**TP-LINUX : Gestion des utilisateurs, des fichiers et des droits d'accès**

**Objectif** : Gérer les utilisateurs sur un système Linux
Gérer les droits d'accès aux fichiers
Manipuler les fichiers et les dossiers

Linux est fondamentalement « multi-utilisateurs » et accessible en réseau.
Il est donc normal de créer des profils utilisateurs.
C'est la seule façon de garantir la protection des données des utilisateurs.
C'est la seule façon de limiter le pouvoir des utilisateurs sur le système.
Sans cela, n'importe qui pourrait faire n'importe quoi…

Il existe des systèmes d'Annuaires (SMB, LDAP, NIS, Kerberos …) très sophistiqués.
Ici nous parlerons uniquement de l'annuaire de base (**PAM**) de Linux.

# 1 Fichier /etc/passwd

Ce fichier fait partie du dispositif PAM.
ATTENTION : Il ne faut pas le modifier avec un éditeur comme nano, sous peine de mettre en panne le PAM.

Cependant la commande : `cat /etc/passwd`
Permet de voir la liste des utilisateurs (première colonne avant le signe « : » )
En tapant cette commande, on se rend compte qu'il existe un grand nombre d'utilisateurs créés par Linux et par les logiciels installés (par exemple, l'utilisateur www-data à été créé par Apache)

# 2 Gérer les utilisateurs

## 2.1 GROUPADD : Créer un groupe

Il est souvent plus pratique de créer les groupes (vides) avant de créer les utilisateurs.
Un groupe permet d'affecter des droits d'accès à un ensemble d'utilisateurs.
Un utilisateur peut appartenir à plusieurs groupes (1 groupe principal, X groupes secondaires).

Pour créer un groupe la commande est :

```
sudo groupadd nom_du_groupe
```

## 2.2 USERADD : Créer un utilisateur

Pour créer un utilisateur qui utilisera la console, la commande la plus simple est :

```
sudo useradd -m nom_de_l_utilisateur -p mot_de_passe_crypté
```

« -m » et « -p » sont des OPTION (comme le « -l » de « ls –l »)
« -m » indique à useradd qu'il faut créer le dossier personnel de l'utilisateur dans le dossier /home.
« -p » fixe le mot de passe. Pour ne pas écrire le mot de passe crypté (très long !!) on préfère utiliser la commande « passwd » dans un second temps :

```
sudo passwd nom_de_l_utilisateur
```

Pour créer un utilisateur et l'affecter à un groupe :

```
sudo useradd -m nom_de_l_utilisateur -g groupe_principal -G groupes_secondaires
```

## 2.3 USERMOD : Modifier un utilisateur déjà existant

Par exemple, pour modifier les groupes d'appartenance d'un utilisateur.

Exemple : L'utilisateur louis appartient à un groupe principal « labo » et un groupe « user ».

On veut qu'il soit aussi dans le groupe « video » :

```
sudo usermod louis -G user,video
```

Note : On est obligé de redonner le nom des groupes secondaires.

Sur Raspberry OS, on peut également simplifier en refaisant un « adduser » :

```
sudo adduser louis video
```

L'utilisateur est maintenant dans le groupe « video ». Les autres groupes sont conservés.

## 2.4 USERDEL : supprimer un utilisateur

Supprimer l'utilisateur mais pas son dossier personnel :

```
sudo userdel nom_utilisateur
```

Supprimer l'utilisateur ET son dossier personnel :

```
sudo userdel  -r nom_utilisateur
```

## 2.5 GROUPS : Vérifier les groupes d'un utilisateur

Pour soi même : `groups`

Pour un utilisateur spécifique : `groups nom_utilisateur`

## 2.6 PASSWD : Modifier un mot de passe

Un utilisateur peut modifier son propre mot de passe en tapant : `passwd`

Pour modifier le mot de passe d'un utilisateur, si on a le droit à sudo, :

`sudo passwd robert` « robert » est le nom de l'utilisateur

## 2.7 Exercice 1

Travail :

1. En utilisant les commandes *useradd* et *groupadd*, créez 4 nouveaux utilisateurs (Amael, Mathilde, Yann, Matthieu) et deux groupes (premiere, terminale).
2. Utilisez la commande « passwd » pour modifier leur mot de passe
3. Tester le fonctionnement des utilisateurs en ouvrant d'autres sessions SSH (avec puTTY par exemple) ou avec la commande « su »

## 2.8 Et SUDO dans tout ça ?

Si vous testez une commande « sudo » en étant connecté avec un nom d'utilisateur créé, cela ne fonctionne pas.

