Organizing services in Grouper

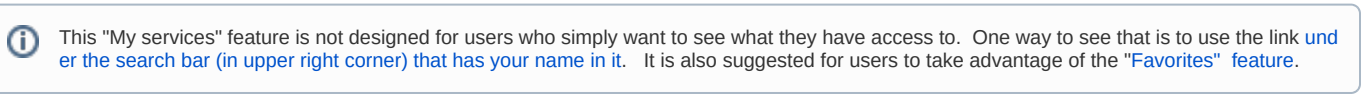
Overview

Grouper has a hierarchical namespace of folders in which to organize Groups, Roles, and Permissions. For large deployments of Grouper, the namespace can make the UI overwhelming for users to find the objects they want to manage.

In Grouper 2.2, the UI makes it possible to filter the registry by a service. This makes it simpler to navigate the namespace and easier to use the UI. Users can find services in the "services widget" on the main UI screen.

Potential Use Cases

- Application owners can tag the main folder of their application as a "Service" so that users can easily find the service in the Grouper registry.
- An admin of a service can go to the UI and see which services they are an admin of, and filter the UI by that service
- A "My Services" thing in the Grouper 2.2 UI that shows users what services they can access, or perhaps can't access, and that shows users that are also a service admin for one or more services an indication that they have that role and maybe even an ability to exercise it.
- A user wants to know whether they are permitted to access a given service, and if not, a step they might take towards (re)establishing their access to it.
- Service Desk staff able to do the same on behalf of a user.
- Report on all services whose access is managed by a given Grouper instance, even if those services aren't all provided by the same IT shop.
- Service names are assigned by multiple people given that permission, and so they must not collide.
- Services could filter the subjects able to be resolved?
- A user of a service can go to the UI and see which services they are a user of, and filter the UI by that service



This "My services" feature is not designed for users who simply want to see what they have access to. One way to see that is to use the link under the search bar (in upper right corner) that has your name in it. It is also suggested for users to take advantage of the "Favorites" feature.

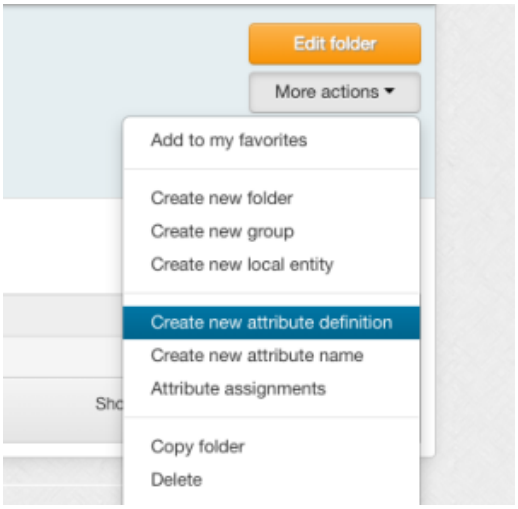
Service design

The design for services is based on the attribute framework. There is an attribute definition type reserved for this, called: service. Originally it was called "domain" but never used, and this has been refactored to "service" in Grouper 2.2+, and is automatically migrated on upgrade. A service attribute is assignable to folders. Any user who can create objects in a folder can create a service attribute and attribute name and assign it to their objects.

