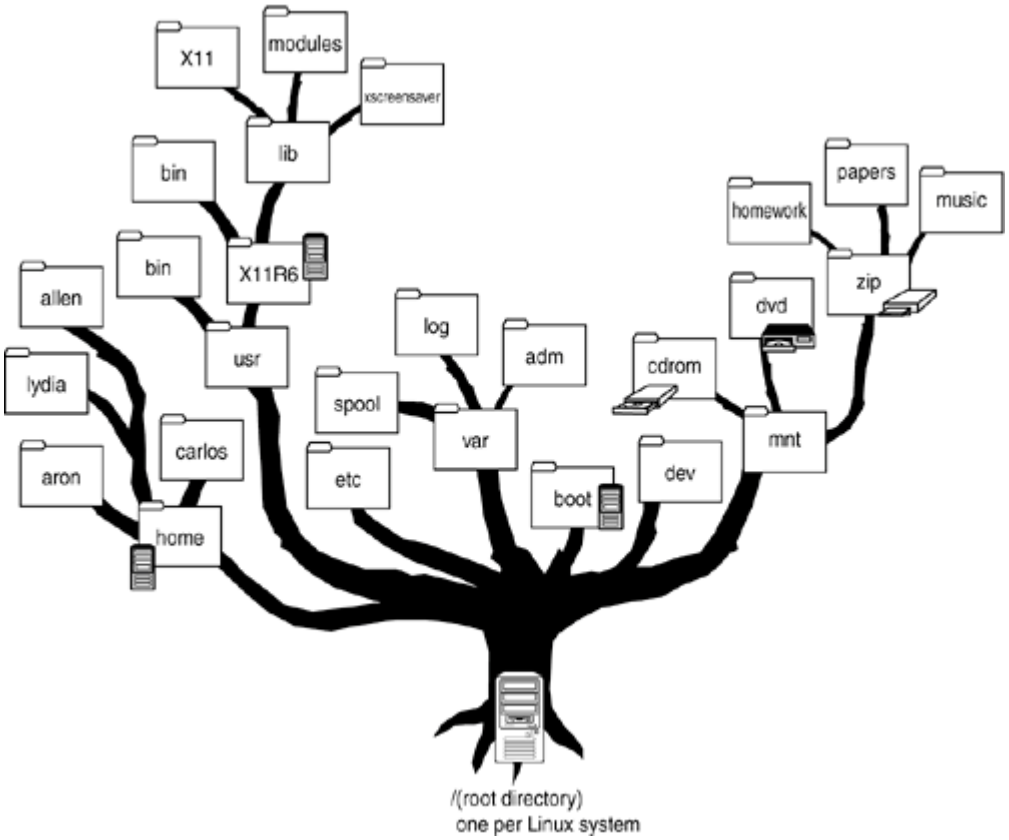
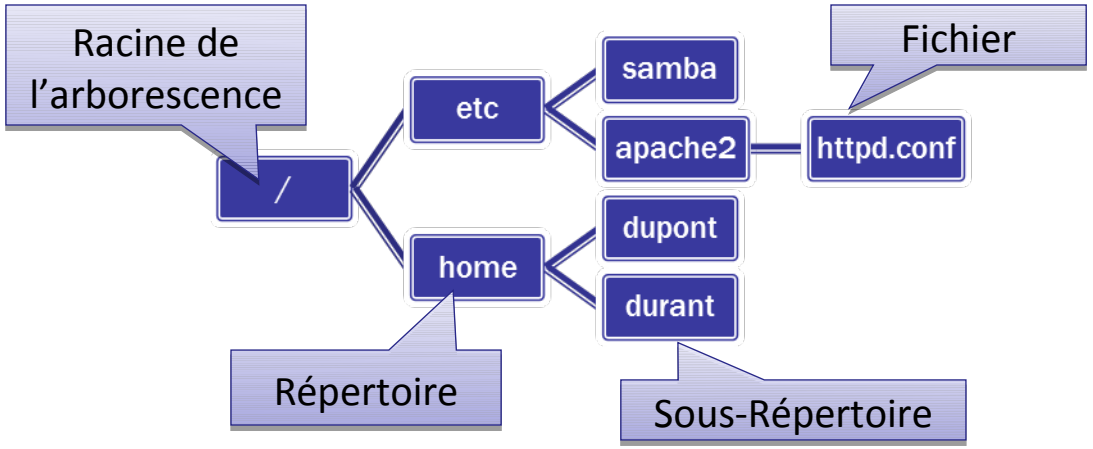
Le système de fichiers LINUX



