Grouper à Rennes1

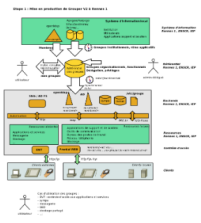
Cette page détaille la mise en oeuvre de Grouper V2 à Rennes1 :

- la classification adoptée pour les groupes,
- les mécanismes d'alimentation à partir des bases métier,
- les principes de mise en oeuvre de la délégation,
- les versements dans les annuaires,
- le déprovisionning.

Contexte -besoins

- baser la politique d'autorisation aux ressources et applications sur Grouper
- gérer des groupes ad hoc (non-institutionnels) en promouvant la délégation de droits à des personnes dans la DSI et hors de la DSI
- utiliser les groupes institutionnels issus des bases "métiers" : structures Harpège et structures des enseignements Apogée
- synchroniser les bases métiers, le référentiel Grouper et les services d'annuaires , en maîtrisant le cycle de vie
- adopter une gestion multi-établissements

Organisation générale



- Grouper 1.6.3 en fin de production
- Grouper 2.2.0, en pré-production, 2.2.1 en production fin Juin 2015
- interfaces casifiées et francisées (liteUI, admin, newUI)
- pas de synchronisation de LDAP vers Grouper
- création des groupes institutionnels dans Grouper par batch java (Harpege) et job Talend (Apogee)
- les groupes sont utilisées par les applications et ressources uniquement via LDAP
  - PSP pour verser de Grouper vers OpenLDAP et Active Directory
  - synchro incrémentale PspChangeLogConsumer toutes les 10 secondes

- grouperLoader pour alimenter
    - tous les groupes institutionnels (Harpege, Apogee)
    - certains rôles applicatifs (Gestimmo, Sifac, WinPaie,...)
- modèle de délégation basée sur l'héritage des privilèges (RuleApi.inheritFolderPrivileges, RuleApi.inheritGroupPrivileges)

Arborescence des groupes

Les choix de noms ont été conditionnés par les rendus des différents services (grouper-ui, annuaires) et la gestion par dossiers. C'est pour cela que nous avons mis dans des branches différentes les services, les UFR et les laboratoires.

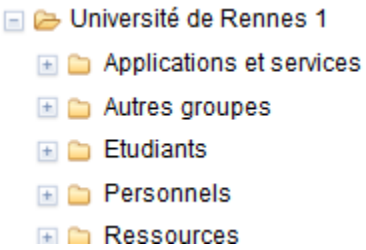
Pour la facilité des utilisateurs finaux, nous avons essayé de garder une sémantique unique à chaque niveau de l'arborescence.

La racine

Le dossier de premier niveau (racine) s'appelle "**Etablissements**" , car nous sommes dans un contexte multi-établissements.

Le niveau établissement

Chaque établissement contient cinq dossiers:

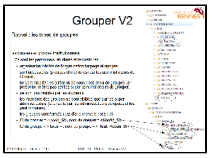


Deux dossiers contiennent les groupes institutionnels (*Etudiants* , *Personnels*) et les trois autres les groupes non-institutionnels (*Applications et services*, *R essources*, *Autres groupes*).

L'identifiant des dossiers (ID Path) : *etu*, *pers*, *app*, *res*, *div*

NB : il n'y a pas de dossier *Administrateurs* à ce niveau : les groupes des administrateurs délégués sont définis dans les dossiers eux-mêmes

Les groupes institutionnels de l'établissement



Les principes suivants sont adoptés pour construire les arborescences des groupes des personnels et des étudiants:

- chaque niveau (une structure, une composante, un laboratoire, une étape, etc...) contient
    - un ou plusieurs sous-dossiers ayant pour *ID <code_SI>* et de nom *<libelle_SI>*
    - le groupe d'ID "*tous*" et de nom "*Tout_<code_SI>*" ayant pour membres directs les sous-groupes de niveau inférieur
- au dernier niveau (la feuille) , le groupe "*Tout_<code_SI>*" qui a pour membres direct les comptes des personnes

avec :

- *<code_SI> : code Harpege ou Apogee* du niveau courant
- *<libelle_SI> :* libellé Harpege ou Apogee du niveau courant

