

Version : **2026.01**

Dernière mise-à-jour : 2025/12/02 15:12

# LDF401 - Gestion des Droits

## Contenu du Module

- **LDF401 - Gestion des Droits**

- Contenu du Module
- Présentation
- Préparation
- LAB #1 - Les Droits Unix Simples
  - 1.1 - La Modification des Droits
    - La Commande chmod
      - Mode Symbolique
      - Mode Octal
    - La Commande umask
  - 1.2 - Modifier le propriétaire ou le groupe
    - La Commande chown
    - La Commande chgrp
- LAB #2 - Les Droits Unix Étendus
  - 2.1 - SUID/SGID bit
  - 2.2 - Inheritance Flag
  - 2.3 - Sticky bit
- LAB #3 - Les Droits Unix Avancés
  - 3.1 - Les ACL
  - 3.2 - Les Attributs Étendus

## Présentation

Dans sa conception de base, Linux utilise une approche sécurité de type **DAC**. Cette approche est maintenue dans la mise en place et l'utilisation des **ACL** et les **Attributs Etendus Ext2/Ext3/Ext4, JFS, ReiserFS, XFS et Btrfs** :

Type de Sécurité	Nom	Description
DAC	<i>Discretionary Access Control</i>	L'accès aux objets est en fonction de l'identité (utilisateur,groupe). Un utilisateur peut rendre accessible aux autres ses propres objets.

## Préparation

Dans votre répertoire personnel, créez un fichier tux.jpg grâce à la commande **touch**:

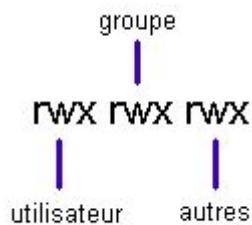
```
trainee@debian12:~$ touch tux.jpg
trainee@debian12:~$ ls -l | grep tux.jpg
-rw-r--r-- 1 trainee trainee 0 Nov 25 12:26 tux.jpg
```



**Important** : Notez que le fichier créé est un fichier **texte**. En effet, Linux ne tient pas compte de l'extension **.jpg**

## LAB #1 - Les Droits Unix Simples

Les autorisations ou droits d'accès en Linux sont communiqués comme suit :



ou r = lecture, w = écriture et x = exécutable

Dans chaque inode est stocké le numéro de l'utilisateur à qui appartient le fichier concerné ainsi que le numéro du groupe. Quand le fichier est ouvert le système compare le numéro de l'utilisateur (UID) avec le numéro de l'utilisateur stocké dans l'inode ( Utilisateur de Référence ). Si ces deux numéros sont identiques, l'utilisateur obtient les droits du propriétaire du fichier. Si les numéros diffèrent, le système vérifie si l'utilisateur est dans le groupe référencé dans l'inode. Si oui, l'utilisateur aura les droits spécifiés pour le groupe. Si aucune condition n'est remplie, l'utilisateur se voit attribuer les droits des «autres».

Les droits pour les répertoires sont légèrement différents :

<b>r</b>	Les éléments du répertoire sont accessibles en lecture ( lister )
<b>w</b>	Les éléments du répertoire sont modifiables ( création et suppression ).
<b>x</b>	Le nom du répertoire peut apparaître dans un chemin d'accès.

## 1.1 - La Modification des Droits

### La Commande chmod

#### Mode Symbolique

Afin de modifier les droits d'accès aux fichiers, on utilise la commande chmod dont la syntaxe est le suivant :

chmod [ -R ] catégorie opérateur permissions nom\_fichier

ou

```
chmod [ -R ] ugoa +-= rwxXst nom_du_fichier
```

où

<b>u</b>	user
<b>g</b>	group
<b>o</b>	other
<b>a</b>	all
<b>+</b>	autorise un accès
<b>-</b>	interdit un accès
<b>=</b>	autorise exclusivement l'accès indiqué
<b>r</b>	read
<b>w</b>	write
<b>x</b>	execute
<b>X</b>	exécution si la cible est un répertoire ou si c'est un fichier est déjà exécutable pour une des <i>catégories</i> (ugo)
<b>s</b>	SUID/SGID bit
<b>t</b>	sticky bit

