

# **Systeme d'exploitation**

La gestion du stockage



**Qu'est-ce qu'un fichier ?**

**Qu'est-ce qu'un dossier ?**

**Qu'est-ce qu'une arborescence ?**



# Sommaire

---

De quoi s'agit-il ?

- 01** L'abstraction
- 02** Les systèmes de fichiers
- 03** L'approche GNU/Linux
- 04** L'approche windows



# L'abstraction



## Métaphore de bureau

---

Ranger ses  
documents

Informations rangées dans un fichier (*file*)

Fichiers rangés dans des répertoires (*directory*)

Fichiers et répertoires identifiés par :

- Emplacement (logique, indépendant du stockage physique)
- Nom (texte quelconque) unique dans l'emplacement

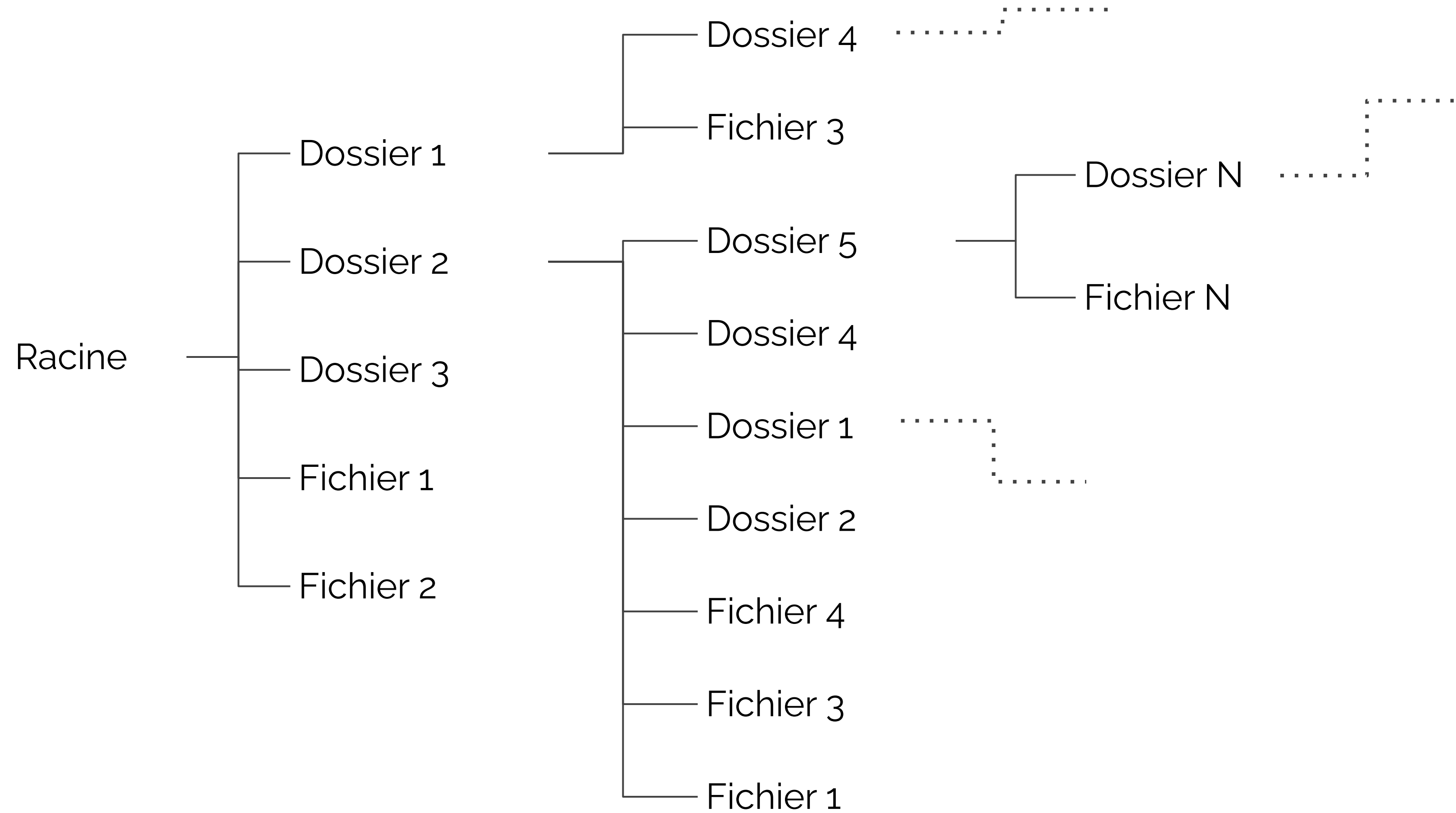
Point de départ : répertoire racine => nom conventionnel