Les groupes des *Personnels*

Ils sont issus d'Harpege et reflètent les différents niveaux des structures:

- *Services* (*ser*) : les groupes des personnels des services centraux et communs
- *Ufr,instituts, écoles* (*ufr*) : les groupes des personnels des composantes d'enseignement
- *Laboratoires* (*lab*) : les groupes des personnels des laboratoires

L'identifiant des dossiers : *ser, ufr, lab*

Les groupes des *Etudiants*

Ils sont issus d'Apogee et sont classés par inscription administrative selon le plan de classement suivant :

> *"2014-2015"* ("2014") > "*Inscriptions Administratives*" ("IA") > *libellé composante (code Apogee)* > *type de diplôme* (*code Apogee*) > *version de diplôme* (code Apogee) > *version d'étape (code Apogee)*

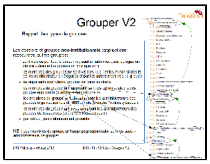
NB : il est prévu à moyen terme de développer la branche "Inscription Pédagogique".

Les groupes non institutionnels

Nous souhaitons que l'administration de ces groupes soit faite par une (ou plusieurs) personne nommée.

La pricipale difficulté pour ces administrateurs est de savoir quand il faut créer un profil applicatif, quand créer un groupe d'accès à une ressource, et surtout quand créer un groupe oragnisationnel ou fonctionnel:

- un groupe "Applications et services" (app) implémente un **rôle** vis à vis d'une **seule** application ou service
- un groupe "Ressources" (ress) contrôle l'accès à une ressource matérielle unique ou un type de matériel
- un groupe "Autres groupes" (div) rassemble des personnes ayant la même fonction, la même mission, le même projet, le même point d'intérêt, faisant partie du même groupe de travail, .. , et auquel est donnée l'autorisation d'accès à plusieurs applications ou ressources

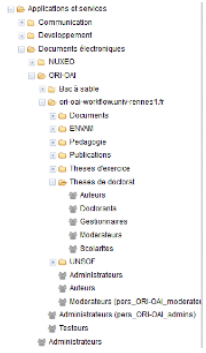


Ils sont classés selon leur type :

profils applicatifs ("*Applications et services*")

- les groupes sont relatifs à une application (service métier, GED, sympa, ...)
- les applications sont classées par domaine ("Communication", "GRH", "Stockage partagé", etc ...)

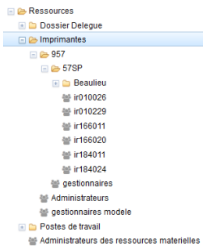
exemple : les rôles pour le référencement des thèses



ressources informatiques ("*Ressources*")

- ces groupes permettent de contôler l'accès/l'utilisation de ressources matérielles
- deux types de ressources actuellement : imprimantes et postes de travail

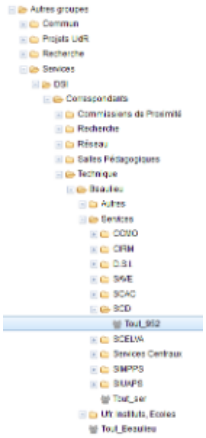
exemple : les groupes d'utilisateurs autorisés pour chaque imprimante de la DSI



les autres groupes ("*Autres groupes*")

- ce sont des groupes transversaux de nature **organisationnelle** (projet, communication, collaboration,...) et **fonctionnelle** (correspondants téléphonie, correspondants techniques de la DSI, ...)
- ils sont internes à une structure (*"Recherche", "Services", "Ufr,instituts,écoles"*) mais peuvent aussi êtres communs à plusieurs structures ("*Communs*", "*Projets UdR*", "*Partages*")

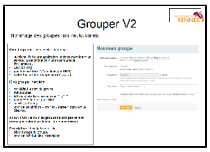
  exemple : les correspondants de la DSI au Service Commun de Documentation



Les règles de nommage des groupes et dossiers non-institutionnels

Les dossiers

Les groupes



Correspondance des champs dans Grouper et des attributs ldap

Correspondance_attributs_grouper_ldap.pdf

Alimentation des groupes

Groupes institutionnels

Les alimentations à partir des bases métiers et la base grouper se font par deux outils développés en interne à Rennes1 :

- le programme java synchroGrpInstit (esup-commons-2) qui permet de créer toute l'arborescence des personnels à partir d'Harpege : les services, les UFR et les laboratoires sous les branches respectives "*Personnels/Services*" , "*Personnels/Ufr,instituts, écoles*" et "*Personnels/Laboratoires*"
- le job Talend qui permet de créer toute l'arborescence des étudiants à partir d'Apogee sous la branche "Etudiants"

Ces batchs lisent les structures Apogée/Harpège et leurs membres et créent des fichiers de lot de commandes à la norme "grouper shell" qui contient les ordres de création/modification des groupes et de leurs membres.

