Reprenez le contrôle à l'aide de Linux !

30 heures Facile

Mis à jour le 29/06/2021

Le contenu de ce cours n'est plus à jour

Nous avons archivé ce cours et n'actualiserons plus son contenu.

Accédez au contenu le plus récent en découvrant ce cours :

SYSTÈMES & RÉSEAUX

Initiez-vous à Linux

Easy 8 heures

Dans ce cours débutant, découvrez Linux : un système d'exploitation gratuit et fascinant qui vous donnera un contrôle sans précédent sur votre ordinateur ! Créé par des passionnés d'informatique, Linux est un vecteur important de la philosophie du libre et l'alternative parfaite à Windows ou macOS.

VOIR LE NOUVEAU COURS

Les utilisateurs et les droits

Linux est un système multi-utilisateurs. Cela signifie que plusieurs personnes peuvent travailler simultanément sur le même OS, en s'y connectant à distance notamment.

Puisque plusieurs utilisateurs peuvent être connectés à Linux en même temps, celui-ci doit avoir une excellente organisation dès le départ. Ainsi chaque personne a **son** propre compte utilisateur, et il existe un ensemble de règles qui disent qui a le droit de faire quoi.
Je vous propose de découvrir tous ces mécanismes dans ce chapitre.

sudo: exécuter une commande en root

Lorsque vous avez installé Ubuntu, on vous a demandé le nom du compte utilisateur que vous vouliez créer. Par exemple dans mon cas j'ai créé l'utilisateur « mateo21 ».
Dans la plupart des distributions Linux on vous proposera de créer un compte utilisateur avec des **droits limités**, comme c'est le cas pour mon compte « mateo21 ».

> Attends, c'est nous qui avons installé Linux mais on n'a pas le droit de faire tout ce que l'on veut dessus ?

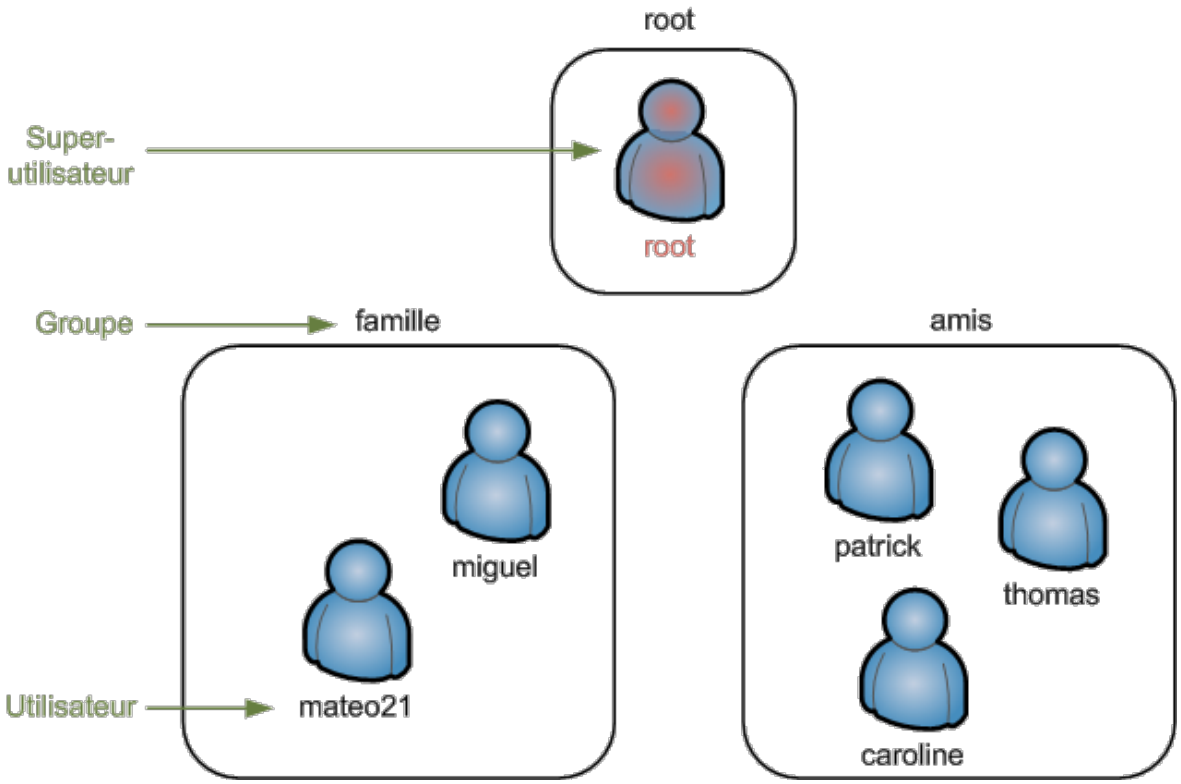
Oui, et c'est une sécurité. Bien sûr, comme vous êtes aux commandes, vous pouvez à tout moment dire : « Bon allez on passe en mode chef-qui-peut-tout-faire ». Mais c'est une sécurité de ne pas avoir le droit de tout faire par défaut, car certaines commandes peuvent être dangereuses pour la stabilité et la sécurité de votre ordinateur. Avoir des droits limités, cela signifie aussi qu'on s'empêche par exemple d'exécuter la « commande de la mort qui tue » qu'on a vue dans le chapitre précédent (`rm -rf /*`).

