

# Les systèmes d'exploitation

Filière: Génie Informatique  
S2

# **Chapitre 4**

# **La gestion des utilisateurs sous Linux**



## Notion d'utilisateur

Linux est un système multi-utilisateur, c'est-à-dire que plusieurs utilisateurs peuvent avoir un compte et travailler en même temps.

Chaque utilisateur dispose d'un certain nombre de privilèges vis-à-vis le système. Donc, Il a des droits d'accès à des fichiers et parfois il n'a pas ces droits pour d'autres fichiers.

Ces droits concernent généralement la lecture, l'écriture et l'exécution d'un fichier.



## Types des comptes

Les comptes utilisateur ne sont pas tous égaux sur Linux. On distingue trois types :

**Super-utilisateur (root)**

**Comptes systèmes**

**Comptes ordinaires**



## Types des comptes

**Super-utilisateur (root):** c'est l'utilisateur le plus important du système du point de vue de l'administration. Il n'est pas concerné par les droits d'accès aux fichiers. Son **UID** égal à 0 (zéro) lui confère sa spécificité. Ce super-utilisateur aura donc à sa charge les tâches d'administration du système.

**Comptes système:** On trouve sur le système toute une série de comptes qui ne sont pas affectés à des personnes (bin, daemon, sync, apache...). Ceux-ci servent à faciliter la gestion des droits d'accès de certaines applications et démons. Ainsi en lançant le serveur Web sous l'identité du compte "apache", on pourra aisément limiter ses droits d'accès à certains fichiers. Les UID compris entre 1 et 999 sont généralement utilisés pour ces comptes.



## Types des comptes

**Comptes ordinaires:** Tous les autres comptes utilisateur sont associés à des personnes; leur vocation est de permettre à des utilisateurs standard de se connecter. L'**UID** d'un utilisateur sera un nombre supérieur ou égal à 1000.



## Exécuter en tant que Administrateur

Ceci étant un travail d'administration du système, un utilisateur ordinaire ne peut pas accéder à ces droits d'administration.

Pour cela, nous aurons souvent à utiliser la commande **sudo** (Eng – Substitute User DO) qui permet à un utilisateur d'exécuter des commandes qui ne peuvent être utilisées que par le super-utilisateur. Elle s'utilise comme suit :

```
$ sudo commande
```

Le mot de passe de votre compte est alors demandé afin que le système vérifie votre identité



## Exécuter en tant que Administrateur

Mais!!!! On pourrait être tenté d'utiliser le super utilisateur (root) comme une session, afin de ne pas avoir à changer d'utilisateur lorsqu'il faut reconfigurer le système.

C'est possible (et nous verrons comment...) Mais

**Il ne faut pas faire cela !!!**

En effet, avec le super utilisateur, une mauvaise manipulation peut causer la perte irrémédiable de tout ou partie de vos données ou rendre la machine inutilisable !

Mais avec la substitution d'utilisateur par `sudo` c'est plutôt une manière de vous dire et de vous rappeler : **Est ce que vous êtes sûre que vous voulez faire ceci?!**



## Ouvrir un terminal en mode root

Utiliser **sudo** pour exécuter une seule commande ne cause pas un réel désagrément,

mais il peut être désagréable de l'utiliser pour exécuter une longue procédure nécessitant plusieurs interventions en mode super-utilisateur (root).

L'ouverture d'un terminal en mode root permet d'éviter d'avoir à appeler sudo à chacune des étapes de cette procédure, sans avoir à activer l'accès au compte d'utilisateur root.

L'inconvénient de cette méthode est qu'aucune trace des actions posées n'est inscrite dans le journal de sudo (sinon l'ouverture du terminal root lui-même).

Il est déconseillé d'ouvrir un terminal root.



## Ouvrir un terminal en mode root

Pour vous servir d'un terminal root :

Ouvrez une fenêtre de terminal ;

Saisissez la commande suivante : utilisateur@ordinateur:~\$ sudo -i

Saisissez votre mot de passe à l'invite de saisie de mot de passe ;

Exécutez votre série de commandes d'administration ;

Fermez la session root : root@ordinateur:~# exit ou Ctrl+D

```
ubuntu@ubuntu-VirtualBox:~$ sudo -i
[sudo] Mot de passe de ubuntu :
root@ubuntu-VirtualBox:~# exit
déconnexion
ubuntu@ubuntu-VirtualBox:~$
```



## Création d'un compte utilisateur

La commande **useradd** permet d'ajouter un nouvel utilisateur sans spécifier aucune information sur ce compte (UID, mot de passe,...)

Syntaxe :

```
# useradd [options] login
```

Le login définit le nom du compte à créer.

**Exemple:**

```
ubuntu@ubuntu-VirtualBox:~$ useradd NewUser
useradd: Permission denied.
useradd : impossible de verrouiller /etc/passwd ; veuillez réessayer plus tard.
ubuntu@ubuntu-VirtualBox:~$ █
```

Mais pourquoi cela ne marche pas??!!

