Gestion des droits et publication

⚠ Ce document à pour but de garder une trace de tout ce que nous comprenons de la la gestion des droits nuxeo.

Gestion des droits

Relation entre le droit Write et CanAskForPublishing sur une section

CanAskForPublishing correspond au droit "Je peux demander à publier dans une section". C'est ce droit qui est testé pour faire apparaître on non le bouton publier en face de chaque section de publication dans l'onglet publication.
CanAskForPublishing est inclus dans Read (ce point nous interpelle mais la question n'est pas là... pour le moment -cf. ci-dessous-).

Par contre, on se demandait comment se fait-il qu'un utilisateur qui a le droit write obtient aussi le droit CanAskForPublishing.

Le réponse :

- En fait, l'utilisateur n'a pas vraiment un droit Write sur une section. On voit dans l'interface d'admin de la section la possibilité de donner un droit d'"écriture" mais le droit réellement attribué est le droit ReadWrite et ce dernier inclus Read et Write. Et comme Read inclus CanAskForPublishing alors CanAskForPublishing est positionné.

⚠ ESUP-ECM casse ce mode de fonctionnement par défaut et ne corrèle plus le droit CanAskForPublishing au droit Read (Merci aux collègues du Rectorat de Rennes pour leur aide à ce sujet)

CanAskForPublishing est inclus dans Read !

Je viens de faire des tests en nuxeo 5.2 de base (hors esup-ecm).

Voici ce que l'on observe :

- Un user a, par défaut, le droit read sur "sections" et ses fils.
- Comme CanAskForPublishing est inclus dans Read tout user peut donc demande à publier.

On peut donc se demander qu'elle est l'utilité de pouvoir positionner le droit CanAskForPublishing dans la mesure où il faut au minimum avoir le droit read pour pouvoir naviguer dans la section où l'on souhaite publier et que Read inclus CanAskForPublishing. Une question en ce sens a été posée à nuxeo.
Ce qui me semblerait mieux :

1. Le code qui affiche ou non le bouton "publier" est affiché si on a le droit CanAskForPublishing OU Write (et pas seulement CanAskForPublishing comme actuellement).
2. Le droit Read ne devrait pas inclure le droit CanAskForPublishing.

Voici ce qu'il faut faire en attendant si on veut pouvoir donner un droit de lecture sans possibilité de demander une publication :

1. Allow Read
2. Deny CanAskForPublish

Comment forcer des droits sur un document

Voici ce que l'on observe :

- Si un admin crée, dans workspace par exemple, un sous-workspace de nom tmp.
- Sur ce sous sous-workspace tmp il donne les droits de gestion au user toto.
- Maintenant toto se connecte, va sur tmp pour enlever les droits hérités.
- Si admin se reconnecte il ne pourra pas accéder à tmp afin de gérer les droits à nouveau !

Comment faire alors ?
La solution consiste à utiliser nxshell (Malheureusement, ce dernier n'a pas encore de fonction pour gérer les droits ! Par contre, il permet d'exécuter des scrips. Et des exemples de ces scripts existent chez nuxeo) :

1. Récupérer les scripts (cd /tmp ; hg clone http://hg.nuxeo.org/addons/nuxeo-shell-scripts)

2. Utiliser nxshell (cd <rep_nuxeo_52>/nuxeo-shell; ./nxshell.sh -h 127.0.0.1)
3. Dans nxshell faire :
    a. Se déplacer dans les dossiers (cd default-domain/workspaces)
    b. Retrouver UID du document (view tmp)
    c. Forcer les droits sur le document (script --file /tmp/nuxeo-shell-scripts/modifyPermissions.js 4cb62b5b-7b6e-432a-8d46-76271d125ea1)

Notes :

1. Le script modifyPermissions.js remets les droits hérités en place et donne le droit ReadWrite à menbers (passage de [toto:Everything:true, Everyone:Everything:false] à [members:ReadWrite:true])
2. Il me semble qu'il faudra que l'on livre ce script dans notre package. Peut-être en le modifiant afin de préciser à qui on donne le droit ReadWrite.
3. Documenter le tout.

