Chapitre 11 : Données

Gestion des volumes et des fichiers

11.1 – Volumes

Nom donné à un espace de stockage de fichiers sur un disque dur (ou assimilé) : c'est généralement une partition dans un disque

(Plan de ce cours)

- Volumes
  - Partitions (très technique)
  - Formats (à connaître)
  - Montage (à savoir)
  - Contenu d'un volume : liens, fichiers spéciaux...

Structure d'un disque dur

- Vu du BIOS ou du pilote, un disque dur est composé de secteurs = tableaux de N octets avec N=512 en général, sauf SSD : 1024 octets
  - PB : confusion entre secteurs et blocs, voir plus loin
  - Les secteurs sont regroupés en « cylindres » : accès plus rapide sur le même cylindre
- Ces secteurs ont un numéro appelé LBA (logic block address)
- Le BIOS est capable de lire ou écrire n'importe lequel de ces secteurs : accès « aléatoire »

Vue Unix

- Dans un système Unix, un disque dur (ou SSD, ou clé USB, ou CD-ROM, etc.) est représenté par un fichier spécial dans /dev :
  - Disques IDE (PATA), sur les vieilles machines :
    - /dev/hda pour le disque primaire maître,
    - /dev/hdb pour le disque primaire esclave,
    - /dev/hdc pour le disque secondaire maître,
    - /dev/hdb pour le disque secondaire esclave
  - CD-ROM IDE (PATA aussi) : /dev/cdrom0, /dev/cdrom1...

Vue Unix, suite

- Depuis quelques années, on a :
  - Disques et CD/DVD SATA (Serial ATA)
  - Disques SCSI (ancien mais très performant)
  - Clés ou disques amovibles USB

  Tous sont représentés par la même famille de noms :
  - /dev/sda pour le premier disque
  - /dev/sdb pour le deuxième
  - etc.

Vue Unix, fin

- Un disque entier est donc composé de secteurs de 512 octets et vu sous la forme d'un fichier spécial, ex : /dev/sda
  - Lire ou écrire ce fichier = lire ou écrire le disque
  - Le fichier est continu : pas de frontière entre secteurs
- La commande dd permet de lire l'un ou l'autre des secteurs donné par son LBA, exemple :

  `dd if=/dev/sda ibs=512 skip=`*`LBA`*` ...`

  Voir plus loin un exemple complet

Structure d'un disque entier

- Le tout premier secteur s'appelle le Master Boot Record et a un rôle particulier : permettre le démarrage du système d'exploitation
  - Il contient un petit programme en langage machine capable de charger grub (ou winload.exe) appelé *boostrap*
  - La perte du MBR => démarrage impossible
- Le BIOS recopie en mémoire le MBR puis exécute le programme qu'il contient

Affichage du MBR

NB : manip pas utile en soi, mais pour la curiosité :

- Voici comment le consulter :

1) Recopie du MBR dans un fichier

```
dd if=/dev/sda ibs=512 skip=0 count=1 of=mbr.bin
```

- dd : copie directe d'octets entre fichiers
- if = fichier d'entrée à lire, ibs = taille des blocs à lire
- skip = LBA du secteur qu'on veut lire
- count = nombre de blocs de ibs octets à lire
- of = fichier de sortie, qui reçoit les *count*ibs* octets

Affichage, suite

2) Affichage du contenu binaire

```
od -Ax -tx1z -v mbr.bin
```

- Les codes qu'on voit sont un programme en langage machine : EB 48 = JMP 0x4A, FA = CLI, …
  - Le site http://onlinedisassembler.com/ offre un désassembleur en ligne, on lui donne les codes et il indique quelles sont les instructions correspondantes
  - C'est ultra technique (réservé aux hackers)
- Les deux derniers octets sont 55 AA = MBR correcte
- Les 4*16 derniers octets = table des partitions...

11.2 – Partitions

Les disques sont généralement découpés en plusieurs parties

Découpage d'un disque

- Avant, le MBR spécifiait aussi le découpage du disque en 4 zones appelées partitions primaires
  - Une partition = « de tel secteur à tel secteur »
  - Table des partitions dans le MBR (4 partitions)
- Ces délimitations sont dans le MBR : il y a 16 octets par partition, à partir des adresses 1BE, 1CE, 1DE et 1EE :
  - 1er octet : drapeaux (flags), ex : 80 = démarrable
  - 5e octet : type de partition, ex : 83 = Linux
  - Autres octets : LBA du début et LBA de fin

Partitions logiques

- Comme le MBR limitait à 4 partitions, ce qui est trop peu pour un système d'exploitation, on a rajouté la possibilité d'étendre la table :
  - Partitions dites « logiques » ou « étendues »
  - Placées dans le premier bloc de la dernière partition, c'est comme un second MBR
- Problème : si le disque est déjà occupé par 4 partitions primaires et que la dernière n'a pas été prévue en étendue, alors c'est fichu (ou c'est fait exprès).