Seul l'administrateur (root) qui a le droit d'ajouter un nouvel utilisateur

```
ubuntu@ubuntu-VirtualBox:~$ sudo useradd NewUser
```



## Le fichier de gestion des utilisateurs

Pour comprendre la configuration d'un compte utilisateur, nous allons nous intéressé à la façon dont Unix gère ces comptes. Or, tout est fichier en Unix.

Le premier fichier que nous allons voir, est le fichier **/etc/passwd** qui contient les informations relatives aux comptes.

C'est ce fichier que le système consulte lorsque vous vous connectez à votre compte en tapant votre identifiant et mot de passe. Si ce que vous avez tapé n'existe pas dans ce fichier alors vous ne pourrez pas vous connecter. Ce fichier contient des champs de texte séparés par : et qui respecte le format suivant:

**nom\_du\_compte** : **mot\_de\_passe** : **numero\_utilisateur** : **numero\_de\_groupe** :  
**commentaire** : **répertoire** : **programme\_de\_demarrage**



## Le fichier de gestion des utilisateurs

### La signification des champs :

- **nom du compte** = identifiant de l'utilisateur
- **mot de passe** = mot de passe de l'utilisateur. Celui-ci est crypté
- **numéro utilisateur** = le **UID**. Cet identifiant est unique. Les valeurs supérieures à 1000 sont pour les comptes utilisateurs
- **numéro de groupe** = un entier qui identifie le groupe de l'utilisateur. C'est un identifiant unique appelé **GID** (Groupe Identifier).
- **commentaire** = des informations sur l'utilisateur.
- **répertoire** = le répertoire dans lequel se retrouve l'utilisateur après s'être connecté.
- **commande** = le shell par défaut qui sera associé à ce compte.



## Le fichier de gestion des utilisateurs

Ainsi, si nous explorons le contenu de ce fichier, nous trouverons pas mal d'informations sur les différents comptes.

Par exemple les deux et les huit dernières lignes:

```
ubuntu@ubuntu-VirtualBox:~$ head -2 /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
```

```
ubuntu@ubuntu-VirtualBox:~$ tail -8 /etc/passwd
hplip:x:118:7:HPLIP system user,,,,:/var/run/hplip:/bin/false
geoclue:x:119:124:::/var/lib/geoclue:/usr/sbin/nologin
gnome-initial-setup:x:120:65534::/run/gnome-initial-setup:/bin/false
gdm:x:121:125:Gnome Display Manager:/var/lib/gdm3:/bin/false
ubuntu:x:1000:1000:Ubuntu,,,,:/home/ubuntu:/bin/bash
smi:x:1001:1001:SMI,,,,:/home/smi:/bin/bash
sma:x:1002:1002:SMA,,,,:/home/sma:/bin/bash
NewUser:x:1003:1003::/home/NewUser:/bin/sh
```



## Le fichier de gestion des utilisateurs

Déjà nous remarquons les informations qui manquent au compte nouvellement créé **NewUser**.

Et si nous explorant le répertoire **/home** nous ne trouverons pas le dossier personnel du compte **NewUser**.

```
ubuntu@ubuntu-VirtualBox:~$ ls /home
sma  smi  ubuntu
```



## Options de la commande AddUser

Création d'un compte avec les options de configuration :

```
# useradd -u 2040 -g DepInfo -G S1 -c "commentaire"  
-e 2016-07-01 -s /bin/bash -d /home/newUser newUser
```

- u : pour spécifier manuellement le UID du compte.
- g : pour spécifier le groupe par défaut.
- G : pour spécifier les groupes secondaires
- c : pour affecter un commentaire (nom exacte, adresse Email...)
- e : pour spécifier une date d'expiration de ce compte (à partir de laquelle, le compte ne sera plus accessible)
- s : pour indiquer le shell par défaut
- d : pour spécifier le répertoire personnel
- m : pour que le dossier personnel soit créé



## Modifier les propriétés d'un compte -usermod-

Pour modifier les propriétés d'un compte déjà créé, on peut utiliser la commande **usermod** dont le fonctionnement est très similaire à **useradd**.

Nous effectuons quelques modifications sur le premier compte **newUser** :

```
sudo usermod -c "c'est mon compte" -s /bin/bash newUser
```



## Changer les options par défaut -useradd-

Le second fichier que nous allons étudier est le fichier qui permet de changer les options par défaut de la commande **useradd**.

Nous pourrons faire cela en modifiant directement le fichier **/etc/default/useradd** ou bien en utilisant la commande **useradd** avec l'option **-D**.