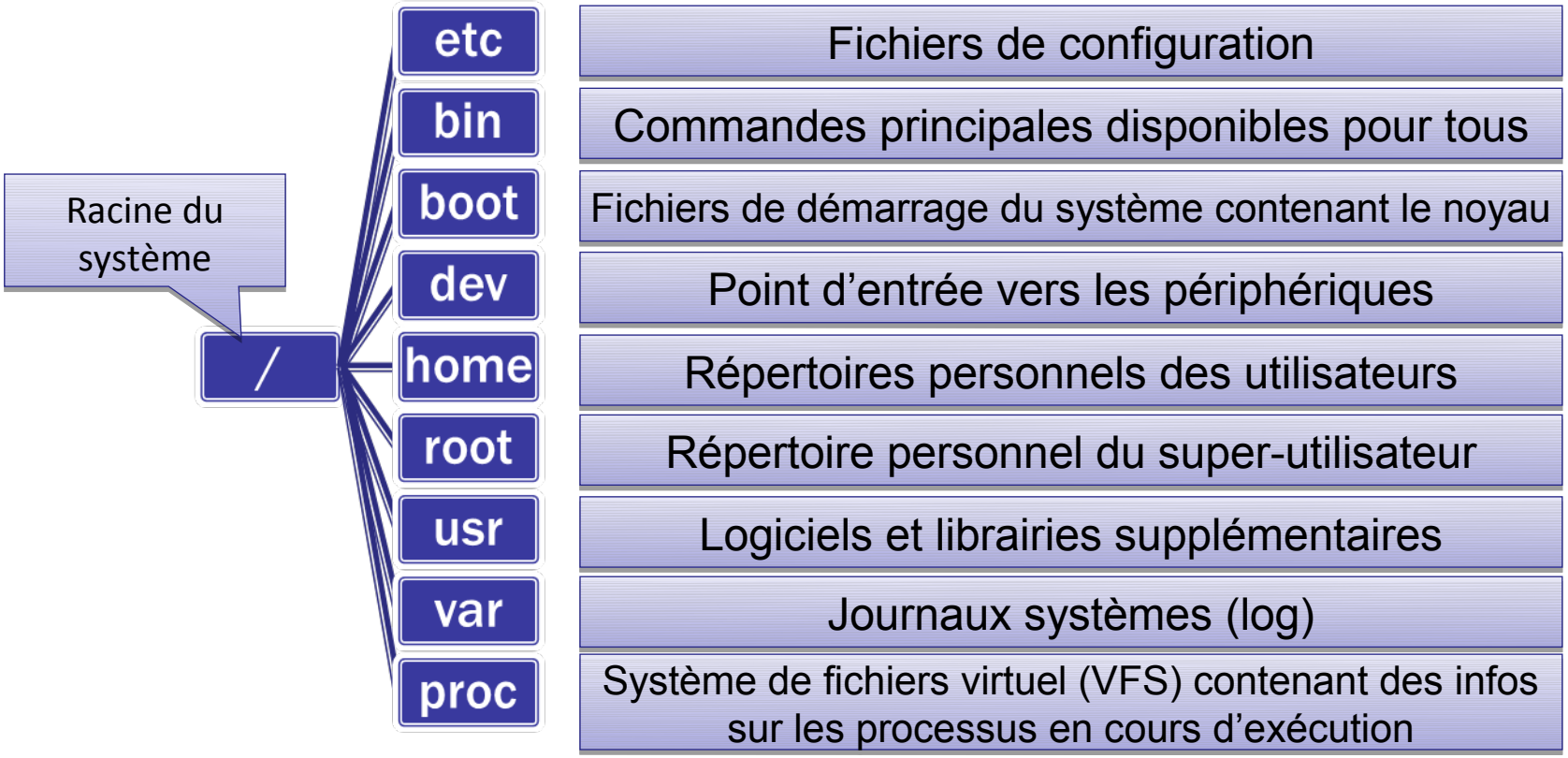
Quelques définitions

- Qu'est-ce qu'un système de fichier ?
  - Organisation physique des données sur un support
    - Sur un disque dur, une clé USB, un DVD, …
- Qu'est-ce qu'une arborescence ?
  - Organisation logique des fichiers sur un ou plusieurs systèmes de fichiers
  - Il s'agit d'une structure de données hiérarchique de type arbre



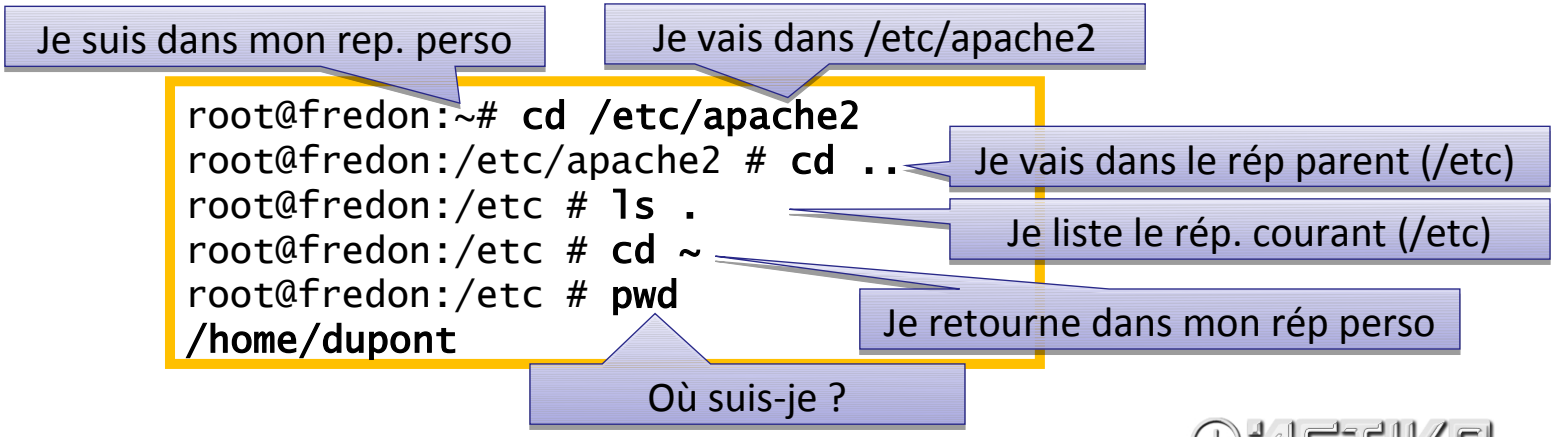
Arborescence Linux

- Voici l'arborescence typique d'un système Linux :



Les symboles associés à l'arborescence

- Différents symboles sont utilisés pour désigner des répertoires
  - Le « . » : Répertoire courant
  - Le « .. » : Répertoire parent
  - Le « ~ » : Répertoire personnel de l'utilisateur courant
- La commande « cd » permet de changer de répertoire
- La commande « ls » permet de lister un répertoire
- La commande « pwd » permet de connaître le rép. courant
- Exemples :

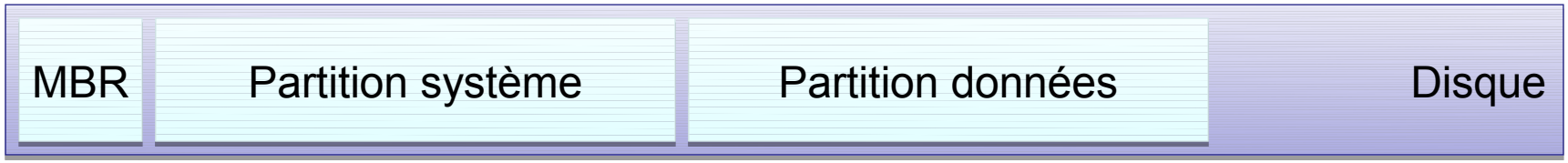


Quelques infos sur les disques durs

- Il est constitué d'un ou de plusieurs plateaux
  - Chaque plateau est divisé en pistes (tracks)
  - Chaque piste est divisée en secteurs (sectors)
  - Le cylindre est formé par les pistes de même rayon sur chaque plateau
- Le formatage est effectué à 2 niveaux
  - En usine : le formatage **bas niveau** des pistes et secteurs
  - Par l'utilisateur : Effacement ou réécriture des données
    - Lors de l'installation du système d'exploitation par exemple
- La capacité d'un disque dépend
  - De la taille des secteurs
  - Du nombre de cylindres et donc du nombre de plateaux

L'organisation du disque

- Organisation typique du poste de travail



- Le **M**aster **B**oot **R**ecord est situé dans les 1er secteurs du disque
- Il est constitué de 2 parties :
  - La table des partitions
  - Le programme d'amorçage qui charge le noyau du système
- Plusieurs types de partitions
  - Principale
  - Etendue
  - Logique

Les partitions

- Les partitions principales
  - Au maximum de 4
  - Accepte tout type de système de fichiers
- Les partitions étendues
  - Destinées à contenir des partitions logiques et non un système de fichiers
  - Nécessitent au moins une partition principale
- Les partitions logiques
  - Contenues dans une partition étendue
  - Accepte tout type de système de fichiers
- Exemple permettant d'installer plusieurs systèmes d'exploitation

Prise en charge des disques sous Linux

- Le pointeur spécial /dev permet l'accès aux disques
  - Format des pointeurs sur disque :

- Types de bus
  - hd : Périphériques IDE
  - sc : Périphériques SCSI
  - sd : Périphériques SATA
- Exemples
  - /dev/hda1 :
    - Partition 1 sur le 1er disque IDE
  - /dev/sdb2 :
    - Partition 2 sur le 2ème disque Sata

Les formats des systèmes de fichiers (1)