C'est normal !!! L'utilisateur PI a le privilège « sudo » et pour des raisons de sécurité, les autres ne l'ont pas.

> Note : Sur les versions Linux avec « sudo », l'utilisateur ROOT existe mais n'est pas activé pour des raisons de sécurité.

Pour qu'un utilisateur ait le droit « sudo », il doit faire partie du groupe « sudo ».

Il faut bien réfléchir avant d'accorder ce privilège.

# 3 Gérer les fichiers et les dossiers

Rappel : Dans une séance précédente, vous avez appris les commandes : `cd, ls, pwd, nano`

## 3.1 MKDIR : Créer un dossier

```
mkdir  /home/travail
```

Cette commande crée un dossier « travail » dans le dossier « /home ». Le dossier « /home » doit déjà exister.

On peut obtenir le même résultat en séparant les commandes :

```
cd /home
mkdir travail
```

## 3.2 RMDIR : Supprimer un dossier vide

ATTENTION : IL N'Y A PAS DE CORBEILLE SOUS LINUX !!! Les fichiers seront définitivement perdus.

```
rmdir  /home/travail
```

Si le dossier n'est pas vide, il faut utiliser avec prudence la commande :

```
rm -r /home/travail
```

## 3.3 Exercice 2

1. Créer le dossier /home/travail. Le dossier */home* existe déjà.
2. Dans /home/travail, créez 2 dossiers : *classe1* et *classe2*.

## 3.4 RM : supprimer un fichier

ATTENTION : IL N'Y A PAS DE CORBEILLE SOUS LINUX !!! Les fichiers seront définitivement perdus.

```
rm  toto.txt
```

## 3.5 MV : Déplacer / Renommer un fichier ou un dossier

Renommer un fichier ou un dossier :

```
mv  nom_actuel.txt nouveau_nom.txt
```

Déplacer un fichier ou un dossier :

```
mv  nom_actuel.txt nouveau_dossier/nouveau_nom.txt
```

## 3.6 CP : Copier un fichier ou un dossier

Copier un fichier :

```
cp  nom_actuel.txt nouveau_nom.txt
```

Copier un dossier :

```
cp -R  nom_actuel nouveau_nom
```

## 3.7 CAT : Afficher le contenu d'un fichier

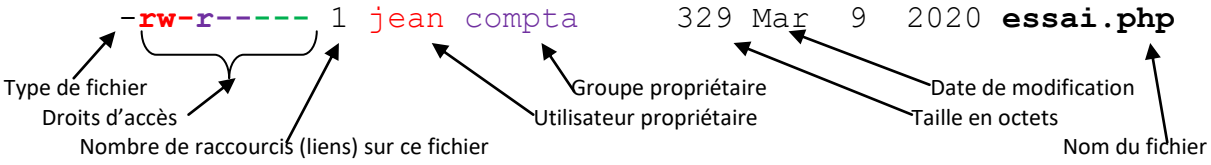
Par exemple pour visualiser le contenu du fichier qui contient le nom des utilisateurs (/etc/passwd) :

```
cat /etc/passwd
```

# 4 Gérer les droits d'accès

## 4.1 Les droits d'accès

On peut visualiser les droits d'accès aux fichiers par la commande LS –L écrite en minuscule !!!!
Exemple pour 1 fichier au hasard :




Dans l'exemple ci-dessus : *Jean* peut lire et modifier (rw-) « essai.php », le groupe *compta* ne peut que le lire (r--), et les *autres* n'ont aucun accès à ce fichier (---) .

