Une **ACL**, ou *Access Control List* (en anglais : « liste de contrôle d'accès ») est, pour définir simplement la notion, une liste de **permissions** sur un fichier, un répertoire ou une arborescence, ajoutée aux permissions « classiques » (c'est-à-dire, techniquement, les permissions POSIX.1) de ce fichier. Ces permissions concernent des utilisateurs et/ou des groupes définis. La gestion des ACL sous GNU/Linux s'inspire de la norme POSIX 1003.1e (projet 17) mais ne la respecte pas entièrement.

Au moyen des ACL, on peut étendre le nombre d'utilisateurs et de groupes ayant des droits sur un même fichier. Rappelons que, dans le monde UNIX, chaque fichier ne peut normalement indiquer des permissions que pour *un seul utilisateur* et *un seul groupe*, qui s'opposent à une unique catégorie correspondant à « tous les autres » (ou « le reste du monde »). Avec les ACL, on peut (entre autres) ajouter à un fichier d'autres utilisateurs et groupes et définir leurs droits séparément. On se rapproche ainsi du système de permissions pratiqué sur les plate-formes NT (de nombreuses différences subsistent, cependant).

Les ACL sont très utiles (voire indispensables) dans des environnements informatiques axés sur le travail collaboratif et mutualisé ; de même, leur utilisation avec **SAMBA** permet d'en étendre les capacités.

> Dans les exemples qui suivent, les commandes peuvent être tapées en root ou en utilisateur normal, selon la portée des modifications : pour modifier les droits d'un fichier qui n'est pas possédé par un utilisateur ou pour ajouter des permissions à un autre utilisateur ou encore pour ajouter aux ACL un groupe dont l'utilisateur ne fait pas partie, il faut être root (pour les utilisateurs Ubuntu, il faut préfixer les commandes par **sudo**).

Sommaire

Exemple pratique

Soit un fichier */var/www/index.php* (page d'index d'un site web, par exemple) dont les droits sont les suivants :

```
ls -l /var/www/index.php
  -rw-r-----  1 root www-data 18 2005-09-11 11:24 /var/www/index.php
```

En d'autres termes, root en est le propriétaire ; il peut le lire et le modifier ; le fichier est aussi accordé au groupe *www-data* (celui sous lequel, par exemple, tourne le **serveur web**), dont les membres peuvent le lire mais pas le modifier. Quant au reste du monde, il ne peut pas y accéder (le fichier contient des informations confidentielles telles qu'un mot de passe à une base de données **MySQL**).

Imaginons qu'on veuille rendre le fichier accessible en lecture aux utilisateurs jean et luce, en lecture et écriture à khadija et alice. On pourrait à la rigueur faire entrer jean et luce dans le groupe *www-data* mais cela introduirait une faille de sécurité car *www-data* peut aussi accéder à des données qui ne les concernent pas. Il n'est en tout cas rationnellement pas prudent d'ajouter khadija et alice au groupe *root*. On ne peut non plus changer les permissions (les mettre en lecture et écriture pour tout le monde) ou modifier le propriétaire et le groupe.

Les ACL sont là une solution pratique et facile à gérer dans ce cas ; il suffit d'ajouter des permissions à l'ACL du fichier (grâce à des commandes décrites plus bas) pour qu'il se présente ainsi :

```
root: rw-
www-data: r--
khadija: rw-
alice: rw-
jean: r--
luce: r--
reste du monde : ---
```

Mise en place

Selon la version du noyau utilisée, le support des ACL peut ou non être déjà compilé. S'il l'est, il faut aussi que la partition contenant les fichiers dont on veut étendre les droits soit montée avec l'option idoine.