# Arborescence de fichiers



Structure  
branches - noeuds  
- feuilles





## Pseudo dossier . et ..



Des histoires de  
points

Chaque dossier contient 2 pseudo dossiers particuliers

- Le dossier .. désigne le **dossier parent**
- Le dossier . désigne le dossier **lui-même**



## Pseudo dossier . et .. (suite)

Des histoires de points

```
wilder@host:~$ ls -a
wilder@host:~$ ls -A
wilder@host:~$ ls -A | wc -l

wilder@host:~$ mkdir test1 && cd test1;touch file1 file2 file3
wilder@host:~/test1$ ls
wilder@host:~/test1$ rm *
wilder@host:~/test1$ ls
wilder@host:~/test1$ ls -a

wilder@host:~/test1$ cd ..
wilder@host:~$
```





# Chemin d'accès



Aiguille et botte  
de foin

## Notion de chemin

- Point de départ
  - Chemin absolu => depuis la racine
  - Chemin relatif => depuis le dossier courant
- Étapes => dossiers
- Point d'arrivée (dossier ou fichier)
- Format => séparateur d'étapes





# Anatomie d'un chemin absolu



Depuis la racine

**<racine>[<séparateur><dossier> ...][<séparateur>|<séparateur><fichier>]**

Exemples GNU/Linux :

- /home/wilder/ : le dossier personnel de wilder
- /var/log/auth.log : le fichier journal de connexion

Exemple Windows :

- C:\Users\wilder : le dossier personnel de wilder
- C:\Windows\system32\notepad.exe : le programme Bloc-notes



# Notion de dossier courant



Sa place dans  
l'arborescence

Chaque processus est associé à un répertoire :

- Point de départ des chemins relatifs
- Dynamique

Par exemple : votre shell (affiché dans le prompt)



# Anatomie d'un chemin relatif



Ne pas repartir du  
début

**[<dossier><séparateur> ...] <fichier> | <dossier> [<séparateur>]**

Exemples GNU/Linux :

- local/bin/
- .ssh/id\_ed25519.pub

Exemple Windows :

- Quêtes\PowerShell\
- Pictures\Photos\me.jpg



## La variable d'environnement “path”

---

Où sont les  
commandes

Certaines *commandes* shell sont des programmes (donc des fichiers).

Via quels chemins le shell va-t-il chercher ces commandes ?

=> Par la valeur d'une variable d'environnement spécifique

Sur GNU/Linux, variable **PATH**. Plus d'info : [article sur linuxhint.com](https://linuxhint.com/path/)

En PowerShell, variable **\$env:path** : [doc officielle](#)



# Les systèmes de fichiers



# Systeme de gestion de fichiers



FS

Objectifs :

- Fournir une arborescence abstraite répertoires/fichiers
- Organiser l'information sur un périphérique de stockage
- Traite périphériques de stockage par blocs (disque, partitions)



