Administration système - TP 4

Auteurs :

BIARD Gauthier

MARION Audran

Le 28/02/2020

Exercice 1. Gestion des utilisateurs et des groupes

1. Commencez par créer deux groupes groupe1 et groupe2

Dans le bash, nous exécutons les commandes :

```
sudo groupadd groupe1
sudo groupadd groupe2
```

> Nous avons du exécuter la commande en *sudo* car nous n'avions pas les droits sinon.

Afin de vérifier que les groupes ont bien été ajoutés, nous exécutons :

```
cat /etc/group | tail -2
```

En retour nous obtenons :

```
groupe1:x:1001:
groupe2:x:1002:
```

Nos groupes ont donc bien été créés.

2. Créez ensuite 4 utilisateurs u1, u2, u3, u4 avec la commande useradd, en demandant la création de leur dossier personnel et avec bash pour shell

Nous avons exécuter les commandes suivantes :

```
sudo useradd -mksd u1
sudo useradd -mksd u2
sudo useradd -mksd u3
sudo useradd -mksd u4
```

3. Ajoutez les utilisateurs dans les groupes créés :

- **u1, u2, u4 dans groupe1**
- **u2, u3, u4 dans groupe2**

Afin d'ajouter les différents utilisateurs à leurs groupes, il faut exécuter :

```
usermod -a -G groupe1 u1
usermod -a -G groupe1 u2
usermod -a -G groupe1 u4
usermod -a -G groupe2 u2
usermod -a -G groupe2 u3
usermod -a -G groupe2 u4
```

> -a -G permet d'ajouter un utilisateur existant à un groupe déjà existant. (Nous pouvons également rajouter d'autres groupes après le premier groupe dans la même ligne de commande)

4. Donnez deux moyens d'afficher les membres de groupe2

Le premier moyen d'afficher les membres de groupe2 est la commande :

```
grep groupe2 /etc/group
```

Dans ce cas, en retour nous avons :

```
groupe2:x:1002:u2,u3,u4
```

Le deuxième moyen d'afficher les membres de groupe2 est la commande :

```
members groupe2
```

> Il faut pour cela installer le paquet *members* : `sudo apt install members`

Dans ce cas, en retour nous avons :

```
u2 u3 u4
```

5. Faites de groupe1 le groupe propriétaire de /home/u1 et /home/u2 et de groupe2 le groupe propriétaire de /home/u3 et /home/u4

```
sudo chgrp groupe1 /home/u1
sudo chgrp groupe1 /home/u2
sudo chgrp groupe2 /home/u3
sudo chgrp groupe2 /home/u4
```

6. Remplacez le groupe primaire des utilisateurs :

- **groupe1 pour u1 et u2**
- **groupe2 pour u3 et u4**

Pour changer le groupe primaire des utilisateurs u1 et u2 :

```
sudo usermod -g groupe1 u1
sudo usermod -g groupe1 u2
```

Pour changer le groupe primaire des utilisateurs u3 et u4 :

```
sudo usermod -g groupe2 u3
sudo usermod -g groupe2 u4
```

Nous pouvons vérifier à quels groupes appartiennent les différents utilisateurs :

```
gauthieraudran@serveur:~$ groups u1
u1 : groupe1
gauthieraudran@serveur:~$ groups u2
u2 : groupe1 groupe2
gauthieraudran@serveur:~$ groups u3
u3 : groupe2
gauthieraudran@serveur:~$ groups u4
u2 : groupe2 groupe1
```

7. Créez deux répertoires /home/groupe1 et /home/groupe2 pour le contenu commun aux groupes, et mettez en place les permissions permettant aux membres de chaque groupe d'écrire dans le dossier associé.

Pour le groupe 1 :

```
mkdir /home/groupe1
chgrp groupe1 /home/groupe1
chmod g+w /home/groupe1
```

Pour le groupe 2:

```
mkdir /home/groupe2
chgrp groupe2 /home/groupe2
chmod g+w /home/groupe2
```

> Tout d'abord, nous créons le dossier, puis nous transférons la possession du groupe et enfin nous donnons le droit d'écriture au groupe.