Noyau

Il faut d'abord savoir si le noyau a été compilé avec le support des ACL. Le plus simple est de le vérifier dans le fichier de configuration du noyau, fichier normalement situé sous */boot*. Pour ce faire, utiliser la **commande *grep*** :

```
grep ACL /boot/config-version-du-noyau
```

Elle doit renvoyer la ligne suivante :

```
CONFIG_FS_POSIX_ACL=y
```

pour signaler que le support général des ACL est présent, et plusieurs lignes du type

```
CONFIG_SystèmeDeFichiers_FS_POSIX_ACL=y
```

où ***SystèmeDeFichiers*** peut recevoir les valeurs (pour un noyau 2.6.8-2-386 à la date de rédaction) *EXT2*, *EXT3*, *JFS* et *XFS*. On peut aussi utiliser les ACL avec les systèmes de fichiers *IBM JFS*, *ReiserFS*, *SGI XFS* et *NFS*. Leur implémentation peut nécessiter de patcher le noyau. Noter que les ACL ne sont pas possibles avec des systèmes de fichiers comme *vfat* qui ne gèrent aucun type de permissions.

Si la valeur des options n'est pas correcte, vous devez **recompiler votre noyau**. N'oubliez pas de prévoir au moins un système de fichiers pour lequel les ACL seront permis.

Systèmes de fichiers/montage des partitions

Quand le noyau est disposé à gérer les ACL, on doit préparer les partitions montées dans un système de fichiers adapté (par exemple, il est exclu de vouloir utiliser ces permissions avec du *vfat*).

Montage et démontage à la volée

Il faut monter les partitions voulues avec l'option *acl*. Par exemple :

```
mount -t ext3 -o defaults,acl /dev/hda2/ /var/www/
```

Si la partition est déjà montée, on peut modifier ces paramètres à la volée :

```
mount -o remount,acl /var/www/
```

Configuration automatique

L'inscription dans ***/etc/fstab*** des options de gestion des ACL est recommandée quand leur utilisation est régulière. Par exemple, notre même couple *partition* / *point de montage* serait déclaré ainsi :

```
/dev/hda2       /var/www            ext3    defaults,acl 0       0
```

À chaque montage automatique des partitions, le support des ACL sera activé.

Commandes

Il existe deux commandes essentielles : l'une pour manipuler l'ACL d'un fichier (*setfacl*) et l'autre pour la consulter (*getfacl*). Les commandes traditionnelles *chmod* et *chown* ne peuvent accéder aux ACL.

Ces deux commandes nécessitent, sous Debian (et distributions dérivées, comme Knoppix ou Ubuntu), l'installation du paquetage « acl ». Pour l'installer :

```
[sudo] apt-get install acl
```

(ajout de `sudo` pour Ubuntu)

Pour les distributions à base de RedHat (donc aussi Fedora, Mandriva), il faut installer les paquetages `acl.*.rpm` et `libacl1.*.rpm` (leur nom contient leur numéro de version).

setfacl

Le nom de la commande se comprend *set file's ACL* (« régler l'ACL du fichier »). Elle possède de nombreuses options dont il convient de prendre connaissance en consultant la page de manuel (`man setfacl`). La commande fonctionne bien sûr aussi de manière récursive (option `-R`) :

```
setfacl -Rm u:khadija:rw /var/www/
```

modifie l'ACL de tous les fichiers situés sous */var/www/* en attribuant une permission de lecture et d'écriture à l'utilisateur khadija.

Ajouter des permissions

La syntaxe fondamentale est simple. La commande `setfacl -m u:khadija:rw /var/www/index.php` modifiera (`-m`) l'ACL de */var/www/index.php* en attribuant à l'utilisateur (préfixe `u:`) khadija les droits `rw` et en lui refusant le droit d'exécution (qui n'a pas été mentionné dans la commande).

Les principaux paramètres à connaître sont :

- **préfixes :**
  - `u:` (droits pour un **u**tilisateur, nommé ou désigné par son uid) ;
  - `g:` (droits pour un **g**roupe, nommé ou désigné par son gid) ;
  - `o:` (droits pour ***o**ther*, le reste du monde) ;
- **permissions :** elles sont codées dans l'ordre `r`, `w` et `x` ou `X` (ce dernier représentant, comme avec *chmod*, le droit d'entrée dans les répertoires ou celui d'exécution pour les fichiers qui ont déjà un marqueur `x`). On les remplace par `-` pour une interdiction explicite. Ne pas mentionner un droit revient aussi à une interdiction : `setfacl -m u:khadija:w /var/www/index.php` et `setfacl -m u:khadija:-w- /var/www/index.php` reviennent au même.