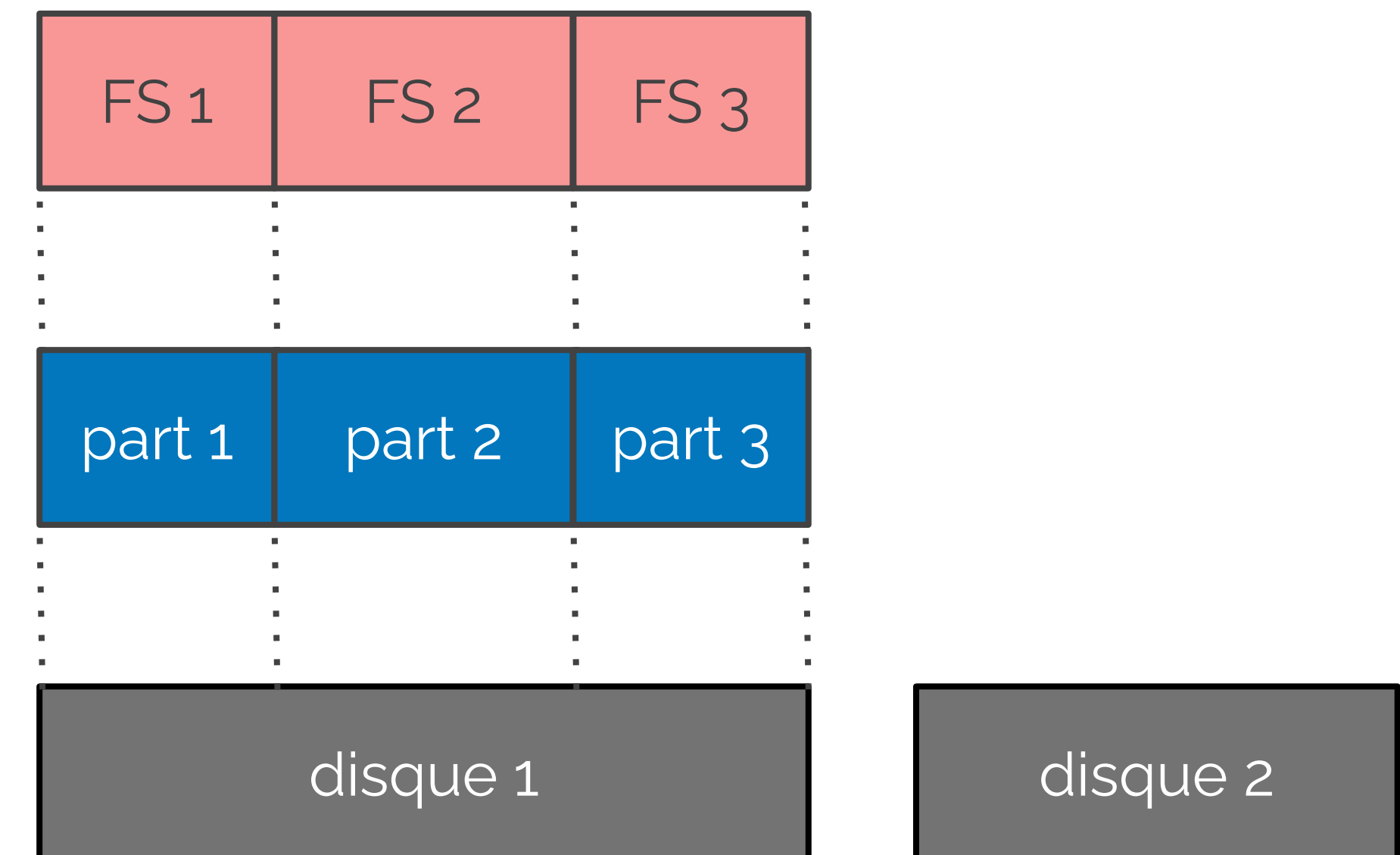


# Approche classique



Disques -  
Partitions - FS

- 1 disque découpé en 1 ou des partitions
- 1 partition associée à 1 FS
- Taille partition  $\leq$  taille disque
- Taille FS = Taille partition







## Métadonnées d'un FS

---

Des données sur  
les données

Métadonnées globales :

- Paramétrage du système de fichiers
- Blocs disponibles/occupés
- Blocs corrompus
- Emplacement de la racine
- Emplacement du boot
- etc.

Métadonnées par fichier/dossier :

- Nom
- Taille
- Identifiants de propriétaire (Utilisateur/Groupe)
- Droits d'accès
- Dates (création, accès, modification)
- Compteur de liens
- etc.



# Liens physiques et symboliques

---

Hard & Sym links

Objectif : fournir l'accès au même fichier depuis différents répertoires

L'approche [hard link](#) :

- S'appuyer sur le système de fichier - un seul fichier réel
- Compteur de lien => quand supprimer réellement le fichier

L'approche [lien symbolique](#) :

- Fichier particulier - renvoi sur un autre chemin
- Indépendant du fichier réel => lien cassé



# FAT

Ancien mais  
compatible

La **FAT** (*File Allocation Table*) a été créée par Microsoft en 1977 pour MS-DOS.