Nous allons d'abord commencer par voir comment sont organisés les utilisateurs sous Linux, puis nous verrons comment devenir le « chef ».
Plus loin dans le chapitre, nous apprendrons à créer et supprimer des utilisateurs en ligne de commande.

L'organisation des utilisateurs sous Linux

On peut créer autant d'utilisateurs que l'on veut, eux-mêmes répartis dans des groupes.
Il y a un utilisateur « spécial », **root**, aussi appelé superutilisateur. Celui-ci a tous les droits sur la machine.

Vous pouvez voir ce que ça donne sur la figure suivante.



Au départ, chez moi, deux utilisateurs sont créés : root et mateo21.

On ne se connecte en root que très rarement, **lorsque c'est nécessaire**. Certaines commandes de Linux que nous allons voir dans ce chapitre ne sont accessibles qu'à root.
Le reste du temps, on utilise le compte « limité » que l'on a créé (mateo21 dans mon cas).

Cette simple protection permet de largement limiter les dégâts en cas de fausse manipulation, de virus sur votre PC, etc. En effet, un virus ne peut rien faire de plus que vous quand vous êtes connectés avec des droits limités. En revanche, si vous êtes en root il pourra tout faire, même détruire votre ordinateur.
Sous Windows, vous êtes toujours connectés en administrateur par défaut (équivalent de root), ce qui explique pourquoi les virus y sont si dangereux.

Exception : Ubuntu est une des rares distributions à interdire de se connecter (logger) en root. Le compte root existe mais vous n'y avez pas accès directement. Nous allons voir que ce n'est pas un problème puisqu'on peut y accéder indirectement.
Les développeurs d'Ubuntu justifient ce choix car ils considèrent, à juste titre, qu'il est dangereux de laisser le compte root entre les mains d'un débutant. Moi-même sur d'autres distributions j'ai tendance à désactiver l'accès direct à l'utilisateur root.

`sudo` : devenir root un instant

Par défaut, vous êtes connectés sous votre compte limité (mateo21 pour ma part).
Il est impossible sous Ubuntu de se connecter directement en root au démarrage de l'ordinateur. Comment faire alors pour exécuter des commandes que seul root a le droit d'exécuter ?

On peut devenir root **temporairement** à l'aide de la commande `sudo`.
Cette commande signifie « Faire en se substituant à l'utilisateur » : **S**ubstitute **U**ser **DO**.

Écrivez donc `sudo` suivi de la commande que vous voulez exécuter, comme ceci :

```
sudo commande
```

On vous demandera normalement votre mot de passe (au moins la première fois) pour exécuter la commande. Ce mot de passe est le même que celui de votre compte utilisateur limité.

Par exemple, vous pouvez exécuter un simple `ls` avec les droits root (vous ne risquez rien, rassurez-vous) :

```
mateo21@mateo21-desktop:/home$ sudo ls
[sudo] password for mateo21:
autredossier   Desktop    Examples  Images  Modèles     Musique  tutos
autresanimaux  Documents  images    log     mondossier  Public   Vidéos
```

Comme vous le voyez, on vous demande d'abord le mot de passe, par sécurité.
Faire un `ls` en tant que root n'apporte rien de bien spécial, c'était simplement pour avoir un exemple « sûr » avec lequel vous ne risquez pas d'endommager votre ordinateur.