On peut construire des commandes plus complexes en enchaînant les entrées dans l'ACL :

```
setfacl -m u:khadija:rw,g:site1:r--,o:--- /var/www/index.php
```

définit des permissions dans l'ACL de */var/www/index.php* pour l'utilisateur khadija, le groupe *site1* et le reste du monde.

Cette commande permet aussi de modifier les permissions classiques (et remplace dans ce cas *chmod*) : l'utilisateur, le groupe et le reste du monde initiaux du fichier sont simplement désignés par le préfixe (`u:`, `g:`, `o:`) suivi d'un nom vide : si un fichier *index.php* appartient à luce:*www-data* avec les droits `r--r-----`, pour donner à l'utilisateur et le groupe les droits en lecture et écriture il suffit d'une commande `setfacl -m u::rw,g::rw /var/www/index.php`. Si l'utilisateur et le groupe possèdent déjà un droit qui ne serait pas mentionné dans la commande *setfacl*, ce droit sera annulé. Soit le fichier *index.php* avec les droits `rw-r-----` pour luce:*www-data*. La commande `setfacl -m u::r,g::x index.php` modifiera les droits à `r----x---` pour pour luce:*www-data*.

Noter qu'un fichier dont seules les permissions classiques ont été altérées par *setfacl* au lieu de *chmod* ne reçoit pas pour autant une ACL. De fait, il n'est pas référencé par `ls -l` comme fichier à ACL (marqueur `+` voir plus bas).

Droits par défaut et héritage des droits étendus

Les droits étendus d'un objet parent ne sont pas automatiquement hérités par les objets contenus. Par exemple, si un répertoire (root:*www-data*, `rwxr-x-r-x`) possède une ACL `u:luce:rwx`, un fichier créé à l'intérieur (ou déjà présent avant l'adjonction de l'ACL) ne reçoit pas cette ACL et ses droits sont ceux impliqués par l'***umask*** défini.

On peut modifier ce comportement en ajoutant, **aux répertoires seulement**, un attribut *default*, codé `d:`, qui se transmet à tous les fichiers créés dans le répertoire après l'ajout de l'ACL par défaut. Par exemple, `setfacl -m d:u:luce:rwX /var/www` donne à luce les droits de lecture et écriture (ainsi qu'« exécution » quand il s'agit de répertoires) pour tous les fichiers qui seront créés sous */var/www* à partir de ce moment, jusqu'à ce que cette ACL « par défaut » soit annulé ou remplacé.

Retirer des permissions

Pour annuler tout ou partie d'une ACL :

```
setfacl -b /var/www/index.php
```

ôte tout le contenu de l'ACL du fichier, tandis que

```
setfacl -x u:khadija,g:site1 /var/www/index.php
```

retire les permissions propres à khadija et au groupe *site1*.

Les permissions ACL par défaut d'un répertoire (`d:`) s'annulent par `setfacl -k`.

Le masque

Le masque est une synthèse des valeurs les plus permissives que possède un fichier doté d'une ACL. Les droits de l'utilisateur fondamental ne sont cependant pas pris en compte. Le masque est calculé automatiquement :

```
chown luce:www-data index.php
chmod 640 index.php
ls -l index.php
   -rw-r-----  1 luce www-data 5055 2005-10-16 18:53 index.php
getfacl index.php
   # file: index.php
   # owner: luce
   # group: www-data
   user::rw-
   group::r--
   other::---
```

Ce fichier n'a pas d'ACL donc pas de masque.

```
setfacl -m u:jean:rw,g:web:rw index.php
getfacl index.php
   # file: index.php
   # owner: luce
   # group: www-data
   user::rw-
   user:jean:rw-
   group::r--
   group:web:rw-
   mask::rw-
   other::---
```

Maintenant que le fichier possède une ACL, il a reçu un masque : les permissions les plus élevées (utilisateur exclu) étant `rw`, c'est aussi la valeur du masque.