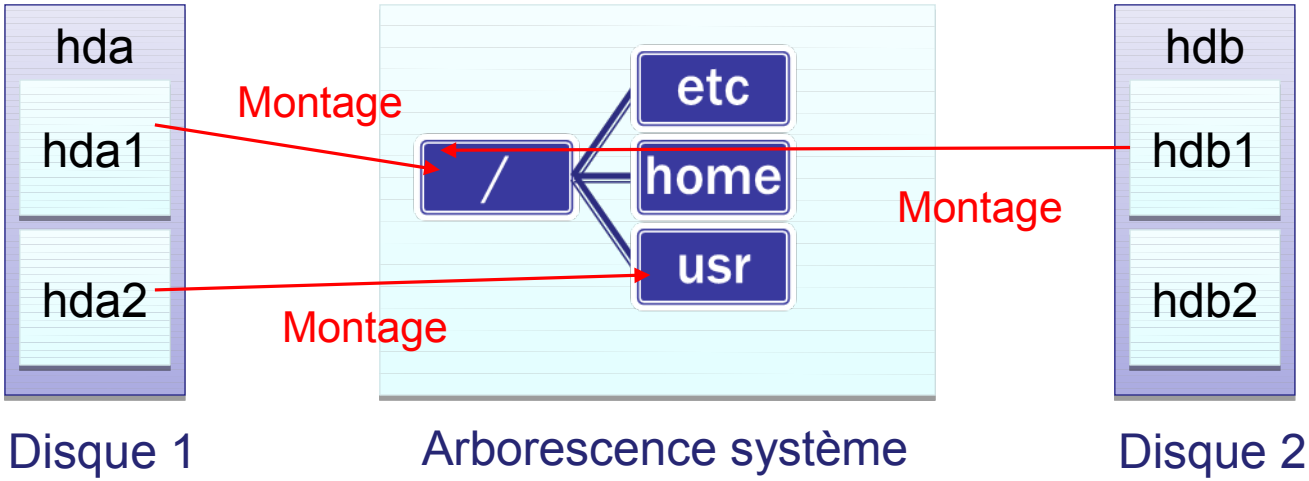
- À chaque système est associé un format
  - Définit la structure des données sur le support
- Sous Linux
  - ext2, ext3, jfs, xfs
  - ext3 est la plus courante pour Linux
- Sous Windows
  - fat, fat32, ntfs
  - Nfts est utilisé sous windows XP et Vista
- Toujours préférer un système de fichier « journalisé »
  - Chaque séquence de lecture/écriture est d'abord inscrite dans un journal avant d'être effectuée
    - Si le système se bloque pendant la séquence, elle sera achevée après le redémarrage
  - On évite les erreurs dans le système de fichiers

Les formats des systèmes de fichiers (2)

- Le format « swap » est utilisé comme « mémoire virtuelle »
  - Dans le cas où la mémoire vive est saturée
  - Par le système pour améliorer les performances
  - La taille du « swap » est fixée au double de la mémoire vive
    - Si 512Mo de mémoire vive -> 1024Mo de swap
- Linux peut lire la plupart des formats
  - Notamment Ceux de Windows : NTFS, FAT, FAT32
- Avant d'être utilisé, un disque doit être partitionné
  - A l'aide de la commande « fdisk » si Linux est déjà installé
  - Par le programme d'installation sinon (dépend de la distribution)
- Il faut ensuite créer un système de fichier
  - Avec l'utilitaire générique « mkfs »
    - mkfs.ext2, mkfs.ext3, mkfs.xfs, …

Points de montage (1)

- Sous Linux, « Tout est fichier »
  - L'arborescence est construite à partir de « points de montage »
- Un point de montage est une association entre une partition physique et l'arborescence du système



- Avantages
  - Mettre à l'abri certaines données stratégiques comme /home
  - La défaillance du disque hdb n'entraîne pas une réinstallation totale

Points de montage (2)

- Tant qu'ils ne sont pas effectués, le système de fichiers est inaccessible
- Ils sont réalisés automatiquement au démarrage du système
  - Seulement ceux qui figurent dans le fichier « /etc/fstab »
- Il est possible de créer un point de montage manuellement
  - Pour les clés USB ou le CDROM par exemple
  - En utilisant la commande « mount »
  - Et « umount » pour supprimer le point de montage

```
root@fredon:~# mount /dev/hdd /mnt/cdrom
root@fredon:~# umount /dev/hdd
```

- Une partition est associée à un système de fichiers
  - Il faut parfois préciser le type de ce système
    - ext2, ext3, xfs, swap, jfs, iso9660, vfat, …

Mise en place d'un nouveau disque (1)

- 3 étapes sont nécessaires
  - Création d'une partition : **fdisk**
  - Initialisation du système de fichiers : **mkfs**
  - Création d'un point de montage : **mount**
- 1) Création d'une partition
  - Partition principale de 100Mo avec « fdisk » sur /dev/sda

```
root@fredon:~# fdisk /dev/sda
Commande (m pour l'aide): n
Action de commande
   e   étendue
   p   partition primaire (1-4)
p
Numéro de partition (1-4): 1
Premier cylindre (1-26, par défaut 1):
Utilisation de la valeur par défaut 1
Dernier cylindre ou +taille or +tailleM ou +tailleK (1-26, par défaut 26): +100M
Commande (m pour l'aide): w
La table de partitions a été altérée!
Appel de ioctl() pour relire la table de partitions.
Synchronisation des disques.
Commande (m pour l'aide): q
```

Mise en place d'un nouveau disque (2)

- Vérification de la partition sous « fdisk »

```
Commande (m pour l'aide): p
Périphérique Amorce    Début         Fin      Blocs     Id  Système
/dev/sda1                 1          13      104391   83  Linux
```

- 2) Initialisation du système de fichiers
  - Avec la commande « mkfs »

```
root@fredon:~# mkfs.ext3 /dev/sda1
mke2fs 1.40.8 (13-Mar-2008)
Étiquette de système de fichiers=
Type de système d'exploitation : Linux
Taille de bloc=1024 (log=0)
Taille de fragment=1024 (log=0)
26104 i-noeuds, 104388 blocs
5219 blocs (5.00%) réservés pour le super utilisateur
Premier bloc de données=1
Nombre maximum de blocs du système de fichiers=67371008
13 groupes de blocs
8192 blocs par groupe, 8192 fragments par groupe
2008 i-noeuds par groupe
Superblocs de secours stockés sur les blocs :
        8193, 24577, 40961, 57345, 73729
Écriture des tables d'i-noeuds : complété
Création du journal (4096 blocs) : complété
Écriture des superblocs et de l'information de comptabilité du système de
fichiers : complété
```

Mise en place d'un nouveau disque (3)

- Création du point de montage
  - De type ext3
  - Avec la commande « mount »

```
root@fredon:~# mount -t ext3 /dev/sda1 /mnt/toto/
```

  - Le répertoire « /mnt/toto » doit déjà exister

- Vérification du point de montage avec la commande « df »

```
root@fredon:~# df
Sys. de fich.          1K-blocs       Occupé Disponible Capacité Monté sur
/dev/sdc1             7913216   2716248   4798156  37% /
varrun                 127856       108    127748   1% /var/run
varlock                127856         0    127856   0% /var/lock
udev                   127856        64    127792   1% /dev
devshm                 127856        12    127844   1% /dev/shm
lrm                    127856     38176     89680  30% /lib/modules/2.6.24-16-generic/vol..
/dev/sdd1             8045180   6987320   1057860  87% /media/GEN-USB
/dev/sda1              101086      5664     90203   6% /mnt/toto
```