Livré dans esup-ecm-1.0 : voir ici http://www.esup-portail.org/display/PROJESUPECM/FAQ

Comment fonctionne le paramètre defaultAdministratorId ?

DefaultAdministratorId est utilisé par le point d'extension userManager de UserService.
La valeur de DefaultAdministratorId est stockée dans UserManagerDescriptor.rootLogin puis dans UserManagerImpl.defaultRootLogin
Ensuite, à chaque clic, la méthode makePrincipal de UserManagerImpl est appelée. C'est elle qui va ajouter le groupe **administrators** au user courant si ce dernier correspond à DefaultAdministratorId. Ceci avec le code suivant :

```
// Create a default admin if needed
if (defaultRootLogin \!= null && defaultRootLogin.equals(principal.getName())) {
  virtualGroups.add(SecurityConstants.ADMINISTRATORS);
}
principal.setVirtualGroups(virtualGroups);
```

Pourquoi, par défaut, les utilisateurs ont le droit de lire les workspaces et les sections ?

Deux choses concourent à cela :

1) Comme déjà dit, ci-dessus, à chaque clic, la méthode **makePrincipal** de **UserManagerImpl** est appelée. Cette méthode positionne aussi les groupes auxquels appartient l'utilisateur courant. Elle ajoute à cette liste de groupe le groupe **defaultGroup** définit par le point d'extension **userManager** de **UserService**.
Ce **defaultGroup** est généralement défini à **members**.

2) La méthode **addRootACP** de **org.nuxeo.ecm.core.storage.sql.SessionImpl** est appelée à la première utilisation de nuxeo quand il faut créer l'arborescence initiale. Cette méthode positionne les droits suivants sur le node root :

SecurityConstants.EVERYTHING à **SecurityConstants.ADMINISTRATORS** (le groupe **administrators**)
SecurityConstants.EVERYTHING à **SecurityConstants.ADMINISTRATOR** (le user **administrator**)
SecurityConstants.READ à **SecurityConstants.MEMBERS** (le groupe **members**)
SecurityConstants.VERSION à **SecurityConstants.MEMBERS** (le groupe **members**)

Le node root est un document nuxeo qui est le père de **default-domain**. On ne le voit même pas dans l'interface JFS de nuxeo.

C'est le troisième de ces 4 droits qui explique pourquoi les utilisateurs ont le droit de lire les workspaces et les sections.

Notes :

- Je n'ai pas cherché à comprendre pourquoi on ne voit pas toujours, dans la zone « droit hérités » les 4 droits cités ci-dessus dans l'interface de gestion des droits de nuxeo.
- Je qualifierais le groupe membres de groupe par défaut choisi. Choisi, dans le sens où on le configure dans le point d'extension. Il n'en est pas de même pour SecurityConstants.EVERYONE (groupe everyOne) qui est un groupe qui est systématiquement ajouté à l'utilisateur courant via la méthode getPrincipalsToCheck de org.nuxeo.ecm.core.security.SecurityService. Ce groupe est un groupe par défaut interne à nuxeo. Il permet, par exemple, de garantir à nuxeo que si, applicativement, il est positionné un droit deny à everyone alors personne ne pourra passer.

Publication et demande de publication

Dans les choses que j'ai pu vérifier pour le moment:
--> on peut publier plusieurs versions dans différentes sections. Le modérateur pourra publier/refuser séparément les différentes demandes sans conflit entre elles
--> pour définir un nouveau modérateur d'une section, je n'ai pas encore la réponse étant donnés les problèmes de droit non résolus.
Ce qui est sûr est que les droits "Gérer tout" et "Ecriture" permettent tous les deux de voir les documents en attente de publication dans les sections. A voir une fois la solution trouvée pour les problèmes de droits si "Gérer tout" et "Ecriture" permettent tous les deux la modération.
--> si un user est ajouté comme admin d'une section alors que des documents sont déjà en attente de publication, celui-ci ne pourra pas les modérer. A priori les modérateurs sont désignés au moment où la demande de publication est faite