L'intérêt du masque est de pouvoir limiter d'un coup toutes les permissions d'un fichier (étendues ou non), sauf celles du propriétaire ; on utilise pour cela le préfixe `m:` suivi du droit maximal à accorder :

```
getfacl index.php
    # file: index.php
    # owner: luce
    # group: www-data
    user::rw-
    user:jean:rw-
    group::r--
    group:web:rw-
    mask::rw-
    other::---
setfacl -m m:r index.php
getfacl index.php
    # file: index.php
    # owner: luce
    # group: www-data
    user::rw-
    user:jean:rw-               #effective:r--
    group::r--
    group:web:rw-               #effective:r--
    mask::r--
    other::---
```

Les valeurs modifiées sont indiquées par le commentaire « *effective:* » suivi des permissions effectives après l'application du masque (ici, jean et *web* n'ont plus que le droit `r`, la situation reste la même pour *www-data*).

> L'existence même d'un masque renvoie au fonctionnement profond des ACL. Pour en comprendre l'utilité réelle sans se limiter à l'application pragmatique qui en est donnée ici, on se reportera à ***POSIX Access Control Lists on Linux***.

getfacl

Cette commande suivie d'un nom de fichier affiche l'ACL de ce fichier (*get file's ACL* « récupérer l'ACL du fichier »). Par exemple :

```
getfacl /var/www
  # file: var/www
  # owner: root
  # group: www-data
  user::rwx
  user:luce:rwx
  group::rwx
  mask::rwx
  other::r-x
  default:user::rwx
  default:user:khadija:rwx
  default:group::rwx
  default:group:www-data:r-x
  default:mask::rwx
  default:other::r-x
```

On voit qu'outre les droits traditionnels attribués à root:*www-data* (droits indiqués après `user::` et `group::`) sont aussi définis :

- des droits complets pour luce (`user:luce:rwx`) ;
- une permission ACL par défaut donnant des droits complets à khadija sur tous les nouveaux fichiers créés sous */var/www/* (`default:user:khadija:rwx`) ;
- une autre permission ACL par défaut donnant des droits de lecture et d'exécution au groupe *www-data* sur les mêmes fichiers (`default:group:www-data:r-x`).

Noter que `user::`, `group::` et `other::` représentent le triplet *utilisateur* / *groupe* / *reste du monde* des permissions classiques. Appliquer cette commande sur un fichier qui n'a pas d'ACL définie donne les mêmes informations que `ls -l`, dans un format différent :

```
setfacl -b index.php # retirer les ACL pouvant exister
ls -l index.php
  -rw-r-----  1 root www-data 5055 2005-10-16 18:53 index.php
getfacl index.php
  # file: index.php
  # owner: root
  # group: www-data
  user::rw-
  group::r--
  other::---
```

ls, *cp* et *mv*

Ces commandes doivent pouvoir lister, copier et déplacer les ACL en même temps que les fichiers. Pour les deux premières commandes, il faut préciser explicitement que l'on veut afficher/conserver les droits (ce qui est aussi le cas quand on ne travaille que sur les droits classiques) : `ls -l`, `cp -a`. La commande *mv*, quant à elle, préserve toujours les droits.

Quand les droits étendus ne peuvent être conservés (déplacement ou copie vers un système de fichier qui n'est pas configuré pour les recevoir ou utilisation d'une version de *cp* trop ancienne), un message d'avertissement en informe l'utilisateur. Par exemple :

```
setfacl -m u:luce:rw index.php
cp -a index.php /mnt/vfat
  cp: preserving permissions for `/mnt/vfat/index.php': Opération non supportée
```

Noter qu'un fichier comportant une ACL qu'on veut lister par `ls -l` n'affiche qu'un `+` à la suite de ses permissions. Seule la commande *getfacl*, pour l'instant, permet d'avoir connaissance du détail. Par exemple :

```
setfacl -m u:khadija:rw /var/www/index.php
ls -l /var/www/index.php
  -rw-rw----+ 1 khadija www-data 5055 2005-10-16 18:53 /var/www/index.php
```

Avec `-rw-rw----+`, on sait que le fichier possède une ACL (`+`), sans en connaître les constituants.

