

4 - Permissions, propriétaires et comptes

Lire et modifier les permissions, les propriétaires, les utilisateurs et les groupes.

- [4.1 - Comprendre les permissions Linux](#)
- [4.2 - Modifier les droits avec chmod](#)
- [4.3 - Propriétaires, groupes, chown et chgrp](#)
- [4.4 - Gérer les utilisateurs et les groupes](#)

4.1 - Comprendre les permissions Linux

Chaque fichier possède un propriétaire, un groupe et des permissions pour trois catégories : **utilisateur** (u), **groupe** (g) et **autres** (o).

Lire les droits

```
ls -l fichier.txt
ls -ld dossier
stat fichier.txt
```

Exemple :

```
-rw-r----- 1 alex informatique 1250 août 7 10:00 rapport.txt
```

Décomposition :

Bloc	Signification
-	fichier ordinaire
rw-	propriétaire : lecture et écriture
r--	groupe : lecture
---	autres : aucun droit

Signification sur un fichier

Droit	Lettre	Effet
lecture	r	lire le contenu
écriture	w	modifier le contenu
exécution	x	lancer le fichier comme programme

Signification sur un dossier

Droit	Effet
<code>r</code>	lister les noms présents
<code>w</code>	créer, renommer ou supprimer des entrées
<code>x</code>	traverser le dossier et accéder aux éléments connus

Les permissions d'un dossier expliquent pourquoi un utilisateur peut parfois supprimer un fichier qu'il ne peut pas modifier : la suppression agit sur l'entrée du dossier parent.

Tester un accès

```
test -r fichier && echo "lisible"
test -w fichier && echo "modifiable"
test -x script.sh && echo "exécutable"
```

Permissions par défaut et umask

```
umask
umask -S
```

La valeur `umask` retire des droits lors de la création. Une valeur courante `022` produit généralement des fichiers `644` et des dossiers `755`. Une valeur `027` limite davantage l'accès des autres utilisateurs.

Modification temporaire dans le shell courant :

```
umask 027
```

Au-delà des droits classiques

Un `+` après les permissions dans `ls -l` peut signaler des ACL :

```
getfacl fichier.txt
```

Les systèmes avec SELinux ou AppArmor peuvent aussi refuser une opération malgré des droits Unix apparemment corrects. Avant d'appliquer `chmod 777`, identifier le propriétaire, les droits du dossier parent, les ACL et le contexte de sécurité.

`777` donne lecture, écriture et exécution à tout le monde. Ce n'est presque jamais une bonne correction : cela masque la cause et augmente les risques.

4.2 - Modifier les droits avec chmod

`chmod` modifie les permissions d'un fichier ou d'un dossier. Il accepte une notation symbolique, plus lisible, ou une notation numérique.

Notation symbolique

Catégories : `u` propriétaire, `g` groupe, `o` autres, `a` tous.

Opérations : `+` ajouter, `-` retirer, `=` définir exactement.

```
chmod u+x script.sh      # ajouter l'exécution au propriétaire
chmod g+w partage/       # autoriser l'écriture au groupe
chmod o-r fichier.txt     # retirer la lecture aux autres
chmod u=rw,g=r,o= fichier # définir précisément les droits
chmod a+r documentation.pdf # lecture pour tous
```

Notation numérique

Chaque droit possède une valeur : lecture `4`, écriture `2`, exécution `1`.

Valeur	Droits	Signification
7	<code>rwX</code>	lecture, écriture, exécution
6	<code>rw-</code>	lecture et écriture
5	<code>r-X</code>	lecture et exécution
4	<code>r--</code>	lecture seule
0	<code>---</code>	aucun droit

```
chmod 644 fichier.txt      # rw-r--r--
chmod 600 secret.txt       # rw-----
```

```
chmod 755 script.sh          # rwxr-xr-x
chmod 750 dossier-prive      # rwxr-x---
```

Appliquer récursivement

```
chmod -R u+rwX,g+rX,o-rwx projet/
```

Le `X` majuscule ajoute l'exécution uniquement aux dossiers et aux fichiers déjà exécutables. Il évite de rendre exécutables tous les documents ordinaires.

Pour régler séparément fichiers et dossiers :

```
find projet -type d -exec chmod 750 {} +
find projet -type f -exec chmod 640 {} +
```

Commencer par afficher les cibles :

```
find projet -type f -print
```

Cas courant : clé SSH

```
chmod 700 ~/.ssh
chmod 600 ~/.ssh/authorized_keys
chmod 600 ~/.ssh/id_ed25519
chmod 644 ~/.ssh/id_ed25519.pub
```

Vérifier

```
stat -c '%A %a %U:%G %n' fichier.txt
namei -l /chemin/vers/fichier.txt
```

`namei -l` affiche les droits de chaque composant du chemin, ce qui aide à repérer un dossier parent non traversable.