par exemple la commande suivante donnera aux autres l'accès en écriture sur le fichier tux.jpg :

```
trainee@debian12:~$ chmod o+w tux.jpg

trainee@debian12:~$ ls -l | grep tux.jpg
-rw-r--rw- 1 trainee trainee 0 Nov 25 12:26 tux.jpg
```

tandis que la commande suivante ôtera les droit d'accès en écriture pour l'utilisateur et le groupe :

```
trainee@debian12:~$ chmod ug-w tux.jpg

trainee@debian12:~$ ls -l | grep tux.jpg
-r--r--rw- 1 trainee trainee 0 Nov 25 12:26 tux.jpg
```



**Important** : Seul le propriétaire du fichier ou root peuvent modifier les permissions.

## Mode Octal

La commande chmod peut également être utilisée avec une représentation octale ( base de 8 ). Les valeurs octales des droits d'accès sont :

r	w	x	r	w	x	r	w	x
400	200	100	40	20	10	4	2	1
<hr/> Utilisateur			<hr/> Group			<hr/> Other		



**Important** : Ainsi les droits rwx rwx rwx correspondent à un chiffre de 777.

La commande chmod prend donc la forme suivante:

```
chmod [ -R ] mode_octal nom_fichier
```

La commande suivante correspond donc à l'attribution des droits : rw- r- r- :

```
trainee@debian12:~$ chmod 644 tux.jpg
trainee@debian12:~$ ls -l | grep tux.jpg
-rw-r--r-- 1 trainee trainee 0 Nov 25 12:26 tux.jpg
```



**Important** : Les droits d'accès par défaut lors de la création d'un objet sont :



Répertoires	rwx rwx rwx	777
Fichier normal	rw- rw- rw-	666

Les options de cette commande sont :

```
trainee@debian12:~$ chmod --help
Usage: chmod [OPTION]... MODE[,MODE]... FILE...
      or: chmod [OPTION]... OCTAL-MODE FILE...
      or: chmod [OPTION]... --reference=FILE FILE...
Change the mode of each FILE to MODE.
With --reference, change the mode of each FILE to that of FILE.

-c, --changes          like verbose but report only when a change is made
-f, --silent, --quiet  suppress most error messages
-v, --verbose          output a diagnostic for every file processed
--no-preserve-root    do not treat '/' specially (the default)
--preserve-root       fail to operate recursively on '/'
--reference=FILE      use FILE's mode instead of MODE values
-R, --recursive       change files and directories recursively
--help                display this help and exit
--version             output version information and exit
```

Each MODE is of the form '[ugoa]\*([-=]([rwxXst]\*|[ugol]))+|[-=][0-7]+'.

GNU coreutils online help: <<https://www.gnu.org/software/coreutils/>>

Full documentation <<https://www.gnu.org/software/coreutils/chmod>>

or available locally via: info '(coreutils) chmod invocation'

## La Commande umask

L'utilisateur peut changer sa masque de permissions défaut lors de la création d'objets en utilisant la commande umask.

La valeur par défaut de l'umask sous RHEL/CentOS est différente pour un utilisateur normal et pour root :

```
trainee@debian12:~$ umask
0022

trainee@debian12:~$ su -
Password: fenestros

root@debian12:~# umask
0022

root@debian12:~# exit
logout
```

Par exemple dans le cas où l'utilisateur souhaite que les fichiers créés dans le futur comportent des droits d'écriture et de lecture pour l'utilisateur mais uniquement des droits de lecture pour le groupe et pour les autres, il utiliserait la commande :

```
$ umask 022 [Entrée]
```

avant de créer son fichier.

umask sert à enlever des droits des droits maximaux :