Vérification d'un système de fichiers

- Si le doute s'installe quant à l'intégrité du système de fichiers
  - Une vérification s'impose
  - Il faut parfois « démonter » le système en préalable à la vérification
- Utilisation de la commande générique « fsck »
  - Fsck.ext3, fsck.vfat, fsck.ext2, fsck.nfs, …
- Exemple :
  - Supprimer d'abord le montage avec « umount »

```
root@fredon:~# fsck /dev/sda1
fsck 1.40.8 (13-Mar-2008)
e2fsck 1.40.8 (13-Mar-2008)
/dev/sda1: clean, 2136/26104 files, 17541/104388 blocks
```

  - Effectuer le montage après vérification pour accéder au système de fichiers

Le répertoire spécial « /proc » (1)

- Répertoire spécial n'existant pas physiquement sur le disque
- « /proc » est un pseudo-système de fichiers mis à jour en temps réel par le noyau
  - Chaque processus en cours d'exécution y dispose d'un sous répertoire
    - Le nom de ce sous-répertoire correspond au PID du processus
    - Des informations importantes sur le processus y sont stockées
      - Fichiers et mémoires utilisées par le processus
- Les fichiers de /proc sont en :
  - Lecture seule : Permet d'obtenir des infos sur les processus
    - Exemple : **/proc/cpuinfo** (Infos sur le processeur)
  - Ecriture : Permet de modifier des paramètres du noyau
    - Exemple : **/proc/sys/net/ipv4/ip_forward** (Activation du routage)

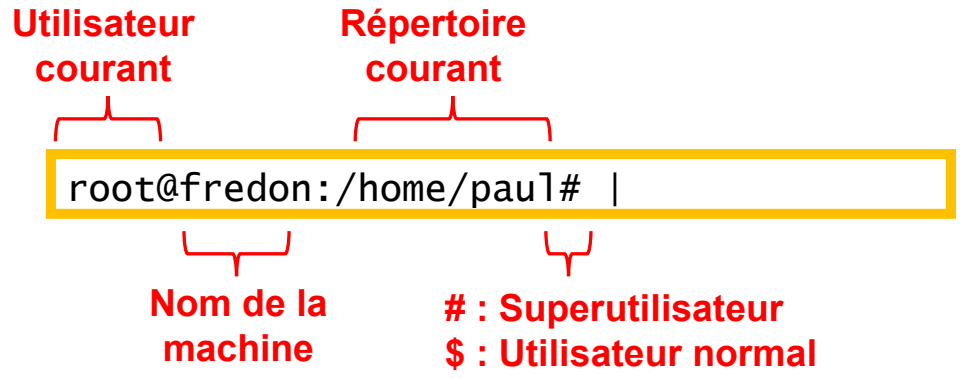
Le répertoire spécial « /proc » (2)

- Quelques fichiers intéressants à consulter
  - **cpuinfo** : informations sur le(s) processeur(s)
  - **meminfo** : utilisation de la mémoire
  - **ioports** : Adresses physiques des différents périphériques matériels
- Visualiser le fichier « /proc/cpuinfo » avec la commande « cat »

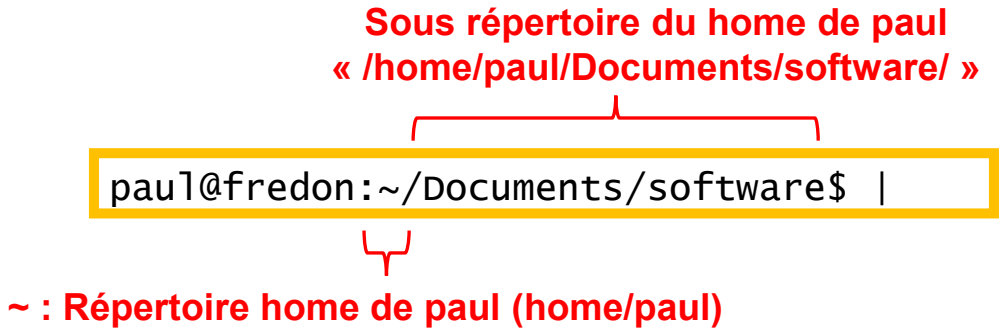
```
root@fredon:~# cat /proc/cpuinfo
processor : 0
vendor_id : GenuineIntel
cpu family      : 6
model           : 23
model name      : Intel(R) Core(TM)2 Duo CPU     T9300  @ 2.50GHz
stepping  : 8
cpu MHz         : 2500.585
cache size      : 6144 KB
fdiv_bug  : no
hlt_bug         : no
f00f_bug  : no
coma_bug  : no
…
```

Comment se repérer dans le système de fichiers ?

- La ligne de commande donne des informations :



- Autre exemple



Où suis-je, où vais-je ?

- La commande « **pwd** » permet de savoir quel est le répertoire courant
- La commande « **ls** » permet de lister les fichiers contenus dans un répertoire
- La commande « **cd** » permet de changer de répertoire
- Les symboles suivants ont une signification particulière :
  - « . » : Le point désigne le répertoire courant
    - Exemple : Exécuter un script depuis le répertoire courant

```
root@fredon:/home/paul# ./script.sh
```

  - « .. » : Les 2 points désignent le répertoire parent
    - Exemple : Se déplacer dans le répertoire parent

```
root@fredon:/home/paul/Docs# cd ..
```

  - « ~ » : Désigne le répertoire home de l'utilisateur courant

```
root@fredon:/home/paul/Docs# cd ~
```

Chemin relatif et absolu

- Il existe 2 méthodes pour spécifier un chemin dans le système de fichiers
  - Chemin relatif : Dépend du répertoire courant
  - Chemin absolu : Débute à la racine du système (« / »)
- Exemples

relatif		absolu
`root@fredon:/home/paul# cd Documents`	=	`root@fredon:/home/paul# cd /home/paul/Documents`
`root@fredon:/home/jean# cd Documents` `root@fredon:/home/jean/Documents/# |`	≠	`root@fredon:/home/jean# cd /home/paul/Documents` `root@fredon:/home/paul/Documents/# |`
`root@fredon:/etc/apache# cd ../` `root@fredon:/etc/# |`	=	`root@fredon:/etc/apache# cd /etc` `root@fredon:/etc/# |`
`paul@fredon:~ # cd Docs` `paul@fredon:~/Docs# |`	≠	`jean@fredon:/etc# cd /home/paul/Docs` `root@fredon :/home/paul/Docs # |`

- Attention aux chemins relatifs à l'intérieur d'un script
  - Le script peut-être exécuté depuis n'importe où
  - Le répertoire courant est donc différent à chaque fois

Exercices

- Dans quel répertoire je suis situé ?

```
paul@fredon:~/Documents/rep1$
```

  - Réponse : Dans le répertoire « /home/paul/Documents/rep1 »

- Par quel chemin relatif équivalent peut-on remplacer celui-ci ?

```
paul@fredon:/etc$ cd /etc/ppp/peers/
```

  - Réponse : « ppp/peers »

- Que m'indiquera le résultat de la commande suivante ?

```
mathilde@fredon:~/Documents/pub$ pwd
```

  - Réponse : « /home/mathilde/Documents/pub »

- Commande la plus courte possible pour revenir dans le répertoire rep2, situé à la racine de mon répertoire « home » ?

```
paul@fredon:~/rep1/sousrep1$
```

