

Systèmes d'exploitation

Module SysExp

2ème cours - UNIX

1

Principes d'un File System

- Un File System Unix manipule des inodes
- Un inode permet au FS de faire correspondre un ou plusieurs fichiers (le côté utilisateur) à un ou plusieurs blocs sur le disque dur (le côté matériel)
- Lorsqu'un nouveau fichier est créé dans le système, un numéro de inode libre est réservé et à ce numéro sont associés le/les blocs correspondants

4

Système de fichiers UNIX

- Objectifs principaux
 - assurer la subsistance des données sur un support informatique (aujourd'hui disques durs) en rendant transparent l'accès aux blocs de données
 - hiérarchisation des informations de manière à faciliter l'enregistrement et l'accès aux données
 - système permettant d'assurer la sécurité de l'accès aux données, par l'utilisation de droits d'accès
- Exemples: ufs, ext3, reiserfs, hfs+, zfs

2

Exemple de table d'inodes

n°	blocs
1	43, 10, 20, 7
2	3
3	11, 13, 45
4	
5	8, 18, 21, 37, 25
6	5, 41
7	
...	

5

Principes d'un File System

- Le FS est un ensemble de méthodes (fonctions, procédures) qui permettent au système d'exploitation de stocker des fichiers
- Mais c'est surtout une structure de données (table des inodes), stockée au tout début de chaque partition d'un disque dur
- Cette structure est une sorte de carte géographique qui indique ce que contient chaque bloc de la partition du disque dur

3

Fichiers sous UNIX

- Fichiers ordinaire
- Répertoires
- Fichiers spéciaux pour l'accès aux périphériques
 - fichiers de type bloc
 - fichiers de type caractères
- Liens symboliques & Pipes nommés (fifo)

6

Fichier ordinaire

- Contient des données de type caractère, sous forme d'octets
- Ces données sont stockées physiquement sur le disque dans les blocs de données, par "tranches" de 1024 octets (taille d'un bloc)
- Grâce à la structure de données du FS (inodes), le système d'exploitation peut savoir où sur le disque (dans quels blocs) se situe chaque partie du fichier

7

Fichiers spéciaux

- Au lieu de faire référence à des blocs sur le disque dur, les fichiers spéciaux (type bloc ou caractères) permettent l'accès direct à des adresses mémoire du système
- Ils permettent d'avoir accès aux gestionnaires de périphériques du système
- Cette méthode permet de garantir le postulat à la base de UNIX: tout est fichier

10

Répertoire

- Contient une liste associant à chaque ligne un nom avec un numéro d'inode
- C'est donc l'appartenance d'un fichier ordinaire (par son inode) à un répertoire qui permet de définir le nom de ce fichier
- Un même fichier ordinaire peut donc être contenu dans plusieurs répertoires différents (liens hard)
- Un répertoire peut bien évidemment contenir d'autres répertoires (structure hiérarchique)

8

Liens symboliques & pipes nommés

- Les liens symboliques sont les correspondants des alias (Mac OS) et des raccourcis (Windows)
- Ils permettent de contourner certaines limitations des liens hard
- Les pipes nommés permettent principalement de faire des échanges de données entre des processus par l'intermédiaire de fichiers nommés

11

Exemple de répertoire

n° inode	nom du fichier
24	.
23	..
26	fichier_1.txt
5	fichier_2.xml
26	fichier_5-idem_1.txt
27	image_1.png
18	répertoire_1
35	script_1.sh

9

Droits d'accès

- Pour garantir la sécurité des données, il existe les droits d'accès
- Ils sont séparés en trois niveaux distincts:
 - les droits du propriétaire (User)
 - les droits du groupe (Group)
 - les droits de tout le monde (Other)
- Ils peuvent être de 3 types différents:
 - lecture (Read) écriture (Write) et exécution (eXecute)