Pour afficher les options par défaut :**useradd -D**

Changement des options par défaut

- ❑ pour changer le répertoire home
  - **useradd -D -b /home\_2**
- ❑ pour changer le groupe par défaut
  - **useradd -D -g dev**
- ❑ pour changer le shell par défaut
  - **useradd -D -s /bin/csh**

```
ubuntu@ubuntu-VirtualBox:~$ useradd -D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/sh
SKEL=/etc/skel
CREATE_MAIL_SPOOL=no
```



## Le fichier des propriétés cachés -shadow-

Le 3ème fichier concernant toujours l'administration des utilisateurs Unix, est : **/etc/shadow**. C'est un fichier sensible (sudo) qui contient les mots de passe et d'autres informations sur les comptes utilisateurs. Essayant de détailler son contenu :

root:!:16719:0:99999:7:::

daemon:\*:15630:0:99999:7:::

...

smi:\$1\$Pv\$Omhf8yD/Pg3sViEtYQtHj/:16719:0:99999:7:::

smi1!:16733:0:99999:7:::

Comme avec le fichier passwd, chaque champ dans le fichier shadow est aussi séparé par deux points “：“, et on trouve les 9 champs suivants:



## Le fichier des propriétés cachés -shadow-

1. Nom d'utilisateur, il doit être présent dans le fichier /etc/passwd.
2. Mot de passe crypté de 13 caractères.
  - Une entrée nulle (::) indique qu'un mot de passe n'est pas demandé pour entrer dans le système.
  - Une entrée (:+:) indique que le mot de passe n'a jamais été initialisé, en conséquence le compte n'est pas encore activé.
  - Une entrée (:{}) indique que le compte n'aura pas de mot de passe et il n'y aura pas la possibilité d'y accéder.
3. Le nombre de jours depuis le 1er Janvier 1970 jusqu'au jour du dernier changement du mot de passe.
4. Le nombre de jours avant que le mot de passe ne puisse être changé (un 0 indique qu'il peut être changé à n'importe quel moment).



## Le fichier des propriétés cachés -shadow-

4. Le nombre de jours après lesquels le mot de passe doit être changé (99999 indique que l'utilisateur peut garder son mot de passe inchangé pendant beaucoup, beaucoup d'années)
5. Le nombre de jours pour avertir l'utilisateur qu'un mot de passe ne va plus être valable (7 pour une semaine entière)
6. Le nombre de jours avant de désactiver le compte après expiration du mot de passe
7. Le nombre de jours depuis le 1er Janvier 1970 pendant lesquels un compte a été désactivé
8. Un champ réservé pour une utilisation future possible



## Changer de mot de passe utilisateur -passwd-

La commande passwd permet de changer le mot de passe d'un utilisateur. L'utilisateur peut changer son mot de passe personnel. Alors que, seul l'administrateur peut changer le mot de passe d'un autre.

```
ubuntu@ubuntu-VirtualBox:~$ sudo passwd user1
Entrez le nouveau mot de passe UNIX :
Retapez le nouveau mot de passe UNIX :
passwd : le mot de passe a été mis à jour avec succès
```

Revoyant l'effet de cette modification dans le fichier **/etc/shadow** :

smil:\$6\$YfrGPySW\$soeegfaoVcd1/AeGswS8Q31YsyN5DBghqxAl/wH3OeLL  
Xr7ZX6M90bWINvvsP15YMloa09A9zO7PCLMgRb92E/:16734:0:99999:7:::  
À partir de là le compte smil est Activé



## La suppression d'un utilisateur -userdel-

La commande qui permet de supprimer un utilisateur :

**userdel user1**

Si on désire aussi supprimer son dossier personnel :

**userdel -r user1**

La commande précédente supprime l'utilisateur **user1** ainsi que son répertoire personnel, cependant un problème demeure: les fichiers appartenant à **user1** et qui se trouvent en dehors du répertoire personnel ne sont pas supprimés. La commande suivante permet de les trouvés et de les supprimés à partir du **UID** de l'utilisateur (on suppose que le **UID** de l'exemple **user1** est **1002**)

**find / -type f -uid 1002 -print -exec rm {} \;**



## Les groupes : Définitions

Un groupe est un ensemble d'utilisateurs pouvant partager des fichiers et des ressources système. Par exemple, des utilisateurs qui travaillent sur le même projet peuvent former une équipe. Une telle équipe est traditionnellement connue comme groupe UNIX.

Chaque groupe doit disposer d'un nom, d'un ID (le GID) et d'une liste des noms d'utilisateur appartenant au groupe. Un GID identifie le groupe en interne sur le système.



## Les groupes : Types

Les deux types de groupes auxquels un utilisateur peut appartenir sont les suivants :

- **Groupe principal** : groupe assigné par le système d'exploitation aux fichiers créés par l'utilisateur. Chaque utilisateur doit appartenir à un groupe principal (par défaut le nom de l'utilisateur est aussi le nom de son groupe principal lors de sa création).
- **Groupes secondaires** : groupes auxquels un utilisateur peut appartenir. Les utilisateurs peuvent appartenir à un nombre maximal de 15 groupes secondaires.

Pour cela, il existe un fichier qui comporte les noms des groupes existants dans votre système.



