Chapitre 2

LE SYSTEME DE FICHIERS D'UNIX

Objectifs:

① Connaître la structure du système de fichiers d'UNIX.

② Manipuler les commandes de base du système de fichier d'UNIX.

Prérequis:

① Les chapitres précédents de ce cours d'UNIX.

Plan

Chapitre 2

LE SYSTEME DE FICHIERS D'UNIX

I/ Structure du système de fichier

Définition:

Un fichier est une suite d'octets, stockées sur une mémoire auxiliaire. Le système UNIX distingue plusieurs types de fichiers.

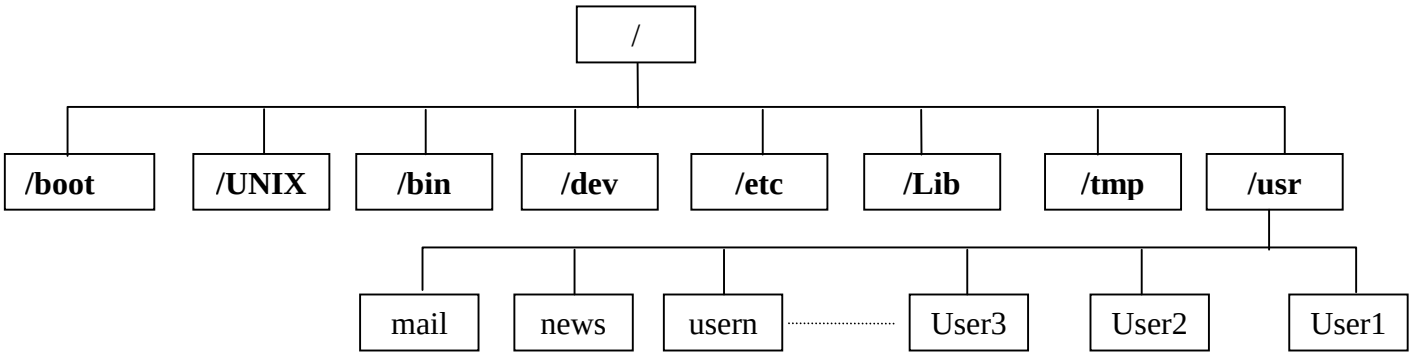
- les fichiers normaux (ordinary files)
- les fichiers répertoires (répertoire ou directory)
- les fichiers spéciaux (special files ou devices)

Chaque fichier possède un nom, un contenu, un endroit où se trouve, sont propriétaire, sa taille et les personnes qui peuvent y accéder.

I-1/ les fichiers normaux : contiennent soit des textes soit des programmes exécutables .

I-2/ les fichiers répertoires: permettent d'organiser l'espace du disque dur. Les fichiers normaux sont regroupés dans des répertoires. Ces répertoires peuvent contenir eux-mêmes des sous-repertoires, des fichiers normaux et des fichiers spéciaux.

I-3/ les fichiers spéciaux : représentent les interfaces avec les périphériques gérés par le système d'exploitation. Exemple: console, imprimante et disque.



/boot : Premier programme exécuté par la machine, il permet de lancer UNIX.

/UNIX : Programme du KERNEL (résident en mémoire). Au fonctionnement, /UNIX est lu, copié dans la mémoire et exécuté.

/bin : Contient les utilitaires et programmes exécutables comme cat, date, wc, who, ...

/dev : Contient les fichiers spéciaux qui représentent des périphériques comme console, ttyxx, lp, ...

/etc : Contient des programmes et fichiers de données pour l'administration du système comme le fichier des mots de passe "passwd" et le fichier des membres de chaque groupe d'utilisateur group.

/Lib : Bibliothèque de programmes et de langages.

/tmp : contient des fichiers temporaires

/usr : /news : Nouveautés dans le domaine d'UNIX

/mail : Boite à lettres

/userx : Répertoire de l'utilisateur userx.

II/ Répertoire de travail

Au moment du login, l'utilisateur est placé sur un répertoire propre à lui, relié au répertoire système /usr. On appelle ce répertoire, le répertoire de départ ou répertoire de connexion.

A partir de sont répertoire de départ, tout utilisateur peut créer ou modifier ses propres fichiers et répertoires qui doivent êtres situés dans la partie inférieure de l'arborescence à partir du répertoire de départ. L'utilisateur peut aussi remonter le système de fichiers jusqu'à la racine ou se positionner sur un répertoire issu de sont répertoire appelé répertoire de travail ou répertoire courant. Chaque fichier ou répertoire est décrit par sont chemin d'accès par rapport au répertoire courant ou par rapport à la raine.