8. Comment faire pour que, dans ces dossiers, seul le propriétaire d'un fichier ait le droit de renommer ou supprimer ce fichier ?

```
find . -type f -print0 | xargs -0 chmod 644
```

> Cette commande nous permet de changer uniquement les permissons des fichiers sans changer les permissions que nous avons appliqué sur les dossiers précédemment.

9. Pouvez-vous vous connecter en tant que u1 ? Pourquoi ?

Nous ne pouvons pas nous connecter en tant que *u1* car le compte n'est pas actif et n'a pas de mot de passe.

> Lorsque nous tapons la commande *su u1*, il faut rentrer un mot de passe mais comme u1 n'en a pas, nous ne pouvons pas nous connecter.

10. Activez le compte de l'utilisateur u1 et vérifiez que vous pouvez désormais vous connecter avec son compte.

Afin d'activer le compte *u1* :

```
gauthieraudran@serveur:~$ sudo passwd u1
New password:
Retype new password:
passwd: password updated successfully
```

> Nous avons choisi *u1* pour mot de passe de *u1*

11. Quels sont l'uid et le gid de u1 ?

Afin d'afficher l'uid et le gid de u1 :

```
gauthieraudran@serveur:~$ id u1
uid=1001(u1) gid=1001(groupe1) groups=1001(groupe1)
```

> Ainsi, l'uid de *u1* est **1001** et son gid est **1001**

12. Quel utilisateur a pour uid 1003 ?

Nous pouvons utiliser la commande :

```
gauthieraudran@serveur:~$ cat /etc/passwd | grep 1003
u3:x:1003:1002::/home/u3:/bin/sh
```

L'utilisateur qui a pour uid *1003* est **u3**

> Afin de n'avoir que le nom de l'utilisatuer, nous pouvons utiliser la commande: *cat /etc/passwd* | *cut -d: -f1,3* | *grep 1003* | *cut -d: -f1* cat /etc/passwd Permet de récupérer le contenu du fichier. cut -d: -f1,3 Permet de récupérer les colonnes 1 et 3. grep 1003 Permet de récupérer les lignes contenant 1003. cut -d: -f1 Permet de récupérer la 1ère colonne qui est le nom d'utilisateur.

13. Quel est l'id du groupe groupe1 ?

Pour récupérer l'id du groupe *groupe1*, il faut exécuter la commande suivante :

```
gauthieraudran@serveur:~$ cat /etc/group | cut -d: -f1,3 | grep groupe1 | cut -d:
-f2
1001
```

L'id de *groupe1* est **1001**

14. Quel groupe a pour guid 1002 ? (Rien n'empêche d'avoir un groupe dont le nom serait 1002...)

Il suffit de faire la commande :

```
getent group 1002
```

Il sagit donc du groupe groupe2 qui a pour guid **1002**.

15. Retirez l'utilisateur u3 du groupe groupe2. Que se passe-t-il ? Expliquez

Afin de retirer l'utilisateur u3 du groupe *groupe2*, il faut exécuter la commande suivante:

```
gauthieraudran@serveur:~$ gpasswd -d u3 groupe2
Removing user u3 from group groupe2
```

Cependant, lorsque nous vérifions à quel groupe appartient *u3*, nous pouvons voir que *u3* appartient toujours à *groupe2*:

```
gauthieraudran@serveur:~$ groups u3
u3 : groupe2
```

16. Modifiez le compte de u4 de sorte que : — il expire au 1er juin 2020 — il faut changer de mot de passe avant 90 jours — il faut attendre 5 jours pour modifier un mot de passe — l'utilisateur est averti 14 jours avant l'expiration de son mot de passe — le compte sera bloqué 30 jours après expiration du mot de passe