  - Réponse : `paul@fredon:~/rep1/sousrep1$ cd ~/rep2`

Les droits des fichiers et répertoires

- Linux est un système multi-utilisateurs
  - Plusieurs utilisateurs se partagent l'espace disque
  - Les fichiers et répertoires d'un utilisateur ne doivent pas être accessibles par les autres
  - Les fichiers de configuration du système doivent être protégés
- Nécessité de spécifier des droits pour chaque fichier/répertoire
  - Plusieurs types de droits : Lecture (R), écriture (W), exécution (X)
  - Ces droits s'appliquent pour 3 groupes d'utilisateurs :
    - Le propriétaire (user) du fichier
    - Le groupe (group) propriétaire (Tous les utilisateurs membre du groupe)
    - Les autres (others). Désigne tous les utilisateurs non membres des 2 précédents
- Les droits sont responsables d'un grand nombre d'erreurs de configuration

Droits : Différence entre fichiers et répertoires

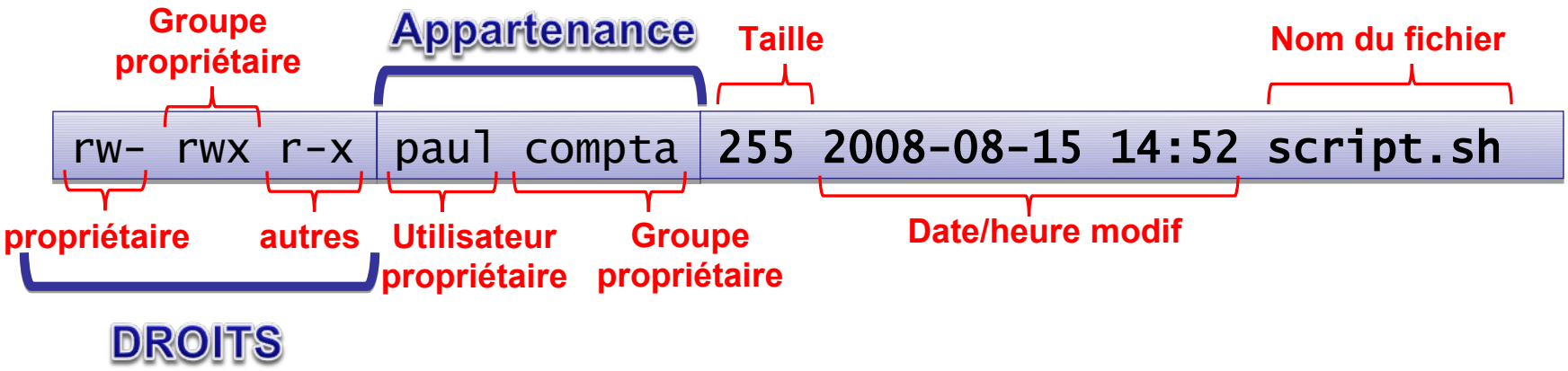
- Nous avons vu qu'il existe 3 types de droits : r, w et x
- Ces droits n'ont pas la même signification pour un fichier que pour un répertoire
- Pour un fichier :
  - **r** : Lecture (afficher)
  - **w** : Ecriture (modification)
  - **x** : Exécution (exécution d'un script)
- Pour un répertoire
  - **r** : Lire le contenu, lister les fichiers (avec ls par exemple)
  - **w** : Modifier le contenu, créer et supprimer des fichiers (avec les commandes « cp », « mv », « rm »)
  - **x** : d'accéder aux fichiers du répertoire. Mais aussi de naviguer dans les sous-répertoires (avec « cd »)
  - En général, lorsque le droit w est accordé, le droit x l'est aussi

Les droits sur les fichiers et répertoires

- La commande « ls -l» permet d'afficher les droits qui s'appliquent

```
root@fredon:/home/paul/Documents# ls -l
total 20
-rw-r--r-- 1 paul paul      0 2008-08-15 14:42 projet.txt
-rw-rw-r-x 1 paul compta 7406 2008-08-15 14:44 rapport2006.ods
-rw-rw-r-- 1 paul paul   7363 2008-08-15 14:44 rapport-activite.odt
-rw-rwxr-x 1 paul compta  255 2008-08-15 14:52 script.sh
```

- Signification des différents champs



Propriétaire : Lecture, écriture
Groupe : Lecture, écriture et exécution
Autres : Lecture et exécution

Remarques sur les droits

- Le droit « w » accordé à un répertoire permet :
  - D'y effacer des fichiers quels que soient le propriétaire et les droits qui s'appliquent à ces fichiers
  - Quand il est donné à un groupe, n'importe quel utilisateur de ce groupe peut supprimer des fichiers (dangereux)
- Les droits ne s'appliquent pas au « super-utilisateur »
  - Il a tous les droits sur tout le système de fichiers
  - C'est une très grande responsabilité puisque sous Linux tout ou presque repose sur les fichiers
    - La tendance évolue vers une utilisation très modérée voire interdite du compte « root »
- Le droit « x » accordé à un répertoire est un préalable indispensable pour exercer des droits sur les fichiers contenus
- L'utilisateur qui crée un fichier en devient le propriétaire
  - Ce fichier aura comme groupe propriétaire, le groupe primaire du propriétaire (Groupe principal auquel appartient le propriétaire)

Exercices

- Quel(s) utilisateur(s) pourra(ont) se déplacer dans le répertoire suivant ?

```
drwxr-x--- 26 paul marketing 4096 2008-08-28 16:11 paul
```

  - Réponse : « paul » et les membres du groupe « marketing »

- Qui pourra créer de nouveaux fichiers dans ce répertoire ?

```
drwxr-xrwx 26 jean marketing 4096 2008-08-28 16:11 bilans
```

  - Réponse : Tout le monde sauf les membres du groupe « marketing »

- Soit le fichier suivant :

```
-rwxr--r-- 26 sarah compta 25140 2008-08-28 16:11 rapport2006.odt
```

  - Situé dans le répertoire suivant :

```
drwxrwxrwx 26 jean compta 4096 2008-08-28 16:11 rapports
```

  - Qui pourra effacer ce fichier ?
    - Réponse : Tout le monde malheureusement !!!

Le super-utilisateur

- Le compte « root » possède tous les droits
  - Celui qui possède le mot de passe root peut tout faire
- La plupart des distributions récentes désactivent le compte « root »
  - Certains utilisateurs peuvent endosser temporairement le rôle du super-utilisateur
    - Avec la commande « sudo » (Super User Do)
      - La commande qui suit le « sudo » sera exécutée en tant que « root »

```
rout@fredon:~/Documents$ sudo commande
```

- En règle générale, on utilise « sudo » que temporairement
  - Pour des tâches administratives
    - Accéder à l'ensemble du système de fichiers
    - Gérer les utilisateurs
    - …

Usurper l'identité d'un utilisateur ?

