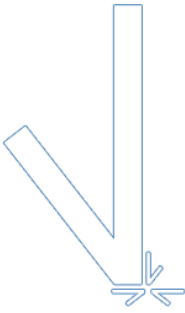


Systemes d'Exploitation



Objectif :

Comprendre comment un système
d'exploitation fonctionne et
comment l'utiliser

Systemes d'Exploitation



Systeme de Fichier

(cours précédent : Systemes d'Exploitation – Utilisateurs et Sessions)

I. Systèmes d'Exploitation

I.5. Système de Fichier



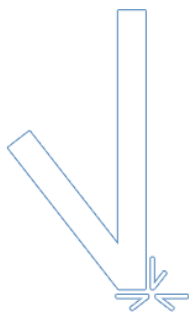
I. Systèmes d'Exploitation

1. Sommaire
2. Introduction
3. Noyau et Pilotes
4. Utilisateurs et Sessions
5. Système de Fichier
 - a. Fichiers
 - b. Inodes
 - c. Partitions
 - d. Format
 - e. Système de Fichier Virtuel
 - f. Montage
 - g. Structure par défaut
6. Permissions et Droits
7. Shell et Utilitaires
8. Gestion de la Mémoire
9. Programmes et Processus
10. Variables d'Environnement
11. Scripting Shell
12. Gestion des Paquets


I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.a Fichiers - fondamentaux

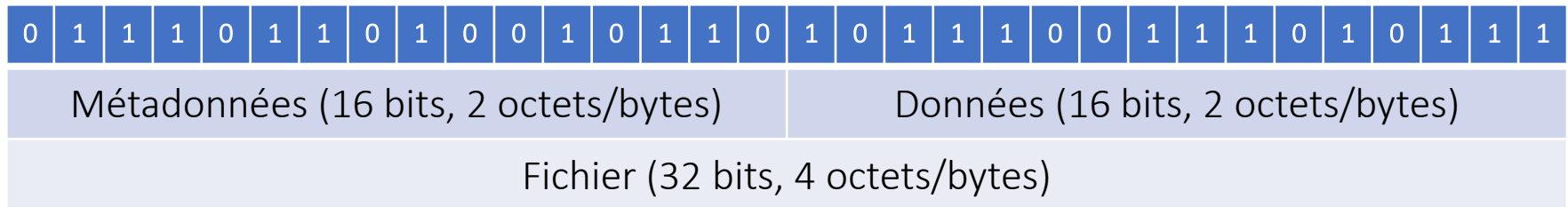


 La mémoire non volatile (disque dur, clé USB, etc.) stocke les informations sous forme d'octets (bytes).

 Ces octets représentent des **données** (texte, son, images, etc.) et des **métadonnées** (données sur les données) qui identifient la **nature des données**.

 Les **métadonnées et les données** ensemble sont appelées **fichier**.

 Exemple :

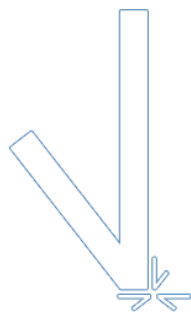


(Ce fichier ne représente pas de données pertinentes, c'est un exemple)

I. Systèmes d'Exploitation

I.5. Système de Fichier

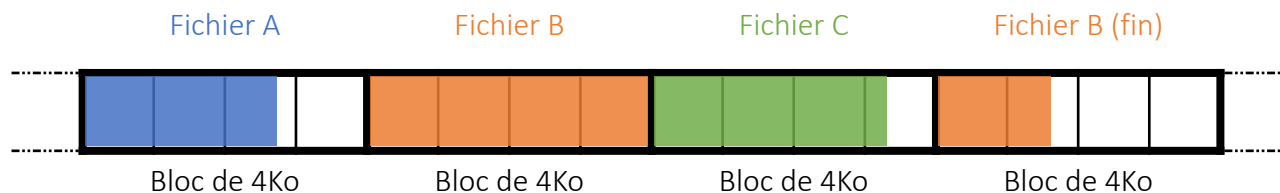
I.5.a Fichiers – stockage de données



- 📏 L'espace disponible sur un périphérique de stockage est généralement divisé en blocs de 4096 octets (4 kilooctets). Ces blocs sont appelés **unités d'allocation** ou **clusters** (groupes).