Le fichier lot de commande est ensuite exécuté par le shell grouper.

Groupes non-institutionnel

Chaque groupe est géré par son administrateur désigné.

De plus en plus de groupes non-institutionnels sont issus d'un base de données tierce (les correspondants téléphonie de Harpege, les contributeurs Nuxeo, les administrateurs Gestimmo, etc...) via grouperLoader : ce sont surtout des rôles applicatifs.

Les autres s'appuient la plupart du temps sur les groupes institutionnels des personnels et étudiants pour constituer la liste de leurs membres qui sera maintenue automatiquement à jour.

Cette bonne pratique évitera des soucis de déprovisionning par la suite, en cas de modification de situation des membres : changement de structure, de catégorie, de mise à disposition ...

Délégation à un administrateur désigné

Les règles adoptées

L'"*administrateur délégué*" pourra :

- créer des sous-dossiers et des groupes dans le dossier qu'il administre,
- administrer les groupes du dossier qu'il administre,
- voir les groupes du dossier qu'il administre
- ajouter/supprimer des membres (personnes et/ou groupes) de ces groupes,
- voir la liste des membres de ces groupes,
- créer des groupes (ouverts ou fermés, fermés par défaut),
- attribuer tous les privilèges à d'autres personnes/groupes,
- voir tous les groupes institutionnels,
- voir les membres des groupes institutionnels,
- les droits d'administration « suivent » dans la sous-arborescence

Le "*gestionnaire délégué*" pourra :

- voir les groupes du dossier qu'il accède,
- ajouter/supprimer des membres (personnes et/ou groupes) de ces groupes, sauf celui des administrateurs délégués,
- voir la liste des membres de ces groupes,
- voir tous les groupes institutionnels (personnels et étudiants),
- voir les membres des groupes institutionnels,
- les droits de modification « suivent » dans la sous-arborescence

Le "*gestionnaire délégué*" ne pourra pas :

- administrer/créer des groupes
- créer des dossiers

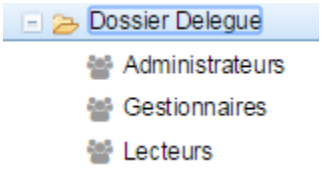
Le « *lecteur* » pourra seulement :

- lire les membres des groupes délégués
- voir les autres groupes délégués
- les droits de lecture « suivent » dans la sous-arborescence

Le membre d'un groupe délégué pourra :

- lire les autres membres du groupe

Le modèle de délégation proposé



La DSI a souhaité garder le controle sur les délégations.

Le demandeur soumet sa requête dans le helpdesk. A la réception la DSI lance un script gsh de génération du dossier qui aura les propriétés suivantes :

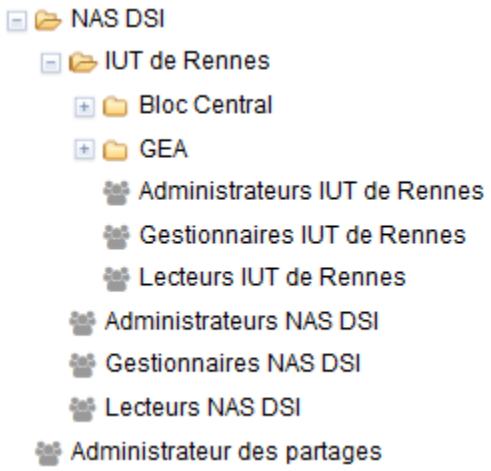
- les administrateurs Grouper administrent le groupe « *Administrateurs* »
- le groupe « *Administrateurs* »
  - administre les groupes « *Gestionnaires* » et « *Lecteurs* »
  - peut créer des groupes et des sous-dossiers dans le dossier délégué
  - contient un seul membre : le demandeur de la délégation (ce peut être une personne ou un groupe)
- le groupe « *Gestionnaires* »
  - peut voir, lire et mettre à jour les membres du groupe « Lecteurs »
  - Il est vide
- Le groupe « *Lecteurs* » est vide