## Les groupes : Fichiers

Le 4<sup>ème</sup> fichier à examiner est le fichier qui gère les groupes : **/etc/group**. On visualisant son contenu, nous pouvons distinguer le nom et le GID des groupes présents sur notre système. En particulier :

**sudo:x:27:ubuntu,smi,sma**

**smi:x:1000:**

**sma:x:1002:**

Remarquons (à partir du fichier **passwd**) que le compte utilisateur principale (ici : **ubuntu**) a pour groupe principal : **ubuntu GID-1000**. Cependant, il appartient à d'autres groupes secondaire comme par exemple **sudo**, et par conséquence il détient les droits d'administration. Nous pouvons changer cela par l'ajout de l'utilisateur **user1** au groupe **sudo** (**sudo:x:27:ubuntu,smi,sma,user1**). Cependant, il toujours **préférable** de changer les groupes secondaires on utilisant la commande :

**sudo usermod -a -G sudo user1**



## Les groupes : Fichiers

D'autres fichiers sont liés aux opérations d'administrations des utilisateurs et des groupes (comme /etc/gshadow, ./etc/sudoers...) Il possible d'avoir plus d'informations sur ces fichiers de configuration on utilisant man section 5 du fichier.

| Fichier      | Description  | Plus d'informations |
|--------------|--|---------------------|
| /etc/passwd  | Information sur les comptes utilisateurs                   | man 5 passwd        |
| /etc/shadow  | information cachée sur les comptes utilisateurs            | man 5 shadow        |
| /etc/group   | Défini les groupes auxquels les utilisateurs appartiennent | man 5 group         |
| /etc/gshadow | Information cachée sur les groupes                         | man 5 gshadow       |
| /etc/sudoers | Liste de qui peut lancer quoi avec <a href="#">sudo</a>    | man 5 sudoers       |



## Les groupes : Commandes

- Pour connaître les groupes d'un utilisateur à l'aide de l'une des commandes :

**groups nom\_utilisateur** ou

**id nom\_utilisateur**

- Pour créer un nouveau groupe :

**groupadd nom\_groupe**

- Pour ajouter un utilisateur à un groupe :

**gpasswd -a nom\_utilisateur nom\_groupe**

- Il est possible de rajouter plusieurs groupes à un utilisateur:

**usermod -aG grp1.grp2 nom\_utilisateur**

- Pour supprimer un groupe :

**groupdel nom\_groupe**



## Options de la commande AddUser

### Exemple

```
ubuntu@ubuntu-VirtualBox:~$ groupadd Informatique
groupadd: Permission denied.
groupadd : impossible de verrouiller /etc/group ; veuillez réessayer plus tard.
ubuntu@ubuntu-VirtualBox:~$ sudo groupadd Informatique
[sudo] Mot de passe de ubuntu :
ubuntu@ubuntu-VirtualBox:~$ sudo groupadd Mathematique
ubuntu@ubuntu-VirtualBox:~$ sudo groupadd Physique
ubuntu@ubuntu-VirtualBox:~$ useradd smil -m
useradd: Permission denied.
useradd : impossible de verrouiller /etc/passwd ; veuillez réessayer plus tard.
ubuntu@ubuntu-VirtualBox:~$ sudo useradd smil -m
ubuntu@ubuntu-VirtualBox:~$ sudo useradd smi2 -m
ubuntu@ubuntu-VirtualBox:~$ sudo useradd smpc1 -g Informatique -m
ubuntu@ubuntu-VirtualBox:~$ sudo useradd smpc2 -g Informatique -m
ubuntu@ubuntu-VirtualBox:~$ sudo useradd smal -g Mathematique -m
ubuntu@ubuntu-VirtualBox:~$ sudo useradd sma2 -g Mathematique -m
ubuntu@ubuntu-VirtualBox:~$ sudo useradd smial -g Mathematique -m
ubuntu@ubuntu-VirtualBox:~$ sudo useradd smia2 -g Mathematique -G Informatique -m
ubuntu@ubuntu-VirtualBox:~$ █
```



## Les groupes : Commandes

### Exemple

```
ubuntu@ubuntu-VirtualBox:~$ id smi1
uid=1005(smi1) gid=1008(smi1) groupes=1008(smi1)
ubuntu@ubuntu-VirtualBox:~$ id smi2
uid=1006(smi2) gid=1009(smi2) groupes=1009(smi2)
ubuntu@ubuntu-VirtualBox:~$ id smpc1
uid=1007(smpc1) gid=1005(Informatique) groupes=1005(Informatique)
ubuntu@ubuntu-VirtualBox:~$ id smpc2
uid=1008(smpc2) gid=1005(Informatique) groupes=1005(Informatique)
ubuntu@ubuntu-VirtualBox:~$ id sma1
uid=1009(sma1) gid=1006(Mathematique) groupes=1006(Mathematique)
ubuntu@ubuntu-VirtualBox:~$ id sma2
uid=1010(sma2) gid=1006(Mathematique) groupes=1006(Mathematique)
ubuntu@ubuntu-VirtualBox:~$ id smial
uid=1011(smial) gid=1006(Mathematique) groupes=1006(Mathematique)
ubuntu@ubuntu-VirtualBox:~$ id smia2
uid=1012(smia2) gid=1006(Mathematique) groupes=1006(Mathematique),1005(Informatique)
ubuntu@ubuntu-VirtualBox:~$ █
```