- 📁 Les fichiers sont stockés sur **1 ou plusieurs blocs non contigus** :

- 📋 Exemple :



Le fichier A est stocké dans 1 bloc, le fichier C est stocké dans 1 bloc. Le fichier B est stocké dans **2 blocs non contigus**.

I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.b inodes - fondamentaux

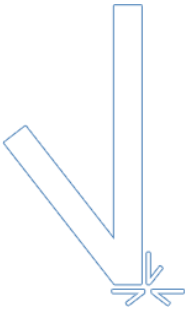


- 📄 Pour rassembler des fichiers stockés de manière non contiguë, nous avons besoin d'un « Sommaire ». Ce système de « Sommaire » doit au minimum se baser sur :
 - 👉 Un **nom** qui identifie le fichier ;
 - 🔗 La **liste des blocs** auxquels accéder pour **reconstituer** les données.
- 📄 La **liste des blocs** qui constituent un fichier est stockée sur un périphérique de stockage dans une structure de données appelée **nœud d'index** ou **inode**.
- 🎯 Un inode contient, au minimum, les informations relatives à l'emplacement d'un fichier sur un périphérique de stockage.

I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.b inodes - fondamentaux



- 📁 Les noms de fichiers sont stockés dans des fichiers qui associent **un nom à un identifiant d'inode**. Ces fichiers spéciaux sont appelés **répertoires**.
- 📄 Chaque nom dans un fichier de répertoire est associé à un identifiant d'inode.

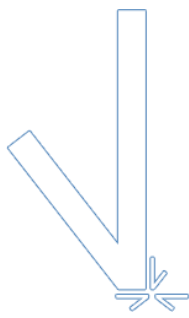
! UN RÉPERTOIRE EST UN FICHIER

- > Un **inode** peut :
 - 🔗 Pointer vers des blocs : c'est un **lien physique**.
 - 🔗 Pointer vers un autre inode : c'est un **lien symbolique**.

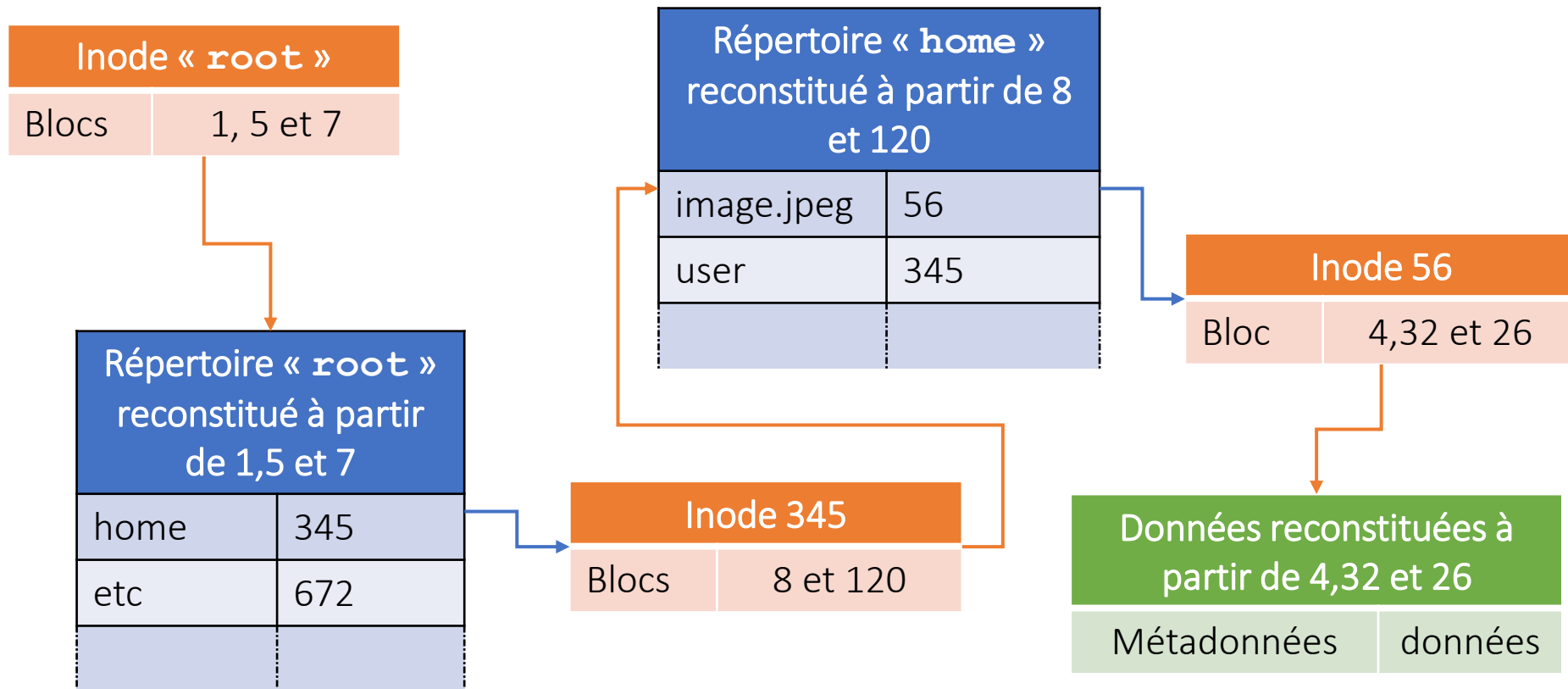
I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.b inodes - fondamentaux



