Table des matières

[ACL 2](#_Toc89321407)

[Commandes 2](#_Toc89321408)

[Commande setfacl 2](#_Toc89321409)

[Droits par défaut et héritage des droits étendus 3](#_Toc89321410)

[Retirait de permissions 3](#_Toc89321411)

[Masque 3](#_Toc89321412)

[Commande getfacl 5](#_Toc89321413)

[Commandes ls, cp et mv 6](#_Toc89321414)

# ACL

Les listes de contrôle d'accès (ACL) fournissent un **mécanisme d'autorisation supplémentaire plus flexible** pour les systèmes de fichiers.   
Il est conçu pour aider avec les **autorisations de fichiers Linux**.   
Les ACL permettent d'**accorder** des **autorisations pour n'importe quel utilisateur ou groupe** à **n'importe quelle ressource de disque**.

## 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, l'installation du paquet acl.   
Pour installer le paquet …  
**>>** sudo apt install acl (Debian)  
**>>** sudo zypper install acl (SUSE)

Pour les distributions à base de RHEL, il faut installer les paquets acl.\*.rpm et libacl1.\*.rpm (leur nom contient leur numéro de version).

### Commande setfacl

Le nom de la commande se comprend comme **set file's ACL**.  
Elle possède de **nombreuses options** dont il convient de prendre connaissance en consultant la page de manuel   
(man setfacl).   
La **commande fonctionne également de manière récursive** (**option -R**).  
**setfacl -Rm u:tux: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 tux**.

#### Ajout de permissions

La commande **setfacl -m u:tux:rw /var/www/index.php** modifiera (-m) l'ACL de /var/www/index.php en attribuant à l'utilisateur (préfixe u:) tux 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 utilisateur, nommé ou désigné par son uid) ;  
  g: (droits pour un groupe, nommé ou désigné par son gid) ;  
  o: (droits pour autres (other) ;
* **Autorisations**  
  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).   
  On les remplace par - pour une interdiction explicite.   
  Ne pas mentionner un droit revient aussi à une interdiction …  
  **setfacl -m u:tux:w /var/www/index.php** et **setfacl -m u:tux:-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:tux:rw,g:finances:r--,o:--- /var/www/index.php** définit des permissions dans l'ACL de /var/www/index.php pour l'utilisateur tux, le groupe finances et les autres .

Cette commande permet aussi de modifier les droits classiques (et remplace dans ce cas chmod) : l'utilisateur, le groupe et les autres initiaux du fichier sont simplement désignés par le préfixe (u:, g:, o:) suivi d'un nom vide.  
Si un fichier appartient à elmo: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. Par exemple le fichier index.php avec les droits rw-r----- pour elmo:www-data, la commande **setfacl -m u::r,g::x index.php** modifiera les droits à r----x--- pour pour elmo:www-data.

**Remarque** …  
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.

## 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:elmo: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:elmo:rwX /var/www** donne à elmo 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.

## Retirait de permissions

Pour annuler tout ou partie d'une ACL, la commande **setfacl -b /var/www/index.php** retire tout le contenu de l'ACL du fichier, tandis que   
**setfacl -x u:tux,g:finances /var/www/index.php** retire les permissions propres à tux et au groupe finances.

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

## Masque

Le masque est une synthèse des valeurs les plus permissives que possède un fichier doté d'une ACL.   
Les droits classiques de l'utilisateur ne sont cependant pas pris en compte.   
Le masque est calculé automatiquement …  
**>>** chown elmo:www-data index.php   
**>>** chmod 640 index.php  
**>>** ls -l index.php  
 -rw-r----- 1 elmo 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:kermit:rw,g:web:rw index.php  
**>>** getfacl index.php  
 # file: index.php

# owner: luce

# group: www-data

user::rw-

user:kermit: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: tux

# group: www-data

user::rw-

user:cookiemonster:rw-

group::r--

group:web:rw-

mask::rw-

other::---

**>>** setfacl -m cookiemonster:rw index.php   
**>>** getfacl index.php  
 # file: index.php

# owner: tux

# group: www-data

user::rw-

user:cookiemonster: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**.

### Commande getfacl

Cette commande suivie d'un nom de fichier affiche l'ACL de ce fichier (get file's ACL).   
Par exemple …  
**>>** getfacl /var/www  
 # file: var/www

# owner: root

# group: www-data

user::rwx

user:elmo:rwx

group::rwx

mask::rwx

other::r-x

default:user::rwx

default:user:kermit: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 elmo (user:elmo:rwx) ;
* une permission ACL par défaut donnant des droits complets à kermit sur tous les nouveaux fichiers créés sous /var/www/ (default:user:kermit: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/autres du monde des permissions classiques. Appliquer cette commande sur un fichier qui n'a pas d'ACL définie donne les mêmes informations que la commande **ls -l**, dans un format différent.   
La commande **setfacl -b index.php** permet de retirer les ACL pouvant exister.

### Commandes 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.

Lorsque 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:kermit: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 elmo 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.