## Les groupes : Commandes

### Exemple

```
ubuntu@ubuntu-VirtualBox:~$ sudo gpasswd -a smil Informatique
Ajout de l'utilisateur smil au groupe Informatique
ubuntu@ubuntu-VirtualBox:~$ id smil
uid=1005(smil) gid=1008(smil) groupes=1008(smil),1005(Informatique)
ubuntu@ubuntu-VirtualBox:~$ sudo usermod -g Informatique smi2
ubuntu@ubuntu-VirtualBox:~$ id smi2
uid=1006(smi2) gid=1005(Informatique) groupes=1005(Informatique)
ubuntu@ubuntu-VirtualBox:~$ sudo usermod -g Physique smpc1
ubuntu@ubuntu-VirtualBox:~$ id smpc1
uid=1007(smpc1) gid=1007(Physique) groupes=1007(Physique)
ubuntu@ubuntu-VirtualBox:~$ sudo gpasswd -a smpc1 Physique
Ajout de l'utilisateur smpc1 au groupe Physique
ubuntu@ubuntu-VirtualBox:~$ sudo gpasswd -a smpc2 Physique
Ajout de l'utilisateur smpc2 au groupe Physique
ubuntu@ubuntu-VirtualBox:~$ id smpc2
uid=1008(smpc2) gid=1005(Informatique) groupes=1005(Informatique),1007(Physique)
ubuntu@ubuntu-VirtualBox:~$ id smpc1
uid=1007(smpc1) gid=1007(Physique) groupes=1007(Physique)
```



## Les groupes : Commandes

### Exemple

```
ubuntu@ubuntu-VirtualBox:~$ sudo deluser smpc2 Informatique
/usr/sbin/deluser: Impossible de retirer un utilisateur de son groupe primaire.
ubuntu@ubuntu-VirtualBox:~$ sudo usermod -g Physique smpc2
ubuntu@ubuntu-VirtualBox:~$ sudo deluser smpc2 Informatique
/usr/sbin/deluser: L'utilisateur « smpc2 » n'est pas membre du groupe « Informatique ».
ubuntu@ubuntu-VirtualBox:~$ id smpc2
uid=1008(smpc2) gid=1007(Physique) groupes=1007(Physique)
ubuntu@ubuntu-VirtualBox:~$ id smpc1
uid=1007(smpc1) gid=1007(Physique) groupes=1007(Physique)
ubuntu@ubuntu-VirtualBox:~$ █
```



## Les droits d'accès

- Linux possède un mécanisme de gestion des droits d'accès. Donc, il est possible pour un propriétaire de fichier de donner ou d'omettre aux autres utilisateur quelques droits.

| Droits d'accès               | Sur les répertoires  | Sur les fichiers   |
|------------------------------|--|--|
| <b>Lire (read ) (r)</b>      | Autorisation de voir le contenu d'un répertoire ou les sous-répertoires.         | Autorisation de voir le contenu du fichier.  |
| <b>Écrire (write) (w)</b>    | Autorisation de créer, modifier, supprimer les fichiers ou les sous-répertoires. | Autorisation aux entités d'ajouter, de modifier, de supprimer le contenu d'un fichier. |
| <b>Executer (execute)(x)</b> | Autorisation d'accéder au répertoire   | Permettre d'exécuter le fichier  |
| (-)                          | Pas d'autorisation   | Pas d'autorisation   |



## Les droits d'accès

```
smi@ubuntu: ~/Documents
smi@ubuntu:~/Documents$ ls -l
total 4
-rw-rw-r-- 1 smi smi    0 Oct 31 07:33 mon_fichier.txt
drwxrwxr-x 2 smi smi 4096 Oct 31 07:36 mon_repertoire
smi@ubuntu:~/Documents$
```

- | **r W X** | **r W -** | **r - -**

**d** | **r W X** | **r W -** | **r - -**

Type de fichier

Droits pour le propriétaire

Droits pour le groupe

Droits pour les autres

Avec:

|   |                    |
|---|--------------------|
| - | Fichier ordinaire  |
| d | Fichier répertoire |
| r | Read (lecture)     |
| w | Write (Ecriture)   |
| x | Exécution          |



## Les droits d'accès

On explique les droits suivants:

- r W X r W - r - -

- ✓ L'utilisateur propriétaire possède tous les droits sur le fichier ordinaire
- ✓ Les utilisateurs du même groupe que le propriétaire possèdent le droit de lire et écrire mais pas exécuter
- ✓ Les utilisateurs autres que ceux du groupe du propriétaire n'ont que le droit de lecture



## Changement des droits d'accès - chmod -

La commande CHMOD (Eng. Change Mode)

Il existe deux façons d'utiliser **chmod**:

- mode symbolique
- mode absolu



## Changement des droits d'accès - chmod -

### CHMOD en mode symbolique

| Opérateur<br>chmod | Signification  | Exemple                     | Résultat  |
|--------------------|--|-----------------------------|---|
| +                  | Ajouter les droits désignés à un fichier ou répertoire   | chmod o+wx mon_fichier.txt  | Ajout des droits de modification et d'exécution au autres utilisateurs            |
| -                  | Supprimer les droits désignés à un fichier ou répertoire | chmod u-x mon_fichier.txt   | Supprime le droit d'exécuter ce fichier pour le propriétaire                      |
| =                  | Attribuer exactement ces droits                          | chmod g=r-x mon_fichier.txt | Donne exactement les droits de lire et d'exécuter pour les utilisateurs du groupe |



## Changement des droits d'accès - chmod -

### CHMOD en mode symbolique

#### Exemples

NB : il est possible de regrouper toutes ces modification en une seul commande :  
**chmod o+wx,u-x,g=rx mon\_fichier.txt**

Pour ajouter le droit d'exécution au propriétaire (**User**)

#### **Chmod u+x fichier**

Pour enlever le droit d'écriture au utilisateurs du groupe(**Group**)

#### **Chmod g-w fichier**

Pour ajouter les droits de lecture et exécution au autres (**Others**)

#### **Chmod o+rx fichier**

Pour affecter les droits de lecture et exécution au propriétaire (**User**)

#### **Chmod u=r-x fichier**



## Changement des droits d'accès - chmod -

### CHMOD en mode absolu

La seconde façon d'attribuer les droits d'accès par chmod, consiste à utiliser des nombres pour chaque ensemble de droits. Nous pouvons calculer ces nombres si on retient la manière simple de transformer un nombre en base binaire vers une base décimale. Ainsi si nous avons ces règles simples à retenir:

- (r, w, ou x) est représenté par 1. (-) est représenté par 0
- Nous supposant que les 3 bits obtenus sont en binaire, puis nous calculons le nombre équivalent en décimale.



## Changement des droits d'accès - chmod -

### CHMOD en mode absolu

Exemples

-    **r w X**    **r w -**    **r - -**

1 1 1    1 1 0    1 0 0

7        6        4

Pour affecter ces droits en mode absolu

**Chmod 764 fichier**



## Changement des droits d'accès - chmod -

### CHMOD en mode absolu

#### Exercice d'application

Expliquer les commandes d'accès suivantes

Chmod 012 fichier

Chmod 234 fichier

Chmod 345 fichier

Chmod 567 fichier

Chmod 000 fichier

Chmod 785 fichier

Chmod 002 fichier

Chmod 234 fichier



## Droits attribués automatiquement à un fichier

Lorsqu'un nouveau fichier est créé, celui-ci obtient automatiquement certains droits. Ces derniers sont définis par défaut dans le fichier de paramétrage de la session : **/etc/pam.d/common-session**.

Il est possible de définir les droits aux fichiers et aux répertoires lors de leur création. Nous utilisons la notion de masque à travers la commande **umask** (Eng - user file creation mode mask, masque de création de fichier par l'utilisateur) .



## Droits attribués automatiquement à un fichier

Premièrement pour savoir quel masque est utilisé par défaut, nous tapons : **umask**

Si on veut afficher les droits par défaut en mode symbolique :  
**umask -S**

```
ubuntu@ubuntu-VirtualBox:~/rep1$ umask
0022
ubuntu@ubuntu-VirtualBox:~/rep1$ umask -S
u=rwx, g=rx, o=rx
ubuntu@ubuntu-VirtualBox:~/rep1$ touch fichier1
ubuntu@ubuntu-VirtualBox:~/rep1$ mkdir répertoire1
ubuntu@ubuntu-VirtualBox:~/rep1$ ls -l
total 4
-rw-r--r-- 1 ubuntu ubuntu    0 oct.  22 10:26 fichier1
drwxr-xr-x 2 ubuntu ubuntu 4096 oct.  22 10:27 répertoire1
ubuntu@ubuntu-VirtualBox:~/rep1$
```



## Droits attribués automatiquement à un fichier

### Explication

**Permission par défaut** = Permission initiale – Masque (022)

**NB**

La permission initiale d'un fichier est **666**

La permission initiale d'un répertoire est **777**

### Exemples

Pour trouver la permission par défaut d'un fichier:

**666**

**rw- rw- rw-**

**—**

**022**

**--- -W- -W-**

**644**

**rw- r-- r--**

Pour trouver la permission par défaut d'un répertoire:

**777**

**rwx rwx rwx**

**—**

**022**

**--- -W- -W-**

**755**

**rwx r-x r-x**



## Droits attribués automatiquement à un fichier

### Changement permanent

Pour changer les droits par défaut d'une manière **permanente**. Il faut:

1. Editer le fichier `/etc/pam.d/common-session`
2. Chercher la ligne ***session optional pam\_umask.so***
3. Changer la ligne par
4. ***session optional pam\_umask.so umask=valeur\_masque***
5. Enregistrer et modifier



## Changement du propriétaire et du groupe

### Les commandes de changement du propriétaire et du groupe d'un fichier

La commande **chown** permet de changer le propriétaire d'un fichier. Pour des raisons de sécurité, seul le **root** peut modifier le propriétaire d'un fichier ou d'un répertoire.