Tout d'abord, nous vérifions les caractéristiques du compte de u4 :

```
gauthieraudran@serveur:~$ chage -l u4
Last password change                                    : Mar 13, 2020
Password expires                                        : never
Password inactive                                       : never
Account expires                                         : never
Minimum number of days between password change          : 0
Maximum number of days between password change          : 99999
Number of days of warning before password expires       : 7
```

Commande	Fonction
chage -E 2020-06-1 u4	change la date d'éxpiration du compte de l'utilisateur
chage -M 90 u4	change la durée maximale de validité du mot de passe
chage -m 5 u4	changer le délai avant de pouvoir rechanger son mot de passe
chage -W 14 u4	change le nombre de jour où l'utilisateur est averti avant l'expiration de son mot de passe

Commande	Fonction
chage -I 30 u4	change le nombre de jours après lequel, suite à l'expiration de son mot de passe, l'utilisateur est bloqué

De ce fait, lorsque nous regardons les caractéristiques du compte après les modifications, nous obtenons :

```
gauthieraudran@serveur:~$ chage -l u4
Last password change                                    : Mar 13, 2020
Password expires                                        : Jun 11, 2020
Password inactive                                       : Jul 11, 2020
Account expires                                         : Jun 01, 2020
Minimum number of days between password change    : 5
Maximum number of days between password change    : 90
Number of days of warning before password expires : 14
```

17. Quel est l'interpréteur de commandes (Shell) de l'utilisateur root ?

On se connecte tout d'abord en tant que root :

```
sudo su
```

Ensuite :

```
root@server:~# printenv SHELL
/bin/bash
```

18. à quoi correspond l'utilisateur *nobody* ?

Il s'agit d'un utilisateur qui existe par défaut et qui ne détient aucun document, n'appartient à aucun groupe de provilèges et qui n'a aucun droits à part ceux d'utilisateur de base. On utilise souvent cet utilisateur pour faire des test sans risquer de faire des erreurs.

19. Par défaut, combien de temps la commande sudo conserve-t-elle votre mot de passe en mémoire ? Quelle commande permet de forcer sudo à oublier votre mot de passe ?

Par défaut, la commande **sudo** conserve le mot de passe pendant 15 minutes.

La commande qui permet de forcer sudo à oublier notre mot de passe est la commande : **sudo -k** (ou la commande **sudo --reset-timestamp**)

> Nous pouvons également lorsque nous sommes connecter en tant que root taper la commande **visudo** et ainsi modifier le fichier :

```
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset,timestamp_timeout=0
Defaults        mail_badpass
Defaults
secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bi
n"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

#includedir /etc/sudoers.d
```

> On a donc rajouter **timestamp_timeout=0**, ce qui permet que le mot de passe ne soit pas sauvegarder lorsque nous faisons un **sudo**

Exercice 2. Gestion des permissions

1. Dans votre $HOME, créez un dossier test, et dans ce dossier un fichier fichier contenant quelques lignes de texte. Quels sont les droits sur test et fichier ?

```
mkdir test
echo 'Ceci est un fichier test' > test/fichier
```

Pour le dossier *test*, nous exécutons la commande : **ls -l**. En retour, nous avons :

```
drwxrwxr-x 2 gauthieraudran gauthieraudran 4096 mars 18 18:55 test
```

> Nous avons donc 775 pour le dossier test.

Pour le fichier *fichier* nous exécutons la commande : **ls -l fichier**. En retour, nous avons :

```
-rw-rw-r-- 1 gauthieraudran gauthieraudran 25 mars 18 18:55
```

> Nous avons donc 664 pour le fichier *fichier*.

2. Retirez tous les droits sur ce fichier (même pour vous), puis essayez de le modifier et de l'afficher en tant que root. Conclusion ?