L'administrateur délégué peut alors créer son arborescence :

- lors de la création d'un sous-dossier par l'administrateur délégué, les privilèges de création de groupes et de sous-dossiers est donné au groupe « *Administrateurs* »
- lors de la création, par l'administrateur délégué, d'un groupe dans un sous-dossier :
  - le privilège d'administration de groupes est donné au groupe « *Administrateurs* »
  - le privilège de mise à jour des membres est donné au groupe « *Gestionnaires* »
  - le privilège de lecture des membres du groupe est donné au groupe « *Lecteurs* »
- et ainsi de suite … (délégation totale sur toute l'arborescence) .

Exemple

Délégation pour la prise en main des partages de l'IUT sur le NAS de l'université



- le groupes des administrateurs Grouper gerde évidemment le droit d'administration sur tous les dossiers et groupes existants et futurs,
- "*Administrateurs NAS DSI*" a le droit d'administration sur tous les dossiers et groupes existants et futurs du dossier *"NAS DSI*" ,
- "*Administrateurs IUT de Rennes*" a le droit d'administration sur tous les dossiers et groupes existants et futurs du dossier *"IUT de Rennes*" ,
- "*Gestionnaires NAS DSI*" a le droit de mise à jour sur tous les groupes existants et futurs du dossier *"NAS DSI*" ,
- "*Gestionnaires IUT de Rennes*" a le droit de mise à jour sur tous les groupes existants et futurs du dossier *"IUT de Rennes*" ,
- "*Lecterus NAS DSI*" a le droit de voir les membres des groupes existants et futurs du dossier *"NAS DSI*" ,
- "*Lecteurs IUT de Rennes*" a le droit devoir les membres des groupes existants et futurs du dossier *"IUT de Rennes*" .

