Module Linux

Equipe Pédagogique

S. BEN YAALA

H. SLIMANI

Gestion des permissions & ACL

2019-2020

+ Plan

- Rappel sur les permissions de base
- Gestion des droits spéciaux
- Gestion des ACLs

Rappel sur les permissions de base

Chaque fichier est attribué à un utilisateur, un groupe d'utilisateurs et tous les autres utilisateurs :

```
[root@localhost ~]# ll /usr/
total 104
dr-xr-xr-x.  2 root root 16384  7 sept. 04:03 bin
drwxr-xr-x.  2 root root     6 13 mars   2014 etc
drwxr-xr-x.  2 root root     6 13 mars   2014 games
drwxr-xr-x.  3 root root    22  7 sept. 04:00 include

[root@localhost ~]# ll -a /home/ludo/
total 16
drwx------. 2 ludo ludo  79  7 sept. 13:59 .
drwxr-xr-x. 4 root root  30  7 sept. 16:34 ..
-rw-------. 1 ludo ludo  14  7 sept. 13:59 .bash_history
-rw-r--r--. 1 ludo ludo  18 11 janv.  2015 .bash_logout
-rw-r--r--. 1 ludo ludo 193 11 janv.  2015 .bash_profile
-rw-r--r--. 1 ludo ludo 231 11 janv.  2015 .bashrc
```

Modifier le propriétaire d'un fichier

On peut modifier les groupes propriétaires des fichiers et répertoires avec la commande chown (change owner) :

```
# chown [options] [newowner][:newgroup] filenames
```

- Changement du groupe propriétaire

```
[root@localhost ~]# touch fichier
[root@localhost ~]# ls -l fichier
-rw-r--r--. 1 root root 0 9 sept. 12:46 fichier

[root@localhost ~]# chown ludo fichier
[root@localhost ~]# ls -l fichier
-rw-r--r--. 1 ludo root 0 9 sept. 12:46 fichier
```

- Changement de propriétaire d'un répertoire et son contenu

```
[root@localhost ~]# mkdir rep
[root@localhost ~]# for i in 1 2 3 4 5 6 ; do touch rep/fichier$i ; done
[root@localhost ~]# chown -R ludo rep/
```

Modifier le groupe


```
# chown [options] [newowner][:newgroup] filenames
```

- Options:

  -R ou –recursive

  Exemple : $ chgrp g3 F1 (F1 : fichier)

  $ chgrp –R g3 R1 (R1 : répertoire)

Ou bien :

$ chown :g3 F1

Rq: changer le propriétaire et le groupe au même temps:

$ chown ludo:g3 F1

Afficher les droits d'un fichier


```
$ ls -l
```

- colonne 1
  - ex : drwxr--r--

Type code (colonne 1)

caractère	signification
-	fichier
d	répertoire
l	lien symbolique
p	pipe
s	socket
b	périphérique bloc
c	périphérique caractère

Permissions simples

- read – write – execute

  r w x

- user – group – others

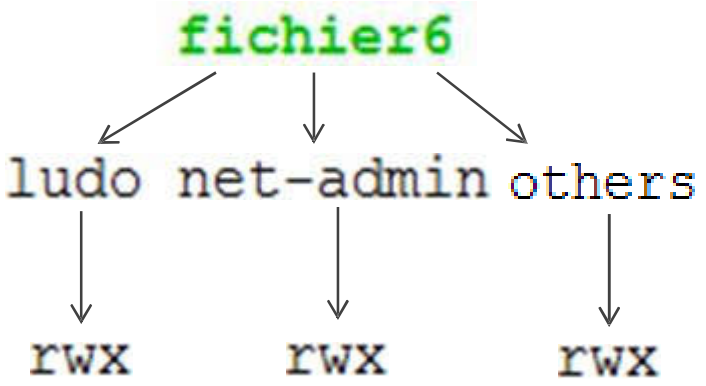
  u g o

- 9 bits
  - représentation octale : r = 4, w = 2, x = 1
  - ex : 754 = rwxr-xr--