Creating services

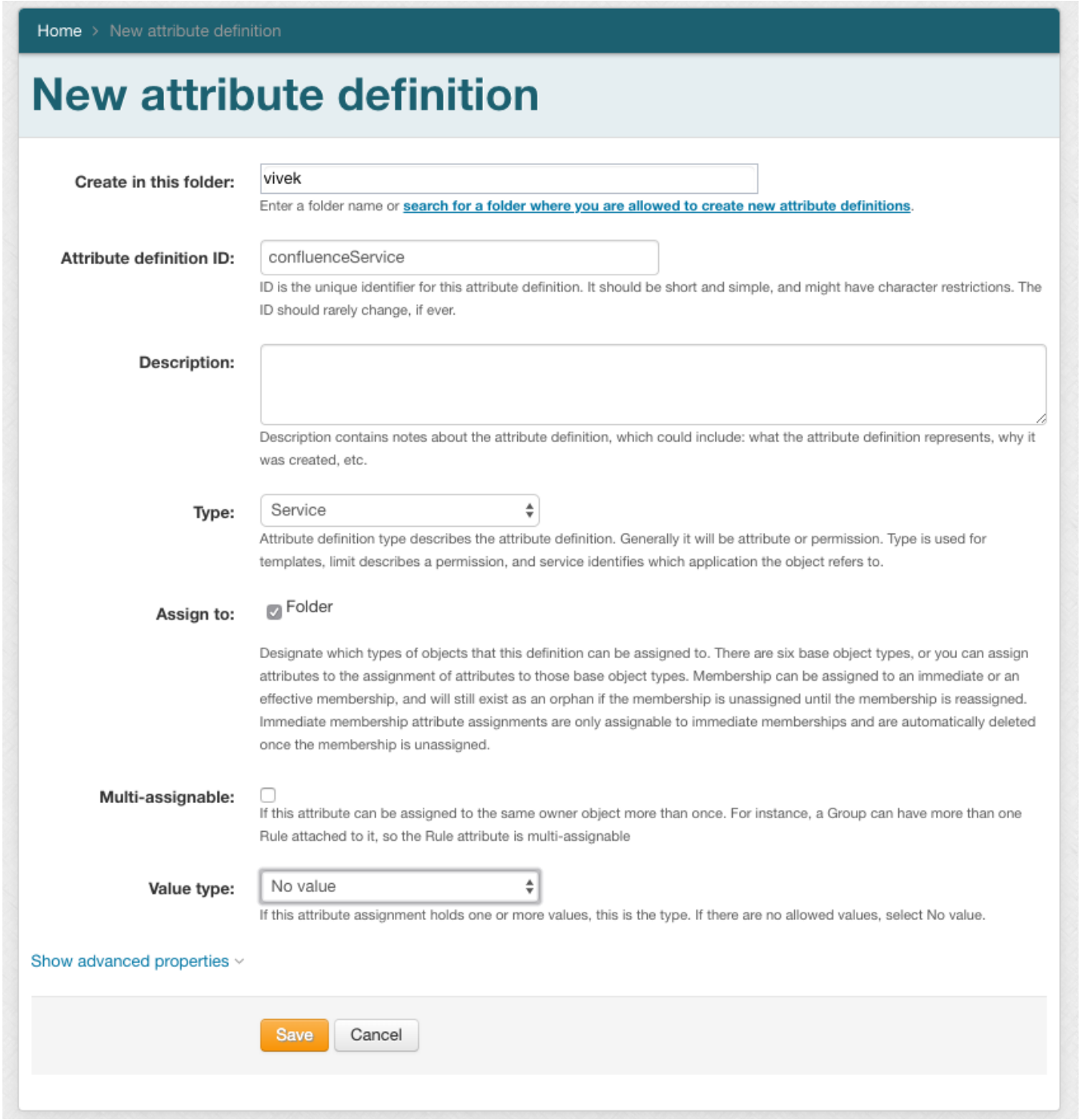
The service attribute definition can be created via GSH or the UI. Here is an example via the UI as of v2.4