- Parfois, le super-utilisateur doit endosser un autre rôle afin d'effectuer une tâche
- Que se passe t-il si le super-utilisateur crée un fichier ?
  - Ce fichier lui appartient
  - Selon les droits qui s'appliquent, les autres utilisateur ne peuvent pas y accéder
- Que faire alors ?
  - Se faire momentanément passer pour un autre utilisateur
  - Avec la commande « su »

```
root@ubuntu:~# su gen
gen@ubuntu:~$ touch toto.txt
gen@ubuntu:~$ ls -l
total 32
drwxr-xr-x 2 gen gen 4096 2009-08-22 14:49 Modèles
-rw-r--r-- 1 gen gen    0 2009-08-31 05:47 toto.txt
drwxr-xr-x 2 gen gen 4096 2009-08-22 14:49 Vidéos
gen@ubuntu:~$ exit
exit
root@ubuntu:~#
```

Je me fais passer pour « gen »

Le fichier créé appartient à « gen »

Je reprends mon rôle initial

Le masque de protection: umask

Le masque de protection de fichier permet de définir les droits par défaut de tout fichier créé.

Manipulation

Ce masque se comporte comme un filtre et utilise la notation numérique.
On parle de filtre car il ne contient pas la série des 3 chiffres octaux correspondants aux droits à allouer aux fichiers, mais celle correspondant aux droits à ne pas allouer.

Le système Unix affecte à un fichier les droits globaux résultant de la soustraction des droits maximaux 777 par le masque de protection.

Exemple : si le masque de protection vaut 037 alors 740 (=777-037) seront les droits alloués à tout nouveau fichier.

La commande permettant de définir un nouveau masque de protection est umask.

Syntaxe : umask droits

Exemple : umask 037

777 = rwx rwx rwx = 111 111 111
- 037 = --- -wx rwx = 000 011 111
= 740 = rwx r-- --- = 111 100 000

D'après cet exemple, tout nouveau fichier aura les droits 740 (rwxr-----) car le masque de protection vaudra 037 (----wxrwx).

Pour connaître la valeur du masque de protection, tapez umask sans attribut.

Les droits étendus : le SUID

- Permet de bénéficier de droits supplémentaires lors de l'exécution d'une commande
  - Un utilisateur quelconque peut alors avoir des droits supplémentaires seulement s'il exécute la commande ayant le SUID
- Exemple de la commande « passwd »
  - Elle permet de modifier son mot de passe
  - « passwd » doit écrire dans le fichier « /etc/shadow » et pourtant :

```
linux:~# ls -l /etc/shadow
-rw-r----- 1 root shadow 700 2007-12-04 18:39 /etc/shadow
```

Aucune permission d'écriture sur ce fichier

```
linux:~# ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 28480 2007-02-27 08:53 /usr/bin/passwd
```

La commande aura les droits du super-utilisateur même si n'importe quelautre utilisateur lance son exécution

En clair un utilisateur de type /homr/toto pourrait lui même redefinir son mots de passe une fois loggé :

toto@server $ passwd mots_de_passe_de_toto

Les droits étendues: SUID

Manipulation

Pour ajouter un droit SUID à un programme, utiliser la commande chmod de la même manière que vous procéderiez pour un droit normal.

Notation symbolique

Syntaxe : chmod u+s fichier-exécutable.

Exemple : chmod u+s data/hotprog.

Notation numérique

La valeur numérique d'un SIUD est 4000 et s'ajoute à la valeur de la série numérique globale.

Syntaxe : chmod 4+droits fichier-exécutable.

Exemple : chmod 4755 monprog.

Dans cet exemple on donne les droits rwsr-xr-x au fichier monprog. Ne pas oublier que le droit SUID s'affiche à la place du droit en exécution du propriétaire sans que ce dernier soit supprimé!

Nota : la présence du droit SUID suppose la présence du droit en exécution qui permet de lancer le fichier exécutable.

Les droits étendus : le SGID

- Identique au SUID mais appliqué au groupe propriétaire
  - La commande obtiendra les droits du groupe propriétaire s'il elle est exécutée par un autre utilisateur
- Attention, appliquée à un répertoire, le SGID :
  - Modifie le groupe propriétaire d'un fichier créé dans le répertoire
  - Ce ne sera plus le groupe primaire du propriétaire
    - Mais plutôt le groupe propriétaire du répertoire
  - Il y a donc un mécanisme d'héritage entre le répertoire et les fichiers nouvellement créés qu'il contient
- Exemple :

```
drwxrws---   2 root compta  4096 2008-08-24 13:05 docs-compta
```

SGID positionné sur « docs-compta »

```
-rw-r--r-- 1 paul compta 0 2008-08-24 13:09 nouveau.txt
```

Le fichier nouvellement crée par paul appartient au groupe « compta »

Les droits étendues: SGID

Manipulation

La valeur numérique du droit SGID est 2000, il est symbolisé par la lettre s et est affiché à la place du droit d'exécution du groupe.

Notation symbolique

Syntaxe : chmod g+s fichier-exécutable|répertoire.

Exemple : chmod g+s data/.

Notation numérique

Syntaxe : chmod 2+droits fichier-exécutable|répertoire.

Exemple : chmod 2755 monprog.

Dans cet exemple on donne les droits rwxr-sr-x au fichier monprog. Ne pas oublier que le droit GUID s'affiche à la place du droit en exécution du groupe sans que ce dernier soit supprimé!

Les droits étendues: Sticky Bit

Le droit Sticky Bit (appellé aussi bit collant) est alloué à la catégorie autres d'un répertoire.

Il permet d'interdire à tout utilisateur (sauf le root) de supprimer un fichier dont il n'est pas le propriétaire, quelque soient ses droits.

Si le répertoire en question est accessible en écriture par n'importe quel utilisateur (rwxrwxrwx), n'importe qui peut poser ce bit collant qui protège tous les fichiers d'une suppression ou modification de la part d'un utilisateur autre que son propriétaire.

Ce bit collant permet donc d'aller à l'encontre du droit en écriture d'un répertoire dont héritent les fichiers du répertoire.

Sa valeur numérique est 1000 et est représenté symboliquement par t.

Exemple : chmod 1755 tmp/.

Exemple : chmod a+t tmp/.

Ce droit s'affiche en lieu et place du droit en exécution de la catégorie autres.

Modifier les droits avec « chmod »

- La commande « chmod » permet de modifier les droits :
  - 2 syntaxes différentes

```
chmod [OPTION]... MODE[,MODE]... FILE...
```

```
chmod [OPTION]... OCTAL-MODE FILE...
```

- Mode symbolique :
  - Basé sur des symboles (ugoa) et des opérateurs (+,-,=)
  - **u** (user), **g** (group), **o** (others), **a** (all users)
  - **+** (Ajouter le droit), **-** (Retirer le droit), **=** (Ajouter le droit et retirer tous les autres)
  - Exemple (Ajoute le droit d'exécution au propriétaire) :

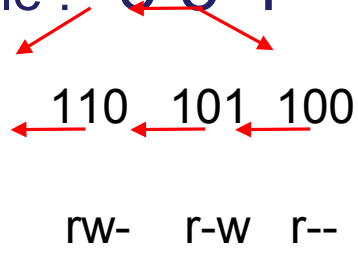
```
chmod u+x rapport.txt
```

- Mode octal :
  - Basé sur des nombres de 0 à 7
  - A chaque bit de la traduction binaire correspond un droit
  - Exemple (rw- rw- r--) :

```
chmod 664 rapport.txt
```

Mode octal de « chmod »

- Les droits sont représentés par un nombre octal (Base 8)
  - De 1 à 7
- La représentation binaire (base 2) donne le détail des droits
  - Exemple : 6 5 4