- Noms de fichiers 8.3 ASCII (étendu à 255 en unicode par vFAT)
- Dernière version FAT32 (Windows 95)

Limitations :

- Fichiers de 4 Gio max
- Partitions de 16 Tio max
- Pas de support des droits d'accès

Avantage : supporté par tous les OS

Fréquemment utilisé pour les stockages amovibles (clés USB par ex.)



# NTFS



The Windows FS

Le **NTFS** (*New Technology File System*) est le successeur de FAT sur Windows NT et 2000.

- Meilleures capacités (taille et nombre de fichiers, taille de volume)
- Support des ACL (Access Control List) avancées
- Journalisation
- Compression
- Chiffrement



# Extended File System

---

The Linux FS

Principal système de fichier GNU/Linux

Actuellement version 4 (ext4)

- Limite la fragmentation ([extent](#))
- Défragmentation en ligne
- Journalisation
- Compression
- Chiffrement



# L'approche GNU/Linux



# Spécificité de l'approche Linux



Tout est fichier

Racine (et arborescence) unique : /

- Indépendant des périphériques de stockage
- Notion de montage

Fichier = ressource

- Pseudo fichiers qui représentent des ressources du système



## L'arborescence classique

---

chaque chose à sa place

**/bin** : exécutables utilisateurs essentiels

**/boot** : fichiers de démarrage et noyaux (linux)

**/dev** : pseudo fichiers correspondant aux périphériques (device)

**/etc** : fichiers de configuration du système

**/home** : répertoires personnels des utilisateurs

**/lib** : bibliothèques partagées

**/mnt /media /cdrom** : montage des FS temporaires

**/opt** : installation des programmes non standard (hors dépôt)





## L'arborescence classique (suite)



chaque chose à sa  
place

**/proc** : pseudo fichiers correspondant aux processus

**/root** : répertoire personnel de root

**/sbin** : exécutable pour l'administration

**/sys** : configuration actuelle du système

**/tmp** : données temporaires

**/usr** : programmes et ressources standards

**/var** : données variables



# Outils de partitionnement

---

L'embarras du  
choix

CLI :

- fdisk
- cfdisk
- parted...

GUI :

- gparted
- gnome-disks
- partitionmanager (KDE)...



# Nomenclature classique



Storage devices

Fichiers dans **/dev** :

- Disques IDE : **/dev/hdX** avec X = a, b, c...
  - Partitions : **/dev/hdXP** avec P = 1, 2, 3
- Disques SATA : **/dev/sdX** avec X = a, b, c...
  - Partitions : **/dev/sdXP** avec P = 1, 2, 3



## Nomenclature classique (suite)



Storage devices

- Disques nvme : **/dev/nvmeYnZ** avec  $Y = 0, 1, 2...$  et  $Z = 1, 2, 3...$ 
  - Partitions : **/dev/nvmeYnZpP** avec  $P = 1, 2, 3$

Numérotation dans l'ordre de la détection.

Un disque peut changer de nom !



# Outils de formatage

---

Approche unifiée

Outil spécifique par type de système de fichier

Une commande : **mkfs**

ex pour formater en ext4 : **mkfs.ext4**

À la création :

- Association d'un UUID
- Choix d'une étiquette (label)
- Choix des paramètres



# Outils de formatage complémentaires

---

Approche unifiée

Outils complémentaires :

- **fsck** : vérifier
- **resize2fs** : redimensionner
- **e2label** : changer l'étiquette
- **badblock** : recherche blocs défectueux
- **tune2fs** : paramétrage



## Étiquettes et identifiants uniques

---

Identification  
stable

Chaque système de fichier (partition) a un **UUID unique et stable**.

