



## MAI – L2/S4 **systèmes d'exploitation**

Cheikh Sidy Mouhamed CISSE Enseignant Associé en Informatique à l'UVS



# Chapitre1 : Système de gestion de fichiers

---

**Objectifs spécifiques :** A la suite de ce chapitre, l'étudiant doit être capable de:

1. Se

# PLAN

---

- Fichier, arborescence et chemin
- Navigation dans le système de fichier
- Gestion des fichiers et répertoires
- Recherche de fichiers
- Gestion des droits sur les fichiers

# Fichier, arborescence et chemin

---

- ❑ **Un fichier** : Collection de données stockées sur disque et qui peut être manipulées comme une seule unité par son nom
- ❑ **Un répertoire** : Fichier qui contient les références d'autres fichiers ou répertoires
- ❑ **Arborescence** : L'ensemble des fichiers et répertoires constituent de Linux l'arborescence de Linux.
- ❑ **Racine (/)** : Le répertoire père de tous les autres répertoires (Prononcer 'Root')

# Fichier, arborescence et chemin

❑ Rôle de quelques répertoires de l'arborescence de Linux :

<b>/</b>	: Racine du système	<b>/sbin</b>	: Commandes de base nécessaires à l'administration système
<b>/bin</b>	: contient les commandes de bases (grep, mount, cp . . . ).	<b>/tmp</b>	: Répertoire temporaire. Contenu susceptible d'être supprimé à chaque redémarrage
<b>/dev</b>	: contient les fichiers spéciaux correspondant aux périphériques	<b>/usr</b>	: Répertoire d'installation des logiciels
<b>/etc</b>	: contient la plupart des fichiers de configuration.	<b>/var</b>	: des données fréquemment réécrites, comme les logs.
<b>/home</b>	: contient les répertoires personnels des utilisateurs	<b>/mnt</b>	: ou <b>/media</b> montage de périphériques de stockage (clés USB ou autres)
<b>/lib</b>	: contient les principales bibliothèques partagées		
<b>/root</b>	: le répertoire personnel et privé de root		

# Fichier, arborescence et chemin

---

- ☐ Les chemins permettent de définir un emplacement au sein du système de fichiers.
- ☐ C'est la liste des répertoires et sous répertoires empruntés pour accéder à un endroit donné de l'arborescence jusqu'à la position souhaitée (répertoire, fichier).
- ☐ Un nom de fichier est ainsi généralement complété par son chemin d'accès.

# Fichier, arborescence et chemin

---

- ❑ C'est ce qui fait que le fichier toto du répertoire rep1 est différent du fichier toto du répertoire rep2.
- ❑ Le FS d'Unix étant hiérarchique, il décrit une arborescence.

# Fichier, arborescence et chemin

---

❑ On distingue des types de chemins d'accès aux fichiers parmi lesquels:

➤ **Les chemins d'accès absolus:** Pour tout fichier présent dans le système, il existe un chemin de la racine (/) et aboutissant à ce fichier. La liste des nœuds rencontrés identifie sans aucune ambiguïté le fichier. Le chemin ainsi parcouru s'appelle chemin absolu.

❑ Commencent obligatoirement par /.

❑ `/home/toto/Docs/Backup/fichier.pdf`



# Fichier, arborescence et chemin

---

- ❑ Un chemin absolu ou complet :
  - démarre de la racine, donc commence par un /,
  - décrit tous les répertoires à traverser pour accéder à l'endroit voulu,
  - ne contient pas de . ni de ..

# Fichier, arborescence et chemin

---

- ❑ **Les chemins d'accès relatifs:** La référence relative d'un fichier est la liste des nœuds rencontrés pour aboutir à ce fichier à partir de l'endroit où on se trouve (working directory).

**Ne commencent jamais par /**

- ❑ Il faut souvent utiliser deux entrées particulières de répertoires :
  - Le point . représente le répertoire courant, actif. Il est généralement implicite.
  - Les doubles points .. représentent le répertoire de niveau inférieur.

# Fichier, arborescence et chemin

---

- ❑ **Documents/Photos** est un chemin relatif : le répertoire Documents est considéré comme existant dans le répertoire courant ;
- ❑ **./Documents/Photos** est un chemin relatif parfaitement identique au précédent, sauf que le répertoire actif (courant) est explicitement indiqué par le point.
- ❑ « **./Documents** » indique explicitement le répertoire Documents dans le répertoire actif;
- ❑ **/usr/local/./bin** est un chemin relatif : les .. sont relatifs à /usr/local et descendent d'un niveau vers /usr. Le chemin final est donc /usr/bin.