Interaction avec KDE

Le gestionnaire de bureau **KDE** permet, depuis la version 3.5, de visualiser et manipuler les ACL des fichiers. Il suffit de cliquer avec le bouton droit de la souris sur un fichier et de sélectionner l'onglet « Droits d'accès » puis « Droits d'accès avancés ». De là, on peut consulter l'ACL du fichier de manière graphique.

Sauvegarde des données

Sauvegarder des données dotées d'ACL nécessite :

- l'utilisation d'un système de fichiers pour le stockage qui soit compatible ;
- et l'utilisation d'un logiciel de sauvegarde qui soit tout autant compatible.

  À titre indicatif, *tar* et *cpio* et *rsync* ne le sont pas (à moins d'être patchés), *star* et *pax* le sont.

Pour contourner le problème de sauvegarde, il est possible d'écrire toutes les ACL dans un fichier qui servira de base à une restauration ultérieure : `getfacl --skip-base -R /dossier/dossier/ > fichier` récupère les informations récursivement et les inscrit dans un simple fichier. La restauration se fait au moyen de `setfacl --restore=fichier`. Il faut, pour qu'elle fonctionne, se placer à la racine contenant l'arborescence, en raison de la notation relative des chemins (d'où le message `Removing leading '/' from absolute path names` que l'on peut souvent lire en tapant des commandes avec ces programmes). Le chemin d'un répertoire */tmp/test* est enregistré comme *tmp/test* : on doit donc, pour restaurer, lancer la commande depuis la racine de */tmp*, c'est-à-dire */*.

Par exemple : le répertoire */tmp/test* contient trois fichiers à ACL. On sauvegarde les ACL avec `getfacl --skip-base -R /tmp/test > acl.acl`. Pour restaurer, on se place à la racine (`cd /`) et on lance `setfacl --restore=acl.acl`. Si on avait lancé la commande depuis */test*, *setfacl* aurait renvoyé les erreurs :

```
setfacl: tmp/test: Aucun fichier ou répertoire de ce type
setfacl: tmp/test/a: Aucun fichier ou répertoire de ce type
setfacl: tmp/test/b: Aucun fichier ou répertoire de ce type
setfacl: tmp/test/c: Aucun fichier ou répertoire de ce type
```

Note concernant la syntaxe de *setfacl*

Les préfixes abrégés peuvent être développés et les permissions codées en octal (avec préfixe *0* optionnel). Ces trois commandes ont donc le même sens :

```
setfacl -m d:u:luce:rw,g:www-data:r,o:- index.php
setfacl -m default:user:luce:6,group:www-data:4,other:0  index.php
setfacl -m default:user:luce:06,group:www-data:04,other:00  index.php
```

Tout au long de ce document, on a parlé de « permissions classiques » opposées à des « permissions étendues » qui leur seraient ajoutées. C'est une simplification : dans les faits, un système de fichiers monté avec le support des ACL change son appréhension globale des permissions, qui sont toutes des ACL, qu'elles soient minimales (utilisateur primaire, groupe primaire, reste du monde) ou étendues. Dans ce dernier cas, c'est la notion de groupe qui est redéfinie : tout utilisateur ou groupe ajouté à l'utilisateur primaire est enregistré avec ses droits dans une classe « groupe » étendue.

Cela explique la nécessité d'un masque : c'est la valeur limite réelle de la classe « groupe », que les entrées qu'elle contient ne peuvent dépasser. Un `ls -l` sur un fichier dont le masque ACL a été abaissé à `r--` alors que les droits du groupe primaire étaient auparavant `rw-` donnera : `-rw-r-----+` et non `-rw-rw----+`. Inversement, si l'on change par *chmod* les permissions du groupe principal, cela revient à changer le masque, donc les permissions de tous les utilisateurs et groupes ajoutés.

Documents annexes

- **Articles de Léa :**
  - les **permissions** « classiques » POSIX.1 ;
  - les **attributs étendus** des systèmes de fichiers *ext2* et *ext3* ;
- **pages de manuel :**
  - `man acl` ;
  - `man setfacl` ;
  - `man getfacl`.
- **sites externes :**
  - **Les ACL POSIX** ;
  - **Comment fonctionnent les ACL POSIX sous GNU/Linux** ;
  - **POSIX Access Control Lists on Linux** (anglais).

Copyright >>