Click on Create new attribute definition

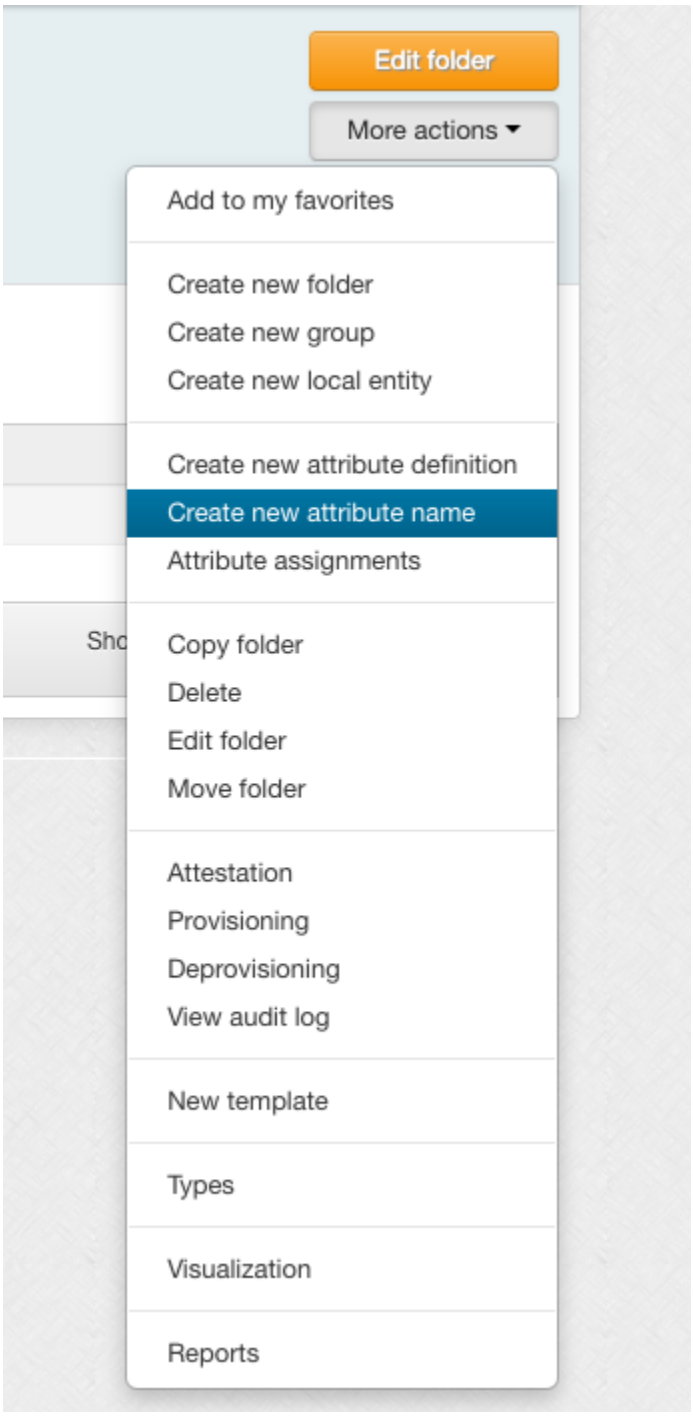


---

