**Les permissions sur les fichiers et les répertoires**

Nous avons vu que l'option -l de la commande ls permet d'afficher au long les attributs des fichiers, dont les droits d'accès. Voyons en détails comment interpréter les droits d'accès.

Au début de chaque ligne générée par ls -l, nous retrouvons une suite de dix caractères.

Le premier caractère identifie le type du fichier.

Les types de fichier les plus communs

* - les fichiers ordinaires
* d les répertoires
* l les liens symboliques

Voici les autres types de fichier

* les fichiers spéciaux en mode bloc (b) – ex : un disque dur, CD rom
* les fichiers spéciaux en mode caractère (c)
* les pipe-lines (p)
* les prises réseau (s, pour socket)

Chaque partie de trois caractères correspond à une série de permissions pour le propriétaire, le groupe et les autres utilisateurs respectivement. La lettre "r" représente la permission de lecture (4 en octal), la lettre "w" représente la permission d'écriture (2 en octal), et la lettre "x" représente la permission d'exécution (1 en octal). Le tiret "-" indique l'absence de permission (0 en octal).

Donc, en convertissant "r--r-xrwx" en octal, nous obtenons :

* La première partie "r--" : 4 (lecture) + 0 (absence d'écriture) + 0 (absence d'exécution) = 4 en octal
* La deuxième partie "r--" : 4 (lecture) + 0 (absence d'écriture) + 0 (absence d'exécution) = 4 en octal
* La troisième partie "r-x" : 4 (lecture) + 0 (absence d'écriture) + 1 (exécution) = 5 en octal
* La quatrième partie "rwx" : 4 (lecture) + 2 (écriture) + 1 (exécution) = 7 en octal

Ainsi, la valeur octale correspondante à "r--r-xrwx" est 457. Veuillez noter que cette représentation est spécifique aux autorisations Linux et ne correspond pas directement à une valeur binaire.

**EXEMPLE**

**-rw-rw-r-- 1 taylor root 280232 Oct 10 16:22 info.txt**

info.txt est donc un fichier ordinaire (-) appartenant à taylor, au groupe root.

Taylor, le propriétaire, a accès en lecture, en écriture, mais pas en exécution (rw-).

Le groupe root a les mêmes droits que Taylor (rw-).

Les autres usagers, n'ont accès qu'en lecture seulement (r--).

Nous verrons que LINUX interprète les droits sur les répertoires de façon un peu différente que pour les fichiers.

**Les autorisations pour les fichiers**

**R**

* on peut lire le contenu d'un fichier

**W**

* on peut modifier le contenu d'un fichier

**X**

* on peut exécuter le contenu d'un fichier (script Shell)

**SUID (fichier exécutable seulement)**

* lors de l'exécution, le processus aura les privilèges du propriétaire du programme et non pas ceux de l'utilisateur
  + Il faut donc être soit un super-user, ou avoir les droits suffisant (SUID)

**exemples:**

/bin/ping

/bin/mount

**Avant d'attribuer l'autorisation SUID à un fichier exécutable vous devez réfléchir à l'impact.**

**SGID (fichier exécutable seulement)**

* lors de l'exécution, le processus aura les privilèges du groupe du programme et non pas ceux de l'utilisateur

**Avant d'attribuer l'autorisation SGID à un fichier exécutable vous devez réfléchir à l'impact.**

**Les autorisations pour les répertoires**

Les droits d'un répertoire ne concernent que le contenu directement sous lui.

**Pour comprendre les permissions sur un répertoire, il faut voir celui-ci comme un catalogue contenant le nom des fichiers et les inodes correspondants.**

**R**

* on peut lister le contenu du répertoire

**W**

* on peut ajouter/renommer/supprimer des fichiers à l’intérieur du répertoire

**X**

* on peut faire un cd dans le répertoire.

**SGID**

* un fichier créé à l’intérieur du répertoire aura le même groupe que le répertoire et non pas le groupe principal de son créateur

**exemple:**

/var/ftp/pub/

**Sticky (pour Linux, cela s’applique seulement pour les répertoires)**

* empêche les utilisateurs d'effacer ou de renommer des fichiers du dossier s'ils ne sont pas les propriétaires de ces fichiers

**La valeur en octale des autorisations**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **utilisateur** | | | **groupe** | | | **autres** | | |
| **r** | **w** | **x** | **r** | **w** | **x** | **r** | **w** | **x** |
| 400 | 200 | 100 | 40 | 20 | 10 | 4 | 2 | 1 |

|  |  |  |
| --- | --- | --- |
| **SUID** | **SGID** | **sticky** |
| **s** | **s** | **t** |
| 4000 | 2000 | 1000 |

**Lorsqu’un utilisateur tente d’accéder à un fichier, les permissions sont vérifiées de gauche à droite**

**Afficher les permissions avec la commande ls**

La commande "**ls -l**" affiche les autorisations sur 9 bits même s’il y en a 12 en réalité.