`sudo su` : devenir root et le rester

Si vous tapez `sudo su` (tout court), vous passerez root indéfiniment.

```
mateo21@mateo21-desktop:/home$ sudo su
[sudo] password for mateo21:
root@mateo21-desktop:/home#
```

Le symbole `#` à la fin de l'invite de commandes vous indique que vous êtes devenus superutilisateur.
Vous pouvez alors exécuter autant de commandes en root que vous le voulez.

Pour quitter le « mode root », tapez `exit` (ou faites la combinaison `Ctrl + D`).

```
root@mateo21-desktop:/home/mateo21# exit
exit
mateo21@mateo21-desktop:~$
```

Et vous voilà redevenus simples mortels.

Sous les autres distributions qu'Ubuntu, écrire « `su` » suffit à passer root.
Il est néanmoins recommandé dans ce cas d'ajouter un tiret en paramètre, c'est-à-dire d'écrire « `su -` ». L'ajout du tiret a pour effet de rendre accessibles certains programmes destinés seulement à root. Par ailleurs, cela vous place directement dans le dossier personnel de root (`/root`).

adduser : gestion des utilisateurs

Maintenant que vous savez passer root (temporairement ou indéfiniment), nous allons pouvoir découvrir des commandes qui sont réservées à root.
`adduser` et `deluser` sont de celles-là. Si vous essayez de les appeler avec votre utilisateur normal, on vous dira que vous n'avez pas le droit de les utiliser. Seul root peut gérer les utilisateurs.

`adduser` : ajouter un utilisateur

La commande `adduser` permet d'ajouter un utilisateur. Vous devez au minimum fournir un paramètre : le nom de l'utilisateur à créer.
Par exemple, pour créer un compte pour Patrick :

```
root@mateo21-desktop:/home# adduser patrick
Ajout de l'utilisateur « patrick »...
Ajout du nouveau groupe « patrick » (1001)...
Ajout du nouvel utilisateur « patrick » (1001) avec le groupe « patrick »...
Création du répertoire personnel « /home/patrick »...
Copie des fichiers depuis « /etc/skel »...
```

Pensez à rajouter un `sudo` devant la commande si vous n'êtes pas déjà root ; pour cela, tapez `sudo adduser patrick`. Moi je n'ai pas eu à le faire car j'ai choisi de rester root indéfiniment en tapant `sudo su` auparavant.
Si vous tentez d'exécuter la commande avec votre compte limité, vous aurez une erreur de ce genre : « `adduser` : Seul le superutilisateur peut ajouter un utilisateur ou un groupe sur le système ».

Le répertoire personnel de patrick est automatiquement créé (`/home/patrick`) et son compte est préconfiguré.
On vous demande ensuite de taper son mot de passe :

```
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
```

Tapez le mot de passe de patrick puis faites `Entrée`. Retapez-le pour valider.
Encore une fois, si vous ne voyez pas d'étoiles `*` quand vous tapez le mot de passe, c'est normal ; c'est une sécurité pour qu'on ne puisse pas compter le nombre de caractères derrière votre épaule.

On vous propose ensuite de rentrer quelques informations personnelles sur patrick, comme son nom, son numéro de téléphone... Si vous voulez le faire, faites-le, mais sinon sachez que vous pouvez taper `Entrée` sans rien écrire ; on ne vous embêtera pas.

```
Modification des informations relatives à l'utilisateur patrick
Entrez la nouvelle valeur ou « Entrée » pour conserver la valeur proposée
        Nom complet []:
```

```
        N° de bureau []:
        Téléphone professionnel []:
        Téléphone personnel []:
        Autre []:
Ces informations sont-elles correctes ? [o/N] o
```

À la fin, on vous demande de confirmer par un « `o` » (oui) que tout est bon. Tapez `Entrée` et ça y est, le compte de patrick est créé !