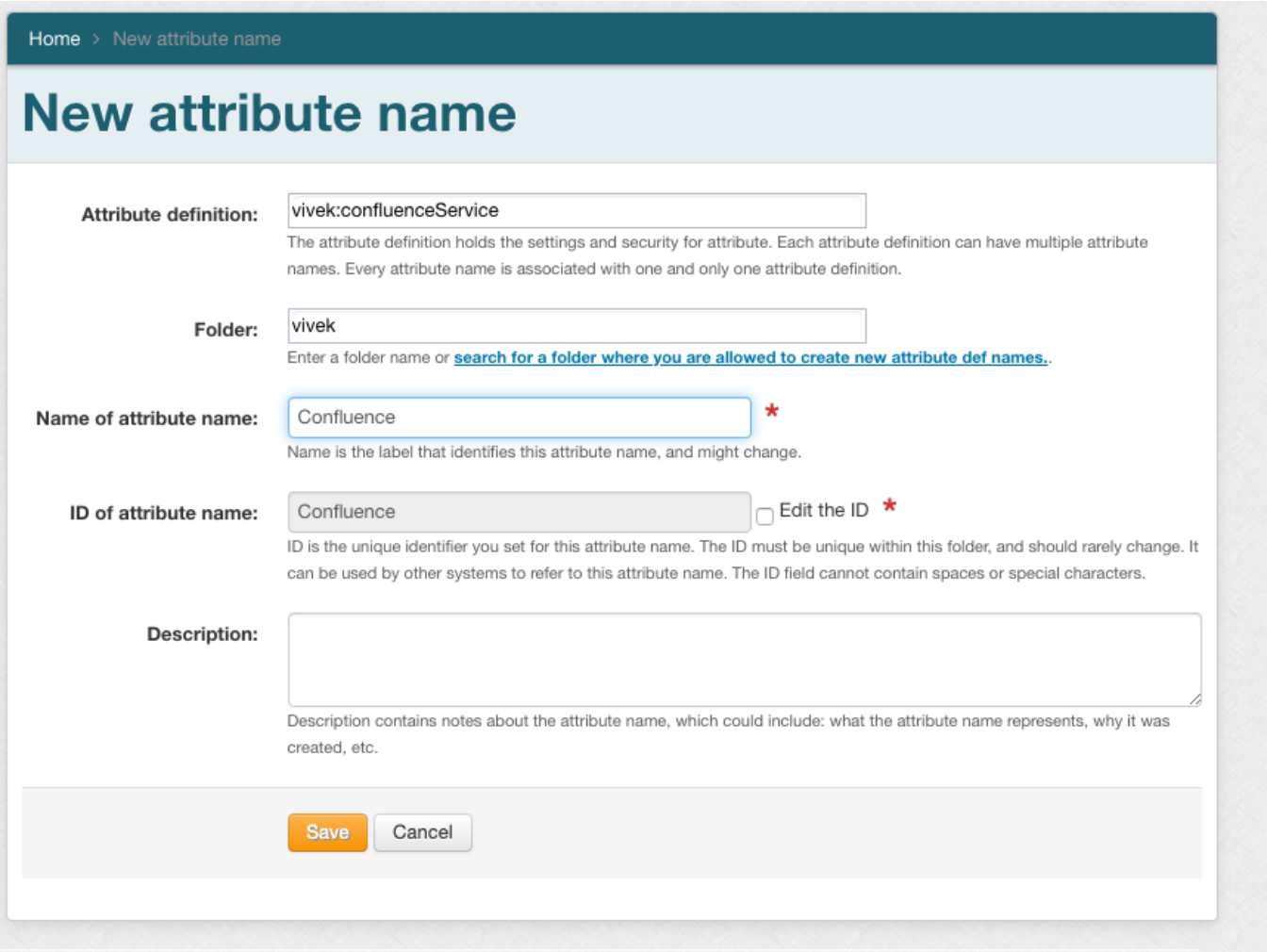
fill out form and save



Click on "Create new attribute name"