Masque maximum lors de la création d'un fichier	rw- rw- rw-	666
Droits à retirer	— -w- -w-	022
Résultat	rw- r- r-	644

Dans l'exemple qui suit, on utilise la commande touch pour créer un fichier vide ayant les nouveaux droits par défaut :

```
trainee@debian12:~$ umask 044
```

```
trainee@debian12:~$ touch tux1.jpg

trainee@debian12:~$ ls -l | grep tux1.jpg
-rw--w--w- 1 trainee trainee 0 Nov 25 15:01 tux1.jpg

trainee@debian12:~$ umask 022

trainee@debian12:~$ umask
0022
```

Les options de cette commande sont :

```
trainee@debian12:~$ help umask
umask: umask [-p] [-S] [mode]
Display or set file mode mask.
Sets the user file-creation mask to MODE. If MODE is omitted, prints
the current value of the mask.
If MODE begins with a digit, it is interpreted as an octal number;
otherwise it is a symbolic mode string like that accepted by chmod(1).
Options:
  -p      if MODE is omitted, output in a form that may be reused as input
  -S      makes the output symbolic; otherwise an octal number is output
Exit Status:
  Returns success unless MODE is invalid or an invalid option is given.
```

## 1.2 - Modifier le propriétaire ou le groupe



**Important** - Le changement de propriétaire d'un fichier se fait uniquement par l'administrateur système - root.

## La Commande chown

Dans le cas du fichier tux.jpg appartenant à trainee, root peut changer le propriétaire de trainee à root avec la commande suivante :

```
trainee@debian12:~$ su -
Password: fenestros

root@debian12:~# cd /home/trainee

root@debian12:/home/trainee# chown root tux.jpg

root@debian12:/home/trainee# ls -l | grep tux.jpg
-rw-r--r-- 1 root      trainee    0 Nov 25 12:26 tux.jpg
```

Les options de cette commande sont :

```
root@debian12:/home/trainee# chown --help
Usage: chown [OPTION]... [OWNER][:[GROUP]] FILE...
      or: chown [OPTION]... --reference=RFILE FILE...
Change the owner and/or group of each FILE to OWNER and/or GROUP.
With --reference, change the owner and group of each FILE to those of RFILE.

-c, --changes      like verbose but report only when a change is made
-f, --silent, --quiet suppress most error messages
-v, --verbose      output a diagnostic for every file processed
      --dereference  affect the referent of each symbolic link (this is
                     the default), rather than the symbolic link itself
-h, --no-dereference  affect symbolic links instead of any referenced file
                     (useful only on systems that can change the
                     ownership of a symlink)
--from=CURRENT_OWNER:CURRENT_GROUP
                     change the owner and/or group of each file only if
                     its current owner and/or group match those specified
```

```
here. Either may be omitted, in which case a match
is not required for the omitted attribute
--no-preserve-root do not treat '/' specially (the default)
--preserve-root    fail to operate recursively on '/'
--reference=RFILE use RFILE's owner and group rather than
                     specifying OWNER:GROUP values
-R, --recursive    operate on files and directories recursively
```

The following options modify how a hierarchy is traversed when the -R option is also specified. If more than one is specified, only the final one takes effect.

```
-H      if a command line argument is a symbolic link
       to a directory, traverse it
-L      traverse every symbolic link to a directory
       encountered
-P      do not traverse any symbolic links (default)

--help   display this help and exit
--version output version information and exit
```

Owner is unchanged if missing. Group is unchanged if missing, but changed to login group if implied by a ':' following a symbolic OWNER. OWNER and GROUP may be numeric as well as symbolic.