`passwd` : changer le mot de passe

S'il était nécessaire de changer le mot de passe de patrick par la suite, utilisez la commande `passwd` en indiquant en paramètre le nom du compte à modifier.

```
root@mateo21-desktop:/home# passwd patrick
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
```

Attention ! Si vous appelez `passwd` sans préciser de compte en paramètre, c'est le mot de passe de l'utilisateur sous lequel vous êtes connecté que vous changerez ! Ainsi, si vous êtes en root, c'est le mot de passe de root qui sera modifié.

`deluser` : supprimer un compte

patrick vous ennuie ? patrick est parti ? Si son compte n'est plus nécessaire (ou que vous voulez vous venger) vous pouvez le supprimer avec `deluser` .

```
deluser patrick
```

Aucune confirmation ne vous sera demandée !

Ne supprimez en aucun cas votre compte utilisateur ! Par exemple, je ne dois surtout pas supprimer le compte mateo21.
En effet, si je le fais, il n'y aura plus que root sur la machine... et Ubuntu interdit de se `logger` en root. Par conséquent, au prochain démarrage de la machine vous ne pourrez pas vous connecter... et vous serez complètement coincés !

Toutefois, cette commande seule ne supprime pas le répertoire personnel de patrick. Si vous voulez supprimer aussi son `home` et tous ses fichiers personnels, utilisez le paramètre `--remove-home` :

```
deluser --remove-home patrick
```

`adduser` et `deluser` sont des commandes qui n'existent que sous Debian et tous ses descendants, dont Ubuntu.
Partout ailleurs on doit utiliser `useradd` et `userdel`, qui sont les commandes Unix traditionnelles fonctionnant partout. Elles font globalement la même chose mais de manière beaucoup plus basique : si vous n'appelez pas `passwd` vous-mêmes, le compte ne sera pas activé et n'aura pas de mot de passe.

addgroup : gestion des groupes

Je vous l'ai dit au début : chaque utilisateur appartient à un groupe.

Oui mais dans ce cas, à quel groupe appartiennent les utilisateurs mateo21 et patrick ? On n'a rien défini, nous !

En effet, si vous ne définissez rien, un groupe du même nom que l'utilisateur sera automatiquement créé : ainsi, mateo21 appartient au groupe mateo21 et patrick au groupe patrick.

On peut le vérifier en regardant à qui appartiennent les dossiers dans `/home` via un `ls -l` :

```
root@mateo21-desktop:~# cd /home
root@mateo21-desktop:/home# ls -l
total 24
drwx------  2 root    root    16384 2007-09-19 18:22 lost+found
drwxr-xr-x 65 mateo21 mateo21  4096 2007-11-15 22:40 mateo21
drwxr-xr-x  2 patrick patrick  4096 2007-11-15 23:00 patrick
```

Souvenez-vous : la 3ème colonne indique le propriétaire du fichier ou dossier ; la 4ème indique le groupe qui possède ce fichier ou dossier.

Ainsi, le dossier mateo21 appartient à l'utilisateur mateo21 et au groupe mateo21.
Même chose pour patrick.
On constatera par ailleurs que `lost+found` appartient à root et qu'il y a un groupe root (root fait donc partie du groupe root).

Bon, mais quel intérêt y a-t-il à ce que tout le monde soit dans son propre groupe, me direz-vous ?
Vous pourriez très bien vous contenter de ce fonctionnement (un utilisateur = un groupe), mais au cas où vous auriez beaucoup d'utilisateurs, je vais quand même vous montrer comment créer des groupes.

`addgroup` : créer un groupe

La commande `addgroup` crée un nouveau groupe. Vous avez juste besoin de spécifier le nom de celui-ci en paramètre :

```
root@mateo21-desktop:/home# addgroup amis
```

```
Ajout du groupe « amis » (identifiant 1002)...
Terminé.
```

Super, mais personne ne fait encore partie de ce groupe. :-(

`usermod` : modifier un utilisateur

La commande `usermod` permet d'éditer un utilisateur. Elle possède plusieurs paramètres ; nous allons en retenir deux :

- **-l** : renomme l'utilisateur (le nom de son répertoire personnel ne sera pas changé par contre) ;
- **-g** : change de groupe.