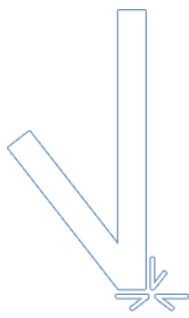
📁 Comment fonctionnent les **inodes** (exemple simplifié : accès à /home/image.jpeg) :



I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.b inodes – commandes utiles



- Les noms et les inodes associés dans un fichier de répertoire peuvent être affichés en utilisant :

ls -i <path to directory>

(ls suivi de l'argument -i et du chemin du répertoire)

- Le contenu d'un inode associé à un nom peut être affiché en utilisant :

stat <path to file or directory>

(stat suivi du nom du fichier ou du répertoire)

- Nous pouvons créer un nouveau lien physique vers un fichier avec :

ln <target filename> <name>

(ln suivi du chemin du fichier et du nom du lien physique)

- Nous pouvons créer un lien symbolique ou symlink vers un fichier avec :

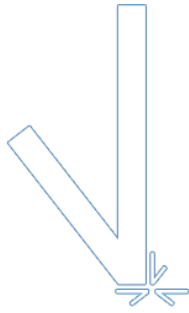
ln -s <target filename> <link name>

(ln suivi de l'argument -s, du chemin du fichier et du nom du lien symbolique)

I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.c Partitions – fondamentaux

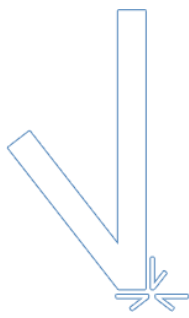


- Les inodes sont stockés **aux côtés** des données des fichiers sur le périphérique de stockage.
 - COMMENT LE SYSTÈME D'EXPLOITATION LOCALISE-T-IL L'INODE « RACINE » ?
- Un bloc de données appelé **SuperBloc** est situé AVANT tout inode ou fichier.
- Le SuperBloc contient les informations nécessaires:
 - Pour localiser l'inode « racine » ;
 - Concernant le nombre maximum d'inodes ;
 - Concernant la taille des blocs (unités d'allocation) ;
 - Concernant le nombre maximum de blocs vers lesquels un inode peut pointer ;
 - ...
- Le SuperBloc contient les métadonnées sur la structure des inodes et des données des fichiers.

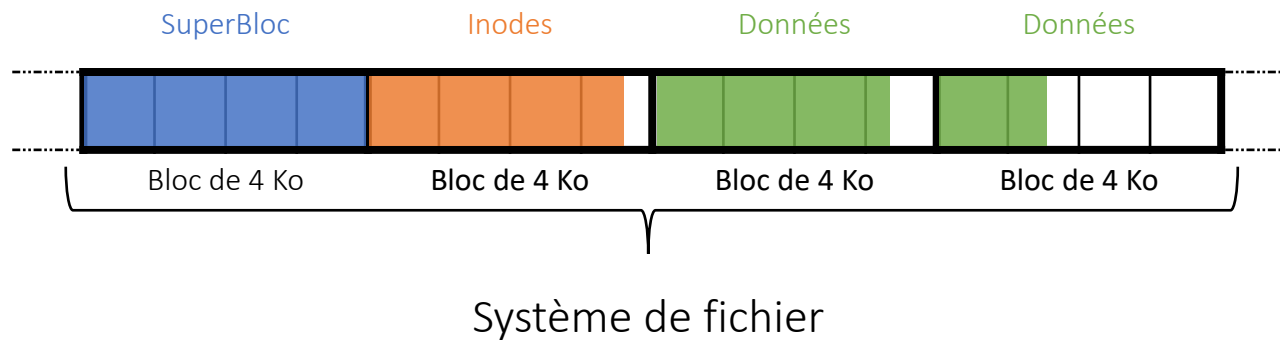
I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.c Partitions – fondamentaux