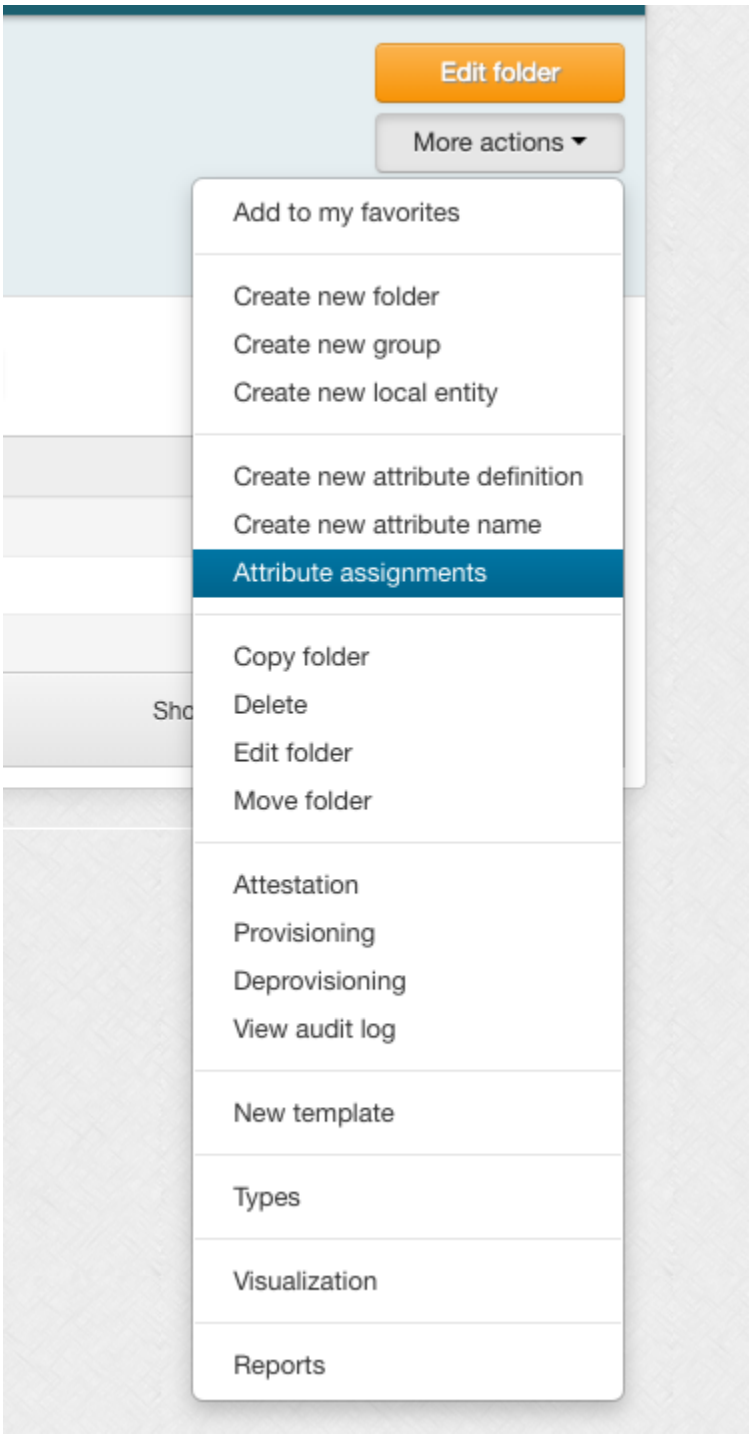


fill out the form and save

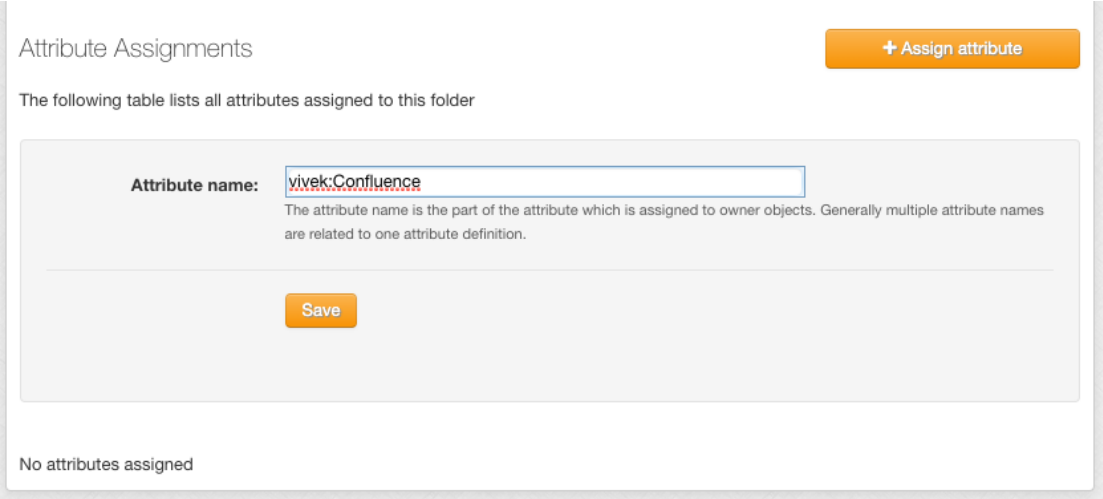


Assign this attribute to the container folder for the application

Click "Attribute assignments"