Partitions avec la norme UEFI

- Dans la norme UEFI, ce n'est plus le MBR qui contient la table des partitions (lui ne contient plus seulement qu'une seule partition pour représenter tout le disque), mais les 33 premiers secteurs du disque :
  - La table est stockée sur les secteurs LBA=2 à 34 et est aussi recopiée à la fin du disque
  - Le secteur LBA=1 est un bloc d'en-tête appelé GPT Header : GUID Partition Table

    => Il peut y avoir 128 partitions au lieu de seulement 4 (ça semble suffisant)

Pourquoi faire des partitions ?

- Comme pour les dossiers : pour séparer les choses
  - Partitions pour le système (/ et /boot)
  - Partition pour les comptes (/home)
  - Partition pour la mémoire virtuelle (swap)
  - Partition pour les sites web hébergés (/var/www)
  - Partition pour les fichiers temporaires (/tmp)
  - Partitions pour archiver les comptes
  - Partitions pour un autre système d'exploitation...

Pourquoi séparer ?

- En cas de panne ou de virus
  - Seules les partitions du système seront à effacer et réinstaller
  - Panne : défaillance du disque, les fichiers de la partition sont perdus mais pas ceux des autres partitions
- En cas de débordement :
  - Ex : serveur FTP : si des utilisateurs déposent trop de fichiers, ça bloque seulement cette partition

Création de partitions

- Plusieurs logiciels permettent de définir les partitions :
  - `fdisk` : outil de base, en ligne de commande et avec un menu simpliste
  - `cfdisk` : interface améliorée pour fdisk
    - Ex : `cfdisk /dev/sda`
  - `gparted` : logiciel très complet et ergonomique

Gestion des partitions

- Le logiciel gparted permet de redimensionner les partitions à volonté :
  - Ex : diminuer l'une et agrandir sa voisine
- Mais ça implique de décaler toutes les données !
  - C'est très long (ex : 12 h pour 2 To à déplacer)
  - C'est très risqué : coupure de courant => partition détruite car incohérente, et irrécupérable !
- Donc : bien réfléchir avant et ne plus y toucher !

Procédure d'installation

- L'installation de Linux est automatisée à l'exception du partionnement des disques :
  - Préserver Windows ou pas
  - Découper l'espace libre
- Ne pas choisir l'installation automatique mais passer en mode manuel => lancement de gparted

Choix de partitionnement

- Windows = 1 partition au moins,
  - C: contient le système, les logiciels et les comptes, très déconseillé de les y laisser => D:
  - Parfois une partition de récupération (totalement inutile si vous n'avez pas les CD d'installation)
- Linux = 3 partitions au moins
  - / contiendra le système, les logiciels
  - /home contiendra les comptes
  - swap pour la mémoire virtuelle, taille ≳ RAM

Partitions vues de Unix

- Chaque partition d'un disque est associée à un fichier spécial : /dev/disque*N* (N : 1, 2...)
  - `/dev/sda1` pour la première partition du 1er disque
  - `/dev/sdc2` pour la deuxième partition du 3e disque
  - etc.
- Ça pose un problème si on échange les disques (branchement sur la carte mère) : le nom de leurs partitions change (sda1 → sdb1)

Nommage par un GUID ou UUID

- Au lieu de nommer les partitions ainsi, on préfère maintenant les identifier par un numéro unique écrit en hexadécimal :
  - un GUID (global identifier) sur Windows, ex :

    B0AA-B27F
  - Ou UUID dans le monde Unix, ex :

    2F0A84C0-B92F-41D1-671A-6466554876F1
  - Ce n° est inscrit au début de chaque partition
- La commande uuidgen affiche un nouvel UUID

La commande blkid

- La commande `blkid` affiche les UUID et le type de formatage des disques connectés au système :

```
lili@localhost$ sudo blkid
/dev/sda1: LABEL="live-rw" UUID="e1314788-3f9e-41fe-
852f-9e2f42de7c9a" TYPE="ext4"
/dev/sr0: LABEL="LILI LiveCD" TYPE="iso9660"
/dev/loop0: TYPE="squashfs"
lili@localhost$
```

Noms UUID des partitions

- Dans ce système de nommage, les volumes sont dans le dossier /dev/disk/by-uuid/ :

  (ce sont des liens vers les /dev/sdXN)

```
lili@localhost$ ls -l /dev/disk/by-uuid/
total 0
lrwxrwxrwx 1 root root 10 nov.  25 08:52 e1314788-3f9e-
41fe-852f-9e2f42de7c9a -> ../../sda1
lili@localhost$
```

Label d'une partition

- Il y a également la possibilité d'utiliser une étiquette (ou *volume label*) pour nommer les partitions : un mot de 16 lettres au maximum
  - Ex : ROOT, HOME, DATA...
- On le définit quand on formate la partition

11.3 – Format de partition

Comment les secteurs d'une partition sont-ils remplis par des fichiers et dossiers ?