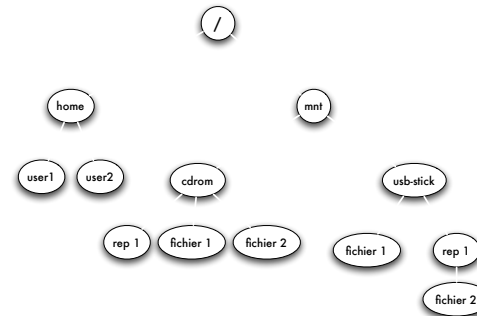
12

Droits d'accès

	Fichiers	Répertoires
Lecture	autorise la lecture du fichier	permet de lister le contenu du répertoire
Écriture	permet la modification du fichier	permet la création et la suppression de fichiers du répertoire Attention: cette permission est valable quels que soit les droits des fichiers!
Exécution	autorise l'exécution du fichier (script ou programme)	permet de se positionner dans le répertoire

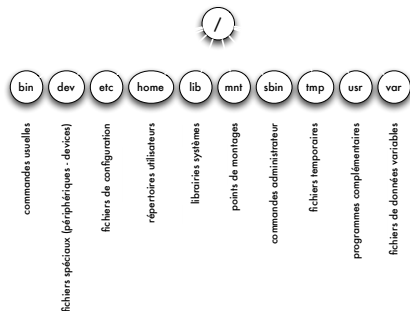
13

Volumes montés



16

Hiérarchie standard UNIX



<http://www.pathname.com/fhs/pub/fhs-2.3.html>

14

conventions de nommage

- sous UNIX, les noms de fichiers peuvent contenir théoriquement n'importe quel caractère.
- cependant, pour éviter d'être coincé, les espaces et caractères accentués ne sont généralement jamais utilisés.
- un fichier commençant par '.' est considéré comme invisible => pas affiché par la commande ls (il faut utiliser l'option -a)

17

Hiérarchie UNIX

- Sous Unix, il n'y a qu'une seule hiérarchie de fichiers. Tout part de la racine (root): /
- Lorsqu'un volume est monté sur un système Unix, il s'ajoute à la hiérarchie de base, généralement dans /mnt
- Tous les fichiers du système sont donc accessibles par la racine.

15

Comment interagir?

- Par une interface graphique
- Par une ligne de commande (shell)

18

Interface graphique

- Exemples: KDE, Gnome, etc.
- Avantages:
 - très user-friendly
 - pas besoin de connaître la structure du système (principes génériques communs à tous les OS)
- Désavantages:
 - accès à distance difficiles (vnc)
 - pas beaucoup de liberté de configuration

19

Fonctionnement

- Un shell est un interpréteur de commande
- Une commande est structurée ainsi:

commande [argument(s)] <retour à la ligne>

- Chaque commande à une syntaxe propre qu'il faut apprendre à connaître

22

Ligne de commande (Shell)

- Exemples: sh, bash, ash, zsh, ksh, csh, tcsh,
- Avantages:
 - accès direct au kernel
 - accès à distance très aisé
 - contrôle de toutes les facettes du système
- Désavantages:
 - phase d'apprentissage indispensable

20

man

- man
 - permet d'obtenir le mode d'emploi d'une commande
- Exemples:
 - man ls => affiche la page de manuel de la commande 'ls'
 - man man => affiche la page de manuel de la commande man

23

Principes d'un Shell

- Par un login (utilisateur + mdp) permet d'avoir accès au système
- Lors du login, l'utilisateur se retrouve dans son dossier utilisateur, à l'intérieur du système de fichiers
- Basé sur un système de commandes (internes et externes)
- Mise en place d'un dialogue dynamique entre l'utilisateur et la machine

21

pwd

- affiche la position actuelle dans le système de fichiers
- lors du login, la position actuelle est automatiquement le répertoire home de l'utilisateur loggé
- Exemple: /home/benoit

24

cd

- permet de se déplacer dans l'arborescence
- positionnement relatif / absolu
- Exemples:
 - `cd` se positionne dans le home
 - `cd ~` idem
 - `cd -` revient à l'endroit précédent
 - `cd /home` se positionne de manière absolue dans le répertoire "home" situé juste en dessous de la racine
 - `cd benoit` se positionne de manière relative dans le répertoire "benoit" du répertoire courant (home)

25

mkdir

- permet de créer un répertoire
- options utiles:
 - `-p` crée les répertoires intermédiaires s'ils n'existent pas
- Exemple:
 - `mkdir -p test/rep1/rep2`

28

ls

- permet de lister le contenu du répertoire courant
- options importantes:
 - `-l` affiche sous forme de liste avec toutes les informations des fichiers
 - `-a` affiche même les fichiers invisibles
 - `-i` affiche les numéros d'inode des fichiers
 - `-h` affiche les tailles de fichiers "humainement" (en octets, Ko, Mo, Go, etc.)