assign newly created attribute to folder

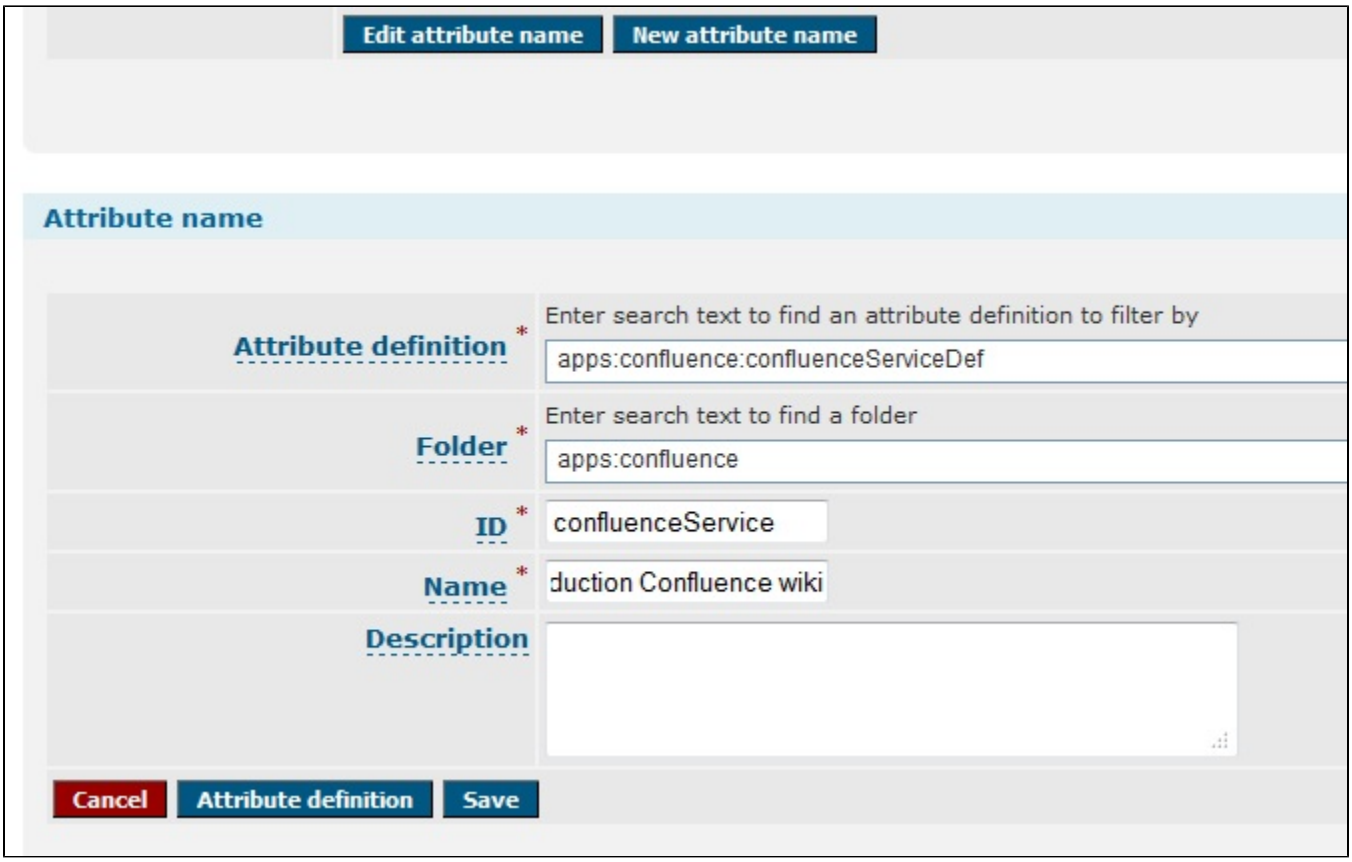


Click save. Now any entities who are members of a group in that folder or subfolder will have that service on the main screen or services screen of the UI. Any entity who has manage privileges on a group in that folder or subfolder, will be considered an "admin" of the service. Note with groups as members and with exclude lists, this can not be 100% accurate.
Here is a GSH example of creating a service:

```
AttributeDef jiraServiceDef = new AttributeDefSave(grouperSession)
      .assignCreateParentStemsIfNotExist(true).assignAttributeDefType(AttributeDefType.service)
      .assignName("apps:jira:jiraServiceDefinition").assignToStem(true).save();

AttributeDefName jiraService = new AttributeDefNameSave(grouperSession, jiraServiceDef)
      .assignCreateParentStemsIfNotExist(true)
      .assignName("apps:jira:jiraService").assignDisplayExtension("Central IT production Jira issue tracker").
save();
```

Here are screenshots on the UI of creating a service definition and name (TODO, IT SHOULD ONLY BE ASSIGNABLE TO FOLDERS!):



Assigning service tags

Service tags can be assigned to folders by GSH, UI, WS, or rule. The service tag applies to the folder, all objects in the folder, and subfolders

Service attribute validation

A Grouper Service attribute definition can not be assigned to objects other than folders. It is a marker attribute (i.e. tag), which has no value. It is single assignable, i.e. you cannot assign the attribute more than once to the same object owner.