Format d'une partition

- Le contenu d'une partition (dossiers et fichiers) est défini par un format :
  - Windows : FAT32, exFAT, NTFS
  - Unix : ext (ext2, ext3, ext4), reiserfs, xfs, btrfs...
  - Commun (CD et DVD) : iso9660, joliet, udf...
- Le format spécifie comment, en interne, il faut écrire un fichier sur les secteurs de la partition
  - Ex : comment ranger les remorques de nombreux camions sur un grand parking

Principe général

- Les secteurs de la partition (512 octets) sont regroupés en *clusters* ou blocs de 1 à 8 Ko en général (2, 4, 8 ou 16 secteurs)
  - Chacun est numéroté 0..N-1 (ex : N = LBA/8)
- Les fichiers sont stockés sur des blocs de préférence consécutifs (mais pas forcément).
  - Quand les blocs ne sont pas consécutifs, on dit que le volume est fragmenté. Cela provient des agrandissements aléatoires de fichiers
  - L'espace libre également peut être fragmenté

Principe général, suite

- Le système mémorise la liste des numéros de blocs que chaque fichier utilise
  - Il faut une structure de données efficace (rapide, peu de place) pour mémoriser ces listes de numéros : liste, arbre, voir en cours de SDD.
- En interne :
  - Un dossier = liste des fichiers qu'il contient
  - Un fichier = liste des numéros de blocs sur lesquels il se trouve

Ce qui différencie les formats

- Une gestion plus ou moins sophistiquée des blocs : réserver des zones de blocs consécutifs pour éviter la fragmentation et accélérer les accès
- Un « journal » (NTFS et ext≥3...) qui permet de récupérer d'une panne en cours d'écriture sur le disque
  - « Je note ce que je vais faire dans le journal, puis je le fais, enfin je vide le journal » : en cas d'interruption, tout reste cohérent

Exemple : ext*N* sur Linux

- ext3 est la version journalisée de ext2
- ext4 est la version améliorée de ext3 (très grands disques et gros fichiers...)
- Toutes séparent une partition en deux zones :
  - Zone des i-nodes : 1 i-node = 1 fichier ou dossier
  - Zone des blocs : contenus des fichiers et dossiers, ils sont séparés en groupes (contiennent des copies des informations importantes, ils sont comme des sous-partitions)
- Ex : telle partition ext3 de 60 Mo contient 14656 i-nodes et 58432 blocs (de 1 Ko) en 8 groupes

i-nodes

- Un i-node représente un fichier ou un dossier
- C'est un struct {
  - Le type : fichier, dossier, raccourci... (voir plus loin)
  - La taille du fichier ou du dossier
  - Les dates de création, modification, accès
  - Les propriétaires et droits (voir cours suivant)
  - Début du fichier (tout s'il est suffisamment petit)
  - N° des blocs contenant le reste du fichier

  }

i-nodes et dossiers

- Un i-node ne contient pas le nom de son fichier, seulement ses caractéristiques
- Ce sont les dossiers qui contiennent les noms
  - L'i-node n°2 contient le répertoire racine /
  - Un dossier = sorte de fichier dont le contenu est un tableau comme celui-ci :
  - Pour le voir : `ls -i`

nom	n° d'i-node
doc.txt	23
WikiSystème.htm	17
tp10	56

Le superbloc

- Le premier Ko (2 premiers secteurs) de la partition permet le démarrage (sorte de MBR)
- Ensuite on trouve un bloc appelé superbloc. On en trouve plusieurs copies plus loin dans la partition (1 par groupe de blocs)
  - Il contient les informations de base : nombre d'i-nodes, nombre de blocs de données, etc.
- Après, on trouve la liste des blocs libres (zone appelée bitmap, 1 bit = bloc libre ou pas)

Formater une partition

- Formater = tout effacer et préparer pour de nouveaux fichiers => bien vérifier avant !
- La commande est :

  `sudo mkfs.`*`type`* `-L` *`label`* `/dev/`*`partition`*

  Ex : `sudo mkfs.ext4 -L HOME /dev/sdb1`

  Ex : `sudo mkfs.xfs -L DATA /dev/sdc3`

  – Option -L : définit le nom (label) de la partition.

  NB : l'UUID ne peut pas être choisi, il est généré automatiquement.

Vérifier une partition

- Il arrive de temps en temps qu'un disque dur commette une erreur de lecture ou d'écriture
  - Des fichiers peuvent être corrompus : contenu altéré (du texte aléatoire à la place)
  - Des dossiers peuvent être incomplets : certains fichiers sont tronqués ou perdus
- Il faut employer un outil de vérification
  - Automatique : tous les 23 démarrages de l'ordinateur ou tous les 30 jours.