Ne pas utiliser `chmod -R 777` comme dépannage universel. Accorder seulement les droits nécessaires au bon utilisateur ou au bon groupe.

4.3 - Propriétaires, groupes, chown et chgrp

`chown` change le propriétaire et éventuellement le groupe. `chgrp` change seulement le groupe.

Examiner l'appartenance

```
ls -l fichier.txt
stat -c '%U:%G %n' fichier.txt
id alex
getent group informatique
```

Changer propriétaire ou groupe

```
sudo chown alex fichier.txt
sudo chown alex:informatique fichier.txt
sudo chown :informatique fichier.txt
sudo chgrp informatique fichier.txt
```

Appliquer récursivement :

```
sudo chown -R alex:informatique /srv/projet
```

Une opération récursive doit cibler un chemin explicite et vérifié :

```
sudo ls -ld /srv/projet
sudo find /srv/projet -maxdepth 2 -printf '%u:%g %p\n'
```

Dossier partagé par un groupe

Créer un dossier collaboratif :

```
sudo mkdir -p /srv/partage
sudo chown root:informatique /srv/partage
sudo chmod 2770 /srv/partage
```

Le chiffre spécial `2` active le bit **setgid** sur le dossier. Les nouveaux éléments héritent alors du groupe `informatique` au lieu d'utiliser systématiquement le groupe principal du créateur.

Vérifier :

```
ls -ld /srv/partage
```

La permission de groupe apparaît avec `s`, par exemple `drwxrws---`.

Bits spéciaux

Bit	Valeur	Usage courant
setuid	4	exécuter un programme avec l'identité de son propriétaire
setgid	2	héritage du groupe sur un dossier
sticky bit	1	chacun ne supprime que ses propres fichiers dans un dossier partagé

Exemple du sticky bit :

```
sudo chmod 1777 /srv/depot-temporaire
```

Le répertoire `/tmp` utilise habituellement ce principe.

Retrouver les identifiants numériques

```
id -u alex
id -g alex
getent passwd alex
getent group informatique
```

Sur un partage NFS ou dans un conteneur, les numéros UID/GID sont souvent plus importants que les noms affichés. Une différence d'identifiants peut provoquer un accès refusé malgré des noms semblables.

4.4 - Gérer les utilisateurs et les groupes

La gestion des comptes varie légèrement selon la distribution. Les commandes ci-dessous couvrent les systèmes Linux courants.

Consulter les comptes

```
id alex
getent passwd alex
getent group informatique
cut -d: -f1 /etc/passwd
```

`getent` est préférable à la lecture directe des fichiers lorsque les comptes peuvent provenir de LDAP, Active Directory ou d'un autre annuaire.

Créer un utilisateur

Sur Debian et Ubuntu, `adduser` guide la création :

```
sudo adduser alex
```

Commande plus générique :

```
sudo useradd -m -s /bin/bash alex
sudo passwd alex
```

Gérer les groupes

```
sudo groupadd informatique
sudo usermod -aG informatique alex
```

```
id alex
```

L'option `-aG` signifie **ajouter aux groupes supplémentaires**. Oublier `-a` avec `usermod -G` peut retirer l'utilisateur de ses autres groupes secondaires.

La nouvelle appartenance s'applique à la prochaine ouverture de session. Pour un test ponctuel :

```
newgrp informatique
```

Verrouiller ou déverrouiller un compte

```
sudo passwd -l alex  
sudo passwd -u alex  
sudo chage -l alex
```

Verrouiller le mot de passe ne termine pas forcément les sessions existantes et n'interdit pas toujours les autres méthodes d'authentification, comme une clé SSH. La procédure de départ d'un utilisateur doit donc inclure sessions, clés, jetons et tâches planifiées.

Modifier un shell

```
chsh -s /bin/bash  
getent passwd "$USER"  
cat /etc/shells
```

Supprimer un compte

Examiner d'abord les fichiers appartenant au compte :

```
sudo find /home /srv -xdev -user alex -print
```

Puis, selon la distribution :

```
sudo userdel alex          # conserver le dossier personnel
sudo userdel -r alex       # supprimer aussi le dossier personnel
```

`userdel -r` est destructif. Sauvegarder ou transférer les données avant de l'utiliser.

Fichiers importants

Fichier	Rôle
<code>/etc/passwd</code>	comptes et shells, sans mots de passe en clair
<code>/etc/shadow</code>	empreintes des mots de passe et expiration
<code>/etc/group</code>	groupes locaux
<code>/etc/sudoers</code>	règles sudo, à modifier avec <code>visudo</code>

Ne jamais éditer `/etc/shadow` manuellement. Utiliser les commandes prévues afin de préserver la cohérence et les permissions.