Exemple de chemin d'accès:

/usr/user2/fich1 :Chemin d'accès de fich1 par rapport à la racine.

fich1 : Chemin d'accès de fich1 par rapport au répertoire de connexion de user2.

/usr/spool :Chemin d'accès du fichier spool par rapport à la racine.

Pour connaître le répertoire courant, ou le répertoire de travail, on dispose de la commande **pwd**.

```
$pwd
```

Pour se déplacer dans l'arborescence et changer de répertoire, on utilise la commande **cd** suivie du chemin d'accès du répertoire.

```
$Cd /usr
$pwd
$cd /usr/spool
$pwd
```

pour remonter d'un niveau dans l'arborescence:

```
$cd ..
$pwd
$cd userx
$pwd
```

III/ La manipulation des répertoires

III-1/création d'un répertoire

Pour créer un répertoire, on utilise la commande **mkdir** suivie du nom du répertoire (ou des noms des répertoires) à créer

```
$mkdir trav
$mkdir temps pers
```

III-2/ contenu d'un répertoire

La commande **ls** permet de lister le contenu d'un répertoire en fichiers et sous-repertoire sans distinction

Exemple:

```
$ls
   fich1
   pers
   temps
   trav
   virus
```

Pour distinguer entre les fichiers et les répertoires on utilise ls avec l'option -p. Dans ce cas les répertoires serons terminés par /

Exemple:

```
$ls
   fich1
   pers/
   temps/
   trav/
   virus
```

Pour obtenir une description plus détaillée du contenu du répertoire, on utilise ls avec l'option -f.

```
$Ls -l
-rw-rw--    1    userx   group   12 May    29    10:64 fich1
drwxrwxr-x  2    userx   group   32 Jun    3     10:45 virus
```

La première colonne représente le mode du fichier:

si le premier caractère est d : répertoire

-: fichier ordinaire

b,c,p: fichier spécial.

Les neufs (9) caractères du champs mode décrivent les permissions accordées à ces fichiers et répertoires (à voir ultérieurement). En suite, on obtient le nombre d'octets dans chaque fichier et répertoire, la et l'heure de sa dernière modification et son nom.

```
$Ls -l  /
```

Dans le cas d'un fichier spécial, le champ indiquant le nombre d'octets est remplacé par deux paramètres, le numéro majeur et le numéro mineur. Le numéro majeur permet de codifier le type du périphérique (exemple : 0 pour console, 1 pour les terminaux tty, 2 pour les disques ....etc.) tandis que le numéro mineur permet de différencier les périphériques de même type par des chiffres distincts.

```
$Ls -l /dev
```

III-3/ suppression d'un répertoire

La suppression se fait par la commande **rmdir**:

Exemple:

```
$rmdir temp
$ls -p
```

La même tâche peut être faite avec la commande **rm** :

Exemple

```
$mkdir temp
$ls -l
$rm -r temp
$ls -l
```

III-4/ taille des répertoires

La commande **du** (**d**isk **u**sage) fournit la taille en espace disque occupée par les répertoires et les sous-repertoires.

Sa syntaxe est :

$ du -option répertoire

les options peuvent êtres:

a: taille de chaque fichier des répertoires en question.

R: générer des messages pour les fichiers ou répertoires non lus

s: seules les tailles totales des répertoires sont données.

Par défaut, la taille de chaque répertoire et sous-repertoire seront données en blocs. La commande **du** interprète chaque bloc de 1024 octets en deux blocs de 512 octets (par exemple, un fichier ou répertoire de 500 octets est interprété comme 2 blocs).

Exemple

```
$du
        1       .pers
        1       .trav
        20      .tp
```

IV/ La manipulation des fichiers

IV-1/ création simple d'un fichier répertoire

Il est possible de créer un fichier ordinaire soit en utilisant un éditeur de texte comme **vi,** soit en utilisant la commande **cat** et la redirection.

Exemple:

```
$ cat  >  nom du fichier
ligne1
ligne2
^d
```

Une fois la ligne est validée, tout ce qui sera saisie sera copié sur le fichier qui peut être fermé et sauvegardé dans le répertoire indiqué en tapant ^d.