Code

Le code du script gsh est basé sur les règles d'héritage de Grouper V2 :

```
// DELEGATION : REGLES D'HERITAGE DES PRIVILEGES
grouperSession = GrouperSession.startRootSession();
//
// PARAMETRESi A RENSEIGNER:
// le dossier a deleguer
chemin = "ur1:app:stockage:nas"
id = "920"
nom = "IUT de Rennes"
dossier = chemin + ":" +id
description = ""
// le demandeur : une personne ou un groupe
demandeur="ur1:pers:ser:957:.....:tous"
// les administrateurs Grouper qui auront le droit d'administrer le groupe des administrateurs delegues
administrateursGrouper="...:adm"


// DOSSIER A DELEGUER
dossierADeleguer = addStem(chemin, id, nom);
dossierADeleguer = StemFinder.findByName(grouperSession, dossier);
RuleApi.reassignStemPrivilegesIfFromGroup(SubjectFinder.findRootSubject(),dossierADeleguer, Stem.Scope.SUB);
RuleApi.reassignGroupPrivilegesIfFromGroup(SubjectFinder.findRootSubject(), dossierADeleguer, Stem.Scope.SUB);
// setStemAttr(dossierADeleguer.getName(),"description",description);
// GROUPES A CREER
// le groupe des administrateurs locaux
administrateurs = addGroup(dossierADeleguer.getName(),"adm","Administrateurs "+ nom);
administrateurs = GroupFinder.findByName(grouperSession, dossierADeleguer.getName()+":adm");
// le groupe des gestionnaires locaux
gestionnaires = addGroup(dossierADeleguer.getName(),"ges","Gestionnaires "+ nom);
gestionnaires = GroupFinder.findByName(grouperSession, dossierADeleguer.getName()+":ges");
//
// le groupes des lecteurs des membres des groupes delegues
lecteurs = addGroup(dossierADeleguer.getName(),"lec","Lecteurs "+ nom);
lecteurs = GroupFinder.findByName(grouperSession, dossierADeleguer.getName()+":lec");
//
// GROUPES INSTITUTIONNELS
// les groupes institutionnels des personnels de Rennes 1
dossierDesPersonnels = StemFinder.findByName(grouperSession, "ur1:pers");
// les groupes institutionnels des etudiants de Rennes 1
dossierDesEtudiants = StemFinder.findByName(grouperSession, "ur1:etu");
//
// GROUPES NON-INSTITUTIONNELS
// les groupes applicatifs de Rennes 1
dossierApplicatifs = StemFinder.findByName(grouperSession, "ur1:app");
// les groupes des ressources de Rennes 1
dossierDesRessources = StemFinder.findByName(grouperSession, "ur1:ress");
// les autres groupes de Rennes 1
dossierAutresGroupes = StemFinder.findByName(grouperSession, "ur1:div");
//
// Droits admin sur le groupe des administrateurs locaux , pour les administrateurs de Grouper
grantPriv(administrateurs.getName(), administrateursGrouper, AccessPrivilege.ADMIN);
// DELEGATION AUX ADMINISTRATEURS LOCAUX
// droit de création de sous-dossiers et de groupes dans le dossier delegue
grantPriv(dossierADeleguer.getName(), administrateurs.getName(), NamingPrivilege.STEM);
grantPriv(dossierADeleguer.getName(), administrateurs.getName(), NamingPrivilege.CREATE);
addMember(administrateurs.getName(),demandeur);
RuleApi.inheritFolderPrivileges(SubjectFinder.findRootSubject(),
 dossierADeleguer , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("stem, create"));
// droit d'administraion des groupes du dossier delegué
grantPriv(gestionnaires.getName(), administrateurs.getName(), AccessPrivilege.ADMIN);
grantPriv(lecteurs.getName(), administrateurs.getName(), AccessPrivilege.ADMIN);
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierADeleguer , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("admin"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierADeleguer , Stem.Scope.SUB, administrateurs.toSubject(),
```

```
Privilege.getInstances("update"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierADeleguer , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("read"));
// visibilités sur les groupes institutionnels et non-institutionnels
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierDesPersonnels , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("read"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierDesPersonnels , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("view"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierDesEtudiants , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("read"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierDesEtudiants , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("view"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierApplicatifs , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("read"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierApplicatifs , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("view"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierDesRessources , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("read"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierDesRessources , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("view"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierAutresGroupes , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("read"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierAutresGroupes , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("view"));
//
// DELEGATION AUX GESTIONNAIRES LOCAUX
// droits de mise à jour/consultation des membres des groupes du dossier delegué
grantPriv(lecteurs.getName(), gestionnaires.getName(), AccessPrivilege.UPDATE);
grantPriv(lecteurs.getName(), gestionnaires.getName(), AccessPrivilege.READ);
grantPriv(lecteurs.getName(), gestionnaires.getName(), AccessPrivilege.VIEW);
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierADeleguer , Stem.Scope.SUB, gestionnaires.toSubject(),
Privilege.getInstances("update"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierADeleguer , Stem.Scope.SUB, gestionnaires.toSubject(),
Privilege.getInstances("read"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierADeleguer , Stem.Scope.SUB, gestionnaires.toSubject(),
Privilege.getInstances("view"));
// droit de création de groupes
dans le dossier:
RuleApi.inheritFolderPrivileges(SubjectFinder.findRootSubject(),
dossierADeleguer , Stem.Scope.SUB, gestionnaires.toSubject(),
Privilege.getInstances("create"));
// visibilités sur les groupes institutionnels et non-institutionnels
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierDesPersonnels , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("read"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierDesPersonnels , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("view"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierDesEtudiants , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("read"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierDesEtudiants , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("view"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierApplicatifs , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("view"));
```

```
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierDesRessources , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("view"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierAutresGroupes , Stem.Scope.SUB, administrateurs.toSubject(),
Privilege.getInstances("view"));
//
// DELEGATION AUX LECTEURS LOCAUX
// droit de consultation du contenu des groupes du dossier delegué
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierADeleguer , Stem.Scope.SUB, lecteurs.toSubject(),
Privilege.getInstances("read"));
RuleApi.inheritGroupPrivileges(SubjectFinder.findRootSubject(),
 dossierADeleguer , Stem.Scope.SUB, lecteurs.toSubject(),
Privilege.getInstances("view"));
```

