Droits « classiques » (non étendus)

Lecture et analyse des droits

Rappel : en faisant un `ls -l` en console, on obtient trois types distincts :

- ➔ les **fichiers** qui commencent par `-`
- ➔ les **dossiers** qui commencent par `d`
- ➔ les **liens symboliques** qui commencent par `l`

Exemples (sorties simplifiées) :

```
$ ls -l /home
drwx------ toto users /home/toto
$ ls -l /etc/password
-rw-r--r-- root root /etc/password
$ ls -l /vmlinuz
lrwxrwxrwx root root vmlinuz -> boot/vmlinuz-4.15.0-66-generic
```

Les liens symboliques sont des fichiers « virtuels », pointant sur des fichiers réels.

Pour information, il existe aussi des liens « durs » (hard links), qui consistent à faire pointer les fichiers identiques vers un même fichier physique, ce qui permet d'économiser beaucoup de place (notamment quand on ne sait pas ranger ses photos, et qu'on recopie 30x la même photo sur son disque dur, par exemple...).

Notre première lettre est toujours suivie de **trois triplets** et de **deux noms** :

- ➔ le nom du **propriétaire** du fichier, auquel s'applique le **premier triplet**,
- ➔ le **groupe** du fichier, auquel s'applique le **second triplet**,
- ➔ le troisième triplet étant les permissions pour le **reste du monde** (comprenez les permissions par défaut).

Rappel : un utilisateur peut faire partie de plusieurs groupes, mais possède toujours **un groupe par défaut**, donné sous forme de numéro dans **/etc/password**, ce numéro étant lié au fichier **/etc/group**, qui contient la liste des groupes standards.

Les triplets sont composés des caractères suivants :

- ➔ `r` : lecture autorisée

➔ w : écriture autorisée

➔ x : exécution pour un fichier / accès pour un dossier

➔ - : si l'autorisation respective est refusée

Par « exécution pour un fichier », on entend que le fichier est en fait un mini-programme écrit sous forme de texte, que l'on nomme un **script**. Les scripts sont écrits par les administrateurs dans différents langages de programmation (script shell, script Python, script PHP, etc).

Exemples (sorties simplifiées) :

Rappel : le dossier **/home** contient les **dossiers des utilisateurs** (l'équivalent de *C:\Users* ou *C:\Documents and settings* sous Windows).

```
$ ls -l /home
drwx------ toto users /home/toto
```

Ici, nous avons un seul dossier utilisateur nommé **toto**, qui appartient à l'usager **toto** et au groupe **users**.

Le propriétaire **toto** a tous les droits sur son dossier personnel, les autres groupes et utilisateurs du système ne pouvant accéder à son dossier (sauf bien entendu l'administrateur **root**, qui a tous les droits par défaut sur les systèmes GNU/Linux et Unices).

```
$ ls -l /etc/password
-rw-r--r-- root root /etc/password
```

Le fichier **/etc/password** est un fichier clé du système puisqu'il contient la liste des utilisateurs et notamment leurs prénoms, noms, droits d'accès, et diverses autres informations, à l'exception des mots de passe, codés dans **/etc/shadow**.

Ici le super-utilisateur **root** a bien entendu les droits de lecture/écriture sur le fichier. Le groupe **root** et les autres utilisateurs sont retreints en lecture seule, et ne peuvent donc ni modifier, ni effacer le fichier.

```
$ ls -l /vmlinuz
lrwxrwxrwx root root vmlinuz -> boot/vmlinuz-4.15.0-66-generic
```

Le fichier **/vmlinuz** est le noyau de démarrage du système, indispensable à son fonctionnement. Ici, le fichier virtuel pointe vers le noyau réel, enregistré dans le dossier **/boot**.

À priori, l'analyse des droits semble démontrer que n'importe quel utilisateur peut effacer ce lien symbolique, mettant en danger le démarrage du système.

En réalité, il n'en est rien. Pour pouvoir effacer ce lien symbolique, il faudrait que l'utilisateur ait le droit d'écriture sur **/** (la racine du système), or **seul root possède ce droit par défaut**.

Les droits en binaire

Une autre manière d'écrire les droits est d'utiliser le binaire avec les valeurs suivantes :

- ➔ **4** : **lecture** autorisée
- ➔ **2** : **écriture** autorisée
- ➔ **1** : **exécution** autorisée pour un fichier / accès autorisé pour un dossier
- ➔ **0** : **aucun droit**

Il suffit alors d'additionner les chiffres pour obtenir le total correspondant pour un triplet.

Le plus souvent, on rencontre les numéros suivants :

```
7 = rwx (lecture/écriture/exécution pour un dossier /
         lecture/écriture/accès pour un dossier)
6 = rw- (lecture/écriture)
5 = r-x (exécution pour un fichier / accès pour un dossier)
4 = r-- (lecture seule)
0 = --- (aucun droit)
```

Avec nos dossiers précédents :

```
$ ls -l /home
drwx------ toto users /home/toto
# /home est donc en 700
$ ls -l /etc/password
-rw-r--r-- root root /etc/password
# /etc/password est donc en 644
$ ls -l /vmlinuz
lrwxrwxrwx root root vmlinuz -> boot/vmlinuz-4.15.0-66-generic
# /vmlinuz est donc en 777
```

Changer les droits / propriétaires / groupes d'un fichier/dossier

Les trois commandes à retenir sont :

➔ **chmod** pour manipuler les **droits d'accès**

➔ **chown** pour changer le **propriétaire**

➔ **chgrp** pour changer le **groupe**

Comme toujours, **man nom_commande** vous donnera l'ensemble des possibilités de ces trois commandes !

Pour simplifier, dans la pratique, on utilise le plus souvent :

```
chmod 755_ou_750_ou_700 mon_dossier
chmod 644_ou_640_ou_600 mon_fichier
chown nom_utilisateur:nom_groupe fichiers_ou_dossiers
chgrp nom_groupe fichiers_ou_dossiers
```

Et quand on veut propager le changement à tous les dossiers/fichiers enfants, on utilise l'option **-R** pour faire le changement en récursif.

Conclusion

Les droits « classiques » sous GNU/Linux sont assez simples à manipuler en ligne de commande. D'autres syntaxes sont possibles mais n'ont pas été étudiées ici, parce que j'estime que **la notation binaire est finalement la plus simple et la plus rapide à l'usage**.

Nous n'avons pas non plus étudié les cas particuliers que sont les bits **setuid** (exécuter en tant que propriétaire du fichier – on pensera en particulier à la commande **passwd** qui doit accéder à **/etc/password** et **/etc/shadow**), **setgid** (exécuter en tant groupe du fichier / propager le groupe dans un dossier) et le **stycky bit** (garder en mémoire un fichier pour le relancer plus rapidement / n'autoriser que le propriétaire à modifier/supprimer les fichiers qui lui appartiennent, matérialisé par la lettre **t** du dossier **/tmp**).

À ces droits « classiques » se greffent encore des droits étendus, via les listes de contrôle d'accès, ou **ACL**, lesquelles se manipulent via les commandes **setfacl** et **getfacl**, et permettent notamment d'étendre les droits sur plusieurs usagers et groupes, comme sous NTFS par exemple.

Ce mécanisme d'ACL étendues devient obligatoire quand on veut créer un **contrôleur de domaine Active Directory** sous GNU/Linux, via le paquet **samba**, pour fixer les bonnes permissions sur les dossiers partagés.