# Fichier, arborescence et chemin

---

- ❑ On parle de **chemin absolu** lorsque les chemins sont référencés par la liste des répertoires et sous répertoires qui permettent d'y accéder depuis la racine.
- ❑ On parle de **chemin relatif** lorsque le chemin est exprimé relativement au répertoire courant (./, ../, ../../).

# Navigation dans le système de fichier

---

- ☐ **dirname** : permet de récupérer le répertoire dans le chemin complet d'un fichier
- ☐ **basename** : récupère au contraire le nom du fichier
- ☐ **pwd (print working directory)** : affiche le répertoire courant

# Navigation dans le système de fichier

---

Exemples :

```
> dirname /usr/include/linux/socket.h  
/usr/include/linux
```

```
> basename /usr/include/linux/socket.h  
socket.h
```

```
> pwd  
/home/toto
```

# Navigation dans le système de fichier

---

**cd ..** : Remonte d'un niveau

**cd /** : Retourne à la racine

**cd -** : Retourne au répertoire précédent

# Navigation dans le système de fichier

---

- ❑ La commande **ls** permet de lister le contenu d'un répertoire (catalogue) en lignes ou colonnes.
- ❑ Les types de fichier sous UNIX sont:
  - "-": Fichier ordinaire. "d": Répertoire. "b": Fichier spécial mode bloc.
  - "c": Fichier spécial mode caractère. "n": Fichier spécial mode réseau.
  - "l": Lien symbolique. "p": Pour une fifo (pipe nommé, named pipe )
- ❑ La commande ls supporte plusieurs paramètres dont voici les plus pertinents



# Navigation dans le système de fichier

---

- ❑ La commande *tree* permet d'afficher la structure arborescente d'un répertoire (sous-répertoires et fichiers ordinaires)

# Gestion des fichiers et répertoires

---

- ❑ Création de répertoires : ***mkdir***

*mkdir [options] nom\_repertoire\_1 nom\_repertoire\_2 ... nom\_repertoire\_n*

- ❑ **Création d'un répertoire (chemin relatif)**

*mkdir un-repertoire un-autre-repertoire un-troisieme-repertoire*

- ❑ **Création d'un répertoire avec un chemin complet**

*mkdir /home/toto/Documentes/un-repertoire*

- ❑ **Création d'un répertoire et créer les répertoires parent s'ils n'existent pas**

*mkdir -p /home/toto/Documentes/ancetre/grand-pere/pere/un-repertoire*

# Gestion des fichiers et répertoires

---

❑ La commande ***mv*** permet

- de déplacer un fichier
- de renommer un fichier (sous Linux renommer un fichier se fait par son déplacement sous un autre nom dans le même répertoire).

## Syntaxe

❑ Déplacement de fichier

- *mv file\_1 file\_2 ... repertoire-destination*

❑ Renommage de fichiers

- *mv nom-originale-de-fichier nouveau-nom*

# Suppression de fichiers et répertoires : rm

---

□ La commande ***rm(remove)*** permet de supprimer des fichiers ou répertoires

➤ *rm [-i] nom-fichier-ou-repertoire-vidé*

➤ *rmdir nom-repertoire-vidé*

*Suppression recursive de repertoire non vide*

➤ *rm [-i] -rf nom-repertoire*

➤ *rmdir --ignore-fail-on-non-empty nom-repertoire*

# Copie de fichiers et répertoires : cp

---

## ❑ Syntaxe

***cp source destination***

## ❑ Quelques options intéressantes :

- -a --archive : copie avec préservation des attributs des fichiers et répertoires
- -i, --interactive : demande avant d'écraser la destination
- -R, -r, --recursive : copie récursive (de fichiers et répertoires)
- -u, --update : Copie seulement si le nouveau fichier est plus récent que la destination
- -v, --verbose : Affiche ce qui est en cours de copie

# Création de fichiers : touch

❑ La commande *touch* permet de :

➤ créer un fichier vide

```
> touch hello_word_file
```

```
> ls -l hello_word_file
```

```
-rw-r--r-- 1 toto toto 0 2011-08-24 18:24 hello_word_file
```

➤ ou de mettre à jour la date du dernier accès et de la dernière modification

```
> ls -l hello_word_file
```

```
-rw-r--r-- 1 toto toto 0 2011-08-24 18:24 hello_word_file
```

```
> touch hello_word_file
```

```
> ls -l hello_word_file
```

```
-rw-r--r-- 1 toto toto 0 2011-08-24 18:27 hello_word_file
```

# Recherche de fichiers avec : find

---

- ❑ La commande ***find*** permet de rechercher un fichier selon plusieurs critères
  - Nom, type de fichier
  - Taille
  - UID,GID
  - Permissions d'accès
  - Heure d'accès
  - Type de système de fichiers