Afin de retirer tous les droits sur ce fichier, nous exécutons la commande :

```
chmod 000 test/fichier
```

Si nous essayons de voir les droits sur ce fichier, nous obtenons en retour :

```
---------- 1 gauthieraudran gauthieraudran 25 mars 18 18:55
```

Nous nous connectons maintenant en tant que root. Ensuite, nous exécutons les commandes suivantes :

```
# echo 'Un autre test' > test/fichier
# cat test/fichier
Un autre test
```

> Nous remarquons que nous pouvons, en tant que root, écrire et lire le fichier. Nous pouvons également supprimer le fichier avec la commande **rm test/fichier** malgrès le fait que toutes les permissions soient retirées.

3. Redonnez vous les droits en écriture et exécution sur fichier puis exécutez la commande echo "echo Hello" > fichier. On a vu lors des TP précédents que cette commande remplace le contenu d'un fichier s'il existe déjà. Que peut-on dire au sujet des droits ?

Afin de redonner les droits, nous exécutons la commande suivante :

```
sudo chmod u+wx test/fichier
```

Si nous regardons les informations sur *fichier* à l'aide de la commande **ls -l test**, nous obtenons en retour :

```
--wx------ 1 gauthieraudran gauthieraudran 0 mars 18 18:55
```

Nous exécutons maintenant la commande demandée :

```
echo "echo Hello" > test/fichier
```

Si nous exécutons la commande **ls -l test**, nous obtenons maintenant en retour :

```
--wx------ 1 gauthieraudran gauthieraudran 11 mars 18 18:55
```

> Les droits d'écriture sont suffisant car nous sommes propiétaire du fichier.

4. Essayez d'exécuter le fichier. Est-ce que cela fonctionne ? Et avec sudo ? Expliquez.

Nous essayons d'éxecuter le fichier :

```
gauthieraudran@serveur:~$ cd test/
gauthieraudran@serveur:~/test$ ./fichier
```

En retour, nous obtenons :

```
bash: ./fichier: Permission denied
```

> Nous ne pouvons pas exécuter le fichier. Nous avons bien le droit d'éxecution sur le fichier mais nous n'avons pas le droit de lecture, c'est pourquoi nous ne pouvons pas exécuter le fichier.

Nous essayons maintenant avec le **sudo** :

```
gauthieraudran@serveur:~/test$ sudo ./fichier
Hello
```

> Lorsque nous exécutons avec **sudo**, nous passons au-dessus des permissions donc nous pouvons exécuter le fichier.

Afin de pouvoir exécuter le fichier sans utiliser **sudo**, nous devons tout d'abord exécuter la commande suivante :

```
sudo chmod u+r fichier
```

Maintenant nous pouvons exécuter le fichier sans avoir besoin d'utiliser **sudo**.

5. Placez-vous dans le répertoire test, et retirez-vous le droit en lecture pour ce répertoire. Listez le contenu du répertoire, puis exécutez ou affichez le contenu du fichier fichier. Qu'en déduisez-vous ? Rétablissez le droit en lecture sur test

Tout d'abord, nous enlevons les droits de lecture sur le répertoire test :

```
chmod u-r test
```

Nous essayons ensuite de lire le contenu du répertoire :

```
gauthieraudran@serveur:~$ ls test
ls: cannot open directory 'test': Permission denied
```

> Nous ne pouvons pas afficher le contenu du dossier *test* car nous n'avons pas les droits de lecture.

Nous essayons maintenant d'afficher et d'exécuter le fichier fichier :

```
gauthieraudran@serveur:~$ cat test/fichier
echo Hello
gauthieraudran@serveur:~$ ./test/fichier
Hello
```

> Nous pouvons donc lire et exécuter le fichier *fichier* car nous n'avons pas changer les permissions sur le fichier. Cependant, nous ne pouvons pas utiliser l'auto-complétion sur le fichier lorsque nous tapons la commande.