Examples:

```
chown root /u          Change the owner of /u to "root".
chown root:staff /u    Likewise, but also change its group to "staff".
chown -hR root /u      Change the owner of /u and subfiles to "root".
```

GNU coreutils online help: <<https://www.gnu.org/software/coreutils/>>
Full documentation <<https://www.gnu.org/software/coreutils/chown>>
or available locally via: info '(coreutils) chown invocation'

## La Commande chgrp

Le même cas de figure s'applique au groupe :

```
root@debian12:/home/trainee# chgrp root tux.jpg
root@debian12:/home/trainee# ls -l | grep tux.jpg
-rw-r--r-- 1 root      root      0 Nov 25 12:26 tux.jpg
```



**Important :** Le droit de supprimer un fichier dépend des droits sur le répertoire dans lequel le fichier est stocké et non des droits du fichier lui-même.

Les options de cette commande sont :

```
root@debian12:/home/trainee# chgrp --help
Usage: chgrp [OPTION]... GROUP FILE...
      or: chgrp [OPTION]... --reference=RFILE FILE...
Change the group of each FILE to GROUP.
With --reference, change the group of each FILE to that of RFILE.

-c, --changes      like verbose but report only when a change is made
-f, --silent, --quiet suppress most error messages
-v, --verbose      output a diagnostic for every file processed
      --dereference  affect the referent of each symbolic link (this is
                     the default), rather than the symbolic link itself
-h, --no-dereference  affect symbolic links instead of any referenced file
                     (useful only on systems that can change the
                     ownership of a symlink)
--no-preserve-root  do not treat '/' specially (the default)
--preserve-root     fail to operate recursively on '/'
```

```
--reference=RFILE  use RFILE's group rather than specifying a
                  GROUP value
-R, --recursive      operate on files and directories recursively
```

The following options modify how a hierarchy is traversed when the -R option is also specified. If more than one is specified, only the final one takes effect.

```
-H      if a command line argument is a symbolic link
          to a directory, traverse it
-L      traverse every symbolic link to a directory
          encountered
-P      do not traverse any symbolic links (default)

--help    display this help and exit
--version  output version information and exit
```

Examples:

```
chgrp staff /u      Change the group of /u to "staff".
chgrp -hR staff /u  Change the group of /u and subfiles to "staff".
```

GNU coreutils online help: <<https://www.gnu.org/software/coreutils/>>
Full documentation <<https://www.gnu.org/software/coreutils/chgrp>>
or available locally via: info '(coreutils) chgrp invocation'

## LAB #2 - Les Droits Unix Etendus

### 2.1 - SUID/SGID bit

Malgré ce que vous venez de voir, dans la première des deux fenêtres ci-dessous, vous noterez que le fichier **passwd** se trouvant dans le répertoire **/etc** possède les permissions **rw- r- r-** et qu'il appartient à **root**. Autrement dit **seul** root peut écrire dans ce fichier. Or, quand un utilisateur normal change son mot de passe, il écrit dans ce fichier. Ceci semble donc être une contradiction.

```
root@debian12:/home/trainee# ls -l /etc/shadow /usr/bin/passwd
-rw-r----- 1 root shadow 1160 Nov 24 17:28 /etc/shadow
-rwsr-xr-x 1 root root  68248 Apr  7  2025 /usr/bin/passwd
```

Pour remédier à cette apparence contradiction, Linux dispose de deux droits d'accès étendus :

- Set UserID bit ( SUID bit )
- Set GroupID bit ( SGID bit )

Quand le SUID bit est placé sur un programme, l'utilisateur qui lance ce programme se voit affecté le numéro d'utilisateur du propriétaire de ce programme et ce pour la durée de son exécution.

Dans le cas du changement de mot de passe, chaque utilisateur qui lance le programme /usr/bin/passwd se trouve temporairement avec le numéro d'utilisateur du propriétaire du programme /usr/bin/passwd, c'est à dire root. De cette façon, l'utilisateur peut intervenir sur le fichier /etc/shadow. Ce droit est indiqué par la lettre s à la place de la lettre x.

La même fonction existe pour le groupe à l'aide du SGID bit.