Commande fsck

- La commande fsck vérifie la cohérence d'une partition

```
sudo fsck -f /dev/hdb1
```

  - Ne pas employer sur une partition montée (cf + loin)
- Les fichiers égarés en cas de corruption de leur dossier sont mis dans lost+found : ils viennent d'i-nodes qui n'étaient plus dans aucun dossier (leur ancien dossier ayant été corrompu)

11.4 – Montage d'une partition

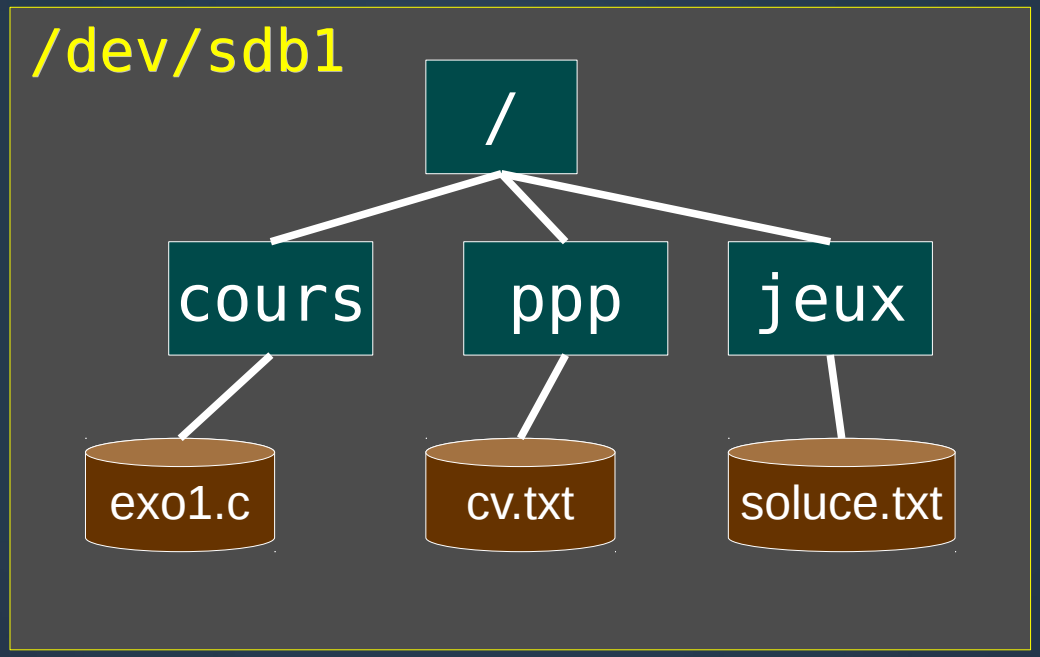
Comment fait-on pour accéder aux fichiers d'une partition ?

Montage d'une partition

- Voici les concepts :
  - Pour accéder aux fichiers d'une partition, on doit la « monter » (mount)
  - Cela fait apparaître son contenu dans l'arbre des fichiers Unix