Nous rétablissons le droit de lecture sur *test* :

```
chmod u+r test
```

6. Créez dans test un fichier nouveau ainsi qu'un répertoire sstest. Retirez au fichier nouveau et au répertoire test le droit en écriture. Tentez de modifier le fichier nouveau. Rétablissez ensuite le droit en écriture au répertoire test. Tentez de modifier le fichier nouveau, puis de le supprimer. Que pouvez-vous déduire de toutes ces manipulations ?

Tout d'abord, nous créons le fichier *nouveau* et le répertoire *sstest* :

```
gauthieraudran@serveur:~$ cd test
gauthieraudran@serveur:~/test$ echo 'nouveau' > nouveau
gauthieraudran@serveur:~/test$ mkdir sstest
```

Ensuite, nous retirons le droit d'écriture au fichier *nouveau* et au répertoire *test* :

```
gauthieraudran@serveur:~/test$ sudo chmod u-w nouveau
gauthieraudran@serveur:~/test$ cd ..
gauthieraudran@serveur:~$ sudo chmod u-w test
gauthieraudran@serveur:~$ echo "Ceci est un test d'écriture" > test/nouveau
- bash: test/nouveau: Permission denied
gauthieraudran@serveur:~$ rm test/nouveau
rm: remove write-protected regular file 'test/nouveau'? yes
gauthieraudran@serveur:~$ ls test
fichier nouveau sstest
```

> En retirant les droits d'écriture au dossier et au fichier, nous en pouvons pas écrire ou supprimer le fichier.

Nous rajoutons le droit d'écriture sur le répertoire :

```
gauthieraudran@serveur:~$ chmod u+w test
gauthieraudran@serveur:~$ echo 'coucou' > test/nouveau
-bash: test/nouveau: Permission denied
$ rm test/nouveau
rm: remove write-protected regular file 'test/nouveau'? yes
$ ls test
fichier  sstest
```

> Lorsque nous remettons les droit d'écriture seulement sur le dossier, nous ne pouvons toujours pas écrire sur le fichier mais nous pourvons tout de même supprimer le fichier (en effet, en faisant cela, nous n'écrivons pas sur le fichier mais sur le répertoire).

7. Positionnez vous dans votre répertoire personnel, puis retirez le droit en exécution du répertoire test. Tentez de créer, supprimer, ou modifier un fichier dans le répertoire test, de vous y déplacer, d'en lister le contenu, etc...Qu'en déduisez vous quant au sens du droit en exécution pour les répertoires ?

```
gauthieraudran@serveur:~$ chmod u-x test
gauthieraudran@serveur:~$ echo 'test' > test/fichier
bash: test/fichier: Permission denied
gauthieraudran@serveur:~$ touch test/test
touch: cannot touch 'test/test': Permission denied
gauthieraudran@serveur:~$ rm test/fichier
rm: cannot remove 'test/fichier': Permission denied
gauthieraudran@serveur:~$ ./test/fichier
bash: ./test/fichier: Permission denied
```

> En supprimant les droits d'éxécution, nous ne pouvons plus créer, ni supprimer, ni modifier et ni exécuter un fichier.

```
gauthieraudran@serveur:~$ ls -l test
ls: cannot access 'test/fichier': Permission denied
ls: cannot access 'test/sstest': Permission denied
total 0
-????????? ? ? ? ?            ? fichier
d????????? ? ? ? ?            ? sstest
```

> Nous pouvons donc toujours lister le contenu du dossier mais nous ne pouvons pas accéder aux éléments. Tout ce que nous pouvons savoir est donc leut nom.