Pour assigner les droits, vous utiliserez la commande chmod :

- chmod u+s nom\_du\_fichier
- chmod g+s nom\_du\_fichier

En base huit les valeurs sont les suivants :

- SUID = 4000
- SGID = 2000

Afin d'identifier les exécutables ayant le SGID ou SUID bit, utilisez la commande suivante :

```
root@debian12:/home/trainee# find / -type f \( -perm -4000 -o -perm -2000 \) -exec ls {} \;
find: '/proc/1675/task/1675/fdinfo/6': No such file or directory
find: '/proc/1675/fdinfo/5': No such file or directory
/usr/bin/su
/usr/bin/pkexec
```

```
/usr/bin/dotlockfile
/usr/bin/fusermount3
/usr/bin/expiry
/usr/bin/crontab
/usr/bin/chsh
/usr/bin/mount
/usr/bin/sudo
/usr/bin/chage
/usr/bin/umount
/usr/bin/chfn
/usr/bin/passwd
/usr/bin/gpasswd
/usr/bin/ssh-agent
/usr/bin/newgrp
/usr/bin/ntfs-3g
/usr/lib/x86_64-linux-gnu/utempter/utempter
/usr/lib/virtualbox/VBoxNetNAT
/usr/lib/virtualbox/VBoxVolInfo
/usr/lib/virtualbox/VBoxNetAdpCtl
/usr/lib/virtualbox/VirtualBoxVM
/usr/lib/virtualbox/VBoxHeadless
/usr/lib/virtualbox/VBoxNetDHCP
/usr/lib/virtualbox/VBoxSDL
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
/usr/lib/polkit-1/polkit-agent-helper-1
/usr/lib/xorg/Xorg.wrap
/usr/sbin/unix_chkpwd
/usr/sbin/pppd
```

## 2.2 - Inheritance Flag

Le SGID bit peut également être affecté à un répertoire. De cette façon, les fichiers et répertoires créés à l'intérieur auront comme groupe le groupe du

répertoire parent. Ce droit s'appelle donc l'**Inheritance Flag** ou le **Drapeau d'Héritage**.

Par exemple :

```
root@debian12:/home/trainee# cd /tmp
root@debian12:/tmp# mkdir inherit
root@debian12:/tmp# chown root:trainee inherit
root@debian12:/tmp# chmod g+s inherit
root@debian12:/tmp# touch inherit/test.txt
root@debian12:/tmp# mkdir inherit/testrep
root@debian12:/tmp# cd inherit; ls -l
total 4
drwxr-sr-x 2 root trainee 4096 Nov 25 15:09 testrep
-rw-r--r-- 1 root trainee    0 Nov 25 15:09 test.txt
```



**Important** : Notez que malgré le fait que root a créé les deux objets, ceux-ci ne sont pas associés avec le groupe **root** mais avec le groupe **trainee**, le groupe du répertoire parent (inherit). Notez aussi que le système a posé le drapeau d'héritage sur le sous-répertoire **testrep**.

## 2.3 - Sticky bit

Il existe un dernier cas qui s'appelle le sticky bit. Le sticky bit est utilisé pour des répertoires ou tout le monde a tous les droits. Dans ce cas, tout le monde peut supprimer des fichiers dans le répertoire. En ajoutant le sticky bit, uniquement le propriétaire du fichier peut le supprimer.

```
# chmod o+t /répertoire
```

ou

```
# chmod 1777 /répertoire
```

Par exemple :

```
root@debian12:/tmp/inherit# mkdir /tmp/repertoire_public; cd /tmp; chmod o+t repertoire_public
root@debian12:/tmp# ls -l | grep repertoire_public
drwxr-xr-t 2 root root 4096 Nov 25 15:10 repertoire_public
```

## LAB #3 - Les Droits Unix Avancés

### 3.1 - Les ACL

Au delà des droits étendus d'Unix, Linux utilise un système d'ACL pour permettre une meilleure gestion des droits sur des fichiers.