La commande **chgrp** permet le changement de groupe pour les fichiers ou répertoires cités, à condition que l'utilisateur fasse partie du nouveau groupe et soit propriétaire de ces fichiers ou répertoires.



## Changement du propriétaire et du groupe

### La commande chgrp

**chgrp** est utilisée pour changer le groupe du fichier ou du répertoire. Le changement de groupe peut être effectué par :

- Le root
- Le propriétaire du fichier si ce dernier est parmi les membres du groupe en question.

**Syntaxe** : **chgrp [options] nouveau\_groupe fichier/répertoire**

*Les options intéressantes :*

**-R** : Changer l'autorisation sur les fichiers qui sont dans les sous-répertoires du répertoire en question.

**-c** : Changer l'autorisation pour chaque fichier.



## Changement du propriétaire et du groupe

### La commande chgrp

#### Exemple

```
ubuntu@ubuntu-VirtualBox:~$ sudo chgrp Informatique fichier1
[sudo] Mot de passe de ubuntu :
ubuntu@ubuntu-VirtualBox:~$ ls -l fichier1
-rw-r--r-- 1 ubuntu Informatique 0 oct. 22 16:33 fichier1
ubuntu@ubuntu-VirtualBox:~$ █
```



## Changement du propriétaire et du groupe

### La commande chown

**chown** est utilisée pour changer le propriétaire et/ou le groupe propriétaire du fichier ou du répertoire.

Le syntaxe de cette command :

**chown [-option] [utilisateur][:groupe] fichier [fichier1 fichier2 ..]**

Elle peut être utilisée pour changer :

- Le propriétaire et le groupe, Seulement le propriétaire, Seulement le groupe (devient alors similaire à **chgrp**)

Les options intéressantes :

**-R** : Modifie tous ses sous-répertoires et ses sous-fichiers d'une manière récursive.



## Changement du propriétaire et du groupe

### La commande chown

#### Exemple

```
ubuntu@ubuntu-VirtualBox:~$ sudo chown :Mathematique fichier1
ubuntu@ubuntu-VirtualBox:~$ ls -l fichier1
-rw-r--r-- 1 ubuntu Mathematique 0 oct. 22 16:33 fichier1
ubuntu@ubuntu-VirtualBox:~$ sudo chown smil:Informatique fichier1
ubuntu@ubuntu-VirtualBox:~$ ls -l fichier1
-rw-r--r-- 1 smil Informatique 0 oct. 22 16:33 fichier1
ubuntu@ubuntu-VirtualBox:~$ █
```



## ACL (Access Control List)

### Limites de la gestion des accès

Nous avons vu que par défaut le système Unix attache trois privilèges à un fichier :

- Les privilèges de l'utilisateur propriétaire (u).
- Ceux du groupe propriétaire (g).
- Ceux des autres (o).

Ainsi le partage de fichiers ne peut se faire qu'à travers le principe de groupe. Imaginant alors la situation suivante:



## ACL (Access Control List)

### Limites de la gestion des accès

Pour qu'un utilisateur X rend un fichier accessible en lecture et écriture à un utilisateur Y et uniquement à Y, il n'y a pas d'autre solution que de créer un groupe G auquel X et Y appartiennent, puis d'accorder les droits de lecture et d'écriture du fichier au groupe G.



## ACL (Access Control List)

### Limites de la gestion des accès

Le problème ici est que la création du groupe n'est possible que pour l'administrateur du système qui seul peut ajouter un nouveau groupe d'utilisateurs.

En outre si un autre utilisateur Z souhaitait rejoindre le projet, il faudrait à nouveau lui faire une demande à l'administrateur pour ajouter le nouveau membre Z au groupe G.



## ACL (Access Control List)

### Introduction aux ACL

Nous vous proposons donc ici de découvrir quelques commandes (ACL) permettant de régler les priviléges plus finement et de manière plus autonome.

Mais, pour travailler avec les ACL, il y a deux prérequis :

- Le noyau doit supporter les ACL.
- Le système de fichier est monté avec l'option ACL

Sachez que les ACL ne peuvent être utilisées que si le noyau les supporte (Sinon, il faut recompiler le noyau...).

Sur Ubuntu, le noyau prend en charge les ACL, mais elles ne sont pas nativement activées.



## ACL (Access Control List)

### Introduction aux ACL

Vérifiant la configuration du noyau à travers cette commande lancée en mode super-user :

```
# grep ACL /boot/config-*
```

La ligne suivante indique que le support général des ACL est présent :

**CONFIG\_\*\*\*\_FS\_POSIX\_ACL=y**

les étoiles sont remplacées par les systèmes de fichiers prises en charge par les ACL (exemples: EXT2,EXT3,EXT4,JFS,...)