Exemple :

```
$ cat fich2
Ce fichier est composé de deux lignes
et sera sert à tester la commande cat.
^d
```

```
$ls -l
-rw-rw-r--   1  userx   group  12   May   29   10:45  fich1
-rw-rw-r--   1  userx   group  68   Jun   3    11:16  fich2
drwxrwxrwx   2  userx   group  32   Jun   3    10:15  temp
```

IV-2/ lecture d'un fichier

Il est possible de lire un fichier ordinaire soit en utilisant un éditeur de texte comme **vi,** soit en utilisant la commande **cat** .

Exemple :

```
$cat fich1
$cat fich1  fich2
```

Dans le cas de fichiers volumineux, on remarque que le défilement sur l'écran est rapide. Donc, il faut y avoir du temps de pause pour l'affichage. Pour cela on peut utiliser la commande **pg**.

pg permet l'affichage de un ou plusieurs fichiers page par page.

```
$pg  nom de fichier
```

pg attend des instructions supplémentaires qui peuvent être:

```
h: voir toutes les possibilités de pg
L: afficher la ligne suivante
<↵>: Afficher page suivante
.: afficher la page courante
f: afficher la page d'après
$: Afficher la dernière page
+iL: Afficher les i lignes suivantes
/chaine/ : Rechercher en avant d'une chaîne
?chaîne?: Rechercher en arrière d'une chaîne
```

Exemple:

```
$pg fich1
<texte>
:h
:L
:+3L
:.
```

:<↵>

:$

:q

$

$pg -5 +15 fich1 : affiche fich1 toutes les 5 lignes à partir de la 15ème ligne.

IV-3/ duplication de fichiers

Pour dupliquer un fichier, il suffit d'utiliser la commande **cp**.

Syntaxe:

$cp source(s) destination

source(s) = un ou plusieurs fichiers

Exemples:

1- copie de deux fichiers sur un autre répertoire en gardant les mêmes noms:

$cp virus fich2 trav

$ls trav

2- Duplication d'un fichier sur le même répertoire de travail avec un nom diffèrent:

$cp fich2 essai.

$ ls -p

$cat essai

3- Duplication d'un répertoire sur un autre:

$cd trav

$ls

$cp * /usr/userx/pers ou $ cp * ../pers

$ls ../pers

$cd ..

Question: *Comment peut-on réaliser le dernier exemple sans changer de répertoire*?

IV-4/ déplacement de fichiers

Pour déplacer un fichier, il suffit d'utiliser la commande **mv** (move).

Syntaxe:

$mv source(s) destination

Exemples:

1- Déplacement d'un fichier d'un répertoire a un autre sans changement de nom:

$mv essai trav/essai

$ls -l

```
$ls -l trav
2- Déplacement d'un fichier d'un répertoire a un autre avec changement de nom:
$mv trav/essai  essai1
$ls -l
$ls -l trav
```

Attention: Si le fichier de destination existe déjà, la commande mv remplacera son contenu par celui du fichier source.

IV-5/ suppression de fichiers

Pour supprimer un fichier, on utilise la commande **rm** (rermove)

Syntaxe:

```
$rm [-option] fichier(s)
```

Exemples:

```
$rm trav/fich2  pers/fich2
$ls -p trav  pers
```

L'utilisation de l'option i permet au système de poser une question de validation avant d'effacer le fichier.

Exemples:

```
$rm -i pers/virus
pers/virus: ? n
$ls pers
$rm -i pers/virus
pers/virus: ? y
$ls pers
```

IV-6/ autres opérations sur les fichiers

La commande file permet de tester sur le contenu d'un fichier. Cette commande permet de deviner le type de fichier invoqué.

Syntaxe:

```
$file  fichier
```

Exemples:

```
$file /bin
/bin : directory
$file /bin/ed
```

```
/bin/ed : cannot open for reading
$file essai1
essai1 : data
```

V/ Conclusion:

Dans ce chapitre nous avons étudié le système de fichier d'UNIX du point de vue de l'utilisateur. En effet, nous avons défini les différents types de fichiers, la structure générale du système de fichier (organisation hiérarchique des répertoires) et les commandes de base pour la manipulation du système de fichier d'UNIX. Par contre, nous n'avons pas évoqué la notion de sécurité qu'offre le système UNIX. Dans le chapitre suivant, on étudiera ce point.