- Le **SuperBloc**, les **Inodes** et les **données** forment ce que nous appelons un **système de fichiers**.

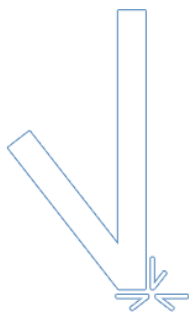


- Plusieurs **systèmes de fichiers** peuvent être stockés sur un périphérique de stockage. Nous pouvons appeler ces sections de stockage des **partitions**

I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.d Format – fondamentaux

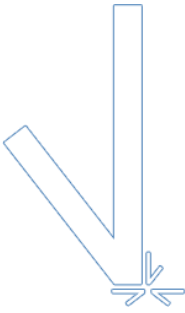


- Chaque **partition** de périphérique de stockage peut avoir un **format de système de fichiers** différent. Les formats les plus connus sont :
 - Ext4 (pour Linux/Unix)
 - Ext3 (pour Linux/Unix)
 - NTFS (par Microsoft)
 - FAT32 (par Microsoft)
 - APFS (par Apple)
 - ...
- Le format indique :
 - Le nombre total de blocs auxquels un inode peut pointer et donc la taille maximale d'un fichier ;
 - Le nombre total de blocs auxquels tous les inodes peuvent pointer et donc la taille maximale du système de fichiers;
 - Le nombre total d'inodes qui peuvent être créés et donc le nombre maximal de fichiers que nous pouvons créer;
 - ...

I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.d Format – fondamentaux

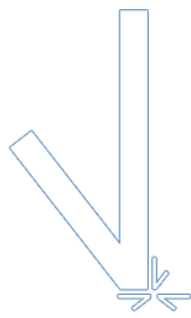


- Le format définit la structure de données du SuperBloc, et ainsi, le système d'exploitation peut obtenir le format en analysant le SuperBloc.
- Vous pouvez consulter [une liste exhaustive des formats et caractéristiques des systèmes de fichiers](#) sur la Wikipédia.

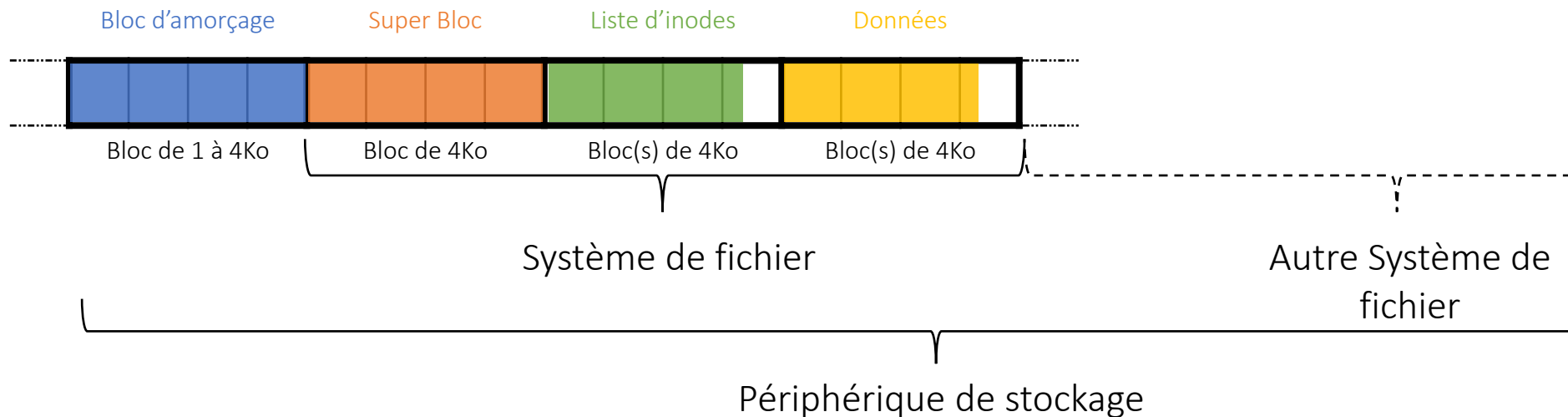
I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.d Format – fondamentaux