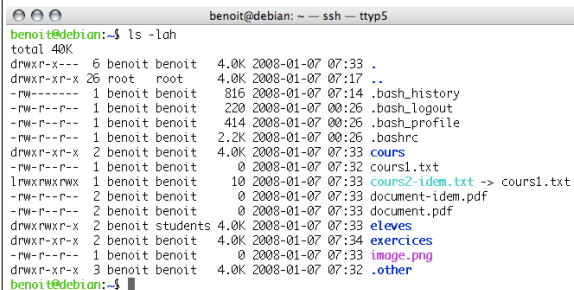
26

rmdir

- permet de supprimer un répertoire
- cette commande ne fonctionne que si le répertoire est vide (ne contient aucun autre fichier)
- options utiles:
 - `-p` supprime les répertoires intermédiaires s'ils sont vides
- Exemple:
 - `rmdir -p test/rep1/test2`

29

listage de fichiers



```
benoit@debian:~$ ls -lah
total 40K
drwxr-xr-x  6 benoit benoit 4.0K 2008-01-07 07:33 .
drwxr-xr-x 26 root  root  4.0K 2008-01-07 07:17 ..
-rw-r--r--  1 benoit benoit 816 2008-01-07 07:14 .bash_history
-rw-r--r--  1 benoit benoit 220 2008-01-07 00:26 .bash_logout
-rw-r--r--  1 benoit benoit 414 2008-01-07 00:26 .bash_profile
-rw-r--r--  1 benoit benoit 2.2K 2008-01-07 00:26 .bashrc
drwxr-xr-x  2 benoit benoit 4.0K 2008-01-07 07:33 cours
-rw-r--r--  1 benoit benoit  0 2008-01-07 07:32 cours1.txt
lrwxrwxrwx  1 benoit benoit 10 2008-01-07 07:33 cours2-idem.txt -> cours1.txt
-rw-r--r--  2 benoit benoit  0 2008-01-07 07:33 document-idem.pdf
-rw-r--r--  2 benoit benoit  0 2008-01-07 07:33 document.pdf
drwxrwxr-x  2 benoit students 4.0K 2008-01-07 07:33 eleves
drwxr-xr-x  2 benoit benoit 4.0K 2008-01-07 07:34 exercices
-rw-r--r--  1 benoit benoit  0 2008-01-07 07:33 image.png
drwxr-xr-x  3 benoit benoit 4.0K 2008-01-07 07:32 .other
benoit@debian:~$
```

27

rm

- permet de supprimer un fichier
- options utiles:
 - `-r` permet de supprimer recursivement des fichiers / répertoires
- cette option est très puissante et dangereuse car elle force les répertoires non-vides à être supprimés!
- la commande `rm -r /` exécutée en tant que root efface tout l'ordinateur!!!

30

mv

- permet de déplacer ou renommer un fichier
- ne sont autorisées que les manipulations sur des fichiers dont les droits correspondent à ceux de l'utilisateur.

31

exercices

- login en ssh sur serveur debian
- pour Mac
 - utiliser ssh & scp
- pour Windows:
 - télécharger putty
 - <http://www.chiark.greenend.org.uk/~sgtatham/putty>
 - PuTTY et PSCP

34

cp

- permet de copier des fichiers ou répertoires d'un endroit à une autre
- option importantes:
 - -r fait une copie récursive (pour les répertoires)
 - -v affiche tous les fichiers copiés sur le shell

32

ln

- permet de créer un lien d'un fichier
- exemple: `ln fichier1.txt fichier2.txt`
- option importantes:
 - -s fait un lien symbolique
- exemple: `ln -s /etc/passwd utilisateurs`

33