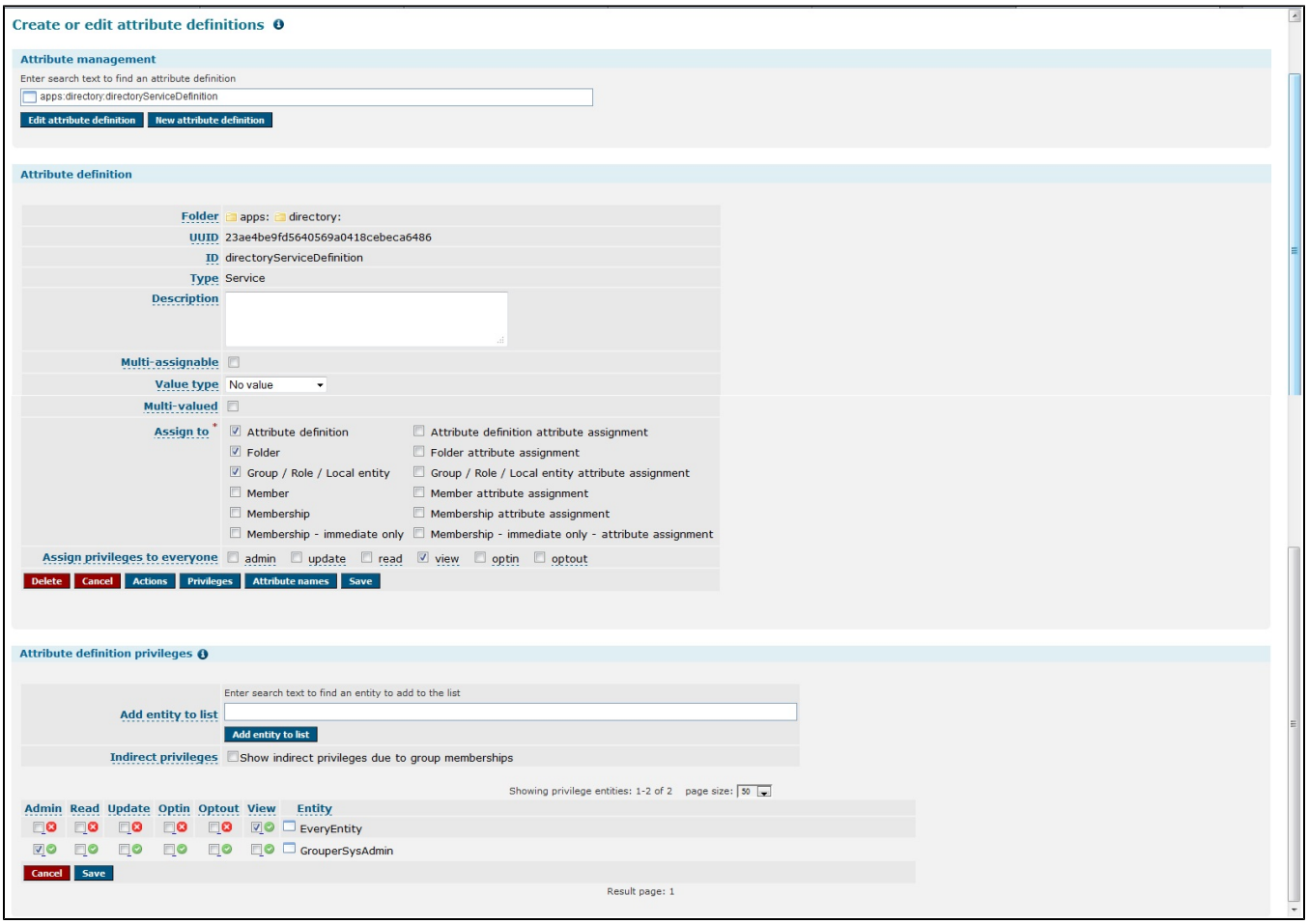
Services privileges

Services have the same privileges that attributes have: READ, VIEW, ADMIN, UPDATE. Note, the VIEW privilege is more complicated, generally you dont have to worry about assigning this, since if someone cannot VIEW the service, but can manage groups in the service, then they still CAN see the service in the UI/WS. This makes using services a lot easier.

GSH/API example of service privileges

```
//the directory is public
directoryServiceDef.getPrivilegeDelegate().grantPriv(SubjectFinder.findAllSubject(), AttributeDefPrivilege.
ATTR_VIEW, false);
```

UI example of service privileges (TODO: NOTE ONLY ASSIGNABLE TO FOLDERS!!!)



Using Grouper services

Services should be exposed by the UI/WS/API. i.e. you should be able to do a GroupFinder filter and restrict the results to a certain service. You should be able to list the services for a user. You could be able to browse the repository and operate comboboxes in the context of a particular service.

API

```
Find members in a service:

      MembershipResult membershipResult = new MembershipFinder().assignServiceId(confluenceService.getId())
          .assignServiceRole(ServiceRole.admin).findMembershipResult();

      List<Member> members = new ArrayList<Member>(membershipResult.members());


Find services for a user:


      Set<AttributeDefName> attributeDefNames = new AttributeDefNameFinder().assignSubject(SubjectTestHelper.
SUBJ0)
        .assignServiceRole(ServiceRole.user).findAttributeNames();
```

WS

- getMemberships: find members in a service: send in the serviceRole (e.g. admin|user), and service lookup (uuid or name of service)
- findAttributeDefNames: find services for a user: find in the serviceRole (e.g. admin|user), and entity lookup (subject id, source id, subject identifier)

Client

- find members in a service:

```
c:\temp> java -jar grouperClient.jar --operation=getMembershipsWs --serviceName=school:apps:wiki --
serviceRole=admin
Index 0: group: school:apps:wiki:groups:admins, subject: jsmith, list: updaters, type: Immediate, enabled: T
Index 1: group: school:apps:wiki:groups:users, subject: hjohnson, list: admins, type: Immediate, enabled: T
```

- find services for a user:

```
c:\temp> java -jar grouperClient.jar --operation=findAttributeDefNamesWs --scope=% --serviceRole=user --
subjectId=jsmith
Index 0: name: school:apps:wiki, displayName: School:Applications:Wiki
Index 1: name: school:apps:pto, displayName: School:Applications:Paid Time Off
```

Metadata

We should have a requirement that there is one service definition per service name, and there could be some built in metadata on the service definition, like who the owner is, a link if people have problems, contact info, description, etc.