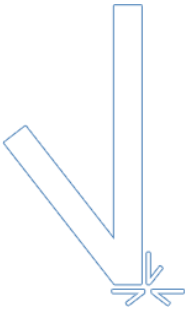
- Le premier bloc du périphérique de stockage est appelé **bloc de d'amorçage (boot block)**. Il amorce le système de fichiers. Ce bloc a le contenu suivant :
 - Il contient les **positions des SuperBlocs des partitions** sur le périphérique de stockage.
 - Si le périphérique de stockage est le périphérique de stockage principal, le bloc d'amorçage contient le **chargeur d'amorçage (boot loader) du noyau**. Un bloc d'amorçage contenant le chargeur d'amorçage est appelé **MBR – Master Boot Record** –.



I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.d Format – commandes utiles



- Nous pouvons vérifier la **liste des partitions** sur les périphériques de stockage avec les utilitaires suivants :

lsblk ou
fdisk -l

(**fdisk** suivi de l'argument **-l**)

- Pour connaître les **formats** des partitions, nous pouvons utiliser l'utilitaire suivant :

blkid

- Pour obtenir des **informations générales** concernant les partitions :

df -T

(**df** suivi de l'argument **-T**)

- Et pour avoir des informations sur le **nombre d'inodes utilisés ou disponibles** :

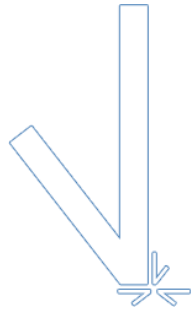
df -i

(**df** suivi de l'argument **-i**)

I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.e Système de fichier virtuel

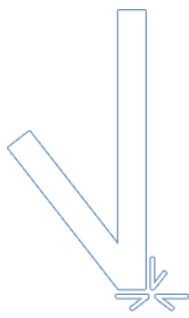


- Toutes les commandes de système de fichiers utilisées jusqu'à présent effectuent des appels système vers le noyau via le commutateur de système de fichiers virtuel – **VFS** (Virtual Filesystem Switch).
- Le **VFS** est la couche d'abstraction qui permet aux utilisateurs et aux applications d'accéder à **divers systèmes de fichiers** via une **interface unique**.
- Le VFS permet d'accéder à **tous les systèmes de fichiers** dans le **même arbre hiérarchique** de fichiers et de répertoires.
- Le VFS expose une API pour :
 - **Monter** (« charger ») un système de fichiers depuis un périphérique de stockage (n'importe quel périphérique de stockage, volatile ou non volatile) ;
 - **Créer** un système de fichiers sur un périphérique de stockage ;
 - **Écrire, lire, ...** des fichiers ;
 - ...