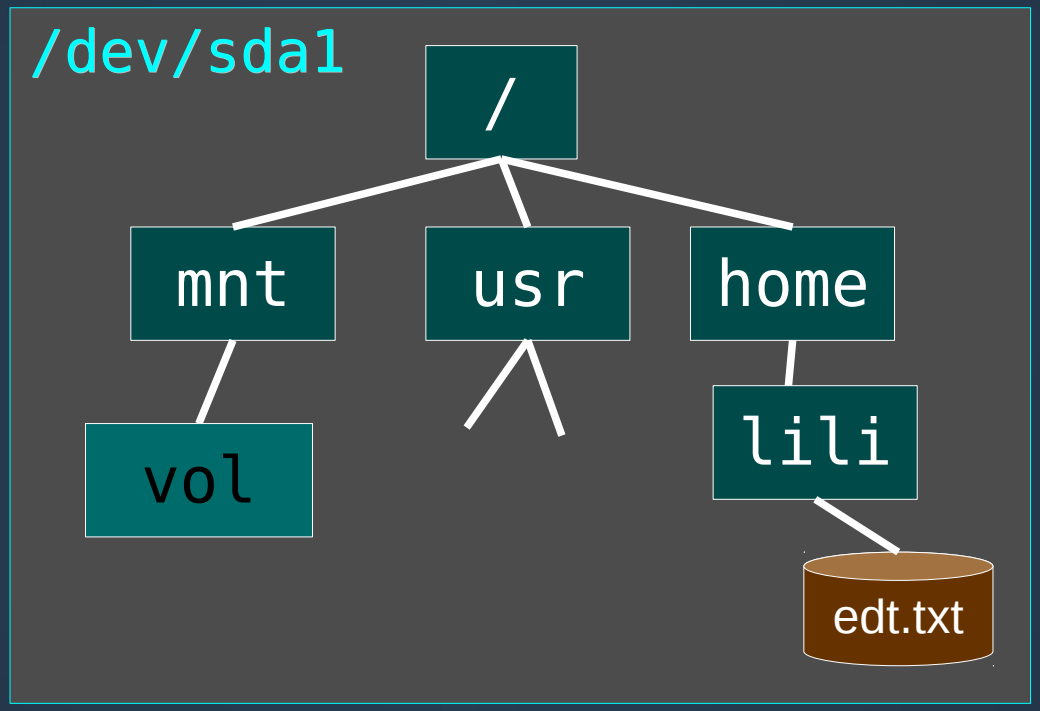
Contenu d'une partition

- Toute partition contient un petit arbre de fichier : racine, dossiers... comme l'arbre Unix



Point de montage

- On doit créer un dossier vide, généralement dans /mnt ou /media, ex : `/mnt/vol`

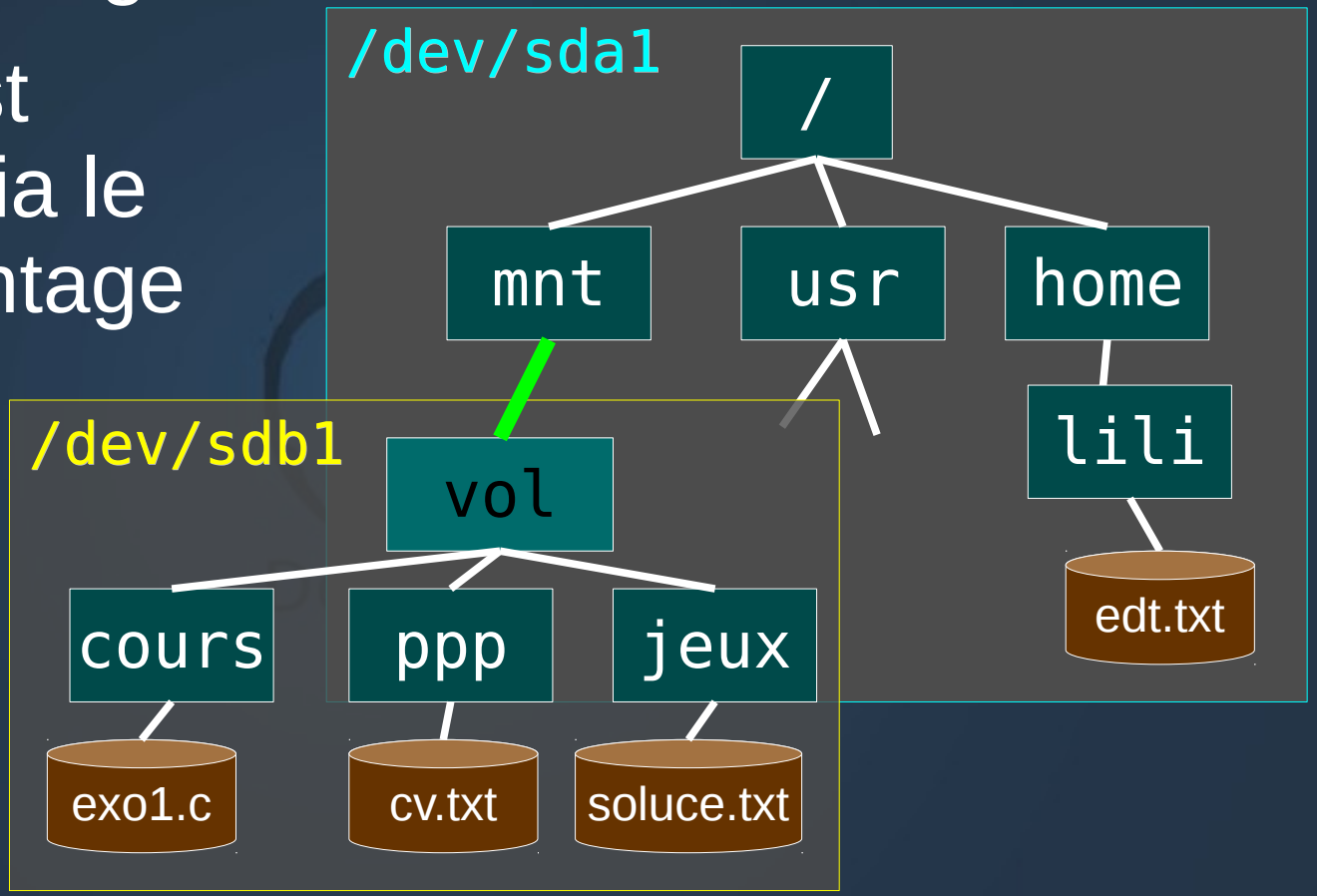


- On va monter `/dev/sdb1` sur `/mnt/vol`

Montage

- Après le montage :

  le volume est accessible via le point de montage



Accès aux fichiers

- L'accès au volume monté est transparent : tout se passe comme si tous les fichiers faisaient partie d'un seul et même arbre
- Le système gère les accès aux partitions :
  - `/home/moi/edt.txt` accès à `/dev/sda1`
  - `/mnt/vol/cours/exo1.c` accès à `/dev/sdb1`
- Attention : on n'a plus accès à l'ancien contenu du point de montage !