Propriétaire : Lecture, écriture
Groupe : Lecture et exécution
Autres : Lecture seulement

- Ce mode permet de modifier tous les droits en même temps
  - A utiliser avec précaution
  - Très efficace pour s'assurer que tous les fichiers ont les mêmes droits
  - Utilisé pour sécuriser les accès des utilisateurs aux fichiers

Exemples d'utilisation de « chmod »

- Retirer le droit d'écriture au propriétaire et au groupe

```
chmod u-w,g-w  /home/paul/secret.txt
```

- Positionner les droits en « rwx r-w --- »

```
chmod 750  /home/paul/secret.txt
```

- Ajouter le droit de lecture aux autres de tous les fichiers

```
chmod  o+r  /home/paul/*
```

- Retirer le droit d'écriture au groupe propriétaire
  - Pour tous les fichiers et répertoires d'un répertoire donné (Option R)

```
chmod -R g-w /home/paul/*
```

- Donner tous les droits à tout le monde (déconseillé)

```
chmod 777 /home/paul/secret.txt
```

Modifier l'appartenance avec « chown »

- La commande « chown » (Change owner) permet de changer l'appartenance
  - Pour le propriétaire ou le groupe propriétaire
- Syntaxe :

```
chown [OPTION]... [OWNER][:[GROUP]] FILE...
```

- Exemples :
  - Modification du propriétaire (paul)

```
chown paul /usr/docs/toto.txt
```

  - Modification du groupe propriétaire (compta)

```
chown :compta /usr/docs/toto.txt
```

  - Modification du propriétaire (jean) et du groupe (direction)
    - Pour tout le contenu du répertoire (Option R - récursif)

```
chown –R jean:direction /usr/docs/rapports/
```

Gestions des comptes utilisateurs

Créer un compte pour un nouvel utilisateur :

**Cela signifie lui permettre d'être connu du poste local, s'y loguer, avoir un accès complet sur son rép. Personnel.
Mais aussi dans une configuration réseau, de pouvoir se connecter à son compte par telnet et ftp,
De pouvoir bénéficier de services réseau de partage distant (sous Linux par NFS
et sous Windows 9x par SMB).**

1 Pour créer l'utilisateur stagex, root passe la commande :

#useradd stagex

**Ceci crée : le répertoire personnel /home/stagex, portant par défaut le nom du compte
une nouvelle entrée dans les 2 fichiers fondamentaux /etc/passwd et /etc/group.
Pour connaitre les options de useradd (indispensable pour gérer les comptes à l'aide de scripts) faire un man useradd**

Exemple2 :
 # useradd -d /home/jojo -s /bin/bash -m jojo

Description des options

-d <dossier> : spécifie le dossier de l'utilisateur (son $HOME)
-s <shell> : spécifie le shell qui sera utilisé au démarrage d'une session en console
-m : crée le $HOME de l'utilisateur et y met les fichiers de configuration du shell de l'utilisateur

On définit le mots de passe de jojo

 # passwd jojo
 # puis entrer le password de jojo à definir

Gestions des comptes utilisateurs

2)Supprimer le compte d'un utilisateur (non connecté), au hasard .. totox.

 # userdel [-r] totox

L'option -r supprime aussi le rép. personnel et les fichiers de l'utilisateur
La commande supprime toute trace de l'utilisateur dans le fichier de configuration : /etc/passwd y compris dans les groupes d'utilisateurs.

3)Modifier le compte de l'utilisateur toto

#usermod [options] totox

usermod -l nouveau_login ancien_login
usermod -s nouveau_shell login (pour changer le shell par défaut de l'utilisateur)
usermod -u nouveau_uid login (pour changer l'uid de l'utilisateur, attention : les permissions ne seront pas modifiée)
usermod -d nouveau_dossier login (pour changer le dossier personnel de l'utilisateur)
usermod -m -d nouveau_dossier login (change le dossier, y copie le contenu de l'ancien et ajuste les permissions)
usermod sans aucun argument pour voir la liste des options possibles

Gestions des groupes utilisateurs

1)Ajout d'un utilisateur à un groupe

Pour ajouter l'utilisateur toto au groupe greta (qui permet d'accéder aux applications graphiques), on va lancer la commande :

 # adduser toto greta

Ou

Editer le fichier /etc/group et d'ajouter une liste d'utilisateurs (séparés par des virgules) sur la ligne du groupe (ou utiliser Linuxconf)

Un groupe est, aussi pour Linux, un ensemble d'utilisateurs qui partagent les mêmes fichers et répertoires.
Nous verrons que les fichiers accordent des droits d'accès réglables à ces groupes.

Chaque utilisateur doit faire partie au moins d'un groupe, son groupe primaire. Celui-ci est défini au moment de la création du compte, et par défaut, l'utilisateur appartient à un nouveau groupe créé, portant son nom.

Ainsi, dans /etc/passwd chaque utilisateur possède un groupe par défaut, précisé par son identifiant gid dans ce fichier.

L'appartenance au groupe primaire n'étant pas exclusive, tout utilisateur peut faire partie de plusieurs autres groupes, appelés ses groupes secondaires.
Mais le rôle joué par le groupe primaire demeure prépondérant, comme nous le verrons dans le système des permissions des fichiers.

Gestions des groupes utilisateurs

2) Pour lister tous les groupes (primaire et secondaires) d'un utilisateur « toto » :

 # groups toto

3) Pour créer un nouveau groupe

 # groupadd greta

4) Supprimer un groupe, au hasard .. encore greta.

 # groupdel greta

Le groupe est supprimé du fichier /etc/group.

Gestions des comptes utilisateurs

Ceux qu'ils faut retenir

1) Tout ce qui concerne la gestion et l'authentification des utilisateurs est inscrit dans un seul fichier :

/etc/passwd

2) La gestion des groupes est assurée par :