Pour l'accès aux fichiers, Linux considère **trois catégories d'utilisateurs** :

- L'utilisateur propriétaire du fichier (fixé par la commande **chown**),
- Les utilisateurs appartenant au groupe propriétaire du fichier (fixé par la commande **chgrp**),
- Les autres utilisateurs...

Pour chacune de ces catégories, trois droits d'accès sont définis : r, w, et x (fixés par la commande **chmod**).
Pour un **fichier**, les droits sont définis comme suit :

- lecture ( r ) Droit de lecture du contenu du fichier
- écriture ( w ) Droit de modifier le contenu du fichier
- exécution ( x ) Droit d'exécuter le fichier (s'il s'agit d'une commande ou d'un shell script...)

Pour un **dossier**, les droits sont définis comme suit :

- lecture ( r ) Droit de lister le contenu du dossier (ls)
- écriture ( w ) Droit de créer ou effacer un fichier
- accès ( x ) Droit de se placer dans le dossier (cd)

## 4.2 CHOWN, CHGRP : Modification des propriétaires :

- Modification du propriétaire : **`chown autre_utilisateur fichier`**
- Modification du groupe d'appartenance : **`chgrp autre_groupe fichier`**
- Modification des 2 en même temps :

**`chgrp autre_utilisateur:autre_groupe fichier`**

NB : L'option –R applique la modification dans les sous-dossiers .
Ex : **`chown –R autre_utilisateur dossier`**

## 4.3 CHMOD : Modification des droits

- Modification des permissions : **`chmod options fichier`**

Les options de la commande chmod peuvent se présenter sous deux formes :
a) Sous forme chiffrée : codage en octal, chaque lettre ayant une valeur numérique.

r = 4 w = 2 x = 1

Exemple : `chmod 750 fichier` donne les droits `rwxr-x---`
Soit : `rwx` (4+2+1=7) : lecture/ écriture / exécution pour l'utilisateur propriétaire
`r-x` (4+0+1=5) : lecture / exécution pour le groupe propriétaire
`---` (0+0+0=0) : aucun droit pour les autres.

NB : L'option –R applique la modification dans les sous-dossiers.
Ex : `chmod -R 750 dossier`

b) Sous forme de lettres :
La lettre **« u »** pour le propriétaire (user) ; « **g** » pour le groupe ; « **o** » pour les autres (others)
Exemple : `u+x` pour ajouter le droit **x** à l'utilisateur
`ugo+rwx` pour tout ajouter à tous

## 4.4 Exercice

- *Amael* et *Mathilde* sont en *premiere* et travaillent dans le dossier *classe1*, *Yann* et *Matthieu* en *terminale* et travaillent dans *classe2*. Réglez les accès des dossiers créés pour que chaque utilisateur accède à son espace et pas à celui des autres.
- Mathilde et Yann travaillent ensemble sur un projet interclasse :
    - Créez un dossier */home/travail/projet* et trouvez une solution pour que chaque utilisateur accède à son travail et pas à celui des autres
- Un utilisateur *jean* a accès à TOUT et peut utiliser la commande **sudo**.

## 4.5 Encore + de réglages ?

Le plus souvent, même s'il est moins sophistiqué que sous Windows, le système de base de gestion des droits est suffisant.

En fait, la sophistication des réglages dépend du TYPE de Système de Fichiers utilisé pour le stockage, et de la version du NOYAU du Système d'Exploitation.

Sous Windows, les types de systèmes de fichiers connus sont FAT32 (ancien) et NTFS (actuel).

Sous Linux, les Systèmes de Fichiers ont évolué aussi : Linux, ext3, ext4, ReiserFS, BtrFS (actuel), …

Actuellement, il est possible d'augmenter les possibilités de réglages grâce

- aux Attributs Spéciaux : droits `t` et `s` (`chmod` avec `1xxx 2xxx 4xxx`)
- aux ACL (Access control List) `getfacl, setfacl`
- aux Attributs étendus : `lsattr`, `chatt` et droits a,A,c,C,d,D,e,F,i,j,P,s,S,t,T,u

Ces possibilités seront étudiées dans une autre séance.