Developing

- There is a view that is hibernated and can be joined to for queries: grouper_service_role_v mapped to Java object: ServiceRoleView
- You can join to this view by group, or attribute, etc. The membership column should have security joined to it:

```
whereClause.append(" and theAttributeDefName.id = theServiceRoleView.serviceNameId ");
   changedQuery = grouperSession.getAccessResolver().hqlFilterGroupsWhereClause(
     grouperSession.getSubject(), byHqlStatic,
     sql, "theServiceRoleView.groupId", AccessPrivilege.READ_PRIVILEGES);

   //fields for the service role
   HibUtils.convertFieldsToSqlInString(serviceRole.fieldsForGroupQuery(), byHqlStatic, whereClause,
"theServiceRoleView.fieldId");
   whereClause.append(" and theServiceRoleView.memberId = :groupMemberId ");
   byHqlStatic.setString("groupMemberId", member.getUuid());
```

API

In the Grouper API you can get services for a user (note, this is for groups in services where the grouper session can read memberships (or admin)

```
Set<AttributeDefName> attributeDefNames = new AttributeDefNameFinder().assignSubject(SubjectTestHelper.SUBJ0)
        .assignServiceRole(ServiceRole.user).findAttributeNames();
```

Here is a web service request:

```
<WsRestFindAttributeDefNamesRequest>
  <scope>%</scope>
  <serviceRole>user</serviceRole>
  <subjectLookup>
    <subjectId>test.subject.0</subjectId>
  </subjectLookup>
</WsRestFindAttributeDefNamesRequest>
```

Web service response:

```
<WsFindAttributeDefNamesResults>
  <attributeDefNameResults>
    <WsAttributeDefName>
      <idIndex>10090</idIndex>
      <extension>jiraService</extension>
      <displayExtension>Central IT production Jira issue tracker
      </displayExtension>
      <displayName>apps:jira:Central IT production Jira issue tracker
      </displayName>
      <name>apps:jira:jiraService</name>
      <uuid>d528f5888e964384be6cc7ed39e3d006</uuid>
      <attributeDefId>05b934189bd342aba0979fafec5e9c07</attributeDefId>
      <attributeDefName>apps:jira:jiraServiceDefinition
      </attributeDefName>
    </WsAttributeDefName>
  </attributeDefNameResults>
  <attributeDefs>
    <WsAttributeDef>
      <idIndex>10022</idIndex>
      <extension>jiraServiceDefinition</extension>
      <name>apps:jira:jiraServiceDefinition</name>
      <uuid>05b934189bd342aba0979fafec5e9c07</uuid>
      <attributeDefType>service</attributeDefType>
      <multiAssignable>F</multiAssignable>
      <multiValued>F</multiValued>
      <valueType>marker</valueType>
    </WsAttributeDef>
  </attributeDefs>
  <resultMetadata>
    <resultCode>SUCCESS</resultCode>
    <resultMessage>Success for: clientVersion: 2.2.0, scope: %,
      splitScope: null, wsAttributeDefLookup: null, attributeAssignType:
      null, attributeDefType: null
      wsAttributeDefNameLookups: null
      wsInheritanceSetRelation: null, pageSize: null, pageNumber: null, sortString: null, ascending:
      null, actAsSubject: null, paramNames:
      , params: null
      , wsSubjectLookup: WsSubjectLookup[subjectId=test.subject.0],
      serviceRole: user
    </resultMessage>
    <success>T</success>
  </resultMetadata>
  <responseMetadata>
    <resultWarnings></resultWarnings>
    <millis>9285</millis>
    <serverVersion>2.2.0</serverVersion>
  </responseMetadata>
</WsFindAttributeDefNamesResults>
```

Questions

1. If you are a member of a group in a service, then should you see it in your service drop down? (yes because it would be too hard to maintain if each user of a service has to be assigned to VIEW the service)
2. Can you see groups you are a member of if you cannot READ that group (My Memberships)? I think that is the current functionality. if it is then it will continue to work