Permissions simples


```
[root@localhost ~]# ls -l rep/
total 0
-rwxr--r--. 1 ludo net-admin 0  9 sept. 12:50 fichier1
-rwxr--r--. 1 ludo net-admin 0  9 sept. 12:50 fichier2
-rwxr--r--. 1 ludo net-admin 0  9 sept. 12:50 fichier4
-rwxr--r--. 1 ludo net-admin 0  9 sept. 12:50 fichier5
-rwxr--r--. 1 ludo net-admin 0  9 sept. 12:50 fichier6
```

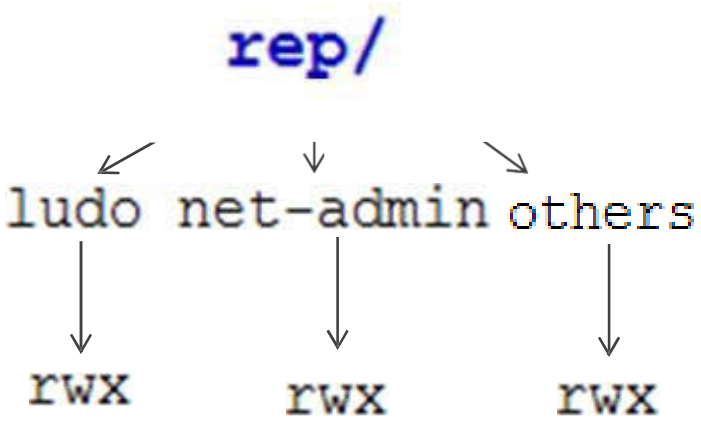
Lecture read — Écriture write — Exécution execute



Permissions simples


```
[root@localhost ~]# ls -ld rep/
drwxr--r--. 2 ludo net-admin 81  9 sept. 13:26 rep/
```

Lecture read — Écriture write — Rentrer dans le répertoire execute



Rappel sur les permissions de base

Deux méthodes d'interprétation des droits :

- de manière symbolique
- de manière octale

Permissions par défaut : umask

- Umask (User file creation mode mask) : restriction des droits d'accès lors de la création d'un fichier/repertoire,
- Les permissions maximum sont :

  -0666 pour la création d'un fichier

  -0777 pour la création d'un répertoire
- Les différents systèmes déterminent les permissions effectives lors du démarrage grâce au programme umask.
- En général la valeur de masque est 022.
- Modification de umask :

  - temporairement : umask nouvelle-valeur

  - ou en éditant le fichier /etc/profile

Modifier les droits d'accès


```
$ chmod [mode] fichier
```

- Exemple :

```
$ chmod u+r,g-x fichier
```

```
$ chmod u=rwx,g=wr,o=r fichier
```

```
$ chmod 764 fichier
```

Droits spéciaux

- **SetUID (SUID):**

- Ce droit spécial ne s'applique qu'à des fichiers (des programmes) et pas des répertoires.
- Permet d'exécuter un prog avec les autorisations (permissions) de celui qui possède le fichier

- **SetGID (SGID) :**

- S'applique aussi bien aux fichiers (programmes) qu'aux répertoires.
- Un prog lancé avec le droit SGID s'exécute avec les droits du groupe du prog.

- **Sticky bit :**

- S'applique aux répertoires.
- Lorsque ce droit est positionné sur un répertoire, il interdit sa suppression à tout utilisateur (autre le propriétaire et le root).

le SUID, (SETUID : droits s, droit 4000)

- Permet de bénéficier de droits supplémentaires lors de l'exécution d'une commande
  - Un utilisateur quelconque peut alors avoir des droits supplémentaires seulement s'il exécute la commande ayant le SUID
- Exemple de la commande « passwd »
  - Elle permet de modifier son mot de passe
  - « passwd » doit écrire dans le fichier « /etc/shadow » et pourtant :

```
linux:~# ls -l /etc/shadow
-rw-r----- 1 root shadow 700 2007-12-04 18:39 /etc/shadow
```