Pour connaître les ACL positionnés sur un fichier, il convient d'utiliser la commande **getfacl** :

```
root@debian12:/tmp# getfacl /home/trainee/tux.jpg
getfacl: Removing leading '/' from absolute path names
# file: home/trainee/tux.jpg
# owner: root
# group: root
user::rw-
group::r--
other::r--
```

Pour positionner des ACL sur un fichier, il convient d'utiliser la commande **setfacl** :

```
root@debian12:/tmp# setfacl --set u::rwx,g::rx,o::-,u:trainee:rw /home/trainee/tux.jpg
```

```
root@debian12:/tmp# getfacl /home/trainee/tux.jpg
getfacl: Removing leading '/' from absolute path names
# file: home/trainee/tux.jpg
# owner: root
# group: root
user::rwx
user:trainee:rwx
group::r-x
mask::rwx
other::---
```



**Important** - Veuillez noter l'apparition de la ligne **mask**. Le mask indique les permissions maximales qui peuvent être accordées à un utilisateur ou un groupe tiers.

Regardez maintenant l'effet des ACL sur un répertoire. Créez le répertoire /home/trainee/rep1 :

```
root@debian12:/tmp# mkdir /home/trainee/rep1
```

Positionnez des ACL le répertoire avec la commande **setfacl** :

```
root@debian12:/tmp# setfacl --set d:u::r,d:g::-,d:o::- /home/trainee/rep1
```

Notez l'utilisation de la lettre **d** pour indiquer une permission *par défaut*.

Créez maintenant un fichier appelé fichier1 dans /home/trainee/rep1 :

```
root@debian12:/tmp# touch /home/trainee/rep1/fichier1
```

Utilisez la commande **getfacl** pour visualiser le résultat :

```
root@debian12:/tmp# getfacl /home/trainee/rep1
getfacl: Removing leading '/' from absolute path names
# file: home/trainee/rep1
# owner: root
# group: root
user::rwx
group::r-x
other::r-x
default:user::r--
default:group::---
default:other::---
```

```
root@debian12:/tmp# getfacl /home/trainee/rep1/fichier1
getfacl: Removing leading '/' from absolute path names
# file: home/trainee/rep1/fichier1
# owner: root
# group: root
user::r--
group::---
other::---
```

Notez que le fichier créé possède les ACL positionnés sur le répertoire rep1.

Dernièrement, les systèmes de sauvegarde classiques sous Linux ne peuvent pas sauvegarder les ACL, sauf l'outil **star**. Si vous n'utilisez pas **star**, il convient donc de sauvegarder les ACL dans un fichier grâce à la commande suivante :

```
root@debian12:/tmp# cd /home/trainee/rep1

root@debian12:/home/trainee/rep1# getfacl -R --skip-base . > backup.acl

root@debian12:/home/trainee/rep1# cat backup.acl
# file: .
# owner: root
# group: root
```

```
user::rwx
group::r-x
other::r-x
default:user::r--
default:group::---
default:other::---
```

La restauration des ACL se fait avec la commande **setfacl** :

```
# setfacl --restore=backup.acl [Entrée]
```

Les options de la commande **getfacl** sont :

```
root@debian12:/home/trainee/repl# getfacl --help
getfacl 2.3.1 -- get file access control lists
Usage: getfacl [-aceEsRLPtpndvh] file ...
  -a, --access          display the file access control list only
  -d, --default         display the default access control list only
  -c, --omit-header    do not display the comment header
  -e, --all-effective   print all effective rights
  -E, --no-effective    print no effective rights
  -s, --skip-base       skip files that only have the base entries
  -R, --recursive       recurse into subdirectories
  -L, --logical         logical walk, follow symbolic links
  -P, --physical        physical walk, do not follow symbolic links
  -t, --tabular         use tabular output format
  -n, --numeric          print numeric user/group identifiers
  --one-file-system     skip files on different filesystems
  -p, --absolute-names  don't strip leading '/' in pathnames
  -v, --version          print version and exit
  -h, --help              this help text
```