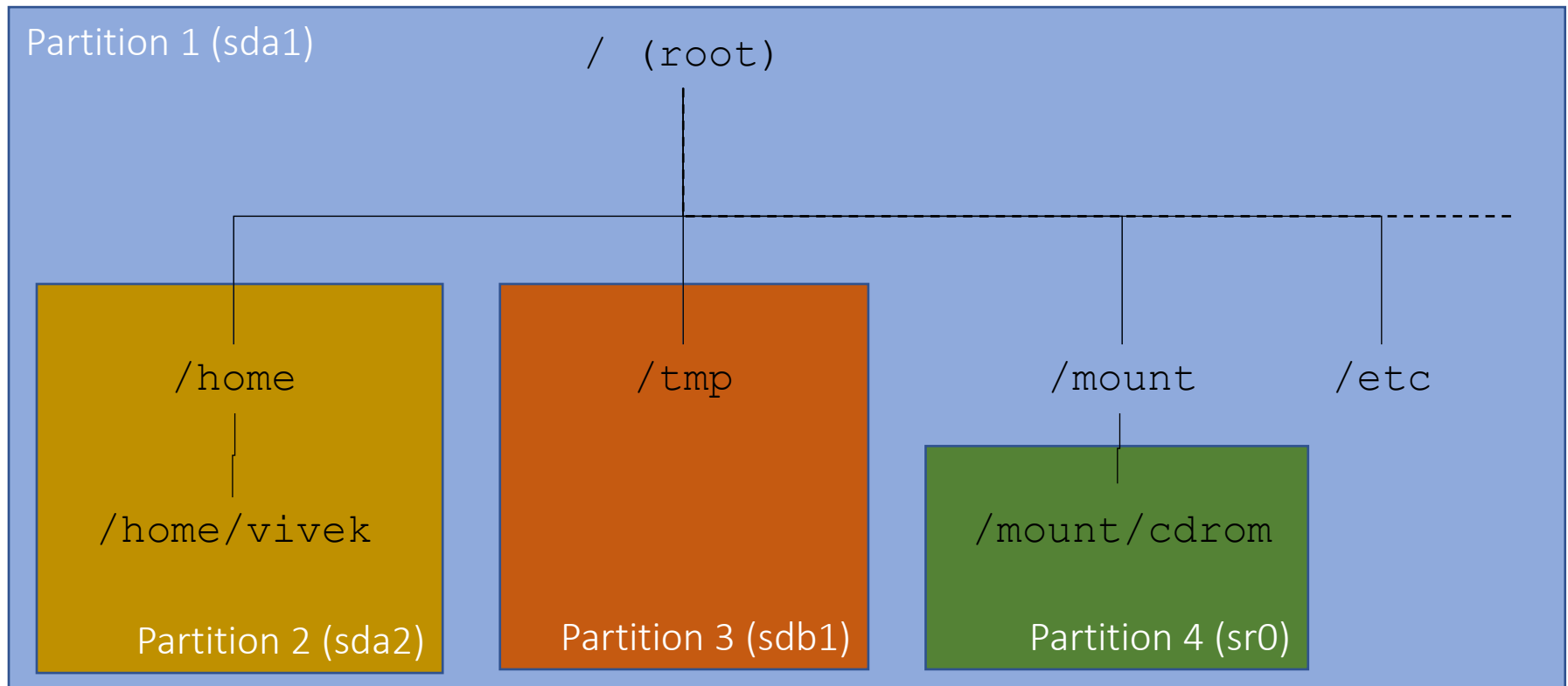
I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.e Système de fichier virtuel



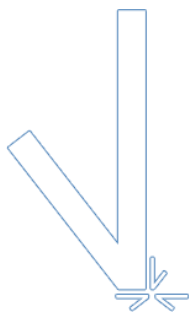
- Exemple simplifié de représentation d'arborescence VFS:



I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.f Montage



- Charger une partition (et donc un système de fichiers) dans le **VFS** s'appelle « **monter** ».
- Avec la configuration appropriée, le système d'exploitation peut effectuer cette opération **automatiquement**. Cette configuration de « montage » statique est écrite dans le fichier `/etc/fstab` :

cat /etc/fstab

(cat suivi du chemin vers le fichier `fstab`)

- Le « montage » peut être effectué **manuellement** avec la commande :

mount <chemin vers la partition> <répertoire cible>

(mount suivi du chemin de la partition et du répertoire cible où la partition sera montée)

- Le « démontage » peut être effectué avec la commande :

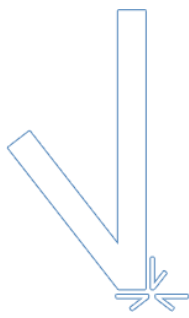
umount <dossier>

(umount suivi du répertoire où la partition est montée)

I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.g Structure par défaut



- Le système de fichiers **du point de vue de l'utilisateur** est un arbre avec les répertoires suivants :
 - `/` : **répertoire racine**, tout est sous `/`
 - `/bin` : répertoire des **binaires** – (programmes) pour tous les utilisateurs.
 - `/sbin` : **binaires système** – pour les utilisateurs administrateurs.
 - `/usr` : **binaires utilisateur** – pour tous ou seulement certains utilisateurs.
 - `/lib` : **bibliothèques** – utilisées par les binaires.
 - `/boot` : **démarrage** – contient le chargeur d'amorçage et le noyau.
 - `/dev` : **périphériques** – chaque fichier sous ce répertoire est une interface d'entrée/sortie vers un périphérique physique ou un pseudo-périphérique (un fichier qui a un comportement spécifique d'entrée/sortie).