/etc/group

3) Les mots de passe cryptés sont maintenant placés dans

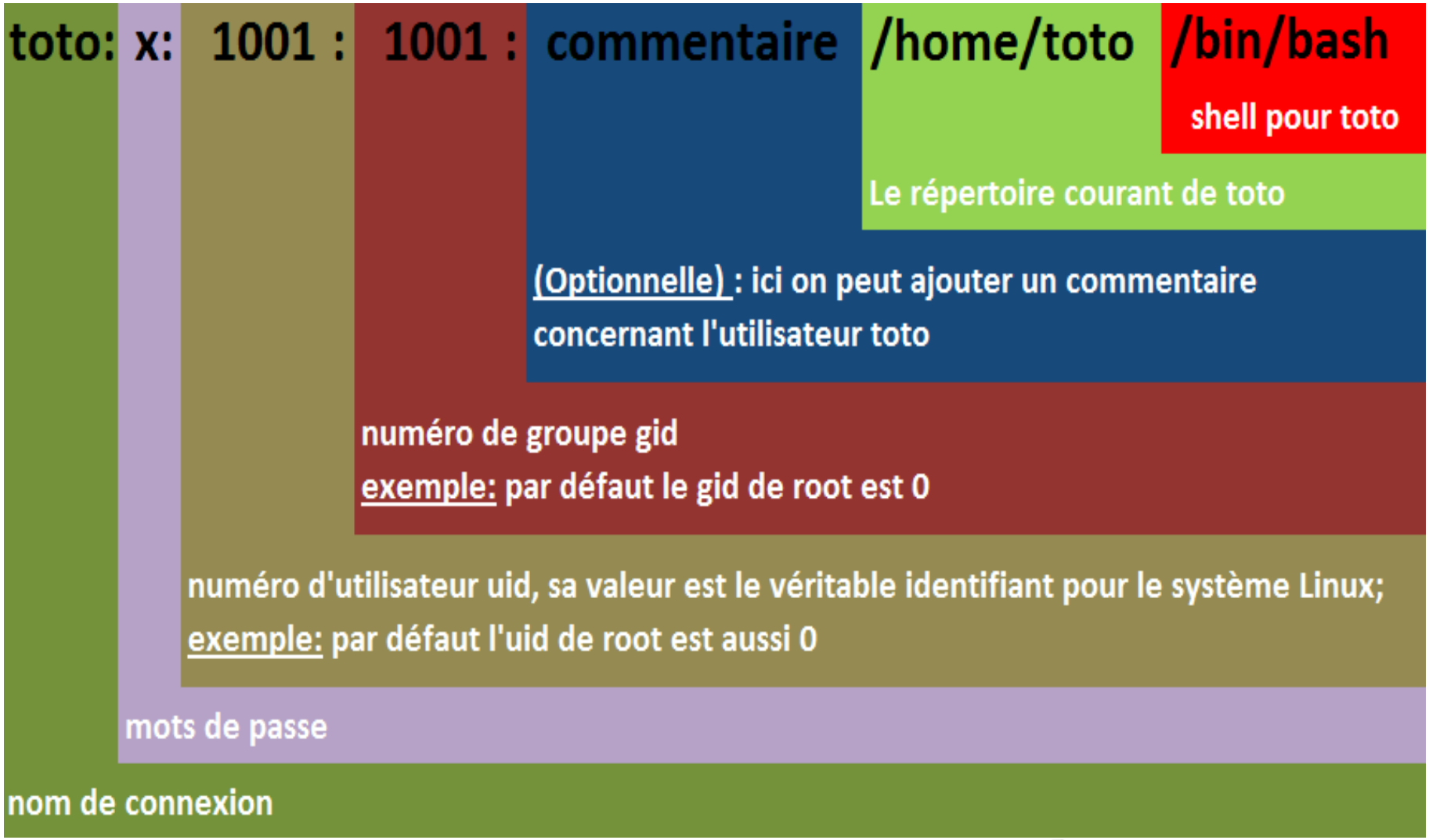
/etc/shadow

par sécurité lisible seulement par root.

Gestions des comptes utilisateurs

Ceux qu'ils faut retenir

Structure de : /etc/passwd



Gestions des comptes utilisateurs

Ceux qu'ils faut retenir

Structure de : /etc/group

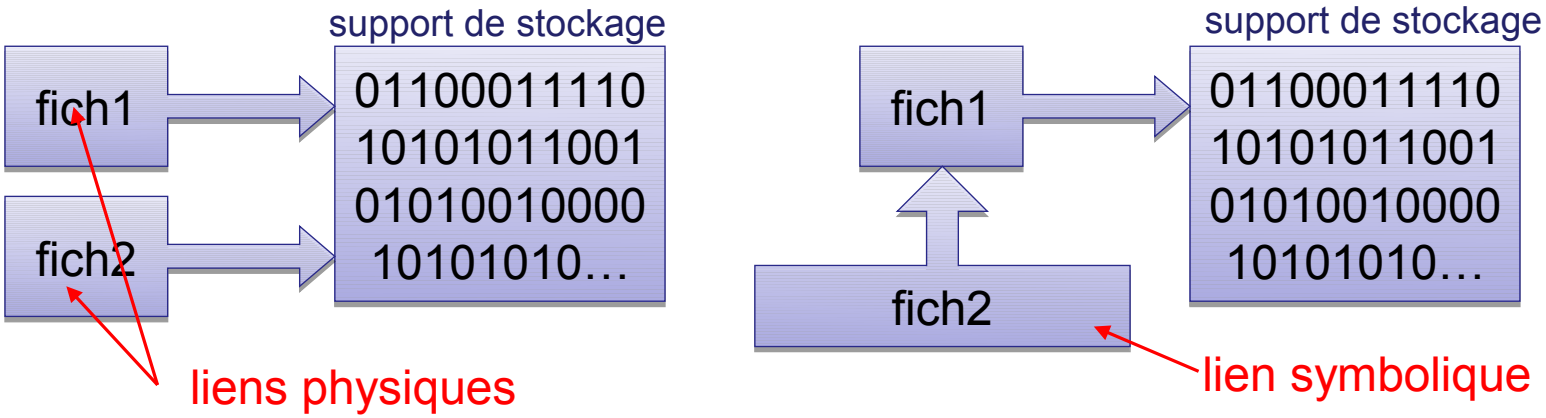
Gestions des comptes utilisateurs

Ceux qu'ils faut retenir

Structure de : /etc/shadow

Les liens

- Un lien est un type spécial de fichier qui fait référence à un autre fichier
- Axe central du fonctionnement de Linux, le lien permet :
  - De créer des raccourcis vers des fichiers existants
    - La compatibilité des logiciels entre les distributions Linux est assurée par les liens
  - D'éviter de stocker plusieurs fois le même fichier dans des répertoires différents
- Un petit dessin :



Les liens symboliques

- Le lien symbolique est une référence vers un fichier cible
  - Lorsque le fichier cible est effacé, le lien est rompu
  - Lorsque le lien est effacé, le fichier cible n'est pas effacé
- Exemple :

```
rout@fredon:~/Documents$ ls -l
total 8
lrwxrwxrwx 1 rout rout   29 2008-08-25 14:23 ip -> /proc/sys/net/ipv4/ip_forward
drwxr-xr-x 3 rout rout 4096 2008-06-02 14:20 software
drwxr-xr-x 3 rout rout 4096 2008-07-29 15:54 vmware-tools
```

Nom du lien

Indique que c'est un lien

Emplacement du vrai fichier

- La commande « ln » avec l'option « -s » est utilisée pour créer un lien symbolique

```
rout@fredon:~/Documents$ ln -s /proc/sys/net/ipv4/ip_forward ip
```

Cible (Target)

Nom du lien (link name)

Les liens physiques

- Un lien physique est associé à un emplacement sur le support de stockage
  - 2 liens peuvent être associés au même « inode »
  - Similaire à la notion de « pointeurs » du langage C
  - Deux liens physiques sont considérés comme 2 fichiers indépendants
    - Même si leur contenu est au même emplacement sur le support
  - Le lien physique est vu comme un fichier régulier
- Créer un lien physique avec la commande « ln » :

nom fichier

```
rout@fredon:~/Documents$ ln /home/paul/Documents/rapport2007-2008.doc rap0708
rout@fredon:~/Documents/essais$ ls -il
total 176
470930 -rw-r--r-- 2 rout rout 84091 2008-08-25 14:48 rap0708
470930 -rw-r--r-- 2 rout rout 84091 2008-08-25 14:48 rapport-annee2007_2008.doc
```

cible

L'« inode » est identique. Il s'agit bien de liens physiques

Nombre de liens vers cet inode. C'est un indice permettant de supposer qu'il s'agit d'un lien