Aucune permission d'écriture sur ce fichier

```
linux:~# ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 28480 2007-02-27 08:53 /usr/bin/passwd
```

La commande aura les droits du super-utilisateur même si n'importe quel autre utilisateur lance son exécution

Le SGID : (SETGID : droits s, droit 2000)

- Identique au SUID mais appliqué au groupe propriétaire
  - La commande obtiendra les droits du groupe propriétaire s'il elle est exécutée par un autre utilisateur
- Attention, appliquée à un répertoire, le SGID :
  - Modifie le groupe propriétaire d'un fichier créé dans le répertoire
  - Il y a donc un mécanisme d'héritage entre le répertoire et les fichiers nouvellement créés qu'il contient
- Exemple :

```
drwxrws---   2 root compta  4096 2008-08-24 13:05 docs-compta
```

SGID positionné sur « docs-compta »

```
-rw-r--r-- 1 paul compta 0 2008-08-24 13:09 nouveau.txt
```

Le fichier nouvellement crée par paul appartient au groupe « compta »

- Créer un utilisateur user1 ayant le groupe primaire ticp et le groupe secondaire tekup
- Connecter vous en tant que user1, créer un répertoire rep1. Quel est le prop et le groupe de rep1
- Changez le groupe propriétaire de rep1 à redhat
- Positionnez le sgid sur rep1
- A qui appartient les fichiers crées dans rep1

Sticky bit : droits t, droit 1000

- Il est applicable sur un répertoire
- Lorsque ce droit est positionné sur un répertoire, il interdit la suppression des fichiers qu'il contient à tout utilisateur (autre que le propriétaire et le root).

- Créer deux utilisateurs student1 et student2 appartent au groupe secondaire tekup
- Se connecter en tant que student 1
- Créer un répertoire rep ayant comme groupe propriétaire tekup
- Positionner le sgid sur rep en tant que root
- Créer un fichier test dans rep en tant que student1
- Donner le droit d'exécution sur les répertoires rep et student1
- Est il possible en tant que student1 de supprimer le fichier test
- Revenir en tant que student 1, positionner le sticky bit sur rep
- Créer un fichier test2
- En tant que student2, essayer de supprimer test2

Représentation des droits spéciaux

- rwsrwsrwt → s: x+suid , S : suid sans x (execute)
- octal suplémentaire
  - 7777
  - SUID = 4, SGID = 2, Sticky bit =1
  - ex : 6744 = rwsr-sr—

Représentation des droits spéciaux

Exemple 1:

$ chmod 4750 F1 → 4750 = rwsr-x---

$ chmod u+s F1

→ Ici , si un utilisateur quelconque 'foulen' exécute F1, il aura les mêmes droits d'exécution que le propriétaire de F1 exécute F1 en tant que root

Exemple 2:

$ chmod 2750 F1 → 2750 = rwxr-s---

$ chmod g+s F1

→ Exécute le fichier F1 en tant que son groupe

Exemple 3:

$ chmod 1750 R1 → 1750 = rwxr-x--t

$ chmod o+t R1

→ Write : écriture + suppression

→ Sticky bit : le répertoire ne peut être supprimé que par le propriétaire ou root → w = modification

Chattr

- Il est possible de protéger tous les fichiers et les répertoires de sorte qu'il sera impossible de les supprimer et les modifier même par le root .

chattr +i nom-fichier

- Pour désactiver cette fonctionnalité

chattr -i nom-fichier

Lsattr pour vérifier

+ Gestion des ACLs

- La norme POSIX définit les droits d'accès sur les fichiers et répertoires à 3 et seulement 3 entités (rwx) pour 3 types d'utilisateurs (ugo) → ces droits sont limités.
- C'est quoi une ACL ?

  - Une ACL, ou Access Control List (en français : « liste de contrôle d'accès »)

  - étendre le nombre d'utilisateurs et de groupes ayant des droits sur un même fichier.

  - très utile (voire indispensable) dans un environnement informatique basé sur un travail collaboratif et mutualisé.

