Linux access control list (ACL)

Les droits sur un fichier/répertoire permettent de restreindre les accès à un fichier/répertoire, suivant un certain nombre de paramètres.
Les trois principaux droits sur des fichiers/répertoires sont:

- la lecture ('r')
- l'écriture ('w')
- l'exécution ('x')

Ces différents droits sont associées à trois types d'utilisateurs :

- Le propriétaire du fichier ('u')
- les utilisateurs appartenant au groupe auquel appartient le fichier/répertoire ('g')
- les autres, qui correspond à tous le monde ('o')

C'est une des bases de la sécurité informatique.
De nombreuses combinaisons sont possibles.
On peut ainsi permettre aux autres utilisateurs la lecture et l'exécution d'un fichier/répertoire, mais pas sa modification, etc.
Ce système a une limite, il ne permet pas de réglages fins.
Il est par exemple impossible de donner les mêmes droits pour partager des fichiers entre plusieurs utilisateurs ou groupes d'utilisateurs, tout en les gardant confidentiels face aux autres.
Pour pallier ces lacunes a été conçu le système des ACL.
un système permettant de faire une gestion plus fine des droits d'accès aux fichiers que ne le permet la méthode employée par les systèmes UNIX/LINUX.

Activation

Les ACL ne sont pas nativement activées sur Linux mais le noyau les prend en charge.
Pour savoir si le noyau Linux prend en charge les ACL :

```
grep ACL /boot/config-$(uname -r)
```

Voici, le résultat :

```
CONFIG_EXT4_FS_POSIX_ACL=y
CONFIG_REISERFS_FS_POSIX_ACL=y
CONFIG_JFS_POSIX_ACL=y
CONFIG_XFS_POSIX_ACL=y
CONFIG_BTRFS_FS_POSIX_ACL=y
CONFIG_FS_POSIX_ACL=y
CONFIG_TMPFS_POSIX_ACL=y
# CONFIG_HFSPLUS_FS_POSIX_ACL is not set
CONFIG_JFFS2_FS_POSIX_ACL=y
CONFIG_F2FS_FS_POSIX_ACL=y
```

```
CONFIG_NFS_V3_ACL=y
CONFIG_NFSD_V2_ACL=y
CONFIG_NFSD_V3_ACL=y
CONFIG_NFS_ACL_SUPPORT=m
CONFIG_CEPH_FS_POSIX_ACL=y
CONFIG_CIFS_ACL=y
CONFIG_9P_FS_POSIX_ACL=y
```

Elle doit retourner y pour le type système de fichier qui vous intéresse.
Le package `acl` doit être déjà installé, si non:

```
apt-get install acl
```

Pour activé sur une partition du système de fichier, modifier votre fichier `fstab` comme cela :

```
vim /etc/fstab
```

```
/dev/mapper/VG0-racine /               ext4    errors=remount-ro,acl 0
1
# /boot was on /dev/sda1 during installation
UUID=57017155-d648-4379-b705-7375d85dfc61 /boot           ext4    defaults
0       2
/dev/mapper/VG0-share /share          ext4
defaults,acl,user_xattr,barrier=1        0       2
/dev/mapper/VG0-swap none            swap    sw              0       0
```

Puis remonter le système de fichier :

```
 mount -o remount /
```

Utilisation :

L'attribution des droits se fait grâce à la commande `setfacl`, la lecture des droits avec `getfacl`
Ainsi les deux commandes suivantes sont équivalentes :

```
chmod u=rw fichier
setfacl -m u::rw fichier
```

Un fichier dont les ACL auront été spécifiés verra s'ajouter un + à la fin de la liste des droits
Voici, un exemple :

```
drwxrwx---+  4 ksh ksh      4096 févr. 17 13:40 test
drwxrwx---+  2 ksh ksh      4096 févr. 23 08:05 test2
drwxrwx---+ 17 ksh ksh      4096 févr.  9 15:43 test3
drwxrwx---   2 ksh ksh     12288 févr. 22 07:51 test4
```

Utilisation de ''getfacl'' :

getfacl permet d'afficher l'ensemble des permissions définies
Voici, un exemple

```
getfacl test/
```

```
# file: test/
# owner: ksh
# group: ksh
user::rwx
user:toto:rwx
group::---
mask::rwx
other::---
default:user::rwx
default:user:toto:rwx
default:group::---
default:mask::rwx
default:other::---
```

Ici on peut voir que le propriétaire du répertoire `test` (ksh) a les droits `rwx` et le login toto `rwx`, les autres utilisateurs n'ont aucun droit.

getfacl -d ... spécifie des acl par défaut, qui ne peuvent s'appliquer qu'aux répertoires. Voici, un exemple :

```
getfacl -d test/
```

```
# file: test/
# owner: ksh
# group: ksh
user::rwx
user:toto:rwx
group::---
mask::rwx
other::---
```

Les permissions effectives sont affichées individuellement pour les users ou groups, Cela apparaît quand le masque est trop restrictif.
Voici, un exemple :

```
getfacl test/
```

```
# file: test/
# owner: ksh
# group: ksh
user::rwx
user:toto:rwx               #effective:r-x
group::---
```

```
mask::r-x
other::---
default:user::rwx
default:user:toto:rwx       #effective:r-x
default:group::---
default:mask::r-x
default:other::---
```

Utilisation de ''setfacl'' :

setfacl permet d'ajouter des permissions
Voici, exemple :

```
setfacl -m u:toto:rwx test/
```

Ici, on ajoute les permissions de lire, écrire et d'exécution sur le répertoire `test` au login `toto`.

On peut cumuler les login et groupe comme cela :

```
setfacl -m u:toto:rwx,u:titi:r-x,g:tata:r-x test/
```

On peut aussi le faire de façon récursive pour l'ensemble d'un répertoire :

```
setfacl -R -m u:toto:rwx,u:titi:r-x,g:tata:r-x test/
```

On peut définir les permissions par défaut :

```
setfacl -R -m u:toto:rwx,d:u:titi:rwx test/
```

Avec cette commande nous avons ajouté les permissions de lire, écrire et d'exécution sur le répertoire `test` au login `toto` et tous les répertoires/fichiers en dessous, mais aussi ajouté les même permission par défaut.

Définir le masque :

```
setfacl -m m::rwx test/
```

Définir les permissions pour le groupe :

```
setfacl -m g::rwx test/
```

Définir les permissions pour le propriétaire :

```
setfacl -m u::rwx test/
```

Définir les permission pour les autres login :

```
setfacl -m o::rwx test/
```

Définir les permissions par défaut :

```
setfacl -m d:u::rwx,g::rwx,o::rwx test/
```

d'ailleurs cette commande équivaut à cela

```
chmod 777 test/
```