Si je veux mettre patrick dans le groupe `amis`, je ferai donc comme ceci :

```
usermod -g amis patrick
```

Et pour remettre patrick dans le groupe patrick comme il l'était avant :

```
usermod -g patrick patrick
```

Il est aussi possible de faire en sorte qu'un utilisateur appartienne à plusieurs groupes. Pour ce faire, utilisez le paramètre `-G` (majuscule).
Exemple : `usermod -G amis,paris,collegues patrick`.
Séparez les noms des groupes par une virgule, sans espace entre chaque nom de groupe.

Faites très attention en utilisant `usermod` ! Lorsque vous avez recours à `-G`, l'utilisateur change de groupe et ce peu importe les groupes auxquels il appartenait auparavant.
Si vous voulez **ajouter** des groupes à un utilisateur (sans perdre les groupes auxquels il appartenait avant cela), utilisez `-a` :
`usermod -aG amis patrick`

`delgroup` : supprimer un groupe

Si vous voulez supprimer un groupe, c'est tout simple :

```
delgroup amis
```

`addgroup` et `delgroup` n'existent que sous Debian et ses dérivés (même remarque que pour `adduser` et `deluser`).
Les commandes « traditionnelles » qui fonctionnent partout sont `groupadd` et `groupdel` , mais elles offrent moins d'options.

chown : : gestion des propriétaires d'un fichier

Seul l'utilisateur root peut changer le propriétaire d'un fichier.
Supposons par exemple que mateo21 possède dans son répertoire personnel un fichier appelé `rapport.txt` .

Voici le résultat d'un `ls -l` pour ce fichier :

```
mateo21@mateo21-desktop:~$ ls -l rapport.txt
-rw-r--r-- 1 mateo21 mateo21 0 2007-11-15 23:14 rapport.txt
```

Petite astuce : comme vous venez de le voir, si on précise un nom de fichier en dernier paramètre de la commande `ls` , on ne verra que ce fichier dans les résultats.
Le joker `*` est là aussi utilisable : `ls -l *.jpg` afficherait uniquement les images JPEG contenues dans ce dossier.

Ce fichier, je souhaite le « donner » à patrick. C'est là qu'intervient la commande `chown` .

`chown` : changer le propriétaire d'un fichier

La commande `chown` , qui doit être utilisée **en tant que root**, attend deux paramètres au moins :

- le nom du nouveau propriétaire ;
- le nom du fichier à modifier.

Cela donne donc :

```
chown patrick rapport.txt
```

On peut voir ensuite que patrick est bien le nouveau propriétaire du fichier :

```
root@mateo21-desktop:/home/mateo21# ls -l rapport.txt
-rw-r--r-- 1 patrick mateo21 0 2007-11-15 23:14 rapport.txt
```

Seulement... il appartient toujours au groupe mateo21 !

`chgrp` : changer le groupe propriétaire d'un fichier

`chgrp` s'utilise exactement de la même manière que `chown` à la différence près qu'il affecte cette fois le

groupe propriétaire d'un fichier.

```
chgrp amis rapport.txt
```

Cette commande affectera le fichier `rapport.txt` au groupe `amis` .

Un petit `ls -l` nous confirmera que `rapport.txt` appartient désormais à patrick et au groupe `amis` :

```
root@mateo21-desktop:/home/mateo21# ls -l rapport.txt
-rw-r--r-- 1 patrick amis 0 2007-11-15 23:14 rapport.txt
```

`chown` peut aussi changer le groupe propriétaire d'un fichier !

Eh oui ! C'est d'ailleurs l'astuce que j'utilise le plus souvent :

```
chown patrick:amis rapport.txt
```

Cela affectera le fichier à l'utilisateur patrick et au groupe `amis` .
Il suffit de séparer par un symbole deux points (« : ») le nom du nouvel utilisateur (à gauche) et le nom du nouveau groupe (à droite).

`-R` : affecter récursivement les sous-dossiers

Très utile aussi, l'option `-R` de `chown` . Elle modifie tous les sous-dossiers et fichiers contenus dans un dossier pour y affecter un nouvel utilisateur (et un nouveau groupe si on utilise la technique du deux points que l'on vient de voir).