Installation des ACLs

- Installation

  #yum install –y acl

- Vérifier la configuration

  #grep ACL /boot/config-3.10.0-229.14.1.el7.x86_64
  CONFIG_EXT4_FS_POSIX_ACL=y
  CONFIG_XFS_POSIX_ACL=y
  CONFIG_BTRFS_FS_POSIX_ACL=y
  CONFIG_FS_POSIX_ACL=y
  CONFIG_GENERIC_ACL=y
  CONFIG_TMPFS_POSIX_ACL=y
  CONFIG_NFS_V3_ACL=y
  CONFIG_NFSD_V2_ACL=y
  CONFIG_NFSD_V3_ACL=y
  CONFIG_NFS_ACL_SUPPORT=m

+ Gestion des ACLs : commandes

- Deux commandes principales :

  - une commande pour manipuler l'ACL d'un fichier :

    - setfacl : (set file's ACL « régler l'ACL du fichier »)

  - une commande pour consulter l'ACL d'un fichier :

    - getfacl : (get file's ACL « récupérer l'ACL du fichier »)

Mise en oeuvre des ACLs

- Afficher les ACLs

```
# getfacl fichier1
# file: fichier1
# owner: root
# group: root
  user::rw-
group::r--
other::r--
```

Mise en oeuvre des ACLs

- Positionner des ACLs

```
#setfacl –m u:ludo:rw fichier1

#setfacl –m g:users:rw  fichier1

#setfacl –m u:ludo:rw,g:users:rw fichier1

#setfacl –m u:ludo:6,g:users:6,o:0 fichier1
```

- Positionner des ACLs de manière récursive

```
setfacl -Rm u:student:rw Tekup/
```

modifie l'ACL de tous les fichiers situés sous Tekup/ en attribuant une permission de lecture et d'écriture à l'utilisateur student.

Mise en oeuvre des ACLs

- Retirer toutes les ACLs

```
#setfacl –b fichier1
```

- Supprimer des ACLs

```
#setfacl –x g:users fichier1
#setfacl –x u:ludo,g:users fichier1
```

- Retirer uniquement les permissions par défaut

```
#setfacl –k fichier1
```

Héritage des ACLs

- Si on applique une ACL à un répertoire, les fichiers créés ensuite dans ce répertoire n'hériteront pas de son ACL.

- Pour rendre l'héritage des ACL possible , il suffit de rajouter le préfixe d: (comme **d**efault) au début de l'ACL :

- Configuration de l'héritage

```
# setfacl -m d:u:ludo:rw,d:g:ludo:rw rep_acl_d/
```

Il est cependant possible de se passer du préfixe d:, dans ce cas, toutes les permissions spécifiées seront des permissions par défaut .

Héritage des ACLs

- Vérification de l'héritage

```
getfacl rep_acl_d/
# file: rep_acl_d/
# owner: root
# group: root
user::rwx
user:ludo:rw-
group::r-x
group:ludo:rw-
mask::rwx
other::r-x
default:user::rwx
default:user:ludo:rw-
default:group::r-x
default:group:ludo:rw-
default:mask::rwx
default:other::r-x
```

+ Sauvegarder et restaurer les ACLs

- Conserver les ACLs lors d'une copie du fichier ou répertoire

```
cp –a file_acl /home/ludo/
```

- Sauvegarde des ACLs dans un fichier

```
getfacl -R rep_acl_d/ >acl.save
```

- Restauration des ACLs

```
setfacl --restore /root/acl.save
```

+ Exemple

1. créez un fichier ~/toto et y écrire quelque chose
2. retirer les droits de lecture et écriture pour tout le monde (groupe et autres) avec chmod
3. avec la commande setfacl, donnez les droits de lecture à votre binôme (exemple : mohamed)
4. un ls -l ~/toto affiche un ' +' à la suite des droits montrant que des ACL ont été ajoutés au fichier
5. Connecter vous en tant que mohamed (su mohamed) et vérifier s'il peut lire ou non le fichier toto.
6. Retirer les ACL sur ~/toto et assurez-vous qu'elles ont disparues