Effectuer un montage

- Pour monter une partition, il faut :
  - Créer ou avoir un dossier vide, ex : /mnt/*dossier* qui définit le point de montage
  - Connaître le fichier spécial /dev/*partition* correspondant à la partition
  - Employer la commande en tant qu'administrateur :

```
mount -t format /dev/partition /mnt/dossier
```

- Ex :

```
sudo mount -t xfs /dev/sdb2 /mnt/vol
```

Liste des montages

- La commande mount sans paramètre montre les montages actuellement en place
  - Affiche des lignes au format :

    partition on *point de montage* type *format* (*options*)

```
lili@localhost$ mount
proc on /proc type proc (rw)
/dev/sdb2 on /home type ext4 (rw)
/dev/sda1 on / type ext4 (rw,errors=remount-ro)
lili@localhost$
```

- À noter que /proc, /sys et autres sont sur des partitions spéciales (virtuelles)

Liste par fichiers spéciaux

- Le fichier `/etc/mtab` affiche aussi la liste des montages actifs. On peut voir les options de montage (rw, users...)
  - La commande mount met à jour ce fichier (sauf si option -n)
- Le fichier /proc/mounts montre une liste similaire
  - Ce fichier est un reflet des tables internes du système

Démonter une partition

- Très simple :
  - Commande `umount` point de montage ou partition
  - NB : c'est `umount` et non pas `unmount`
- Ex :

```
sudo umount /mnt/cd
sudo umount /dev/sdb2
```

Infos sur les partitions

- La commande `df` affiche la liste des partitions et leur taux de remplissage
  - L'option -h les affiche en Mo ou Go

```
lili@localhost$ df
Sys. de fichiers blocks de 1K  Utilisé Disponible Uti% Monté sur
/dev/sda1             19553560 11138404    7398836  61% /
udev                     10240        0      10240   0% /dev
tmpfs                   398500     1080     397420   1% /run
tmpfs                     5120        0       5120   0% /run/lock
tmpfs                  1597540      420    1597120   1% /run/shm
/dev/sdb1             53359108 46587712    4037828  93% /home
lili@localhost$
```

11.5 – Montage permanent

Comment mettre en place un montage définitivement ?

/etc/fstab

- Le fichier `/etc/fstab` contient la liste des montages qui doivent être faits au démarrage du système (option auto) ou qui peuvent être faits ultérieurement (options noauto,user)
- La commande `mount -a` est lancée au démarrage et monte tous les volumes notés auto

Syntaxe de /etc/fstab

- Les lignes de /etc/fstab contiennent 6 « mots » :
  1. Le périphérique associé à la partition, ex : /dev/sdb1
  2. Le point de montage, ex : /
  3. Le format de la partition, ex : ext4
  4. Les options, ex : defaults, noauto, user
  5. Un 0, c'est une option pour l'archivage avec dump
  6. Un 1 pour /, 2 pour les autres : ordre de vérification par fsck, 0 empêche la vérification

Application

- En TP, on va créer une partition sur un autre disque virtuel et monter cette partition automatiquement sur /home

  => les comptes seront sur cette partition

  – Intérêt : pas de perte des données s'il faut réinstaller le système :
    - On peut reformater /dev/sda1 et réinstaller le système
    - On remonte /dev/sdb1 sur /home et on récupère tout

11.6 – Changement de racine

On peut faire en sorte d'ouvrir un shell dans un sous-dossier limité

Système Unix

Un système Unix est :

- Une arborescence / complète
  - /boot, /bin, /lib, /etc, /usr, /home, /tmp, /var...
- Située sur une ou plusieurs partitions montées au démarrage
  - Au moins une partition dite racine pour /

Commande chroot

- On peut définir une autre racine pour un shell ou un processus :
  - Ce shell ou processus croira avoir affaire à un système complet, mais en réalité, ça ne sera qu'une partie de l'arborescence réelle
  - Il faut que cette autre racine soit un dossier contenant un système Unix complet
- NB : Il y a quelques contraintes (/dev, /proc, /sys, /run...) sur les systèmes récents

Exemple

1) On démarre sur le live CD-ROM (comme la première fois en TP)