8. Rétablissez le droit en exécution du répertoire test. Positionnez vous dans ce répertoire et retirez lui à nouveau le droit d'exécution. Essayez de créer, supprimer et modifier un fichier dans le répertoire test, de vous déplacer dans ssrep, de lister son contenu. Qu'en concluez-vous quant à l'influence des droits que l'on possède sur le répertoire courant ? Peut-on retourner dans le répertoire parent avec "cd .." ? Pouvez-vous donner une explication ?

```
gauthieraudran@serveur:~$ chmod u+x test
gauthieraudran@serveur:~$ cd test
gauthieraudran@serveur:~/test$ chmod u-x ../test
gauthieraudran@serveur:~/test$ touch coucou
touch: cannot touch 'a': Permission denied
gauthieraudran@serveur:~/test$ ls
ls: cannot open directory '.': Permission denied
gauthieraudran@serveur:~/test$ rm fichier
rm: cannot remove 'fichier': Permission denied
gauthieraudran@serveur:~/test$ echo 'test' > fichier
bash: fichier: Permission denied
gauthieraudran@serveur:~/test$ cd ssrep
bash: cd: ssrep: Permission denied
gauthieraudran@serveur:~/test$ ls ssrep
ls: cannot access 'ssrep': Permission denied
gauthieraudran@serveur:~/test$ cd ..
gauthieraudran@serveur:~$
```

> Nous pouvons donc en conclure que nos permissions dépendent du dossier dans lequel nous nous trouvons. C'est pourquoi nous ne pouvons pas exécuter de commande dans le dossier dans lequel nous nous trouvons.

> La commande **cd..** fonctionne car nous avons les droits dans le dossier parent. Nous n'avions pas les droits dans le dossier *test* afin d'aller dans le dossier *ssrep*.

9. Rétablissez le droit en exécution du répertoire test. Attribuez au fichier fichier les droits suffisants pour qu'une autre personne de votre groupe puisse y accéder en lecture, mais pas en écriture.

Afin de rétablir le droit d'exécution du répertoire test :

```
chmod u+x test
```

Afin d'ajouter le droit de lecture à quelqu'un du même groupe :

```
chmod g+r test/fichier
```

Afin d'enlever le droit d'écriture à quelqu'un du même groupe :

```
chmod g-w test/fichier
```

10. Définissez un umask très restrictif qui interdit à quiconque à part vous l'accès en lecture ou en écriture, ainsi que la traversée de vos répertoires. Testez sur un nouveau fichier et un nouveau répertoire.

Nous créons le umask puis nous le vérifions :

```
gauthieraudran@serveur:~$ umask 077
gauthieraudran@serveur:~$ umask -S
u=rwx,g=,o=
```

Nous testons maintenant de créer un nouveau fichier et un nouveau répertoire et d'y accéder :

```
gauthieraudran@serveur:~$ mkdir coucou
gauthieraudran@serveur:~$ echo "C'est un test" > coucou/nvfichier
gauthieraudran@serveur:~$ cd coucou
gauthieraudran@serveur:~/coucou$ ls
```

```
nvfichier
gauthieraudran@serveur:~/coucou$ cat nvfichier
C'est un test
```

> Tout fonctionne correctement. Maintenant testons avec un autre utilisateur :

```
gauthieraudran@serveur:~$ su u1
Password:
$ cd /home
$ cd /gauthieraudran
sh: 2: cd: can't cd to /gauthieraudran
```

> Nous ne pouvons donc même pas accéder à notre dossier d'utilisateur car seul nous avons les droits.

11. Définissez un umask très permissif qui autorise tout le monde à lire vos fichiers et traverser vos répertoires, mais n'autorise que vous à écrire. Testez sur un nouveau fichier et un nouveau répertoire.

On définit le umask :

```
gauthieraudran@serveur:~$ umask 022
gauthieraudran@serveur:~$ umask -S
u=rwx,g=rx,o=rx
gauthieraudran@serveur:~$ mkdir coucou2
gauthieraudran@serveur:~$ echo "Ceci est un test" > coucou2/test
```

Nous essayons avec un utilisateur :

```
gauthieraudran@serveur:~$ su u1
Password:
$ cd /home
$ cd /gauthieraudran
$ ls
coucou     installed-package   nudoku           origine-commande2     repo-cpe
coucou2  installed-package2  origine-commande  origine-commande.deb  test
$ cd coucou2
$ ls
test
$ cat test
Ceci est un test
$ mkdir test2
mkdir: cannot create directory 'test2': Permission denied
$ touch test2
touch: cannot touch 'test2': Permission denied
```

> Nous pouvons donc bien naviguer dans les dossiers mais nous ne pouvons ni créer de dossier ni créer de fichier.

