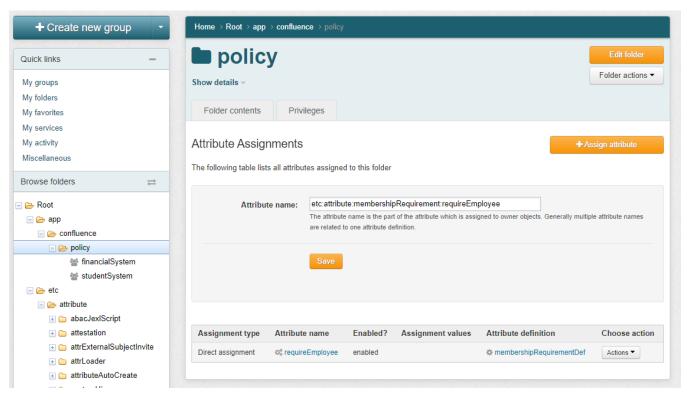
## Assign the attribute to groups or folders

Note, this will remove direct members. Note, be careful about service principals, maybe those need to be in a different manual group?

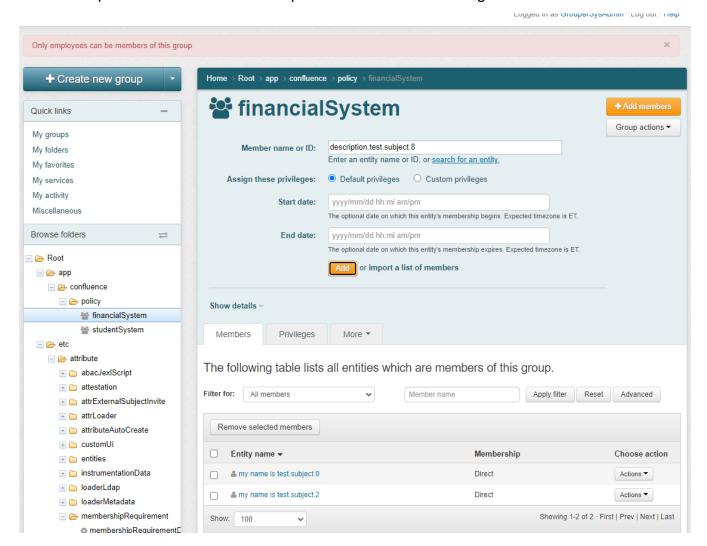
Groups:



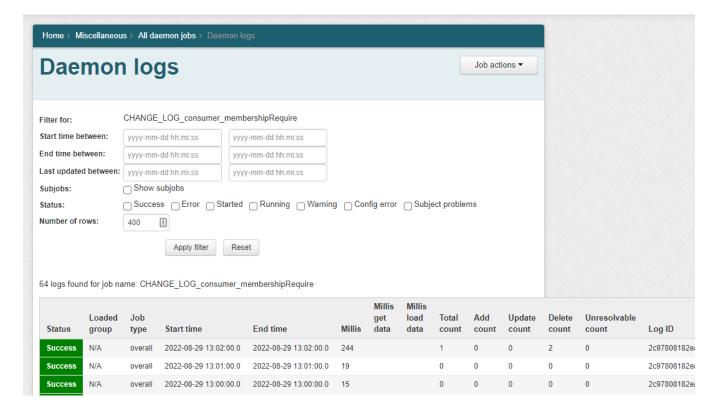
#### Folders



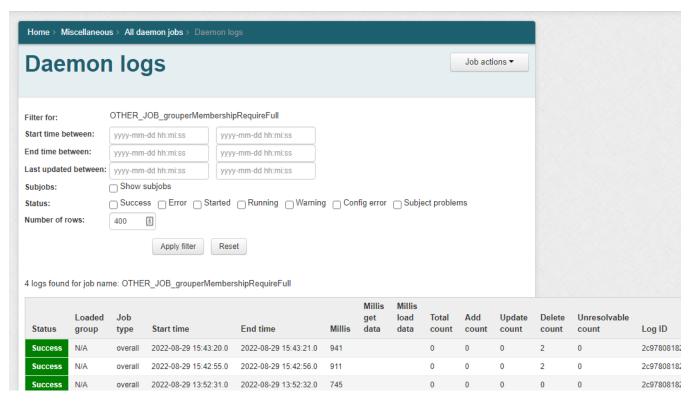
#### Membership hook will veto membership adds if the user is not eligible



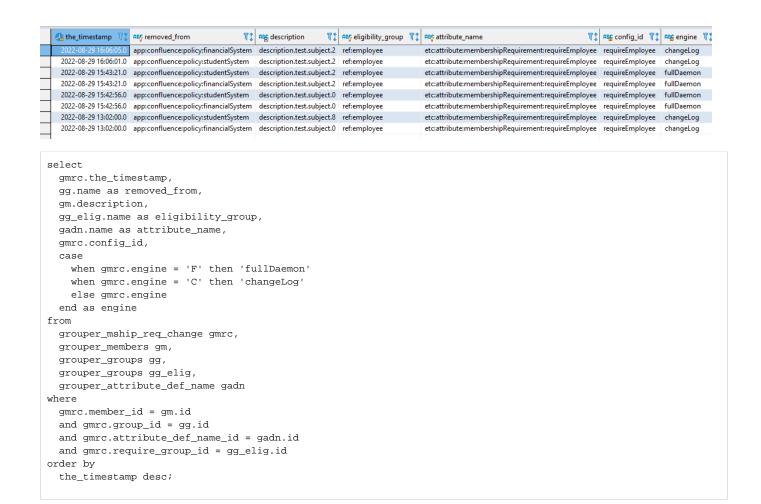
Change log consumer will remove members when no longer eligible



Full sync daemon will make sure everything is correct



When members are removed a record is kept in the grouper\_mship\_req\_change table



#### TO DO

More features can be added to this:

- 1. Notifications (to managers or users)
- 2. Grace periods
- 3. Read-only mode
- 4. Exclude groups which are "exclude" type (doesn't exist yet)
- 5. Exclude groups by regex
- 6. Include only manual groups
- 7. Constrain subject sources
- 8. Remove when membership remove in folder (e.g. job or title changes)
- 9. Loader can restrict ineligible members
  - a. Confirm that JEXL scripted groups are affected correctly