

Assurez-vous aussi que votre noyau ait les modules intégrés pour le support des ACL. Allez dans le répertoire `/usr/src/linux` et saisissez la commande:

make menuconfig

Pour le système de fichiers EXT3, assurez-vous que les modules suivants sont intégrés dans le noyau.

File systems -->

- Ext3 journalling file system support*
- Ext3 extended attributes*
- Ext3 POSIX Access Control Lists*
- Ext3 Security Labels*

Pour ceux qui avait choisi ReiserFS précédemment, assurez-vous à la place que les modules suivant sont intégrés dans le noyau.

File systems -->

- Reiserfs Support*
- Enable reiserfs debug mode (NEW)*
- Stats in /proc/fs/reiserfs (NEW)*
- ReiserFS extended attributes (NEW)*

Si votre noyau ne supporte pas les modules de votre système de fichier, renvoyer les procédures de compilation du noyau dans le *Module 2 – Unité 1*.

Maintenant, il faut que votre système prenne compte de cette nouvelle option, utilisez les commandes,

```
mount -o remount /  
mount -o remount /var
```

Ou simplement redémarrer votre ordinateur.

```
shutdown -r now
```

NOTES



Autorisations ACL(EA)

De base, dans les permissions POSIX, quatre entrées sont utilisées pour l'attribution de permissions: *owner*, *group owner*, *other* et *mask*. C'est ce que nous nommerons des entrées fixes.

Les permissions ACL(EA) pour ACL Extension Attributes, vous permettent d'ajouter des entrées supplémentaires sous les entrées de base utilisateur propriétaire (*owner*) et groupe propriétaire (*group owner*). Vous ne pouvez pas ajouter des entrées supplémentaires sous l'entrée autre (*other*).

Par exemple, pour un répertoire, les permissions de base pourraient ressembler à ceci:

```
drwxr-xr-x 2 imurphy users 4096      May 15 07:37 cybiolab
```

Nous pourrions vouloir ajouter des permissions pour des utilisateurs ou groupes supplémentaires. Ces nouvelles entrées se retrouveraient sous les entrées de base.

Owner: imurphy

Entrée#1: drobbins

Entrée#2: jallison

Group: users

Entrée#3: GentooUsers

Entrée#4: SambaUsers

Others



NOTE: Nous le redirons jamais assez, attribuer toujours vos permissions au niveau des groupes et jamais au niveau des utilisateurs.

NOTES



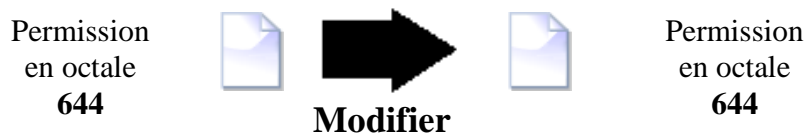
Imaginons la situation où nous avons les caractéristiques suivantes pour deux utilisateurs qui veulent travailler sur un même fichier localement sur un poste Linux.

Utilisateur:	drobbins
Groupe Principal:	GentooUsers
Groupe Suppl:	users

Utilisateur:	jallison
Groupe:	SambaUsers
Groupe Suppl:	users

Un fichier a été créé par l'utilisateur *drobbins*, par conséquent le fichier aura les permissions 644 par défaut. Bon, voici un premier problème, l'entrée "*autre*" a la permission lecture. Il est à vous de définir les permissions nécessaires pour ne pas avoir de gain d'autorisation.

L'utilisateur propriétaire sera *drobbins* et le "*groupe propriétaire*" sera *GentooUsers*, donc l'utilisateur *jallison* ne pourra pas modifier le fichier. Alors pourquoi ne pas mettre les deux utilisateurs dans un groupe commun tel que *users*? Voici le second problème. À la maison cela est utilisable, mais en entreprise mettre le groupe propriétaire à *users* pour tout le monde, ne permet plus de limiter deux groupes avec des besoins distincts de travailler sur une même portion de répertoire.



Conclusion l'héritage des permissions avec les permissions POSIX avec les ACL de base n'est pas fonctionnel. Devrions-nous dire pas fonctionnel ou mal utilisé? Voici une solution, qui est plus une façon d'utiliser la combinaison des permissions POSIX avec la capacité des extensions d'ACL qui va nous permettre d'appliquer correctement des permissions dans un environnement d'entreprise, et augmenter de façon considérable la flexibilité d'attribution de permissions.