Par exemple, si je suis sadique et que je veux donner tout le contenu du dossier personnel de patrick à mateo21 (et au groupe mateo21), c'est très simple :

```
chown -R mateo21:mateo21 /home/patrick/
```

Résultat :

```
root@mateo21-desktop:/home# ls -l
total 24
drwx------  2 root    root    16384 2007-09-19 18:22 lost+found
drwxr-xr-x 62 mateo21 mateo21  4096 2007-11-15 23:19 mateo21
drwxr-xr-x  2 mateo21 mateo21  4096 2007-11-15 23:00 patrick
```

Désormais tous les fichiers à l'intérieur du dossier de patrick appartiennent à mateo21 (je sais, je suis vraiment trop diabolique).

chmod : modifier les droits d'accès

On attaque maintenant la partie la plus « coton » du chapitre si je puis dire : les droits d'accès.

Le fonctionnement des droits

Chaque fichier et chaque dossier possède une liste de droits. C'est une liste qui indique qui a le droit de voir le fichier, de le modifier et de l'exécuter.

Vous avez déjà vu des listes de droits, oui oui ! Lorsque vous faites un `ls -l`, il s'agit de la première colonne :

```
mateo21@mateo21-desktop:~$ ls -l
total 40
drwxr-xr-x 2 mateo21 mateo21 4096 2007-11-13 21:53 Desktop
drwxr-xr-x 2 mateo21 mateo21 4096 2007-11-13 13:46 Documents
lrwxrwxrwx 1 mateo21 mateo21   26 2007-09-19 18:31 Examples -> /usr/share/example-content
drwxr-xr-x 2 mateo21 mateo21 4096 2007-09-25 20:28 images
drwxr-xr-x 2 mateo21 mateo21 4096 2007-10-19 01:21 Images
drwxr-xr-x 3 mateo21 mateo21 4096 2007-09-25 11:11 log
drwxr-xr-x 2 mateo21 mateo21 4096 2007-10-19 01:21 Modèles
drwxr-xr-x 2 mateo21 mateo21 4096 2007-10-19 01:21 Musique
drwxr-xr-x 2 mateo21 mateo21 4096 2007-10-19 01:21 Public
-rw-r--r-- 1 mateo21 mateo21    0 2007-11-15 23:14 rapport.txt
drwxr-xr-x 3 mateo21 mateo21 4096 2007-09-19 19:51 tutos
drwxr-xr-x 2 mateo21 mateo21 4096 2007-10-19 01:21 Vidéos
```

Vous voyez tous ces `d`, `r`, `w` et `x` au début ? Ce sont ce qu'on appelle les droits d'accès du fichier ou dossier.

On peut voir cinq lettres différentes. Voici leur signification :

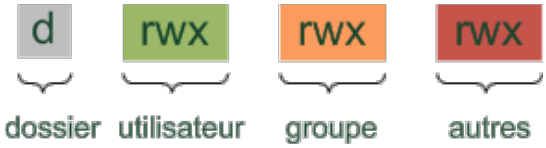
- **d** (Directory) : indique si l'élément est un dossier ;
- **l** (Link) : indique si l'élément est un lien (raccourci) ;
- **r** (Read) : indique si on peut lire l'élément ;
- **w** (Write) : indique si on peut modifier l'élément ;
- **x** (eXecute) : si c'est un fichier, « `x` » indique qu'on peut l'exécuter. Ce n'est utile que pour les fichiers exécutables (programmes et scripts).

  Si c'est un dossier, « `x` » indique qu'on peut le « traverser », c'est-à-dire qu'on peut voir les sous-dossiers qu'il contient si on a le droit de lecture dessus.

Si la lettre apparaît, c'est que le droit existe. S'il y a un tiret à la place, c'est qu'il n'y a aucun droit.

Pourquoi est-ce qu'on voit parfois `r`, `w` et `x` à plusieurs reprises ?

Les droits sont découpés en fonction des utilisateurs (figure suivante).



Le premier élément `d` mis à part, on constate que `r`, `w` et `x` sont répétés trois fois en fonction des utilisateurs :

- le premier triplet `rwx` indique les droits que possède le **propriétaire** du fichier sur ce dernier ;
- le second triplet `rwx` indique les droits que possèdent les autres membres du **groupe** sur ce fichier ;
- enfin, le dernier triplet `rwx` indique les droits que possèdent tous les **autres** utilisateurs de la machine sur le fichier.