Si on attribue **SUID** alors le x de l’utilisateur est remplacé par un s.

exemple: rw**s**r-xr-x

Si on attribue **SGID** alors le x du groupe est remplacé par un s.

exemple: rwxr-**s**r-x

Si on attribue "**sticky**" alors le x des autres est remplacé par un t.

exemple: rwxrwxrw**t**

Si ces autorisations sont attribuées et que le l'autorisation d’exécution correspondant ne l’est pas alors les autorisations s'affichent en majuscules.

exemples:

rw**S**r-xr-x

rwxr-**S**r-x

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

**Comment changer les permissions**

**chgrp (pour changer le groupe d'appartenance d'un fichier)**

chgrp nom\_du\_groupe chemin\_du\_fichier

chgrp a33 /home/etudiant/\*

**chown (pour changer le propriétaire d'un fichier)**

chown nom\_du\_propriétaire chemin\_du\_fichier

chown etudiant /home/etudiant/\*

**chmod (notation symbolique)**

chmod [catégorie d'usager] [+-=] [droits] chemin\_du\_fichier

les catégories

u (user)

g (group)

o (other)

a (all)

+ : ajoute des droits

- : retire des droits

= : modifie les droits existants

droits (r, w, x)

ajoute le droit d'exécution au propriétaire sur toto.txt

chmod u+x toto.txt

enlève le droit d'écriture au groupe sur toto.txt

chmod g-w toto.txt

modife les droits de tout le monde pour rw sur toto.txt

chmod a=rw toto.txt

modifie les droits du propriétaire et de son groupe à r et x et

modifie les droits de tout le monde afin qu'ils n'aient plus d'accès

chmod ug=rx,o= toto.txt

**SUID**

* chmod u+s chemin\_du fichier

chmod u+s /tmp/toto

**SGID**

* chmod g+s chemin\_du\_fichier

chmod g+s /tmp/tata

* chmod g+s chemin\_du\_répertoire

chmod g+s /tmp/info/

**"sticky"**

* chmod o+t chemin\_du\_répertoire

chmod o+t /tmp/test/

**Comment changer les permissions avec les valeurs octales**

chmod *valeur\_octale* chemin\_du\_fichier

chmod 777 /tmp/f1.txt modifie tout le monde à rwx

chmod 666 /tmp/f2.txt modifie tout le monde à rw

chmod 770 /tmp/f3.txt modifie user et group à rwx et other n'a aucun droit

chmod 070 /tmp/f4.txt modifie group à rwx, user et other n'ont aucun droit

chmod 4777 /tmp/f1.sh attribue SUID sur un fichier qui a 777

chmod 2777 /tmp/f1.sh attribue SGID sur un fichier qui a 777

chmod 2777 /tmp/info/ attribue SGID sur un répertoire qui a 777

chmod 1777 /tmp/test/ attribue "sticky" sur un répertoire qui a 777

**La commande umask**

La commande **umask** est le masque de création pour les répertoires et les fichiers.

Pour le root le masque par défaut est 0022.

Pour un utilisateur standard le masque par défaut est 0002.

Si un répertoire a les droits 777 et la valeur du masque est 0022 alors les nouveaux sous-répertoires auront les droits 755 (777 - 022 = 755)

Par défaut, les fichiers ne sont jamais créés avec les droits d’exécution.

**umask -S**

affiche le format du masque sous forme symbolique

u=rwx,g=rx,o=rx

L'utilisateur root peut modifier le fichier /etc/bashrc de façon à ce qu'elle soit exécutée à chaque nouveau shell.

Il est possible de modifier la valeur du masque pour un utilisateur en ajoutant une ligne dans le fichier "/home/e1/**.bash\_profile**".

**Créer un répertoire et donner les droits en une seule commande**

mkdir /tmp/test -m 755

**ACL (Access Control List)**

Les ACL ont, entre autre, un intérêt pour les systèmes de fichiers qui contiennent des ressources partagées par SAMBA. L'administrateur peut alors attribuer les droits d'accès aux ressources partagées avec la même logique que sous Windows.

getfacl

* la commande affiche les ACL

setfacl

* la commande attribue les ACL

**Trouver des fichiers qui ont des autorisations spécifiques**

-perm mode

fichier dont les autorisations d'accès sont fixées exactement au mode indiqueé

-perm -mode

fichier ayant au moins toutes les autorisations indiquées dans le mode

-perm +mode

fichier ayant certaines des autorisations indiquées dans le mode

Voici les fichiers du répertoire /test

rwxr-x--- f0

rwxr-x--x f1

rwxr-x-w- f2

rwxr-x-wx f3

rwxr-xr-- f4

rwxr-xr-x f5

rwxr-xrw- f6

rwxr-xrwx f7

r-xr-xr-x f8

r-xr-x--- f9

Pour trouver les fichiers qui ont exactement les autorisations indiquées 🡺 757

* **find /test -perm 757**

**f7** l'autorisation 757 = 757

Pour trouver les fichiers ayant au moins toutes les autorisations indiquées 🡺 751

* **find /test -perm -751**

**f1** l'autorisation 751 contient 751

**f3** l'autorisation 753 contient 751

**f5** l'autorisation 755 contient 751

**f7** l'autorisation 757 contient 751

Pour trouver les fichiers ayant certaines des autorisations indiquées 🡺 005

* **find /test -perm +005**

**on doit trouver les fichiers qui ont R ou X dans "other"**

**f1** dans la catégorie "other" on a --x

**f3** dans la catégorie "other" on a -wx

**f4** dans la catégorie "other" on a r--

**f5** dans la catégorie "other" on a r-x

**f6** dans la catégorie "other" on a rw-

**f7** dans la catégorie "other" on a rwx

**f8** dans la catégorie "other" on a r-x