## ACL (Access Control List)

```
ubuntu@ubuntu-VirtualBox:~$ sudo grep ACL /boot/config-*
/boot/config-4.15.0-29-generic:CONFIG_EXT4_FS_POSIX_ACL=y
/boot/config-4.15.0-29-generic:CONFIG_REISERFS_FS_POSIX_ACL=y
/boot/config-4.15.0-29-generic:CONFIG_JFS_POSIX_ACL=y
/boot/config-4.15.0-29-generic:CONFIG_XFS_POSIX_ACL=y
/boot/config-4.15.0-29-generic:CONFIG_BTRFS_FS_POSIX_ACL=y
/boot/config-4.15.0-29-generic:CONFIG_F2FS_FS_POSIX_ACL=y
/boot/config-4.15.0-29-generic:CONFIG_FS_POSIX_ACL=y
/boot/config-4.15.0-29-generic:CONFIG_TMPFS_POSIX_ACL=y
/boot/config-4.15.0-29-generic:CONFIG_HFSPLUS_FS_POSIX_ACL=y
/boot/config-4.15.0-29-generic:CONFIG_JFFS2_FS_POSIX_ACL=y
/boot/config-4.15.0-29-generic:CONFIG_NFS_V3_ACL=y
/boot/config-4.15.0-29-generic:CONFIG_NFSD_V2_ACL=y
/boot/config-4.15.0-29-generic:CONFIG_NFSD_V3_ACL=y
/boot/config-4.15.0-29-generic:CONFIG_NFS_ACL_SUPPORT=m
/boot/config-4.15.0-29-generic:CONFIG_CEPH_FS_POSIX_ACL=y
/boot/config-4.15.0-29-generic:CONFIG_CIFS_ACL=y
```



## ACL (Access Control List)

### Introduction aux ACL

Votre système de fichiers est de type : EXT4, donc normalement les ACL sont supportées et activées, mais nous ne pouvons toujours pas les modifier, pour cela nous devons installer le paquet **acl** qui comporte les deux commandes nécessaires pour cette modification, en tapant la commande suivante :

```
# apt-get install acl
```

**NB :**

- 1- Cela peut nécessiter une connexion internet
- 2- Parfois il est nécessaire de remonter les partitions avec l'option **acl**



## ACL (Access Control List)

### Introduction aux ACL

Le paquetage acl installé, permet de disposer de deux nouvelles commandes pour la gestion des ACL sous les trois types d'ACL. C'est deux commandes sont :

**setfacl** (Eng - set file's ACL « régler l'ACL du fichier ») : mise en place de ces autorisations.

**getfacl** (Eng - get file's ACL « récupérer l'ACL du fichier») examen des autorisations d'un fichier.



## ACL (Access Control List)

### Attribution des ACL

La syntaxe:

**setfacl -m u:utilisateur:permissions fichier**

Permet de modifier (c'est le sens de l'option m) les autorisations de l'utilisateur (argument u:). Les autorisations permissions peuvent être définies en utilisant la notation symbolique sur le fichier.

**-m, --modify** - modifier les ACL d'un fichier ou répertoire

**-x, --remove** - supprime des entrées ACLs

**-b, --remove-all** - supprime toutes les entrées ACLs

**-R, --recursive** - application des ACLs de façon récursive

Regardez le manuel man setfacl pour plus de détails.



## ACL (Access Control List)

### Examiner les autorisations associées à un fichier

La syntaxe:

**getfacl [option] fichier**

La commande getfacl nous permet d'afficher les ACL du fichier. Les quelques options :

- R permet de voir les ACLs de façon récursive.
- L pour le suivre des liens symboliques.



## ACL (Access Control List)

### Exemple

```
ubuntu@ubuntu-VirtualBox:~$ getfacl fichier1
# file: fichier1
# owner: smil
# group: Informatique
user::rw-
group::r--
other::r--

ubuntu@ubuntu-VirtualBox:~$ setfacl -m u:smil:rwx fichier1
setfacl: fichier1: Opération non permise
ubuntu@ubuntu-VirtualBox:~$ setfacl -m u:smil:rwx Exercice1
ubuntu@ubuntu-VirtualBox:~$ getfacl Exercice1
# file: Exercice1
# owner: ubuntu
# group: ubuntu
user::rw-
user:smil:rwx
group::r--
mask::rwx
other::r--
```



## ACL (Access Control List)

### Exemple

De la même manière, les droits ACL peuvent être attribuée à un groupe, en remplaçant le u par g

```
ubuntu@ubuntu-VirtualBox:~$ setfacl -m g:Mathematique:rwx Exercice1
ubuntu@ubuntu-VirtualBox:~$ getfacl Exercice1
# file: Exercice1
# owner: ubuntu
# group: ubuntu
user::rw-
user:smil:rwx
group::r--
group:Mathematique:rwx
mask::rwx
other::r--
```



## ACL (Access Control List)

### Exemple

Si nous désirons enlever à smil tout droit ACL sur ce fichier :

À travers cette commande : **setfacl -x u:smil fichier**

Et il redevient un utilisateur normal soumis aux règles classiques.

Vous pouvez aussi supprimer l'ensemble des droits ACL d'un fichier

**setfacl -b fichier**

# **Fin du chapitre 4**