Prenons un cas concret, le fichier `rapport.txt` :

```
mateo21@mateo21-desktop:~$ ls -l rapport.txt
-rw-r--r-- 1 mateo21 mateo21 0 2007-11-15 23:14 rapport.txt
```

Ses droits sont : `-rw-r--r--`

- **-** : le premier tiret indique qu'il ne s'agit pas d'un dossier. S'il y avait eu un `d` à la place, cela aurait indiqué qu'il s'agissait d'un dossier.
- **rw-** : indique que le propriétaire du fichier, mateo21 en l'occurrence, peut lire et modifier (et donc supprimer) le fichier. En revanche, il ne peut pas l'exécuter car il n'a pas de `x` à la fin. Je rappelle que quiconque peut modifier un fichier a aussi le droit de le supprimer.
- **r--** : tous les utilisateurs qui font partie du groupe mateo21 mais qui ne sont pas mateo21 peuvent seulement lire le fichier. Ils ne peuvent ni le modifier, ni l'exécuter. Je reconnais qu'avoir un nom de groupe identique au nom d'utilisateur peut embrouiller : si vous êtes aussi bien organisés que sur mon premier schéma, on parlera plutôt du groupe famille.
- **r--** : tous les autres (ceux qui ne font pas partie du groupe mateo21) peuvent seulement lire le fichier.

En résumé, ces droits nous apprennent que l'élément est un fichier, que mateo21 peut le lire et le modifier et que tous les autres utilisateurs peuvent seulement le lire.

Et root ?
Il a quels droits ?

Souvenez-vous d'une chose : root a TOUS les droits. Il peut tout faire : lire, modifier, exécuter n'importe quel fichier.

`chmod` : modifier les droits d'accès

Maintenant que nous savons voir et comprendre les droits d'accès d'un fichier, nous allons apprendre à les modifier à l'aide de la commande `chmod` .

Une précision importante pour commencer : contrairement aux commandes précédentes, vous n'avez pas besoin d'être root pour utiliser `chmod`. Vous devez juste être propriétaires du fichier dont vous voulez modifier les droits d'accès.

`chmod` est un petit peu délicat à utiliser. En effet, on peut attribuer les droits sur un fichier / dossier via plusieurs méthodes différentes, la plus courante étant celle des chiffres.

Attribuer des droits avec des chiffres (`chmod` absolu)

J'espère que vous êtes prêts pour effectuer quelques additions !
Il va falloir faire un petit peu de calcul mental. En effet, on attribue un chiffre à chaque droit :

Droit	Chiffre
r	4
w	2
x	1

Si vous voulez combiner ces droits, il va falloir additionner les chiffres correspondants.
Ainsi, pour attribuer le droit de lecture et de modification, il faut additionner $4+2$, ce qui donne 6. Le chiffre 6 signifie donc « Droit de lecture et d'écriture ».

Voici la liste des droits possibles et la valeur correspondante :

Droits	Chiffre	Calcul
---	0	0 + 0 + 0
r--	4	4 + 0 + 0
-w-	2	0 + 2 + 0
--x	1	0 + 0 + 1
rw-	6	4 + 2 + 0
-wx	3	0 + 2 + 1
r-x	5	4 + 0 + 1
rwx	7	4 + 2 + 1

C'est compris ?
Avec ça, on peut calculer la valeur d'un triplet de droits. Il faut faire le même calcul pour les droits que l'on

veut attribuer au propriétaire, au groupe et aux autres.

Par exemple, « 640 » indique les droits du propriétaire, du groupe et des autres (dans l'ordre).

- 6 : droit de lecture et d'écriture pour le propriétaire.
- 4 : droit de lecture pour le groupe.
- 0 : aucun droit pour les autres.

Le droit maximal que l'on puisse donner à tout le monde est 777 : droit de lecture, d'écriture et d'exécution pour le propriétaire, pour son groupe et pour tous les autres. Bref, avec un tel droit tout le monde peut tout faire sur ce fichier.
Au contraire, avec un droit de 000, personne ne peut rien faire... à part root, bien sûr.