NOTES



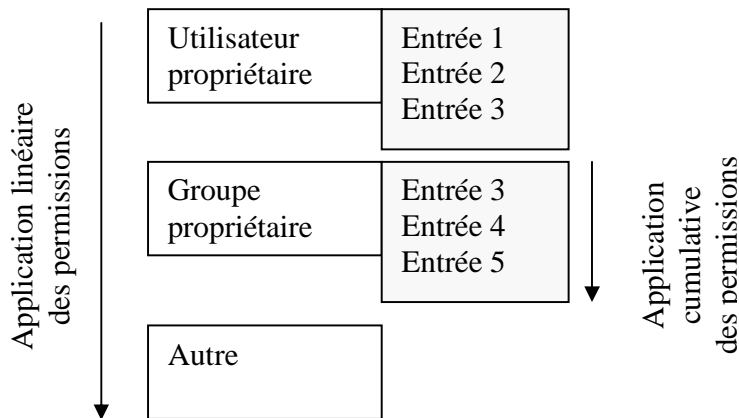
Ordre d'application des autorisations POSIX avec ACL(EA)

L'ordre d'application des autorisations POSIX avec ACL(EA) se font de la façon linéaire pour les permissions POSIX, mais deviennent cumulatives pour les ACL.

- Permissions POSIX
- ACL

L'application des permissions par le système se fera par les étapes suivantes :

1. Le système regarde si l'utilisateur est l'utilisateur propriétaire ou si une entrée sous l'entrée de base utilisateur propriétaire est correspondante.
2. Le système regarde si l'utilisateur fait parti du groupe propriétaire ou une ou plusieurs entrées sous l'entrée de base groupe propriétaire sont correspondantes.
3. Si aucune règle n'est correspondante, le système applique les permissions de l'entrée de base autre.



Pour mieux imaginer le concept, voici un exemple avec un fichier ayant les permissions suivantes,

```

Owner: root          rw-
Group: root          rw-
  Entrée#1: Gentoo   r--
  Entrée#2: Linux    -w-
Others               r-x
  
```

NOTES



Commande `getfacl`

La commande `getfacl` permet d'afficher les listes de contrôle d'accès des fichiers et des répertoires. Cette commande se rapproche un peu de la commande `ls` mais avec le support des ACL.

Voici le tableau des commutateurs les plus utiles avec la commande `getfacl`.

Commutateur	Description
<code>-d</code>	Affiche la liste de contrôle d'accès par défaut.
<code>-R, --recursive</code>	Affiche les ACL de tous les fichiers et répertoires de façon récursive.
<code>--all-effective</code>	Affiche le commentaire des permissions effectives, ce même si elles sont identiques aux permissions défini par les entrées ACL.

Syntaxe:

```
getfacl [options] objet
```

Voici plusieurs exemples d'utilisation de la commande `getfacl`:

Affiche les ACL d'un fichier en particulier.

```
getfacl <fichier>
```

Affiche les ACL de tous les objets d'un répertoire.

```
getfacl *
```

Affiche les ACL de tous les objets d'un répertoire de façon récursive.

```
getfacl -R *
```

Affiche les permissions effectives d'un répertoire.

```
getfacl --all-effective <répertoire>
```

NOTES



Le résultat que vous obtiendrez par la commande `getfacl` peut se lire en trois sections,

- La première section indique le nom de l'objet, l'utilisateur propriétaire ainsi que le groupe propriétaire de l'objet;
- La seconde section représente les permissions POSIX et les ACL qui permettent l'accès à l'objet;
- La troisième section représente les ACL par défaut qui permettent la propagation des permissions aux fichiers ou répertoires qui sont créé à l'intérieur de ce dernier.

Par exemple, voici les permissions d'un répertoire nommé `cybiolab`. Le premier encadré est le résultat de la commande `ls` et le second encadré est le résultat affiché par la commande `getfacl`.

```
ls  
drwxrws----+ 2 root  root 4096 Dec 15 23:23 cybiolab
```

Le résultat de la commande `ls` nous indique que:

- l'utilisateur propriétaire a les permissions POSIX en lecture, écriture et exécution.
- le groupe propriétaire a les permissions POSIX en lecture, écriture et exécution.
- la permission spéciale SGID est appliquée sur l'objet.
- les autres n'ont aucun accès.
- l'utilisateur propriétaire est l'utilisateur `root`
- le groupe propriétaire est le groupe `root`.