Synchronisation Grouper - Services d'annuaires

Grouper - annuaire ldap "global"

L'annuaire appelé "ldap global" est l'annuaire LDAP de référence, utilisé pour l'authentification et les recherche de personnes

La synchronisation se fait avec l'outil "PSP"

Les groupes sont placés dans une branche "ou=groupes", à plat (pas d'arborescence)

Les appartenances indirectes sont traduites en appartenances individuelles directes

La synchronisation met aussi à jour l'attribut "memberOf" de chaque personne

La configuration de PSP est conforme à psp-example-grouper-to-openldap, a ceci-près:

- l'attribut eduMember n'est pas utilisé
- tous les groupes sont "à plat" (mode "flat" au lieu de "bushy"
- la source des membres est unique (c'est l'ou=people de l'annuaire lui-même)

Grouper - annuaire active directory

L'annuaire est utilisé pour la gestion des droits sur des espaces de stockage partagés, pour la configuration des postes de travail , pour la définition des autorisations aux imprimantes

La synchronisation se fait avec l'outil "PSP"

Les groupes sont placés dans une unité organisationnelle "ougroupes", en conservant l'arborescence de la base grouper

L'attribut sAMAccountName reçoit la valeur de l'id_path grouper avec conversion des ":" en "_" (obsolète)

La configuration PSP est tirée de psp-exemple-grouper-to-active-directory.

(Dé)provisionning

Règles de provisionning

Mise à jour de la base Grouper

- nouveau personnel : à la validation du compte Sesame + 1 h max
- nouvel étudiant : à la validation du Sesame + 1h max

Les groupes institutionnels sont tous des groupes grouperLoader branchés sur Harpege et Apogee : l'execution des requêtes SQL se fait toutes les heures

Mise à jour des annuaires

- globale tous les matins pour tous les groupes institutionnels et non-institutionnels : via PSP
- répercussion des modifications en journée : T+n minutes (n de 1 à 10) : se fait via grouperLoader

Règles de déprovisionning

Membres des groupes institutionnels

Ce sont les bases métier qui fixent les règles de déprovisionning. Les membres des groupes institutionnels des personnels et étudiants sont toujours à jour, grâce à grouperLoader.

Les cas de changement de situation peuvent être :

- pour les personnels:
  - changement de structure (mise à disposition, mutation interne)
  - changement de catégorie, fonction, etc ...
  - sortie de l'établissement (retraite, fin de contrat, etc …)
  - fermeture de la structure (laboratoire, service, …)
- pour les étudiants :
  - changement de composante
  - inscription à l'année universitaire N+1 les règles précises restent à écrire (les pratiques peuvent être différentes selon les unités d'enseignement)

Membres des groupes non-institutionnels

- Beaucoup de groupes non-institutionnels sont issus d'un base de données tierce (les correspondants téléphonie de Harpege, les contributeurs Nuxeo, les administrateurs Gestimmo, etc...) : ce sont surtout des rôles applicatifs. Ces groupes sont donc toujours à jour grâce à grouperLoader.
- Comme vu plus haut, le compte d'un utilisateur référencé comme membre de groupes non-institutionnels peut avoir expiré. Il apparaitra alors comme “**entité non trouvée**” dans Grouper. Il ne sera jamais versé comme membre de groupe dans les annuaires (sources des personnes), cependant un nettoyage régulier sera lancé pour supprimer leurs appartenances dans Grouper.

  ```
  ./bin/gsh.sh -usdu -source rennes1:ldap -delete
  ```

- Un script permettra de changer l'année des groupes étudiants référencés dans les groupes non-institutionnels (selon les règles de changement d'année plus haut) .
- Les membres ayant une durée d'appartenance variable: les membres obsolètes (date de fin atteinte) sont supprimés, les nouveaux membres (date de début atteinte) sont ajoutés automatiquement dans la base Grouper, puis dans les annuaires.

NB : les groupes non-institutionnels devenus vides sont conservés dans Grouper et donc dans les annuaires.