## Recherche de fichiers avec : find

## Syntaxe de *find*

```
find [options]... [<search-root>] <search-expression>
```



# Gestion des droits sur les fichiers

---

- ❑ Chaque fichier ou répertoire se voit attribuer des droits qui lui sont propres, des autorisations d'accès individuelles.
- ❑ Lors d'un accès le système vérifie si celui-ci est permis.
- ❑ À sa création par l'administrateur, un utilisateur se voit affecter un UID(User Identification) unique
  - Les utilisateurs sont définis dans le fichier */etc/passwd*
- ❑ De même chaque utilisateur est rattaché à un groupe au moins (groupe principal), chaque groupe possédant un identifiant unique, le GID (Group Identification)
  - Les groupes sont définis dans */etc/group*

# Gestion des droits sur les fichiers

- ❑ À chaque fichier (*inode*) sont associés un UID et un GID définissant son propriétaire et son groupe d'appartenance.
- ❑ On distingue trois cas de figure :
  - UID de l'utilisateur identique à l'UID défini pour le fichier. Cet utilisateur est propriétaire du fichier.
  - Les UID sont différents : le système vérifie si le GID de l'utilisateur est identique au GID du fichier. Si oui l'utilisateur appartient au groupe associé au fichier.
  - Dans les autres cas (aucune correspondance) : il s'agit du reste du monde (others), ni le propriétaire, ni un membre du groupe

# Gestion des droits sur les fichiers

Droit	Signification
<b>Général</b>	
r	Readable (lecture).
w	Writable (écriture).
x	Executable (exécutable comme programme).

# Gestion des droits sur les fichiers

Fichier normal	
r	Le contenu du fichier peut être lu, chargé en mémoire, visualisé, recopié.
w	Le contenu du fichier peut être modifié, on peut écrire dedans. La suppression n'est pas forcément liée à ce droit (voir droits sur répertoire).
x	Le fichier peut être exécuté depuis la ligne de commande

# Gestion des droits sur les fichiers

Répertoire	
r	Les éléments du répertoire (catalogue) sont accessibles en lecture. Sans cette autorisation, ls et les critères de filtre sur le répertoire et son contenu ne sont pas possibles. L'accès individuel à un fichier reste possible si vous connaissez son chemin.
w	Les éléments du répertoire (catalogue) sont modifiables et il est possible de créer, renommer et supprimer des fichiers dans ce répertoire. <b>C'est ce droit qui contrôle l'autorisation de suppression d'un fichier.</b>
x	Le catalogue peut être accédé par CD et listé. Sans cette autorisation il est impossible d'accéder au répertoire et d'agir sur son contenu qui devient verrouillé.

Ainsi pour un fichier :

rwX	r-X	r--
Droits de l'utilisateur, en lecture, écriture et exécution.	Droits pour les membres du groupe en lecture et exécution.	Droits pour le reste du monde en lecture uniquement.

# Gestion des droits sur les fichiers

---

- ❑ La commande `chmod` (change mode) permet de modifier les droits sur un fichier ou un répertoire.
- ❑ Il existe deux méthodes pour modifier ces droits :
  - par la forme symbolique
  - Par la base 8.
- ❑ Seul le propriétaire d'un fichier peut en modifier les droits (plus l'administrateur système).

# Gestion des droits sur les fichiers

---

## ❑ Par symbole

- La syntaxe est la suivante :
- `chmod modifications Fic1 [Fic2...]`
- `u` : modifie les droits de l'utilisateur
- `g` : modifie les droits du groupe
- `o` : modifie les droits des autres



# Gestion des droits sur les fichiers

---

## ❑ Par symbole

- Pour ajouter des droits on utilise le caractère +
- Pour retirer des droits, le caractère –
- Le caractère = permet de ne pas tenir en compte le + et le –
  - `$ chmod g+w fichier1`
  - `$ chmod u=rwx,g=x,o=rw fichier2`
  - `$ chmod o-r fichier3`
- Pour supprimer tous les droits : `$ chmod o=fichier2`



# Gestion des droits sur les fichiers

## ❑ Par base de 8

➤ Voici un tableau récapitulatif

Propriétaire			Groupe			Reste du monde		
r	w	x	r	w	x	r	w	x
400	200	100	40	20	10	4	2	1

# Gestion des droits sur les fichiers

## ❑ Par base de 8

- Pour obtenir le droit final il suffit d'additionner les valeurs.
- Par exemple:
- Pour *rwxrwrw* alors obtenez  $400+200+100+40+20+4+2=766$
- pour *rw-r--r--*  $400+200+40+4=644$
  
- `$ chmod 755 fichier1`
- `$ chmod 644 fichier2`
- Pour verification: `$ ls -l fichier1 fichier2`