Remarquer le `+` à la fin des permissions. Cela indique que des ACL ont été appliqués sur le répertoire.

La commande `getfacl` pour sa part nous donne le résultat suivant sur le même répertoire.

```
getfacl cybiolab  
# file: cybiolab  
# owner: root  
# group: root
```

NOTES



```
user::rwx
user:drobbins:rwx
group::rwx
group: DomainUsers:rwx
mask::rwx
other::---
default:user::rwx
default:user:drobbins:rwx
default:group::rwx
default:group: DomainUsers:rwx
default:mask::rwx
default:other::---
```

Comme indiqué, le résultat obtenu par la commande `getfacl` peut se lire en trois sections,

```
# file: cybiolab
# owner: root
# group: root
```

Cette première section nous indique que:

- le nom de l'objet est `cybiolab`,
- l'utilisateur propriétaire des permissions POSIX est `root`,
- le groupe propriétaire des permissions POSIX est `root`.

```
user::rwx
user:drobbins:r-x
group::rwx
group: DomainUsers:rwx
mask::rwx
other::---
```

La seconde section nous affiche les permissions POSIX et les ACL définis sur l'objet même, soit que:

- l'utilisateur propriétaire a les permissions POSIX en lecture, écriture et exécution.
- L'utilisateur `drobbins` a un ACL qui lui donne les permissions en lecture et exécution.

NOTES



Le groupe propriétaire est root, qu'il a les permissions rwx sur le répertoire et que lorsqu'un utilisateur de ce groupe créera un objet dans ce répertoire ce dernier aura les permissions rw-rw---- pour les fichiers et rwxrwx--- sur les répertoires.

L'utilisateur drobbins a les permissions r-x sur le répertoire et que lorsqu'il créera un objet dans ce répertoire ce dernier aura les permissions rw-rw---- pour les fichiers et rwxrwx--- sur les répertoires.

Le groupe DomainUsers a les permissions rwx sur le répertoire et que lorsqu'un utilisateur de ce groupe créera un objet dans ce répertoire ce dernier aura les permissions rw-rw---- pour les fichiers et rwxrwx--- sur les répertoires.

Aucune permission n'a été attribuée aux autres.

Commande setfacl

La commande *setfacl* permet de définir les listes de contrôle d'accès des fichiers et des répertoires. Cette commande n'a pas les mêmes fonctions que la commande *chmod*, car elle ne peut pas définir de permissions POSIX, mais seulement les permissions dans les ACL.

Voici le tableau des commutateurs les plus utiles avec la commande *setfacl*.

Commutateur	Description
-m, --modify	Permet de modifier les ACL des fichiers ou des répertoires.
-d, --default	Permet de modifier les ACL par défaut des fichiers ou des répertoires.
-r, --recursive	Applique de façon récursive les ACLs sur tous les fichiers et répertoires.
-x, --remove	Permet de supprimer les ACL des fichiers ou des répertoires.
-b, --remove-all	Supprimer tout les entrées ACL de l'objet. Les permissions POSIX ne sont pas affecté par ce commutateur.
-k, --remove-default	Supprime les ACL par défaut (ceux définis avec le commutateur -d).

NOTES



Syntaxe:

```
setfacl [options] [action] [type:qui:permission] objet
```

Pour simplifier la compréhension, voici plusieurs exemples d'utilisation de la commande setfacl.

Définition des permissions en lecture, écriture et exécution pour un utilisateur sur un répertoire.

```
setfacl -m user:<utilisateur>:rwx <répertoire>  
setfacl -m u:<utilisateur>:rwx <répertoire>
```

Remarquer que le type de personne à qui s'applique l'ACL peut être abrégé par la première lettre, soit ugo.

Définition des permissions en lecture et écriture pour un utilisateur sur un fichier.

```
setfacl -m group:<groupe>:rw <répertoire>
```

Définition des permissions en lecture seule pour les autres sur un répertoire.

```
setfacl -m other:r-- <répertoire>
```

Suppression des permissions pour un utilisateur sur un fichier.

```
setfacl -x user:<utilisateur>: <fichier>
```

Définition des ACL par défaut en lecture, écriture et exécution à un utilisateur pour la propagation des permissions à l'intérieur d'un répertoire.

```
setfacl -d -m group:<groupe>:rwx <répertoire>
```

Suppression des ACL par défaut pour un utilisateur sur un répertoire.

```
setfacl -k user:<utilisateur>: <répertoire>
```

NOTES