12. Définissez un umask équilibré qui vous autorise un accès complet et autorise un accès en lecture aux membres de votre groupe. Testez sur un nouveau fichier et un nouveau répertoire.

Nous créons le umask, le vérifions et créons des dossiers de test :

```
gauthieraudran@serveur:~$ umask 037
gauthieraudran@serveur:~$ umask -S
u=rwx,g=r,o=
gauthieraudran@serveur:~$ mkdir coucou12
gauthieraudran@serveur:~$ echo "Ceci est un test" > coucou12/test
```

Nous testons avec un utilisateur :

```
gauthieraudran@serveur:~$ su u1
Password:
$ cd /home/gauthieraudran
$ ls
coucou      installed-package   origine-commande        repo-cpe
coucou12  installed-package2  origine-commande2     test
coucou2   nudoku          origine-commande.deb
$ cd coucou12
sh: 3: cd: can't cd to coucou12
```

> Cela est dû au fait que l'utilisateur n'appartienne pas à notre groupe. Ajoutons-le avec la commande : **sudo usermod -a -G gauthieraudran u1**

```
$ su u1
Password:
$ cd /home/gauthieraudran
$ ls
coucou      installed-package   origine-commande        repo-cpe
coucou12  installed-package2  origine-commande2     test
coucou2   nudoku          origine-commande.deb
$ cd coucou12
$ ls
test
$ cat test
Ceci est un test
$ mkdir test2
mkdir: cannot create directory 'test2': Permission denied
$ touch test2
touch: cannot touch 'test2': Permission denied
```

> En appartenant au groupe *gauthieraudran*, cet utilisateur à le droit de le déplacer dans l'arborescence mais ne peut toujours pas créer de fichier ou de dossier.

13. Transcrivez les commandes suivantes de la notation classique à la notation octale ou vice-versa (vous pourrez vous aider de la commande stat pour valider vos réponses) :

- **chmod u=rx,g=wx,o=r fic**
- **chmod uo+w,g-rx fic en sachant que les droits initiaux de fic sont r--r-x---**
- **chmod 653 fic en sachant que les droits initiaux de fic sont 711**
- **chmod u+x,g=w,o-r fic en sachant que les droits initiaux de fic sont r--r-x---**

Commande	Transcription
chmod u=rx,g=wx,o=r fic	chmod 534 fic
chmod uo+w,g-rx fic en sachant que les droits initiaux de fic sont r--r-x---	chmod 604 fic
chmod 653 fic en sachant que les droits initiaux de fic sont 711	chmod u-x,g-r,o-w fic
chmod u+x,g=w,o-r fic en sachant que les droits initiaux de fic sont r--r-x---	chmod 520 fic

14. Affichez les droits sur le programme passwd. Que remarquez-vous ? En affichant les droits du fichier /etc/passwd, pouvez-vous justifier les permissions sur le programme passwd ?

```
gauthieraudran@serveur:~$ stat -c %A /etc/passwd
-rw-r--r--
```

> Les permissions de ce dossier correspondent à celles qu'ont un dossier par défaut. Le propriétaire de ce dossier est root. Les utilisateurs *group* et *other* ont seulement le droit en lecture, donc ils ne peuvent que lister les utilisateur et connaître leur groupe etc. Cependant, ils n'ont pas le droit en écriture et ne peuvent donc pas l'éditer (ce que nous avons remarquer précédement car il faut être root afin de pouvoir créer un utilisateur).

Pour les plus rapides :

15. Access Control Lists (ACL) : suivez le tutoriel de cette page : https://doc.ubuntu-fr.org/acl.

16. Quotas disques : suivez le tutoriel de cette page : https://doc.ubuntu-fr.org/quota.