Pour changer les droits sur le fichier `rapport.txt`, et être le seul autorisé à le lire et l'éditer, je dois exécuter cette commande :

```
chmod 600 rapport.txt
```

Un petit `ls -l` pour voir le résultat :

```
mateo21@mateo21-desktop:~$ ls -l rapport.txt
-rw------- 1 mateo21 mateo21 0 2007-11-15 23:14 rapport.txt
```

Bingo !
On a bien confirmation que seul le propriétaire du fichier, c'est-à-dire moi, peut le lire et le modifier !

Attribuer des droits avec des lettres (`chmod` relatif)

Il existe un autre moyen de modifier les droits d'un fichier. Il revient un peu au même mais permet parfois de paramétrer plus finement, droit par droit.
Dans ce mode, il faut savoir que :

- **u** = user (propriétaire) ;
- **g** = group (groupe) ;
- **o** = other (autres).

... et que :

- **+** signifie : « Ajouter le droit » ;
- **-** signifie : « Supprimer le droit » ;
- **=** signifie : « Affecter le droit ».

Maintenant que vous savez cela, vous pouvez écrire :

```
chmod g+w rapport.txt
```

Signification : « Ajouter le droit d'écriture au groupe ».

```
chmod o-r rapport.txt
```

Signification : « Enlever le droit de lecture aux autres ».

```
chmod u+rx rapport.txt
```

Signification : « Ajouter les droits de lecture et d'exécution au propriétaire ».

```
chmod g+w,o-w rapport.txt
```

Signification : « Ajouter le droit d'écriture au groupe et l'enlever aux autres ».

```
chmod go-r rapport.txt
```

Signification : « Enlever le droit de lecture au groupe et aux autres ».

```
chmod +x rapport.txt
```

Signification : « Ajouter le droit d'exécution à tout le monde ».

```
chmod u=rwx,g=r,o=- rapport.txt
```

Signification : « Affecter tous les droits au propriétaire, juste la lecture au groupe, rien aux autres ».

Voilà, ouf ! J'ai préféré vous expliquer le fonctionnement à travers des exemples concrets plutôt que de faire un cours théorique sur la syntaxe d'une des utilisations possibles de `chmod` .
Normalement si vous suivez mes exemples vous devriez être capables de tout faire !

Et toujours... `-R` pour affecter récursivement

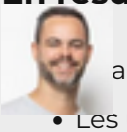
Le paramètre `-R` existe aussi pour `chmod` . Si vous affectez des droits sur un dossier avec `-R` , tous ses fichiers et sous-dossiers récupèreront le même droit.

Si je veux être le seul à pouvoir lire, éditer et exécuter les fichiers de mon répertoire personnel et de tous ses fichiers, j'ai juste besoin d'écrire :

```
chmod -R 700 /home/mateo21
```

C'est tout !

Le professeur



Mathieu Nebra

Entrepreneur à plein temps, auteur à plein temps et co-fondateur d'OpenClassrooms :o)

En résumé

- Chaque personne qui utilise une machine Linux possède un compte utilisateur.
- Les utilisateurs sont classés par groupes.
- Il existe un superutilisateur qui a tous les droits : root. C'est l'administrateur de la machine, le seul à être autorisé à installer des programmes ou effectuer certaines modifications sur le système.
- Certaines commandes ne fonctionnent que lorsqu'on est root et nécessitent donc de se transformer

Découvrez aussi ce cours en...

- Livre PDF
- ro ide de `sudo` . C'est le cas de la commande d'ajout d'utilisateur (`adduser`), de appr n d'utilisateur (`deluser`) ou encore de changement de propriétaire d'un fichier (`chown`).
- On peut modifier les droits d'accès à un fichier avec `chmod` . Il existe trois types de droits : `r` (droit de lecture), `w` (droit d'écriture) et `x` (droit d'exécution).

J'AI TERMINÉ CE CHAPITRE ET JE PASSE AU SUIVANT

OPENCLASSROOMS

Qui sommes-nous ?

Financements

Expérience de formation

Forum

Blog

Presse

OPPORTUNITÉS

Nous rejoindre

Devenir mentor

Devenir coach carrière

AIDE

FAQ

POUR LES ENTREPRISES

Former et recruter

EN PLUS

Boutique

Mentions légales

Conditions générales d'utilisation

MANIPULER LES FICHIERS

NANO, L'ÉDITEUR DE TEXTE DU DÉBUTANT