I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.g Structure par défaut

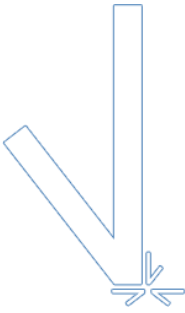


- `/etc` : répertoire des **configurations**. Ce répertoire contient les fichiers de configuration pour les programmes utilisateurs ou liés aux comportements du système d'exploitation.
- `/home` : répertoire des **utilisateurs**. Ce répertoire contient un répertoire par utilisateur du système (sauf l'utilisateur `root`).
- `/root` : répertoire de l'utilisateur `root`.
- `/tmp` : répertoire des fichiers **temporaires**.

I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.g Structure par défaut



- `/media` : répertoire de montage des **médias amovibles**.
- `/proc` : répertoire des **processus en cours d'exécution**.
- `/sys` : répertoire des **variables système** – contient des informations sur le système d'exploitation et les périphériques matériels.
- `/var` : répertoire des **fichiers variables** – contient des fichiers tels que les journaux système, les e-mails et d'autres fichiers qui changent pendant l'exécution du système d'exploitation.
- ...

I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.g Structure par défaut



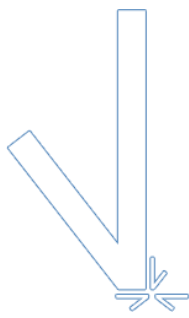
- Dans un système de fichiers hiérarchique, le symbole / est le séparateur entre un répertoire et l'un de ses sous-répertoires.
- Le répertoire racine n'a pas de nom, donc nous faisons référence à ce répertoire en utilisant le symbole /.
- Nous pouvons écrire le chemin vers un fichier ou un répertoire en utilisant une **notation absolue** (en partant du répertoire racine) comme suit :

`/chemin/vers/un/répertoire/ou/un/fichier`

I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.g Structure par défaut



- Nous pouvons écrire le chemin vers un fichier ou un répertoire en utilisant un **chemin relatif** par rapport au répertoire courant. Par exemple :

chemin/vers/un/répertoire/ou/un/fichier

- Pour **lister** le contenu d'un répertoire, nous pouvons utiliser :

ls </chemin/vers/un/répertoire>

(ls suivi d'un chemin de répertoire)

- Pour **changer** de répertoire courant, nous pouvons utiliser :

cd </chemin/vers/un/répertoire>

(cd suivi d'un chemin de répertoire)

- Pour **savoir** dans quel répertoire nous nous trouvons, nous pouvons utiliser :

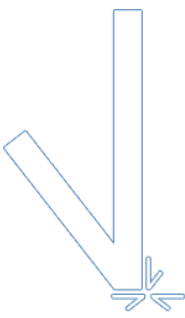
pwd

(print working directory)

I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.g Structure par défaut



- Certains « **alias** » spéciaux correspondant à des chemins absolus spécifiques sont mis à disposition par le VFS :
 - Pour le répertoire courant, c'est le `.`
 - Pour le répertoire parent du répertoire courant, c'est le `..`
 - Pour le répertoire personnel de l'utilisateur actuel, c'est le `~`
- Pour **créer** un **répertoire**, nous pouvons utiliser :

`mkdir <chemin vers le nouveau répertoire>`

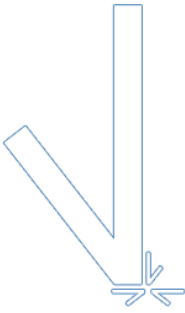
- Pour **supprimer** un **répertoire**, nous pouvons utiliser :

`rmdir <chemin vers le répertoire à supprimer>`

I. Systèmes d'Exploitation

I.5. Système de Fichier

I.5.g Structure par défaut



- Pour créer un fichier, nous pouvons utiliser :

touch <chemin vers le nouveau fichier>

- Pour supprimer un fichier ou un répertoire, nous pouvons utiliser :

rm <chemin vers le fichier à supprimer>

- Pour copier un répertoire ou un fichier, nous pouvons utiliser :

cp <chemin vers le fichier ou répertoire source> <destination>

- Pour déplacer (ou *renommer*) un répertoire ou un fichier, nous pouvons utiliser :

mv <chemin vers le fichier ou répertoire source> <destination>



Permissions et Droits

(voir cours suivant : Systèmes d'Exploitation – Permissions et Droits)