Les options de la commande **setfacl** sont :

```
root@debian12:/home/trainee/repl# setfacl --help
setfacl 2.3.1 -- set file access control lists
Usage: setfacl [-bkndRLP] { -m|-M|-x|-X ... } file ...
  -m, --modify=acl      modify the current ACL(s) of file(s)
  -M, --modify-file=file read ACL entries to modify from file
  -x, --remove=acl      remove entries from the ACL(s) of file(s)
  -X, --remove-file=file read ACL entries to remove from file
  -b, --remove-all      remove all extended ACL entries
  -k, --remove-default  remove the default ACL
  --set=acl             set the ACL of file(s), replacing the current ACL
  --set-file=file       read ACL entries to set from file
  --mask                do recalculate the effective rights mask
  -n, --no-mask         don't recalculate the effective rights mask
  -d, --default         operations apply to the default ACL
  -R, --recursive       recurse into subdirectories
  -L, --logical         logical walk, follow symbolic links
  -P, --physical        physical walk, do not follow symbolic links
  --restore=file        restore ACLs (inverse of `getfacl -R`)
  --test                test mode (ACLs are not modified)
  -v, --version         print version and exit
  -h, --help             this help text
```

## 3.2 - Les Attributs Etendus

Les attributs s'ajoutent aux caractéristiques classiques d'un fichier dans un système de fichiers Ext2/Ext3/Ext4, JFS, ReiserFS, XFS et Btrfs.

Les principaux attributs sont :

Attribut	Description
a	Fichier journal - uniquement l'ajout de données au fichier est permis. Le fichier ne peut pas être détruit
i	Le fichier ne peut pas être modifié, ni être détruit, ni être déplacé. Le placement d'un lien sur le fichier n'est pas permis
s	Le fichier sera physiquement détruit lors de sa suppression

Attribut	Description
D	Répertoire synchrone
S	Fichier synchrone
A	La date et l'heure de dernier accès ne seront pas mises à jour



**Important** - Un fichier synchrone et un répertoire synchrone impliquent que les modifications seront immédiatement inscrites sur disque.

Les commandes associées avec les attributs sont :

Commande	description
chattr	Modifie les attributs
lsattr	Visualise les attributs

Pour mieux comprendre, créez le répertoire **/root/attributs/rep** :

```
root@debian12:/home/trainee/rep1# cd /root
```

```
root@debian12:~# mkdir -p attributs/rep
```

Créez ensuite les fichier **fichier** et **rep/fichier1** :

```
root@debian12:~# touch attributs/fichier
```

```
root@debian12:~# touch attributs/rep/fichier1
```

Modifiez les attributs d'une manière récursive sur le répertoire **attributs** :

```
root@debian12:~# chattr +i -R attributs/
```

Visualisez les attributs de l'arborescence **attributs** :

```
root@debian12:~# lsattr -R attributs
----i-----e----- attributs/rep

attributs/rep:
----i-----e----- attributs/rep/fichier1

----i-----e----- attributs/fichier
```



**Important** - Notez que l'attribut **e** sous Ext4 indique l'utilisation des **Extents**. Cet attribut ne peut pas être enlever avec la commande **chattr**. Les Extents seront couverts dans le cours **Gestion des Disques, des Systèmes de Fichiers et le Swap**.

Essayez maintenant de déplacer le fichier **fichier**. Vous obtiendrez un résultat similaire à celui-ci :

```
root@debian12:~# cd attributs; mv /root/attributs/fichier /root/attributs/rep/fichier
mv: cannot move '/root/attributs/fichier' to '/root/attributs/rep/fichier': Operation not permitted
```

Copyright © 2025 Hugh Norris.

From:  
<https://www.ittraining.team/> - **www.ittraining.team**

Permanent link:  
<https://www.ittraining.team/doku.php?id=elearning:workbooks:debian:11:sec:l101>

Last update: **2025/12/02 15:12**