- **lsblk** : affiche les périphériques de stockage (et UUID)
- **blkid** : similaire mais plus bas niveau
- Liens symboliques dans **/dev/disk/by-uuid**

```
wilder@host:~$ ls -lh /dev/disk/by-uuid/
```

```
total 0
```

```
lrwxrwxrwx 1 root root 15 avril 11 13:59 2e80388d-0014-4fe0-9328-d06296d423f7 -> ../../nvme0n1p1  
lrwxrwxrwx 1 root root 15 avril 11 13:59 9e35d3c3-e1d1-484c-ad58-ba0d8e18f7c0 -> ../../nvme0n1p2  
lrwxrwxrwx 1 root root 15 avril 11 13:59 df91b5ef-b165-424d-843f-b2e212fa27f5 -> ../../nvme0n1p3
```



## Monter / Démonter un FS



Rendre disponible  
via l'arborescence

**mount** : monte un système de fichier dans l'arborescence en précisant :

- Le périphérique à monter (ex : /dev/sda1)
- le chemin de montage (ex : /home)

**umount** : opération inverse

[/etc/fstab](#) : montages automatiques au démarrage

```
# /etc/fstab: static file system information.
# <file system>    <mount point>    <type> <options>          <dump> <pass>
# / was on /dev/nvme0n1p4 during installation
UUID=9e35d3c3-e1d1-484c-ad58-ba0d8e18f7c0 /          ext4 errors=remount-ro 0    1
# /boot was on /dev/nvme0n1p3 during installation
UUID=df91b5ef-b165-424d-843f-b2e212fa27f5 /boot      ext4 defaults          0    2
# swap was on /dev/nvme0n1p2 during installation
UUID=2e80388d-0014-4fe0-9328-d06296d423f7 none      swap sw                0    0
```





## Quelques commandes utiles

---

Back to basics

**pwd** : affiche le répertoire courant

**ls** : affiche le contenu d'un répertoire (ou du répertoire courant)

**cat** : affiche un fichier

**more/less** : affichage paginé

**head/tail** : affichage début/fin d'un fichier

**cp** : copie un fichier/dossier

**mv** : déplace/renomme un fichier/dossier

**rm** : supprime un fichier/dossier



## Quelques commandes utiles (suite)

---

Back to basics

**mkdir** : crée un dossier

**rmdir** : supprime un dossier (vide)

**touch** : mise à jour date d'accès (crée) un fichier

**which/whereis/locate** : trouve un fichier

**diff** : affiche les différences

**wc** : compte les lignes/mots/caractères

**grep** : filtre

**find** : recherche des fichiers

**sed** : édition fichier texte en ligne

**awk** : traitement de fichier par champs



# L'approche Windows



## Spécificité de l'approche Windows

---

Des racines

Chaque partition/système de fichier correspond à une racine :

- Identifiée par une lettre **c:/ d:/ ...**
- Comporte sa propre arborescence de fichiers

C:\ habituellement le disque système



## L'arborescence classique



Chaque chose à sa place

**C:\\$Recycle.Bin** : corbeille du système de fichier c:

**C:\PerfLogs** : journaux de performance

**C:\Program Files** : installation des programmes tiers

**C:\Program Files (x86)** : installation des programmes tiers (32 bits)

**C:\ProgramData** : données globales des programmes

**C:\Recovery** : récupération après incident

**C:\System Volume Information** : métadonnées NTFS et points de restauration

**C:\Users** : répertoire personnel des utilisateurs

**C:\Windows** : installation du système



## Les outils



Need more space ?

CLI :

- **Diskpart** : gestion des disques et partitions
- **Format** : création d'un système de fichier sur une partition

GUI :

- diskmgmt.msc : Gestion de disques

Disques sont automatiquement disponible à chaque démarrage



## Quelques commandes utiles

---

Un peu de  
PowerShell

**Get-Location** (alias pwd) : affiche le répertoire courant

**Get-ChildItem** (alias ls) : affiche (et cherche) le contenu d'un répertoire

**Get-Content** (alias cat) : affiche un fichier

**more** : affichage paginé

**Copy-Item** (alias cp) : copie un fichier/dossier

**Move-Item** (alias mv) : déplace/renomme un fichier/dossier



## Quelques commandes PowerShell utiles



Need more space ?

**Remove-Item** (alias rm et rmdir) : supprime un fichier/dossier

**mkdir** : crée un dossier

**Compare-Object** (alias diff) : affiche les différences

**Where-Object** : filtre





## En résumé

---

A retenir

- Notion de système de fichier
- Mise en place sous GNU/Linux
- Mise en place sous Windows
- Quelques commandes à expérimenter



## **MERCI**

---

pour votre participation.

C'est à vous maintenant.

Des questions ?

Des remarques ?





**Atelier :**

**Jouer avec les partitions sous Linux**