   => la racine du système = le cd-rom

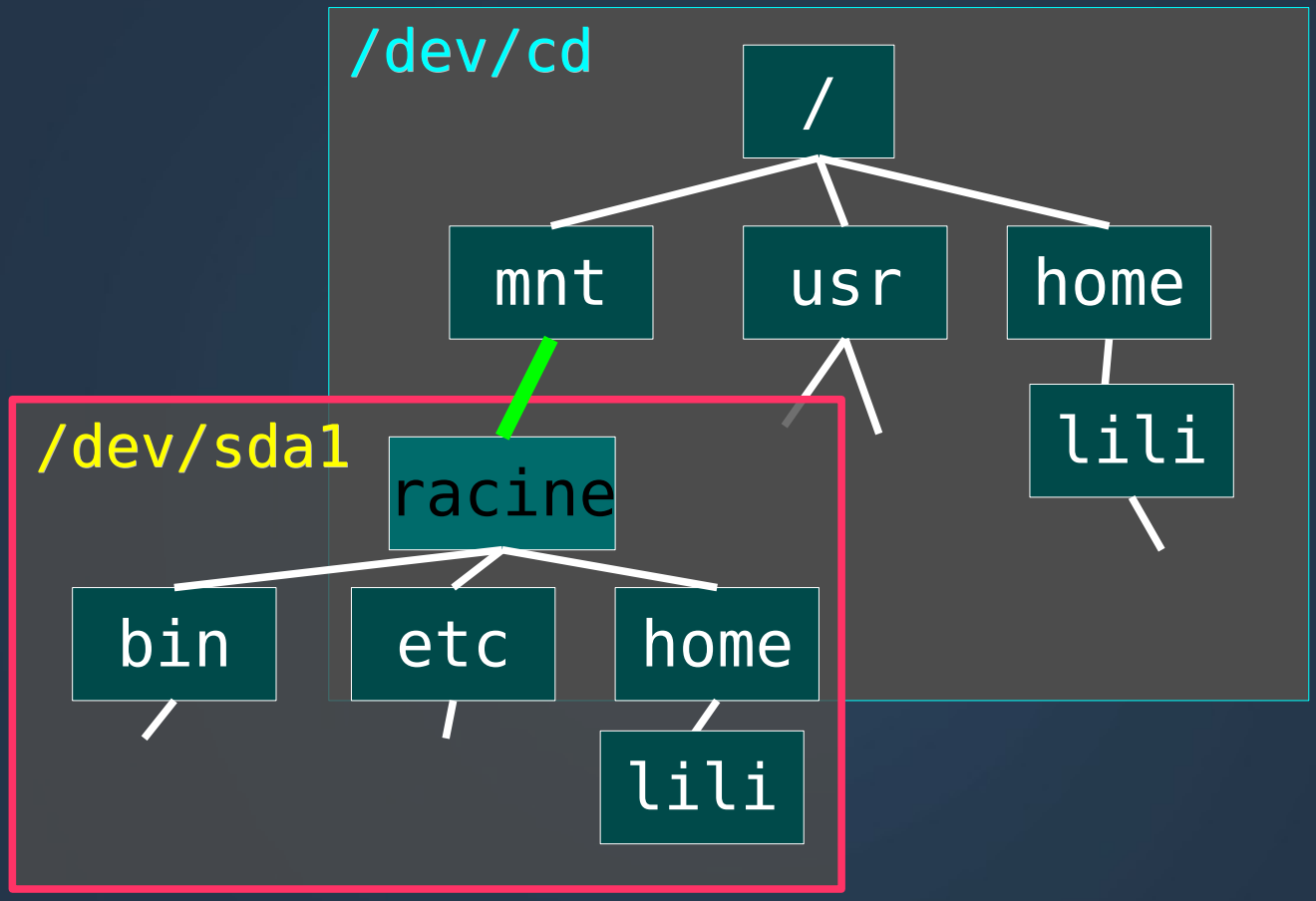
2) On monte /dev/sda1 sur /mnt/racine

3) On change de racine :

```
sudo chroot /mnt/racine
```

=> on se retrouve comme si on avait démarré directement sur /dev/sda1

Illustration du chroot



11.7 – Montage d'un fichier

Certains fichiers peuvent être montés comme des partitions

Monter un fichier

- Certains fichiers ont une structure de volume
  - Les .iso sont des « images » de CD-ROM : ils sont structurés en secteurs, comme les disques
  - On s'intéresse uniquement aux CD de type DATA (pas type musical). On les crée avec un logiciel de gravure, et on enregistre un .iso au lieu de graver
- On utilise l'option `-o loop` de mount :

  `mount -o loop` *`fichier`*`.iso` *`pt_de_montage`*
  - Ex : `sudo mount -o loop data.iso /mnt/iso`

Comment ça marche ?

- L'option -o loop fait appel à un périphérique spécial : `/dev/loop`*N* (*N*=0..7) géré par la commande `losetup` qui simule une partition virtuelle à partir d'un simple fichier :
  - Attachement puis montage :

```
sudo losetup /dev/loop0 fichier.iso
sudo mount /dev/loop0 /mnt/iso
```

  - Démontage puis détachement :

```
sudo umount /dev/loop0
sudo losetup -d /dev/loop0
```

Applications de losetup

- C'est ce mécanisme qui permet de crypter un volume. La commande losetup peut intercaler un « crypteur » entre le fichier et le périphérique :
  - Tout secteur lu est décrypté
  - Tout secteur écrit est crypté
- Mise en place (sudo partout) :

```
modprobe aes
losetup -e aes /dev/loop0 crypto.img
mount /dev/loop0 /mnt/crypto
```

Remarque sur le cryptage

- On fera un peu différemment en TP, on utilisera cryptsetup au lieu de losetup.
- Car, actuellement (Linux noyau 3.x), on doit employer le mécanisme LUKS (Linux Unified Key Setup), mais c'est globalement trop complexe pour être expliqué ici.

11.8 – Fichiers, dossiers, liens...

Quels sont les différents types de fichiers qu'on peut rencontrer dans un arbre ?

Nature des éléments

- La nature des éléments d'un dossier (fichiers, sous-répertoires, etc) est indiquée par la première lettre de la commande `ls -l` :
  - `-` indique un fichier ordinaire
  - `d` indique un dossier
  - etc... (voir la suite du cours)
- Ensuite, on trouve les droits, mais c'est l'objet du prochain cours

Fichiers ordinaires

- Les fichiers ordinaires sont des sortes de chaînes de caractères. On peut y mettre n'importe quel type de contenu pourvu qu'il existe un codeur (format d'enregistrement) et un décodeur (lecteur de format)
  - Par exemple une image jpg : codage interne

Dossier

- Un dossier est simplement une association entre des noms et des numéros d'i-node
- Parmi les noms :
  - `.` représente le dossier lui-même
  - `..` représente le dossier parent
  - Ces éléments sont des liens physiques

Lien physique

- C'est le partage d'un même i-node entre deux fichiers (ou dossiers) dans une partition
  - Deux noms sont associés au même i-node
  - Pas forcément dans le même dossier
- Exemple :
  - `vi double.txt`
  - `more doc.txt`

nom	n° d'i-node
doc.txt	23
WikiSystème.htm	17
tp10	56
double.txt	23

Création d'un lien physique

- La commande `ln source lien` crée une sorte de double du fichier source et l'appelle lien
  - `Ex: ln WikiSysteme.htm double.htm`
- Attention, ça crée en fait *le même contenu* avec *deux noms différents*, ce n'est pas une copie :
  - Modifier l'un c'est modifier l'autre
  - Supprimer l'un ne supprime pas l'autre

Compteur de liens

- Si on supprime l'un des liens sur un fichier, ça ne supprime pas l'autre :
  - Les i-nodes comptent le nombre de liens qui sont sur eux
  - La commande `rm` décrémente le nombre de liens et enlève le couple (nom, n°i-node)
  - Le contenu est détruit si le nombre de liens passe à 0
- Le nombre de liens est affiché par `ls -l`

```
-rw-r--r-- 2 pierre admin  223 août 15 infos
```

Commande `ls -ali`

- La commande `ls -ali` affiche :
  - `-a` tous les fichiers, y compris les cachés
  - `-l` tous les détails
  - `-i` les numéros d'i-nodes
- On voit :
  - N° d'i-node
  - Type -, d...
  - Protections
  - Nombre de liens

Dossiers . et ..

- On pourra vérifier avec ls -ali que `.` et `..` sont des liens physiques vers le dossier lui-même et son parent
  - Cependant il n'est pas possible de créer un lien sur un dossier, l'option -d est bloquée, même en sudo
- Une limite aux liens physiques : un n° d'i-node étant relatif à sa partition, on ne peut pas créer de liens physiques vers des fichiers d'une autre partition

Liens logiques ou symboliques

- Les liens logiques sont des fichiers spéciaux
  - Leur contenu = un nom complet vers un autre fichier ou dossier : la *cible* du lien
  - Supprimer la cible rend le lien invalide
- Création :

  ```
  ln -s cible nom
  ```

  - Crée un raccourci appelé *nom* vers la *cible*
  - Cible : nom relatif ou nom absolu
- `ls -l` affiche la lettre `l` en début de ligne

Autres types d'éléments

- Fichiers spéciaux présents p.ex. dans `/dev`
  - Périphériques caractères (c) : ex : terminaux tels que /dev/pts/N
  - Périphériques blocs (b) : ex : disques et partitions /dev/sda, /dev/sda1
  - Sockets réseaux (s) : ex /dev/log est un dispositif permettant de transmettre des infos entre machines
  - Tubes nommés (p) : ex /dev/initctl est un pipe permettant de communiquer avec le processus init