Linux -- Bases des systèmes d'exploitation

Halim Djerroud

LISV
Laboratoire d'ingénierie
des systèmes de Versailles

UVSQ
université PARIS-SACLAY

révision : 0.1

Plan de la séance

1. Rôles d'un système d'exploitation ; comparaison Windows / Linux.
2. Distributions Linux et usages en R&T.
3. Interface CLI vs GUI, introduction au shell Bash.
4. Arborescence Linux : `/`, `/home`, `/etc`, `/var`, `/usr`.
5. Commandes de base : `pwd`, `ls -al`, `cd`, `cat`, `less`.
6. Utilisation de l'aide : `man`, `apropos`, `whatis`.

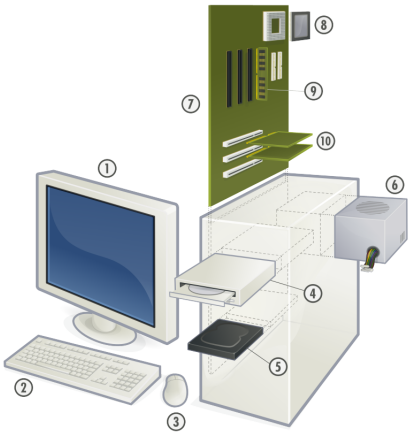
Rôles d'un système d'exploitation

Rôles d'un système d'exploitation ; comparaison Windows / Linux

Objectifs du chapitre :

1. Comprendre les rôles principaux d'un système d'exploitation.
2. Identifier les différences entre Windows et Linux.
3. Situer les usages de Linux dans le contexte R&T.

Architecture d'un ordinateur (intérieur schéma)



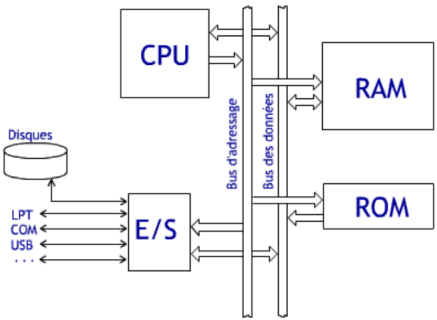


Figure – Source : channelconscience.unblog.fr & courstechinfo.be

Rôle du système d'exploitation

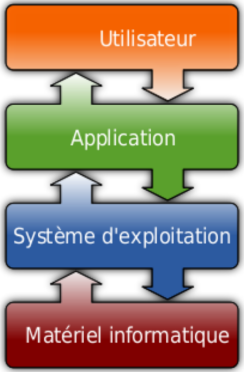


Figure – Source : wikipedia

Procure aux processus :

- ressources de calcul (processeur)
- espace mémoire suffisant (RAM)
- espace sur le disque dur
- utilisation des ressources (files d'attente, priorités)
- pas d'accès concurrents
- permet les communications
- droits d'accès processus/utilisateurs
- analyse son propre fonctionnement
- peut "tuer" un processus

Qu'est-ce qu'un système d'exploitation ?

Premier programme qui est exécuté au démarrage de la machine, après l'amorçage

- Logiciel intermédiaire entre le matériel (*hardware*) et les applications. Ses fonctions principales :
  1. Gestion du matériel (CPU, mémoire, périphériques).
  2. Gestion des fichiers et répertoires.
  3. Gestion des utilisateurs et de la sécurité.
  4. Gestion des applications et processus.

Constitué de :

- **Un noyau (kernel) :** gestion de la mémoire, des processus, des fichiers, des entrées-sorties principales, et des fonctionnalités de communication
- **Un interpréteur de commande (shell = "coquille" par opposition au noyau) :** communication avec le système d'exploitation grâces au commandes en console
- **Le système de fichiers (file system) :** une hiérarchie de dossiers contenant des fichiers
- **Interface(s) graphique(s) :** environnement de bureau"
- **Ensemble de logiciels :** jeux, outils, etc...

Comparaison Windows / GNU/Linux

Windows

- Licence propriétaire (Microsoft).
- Interface graphique (GUI) dominante.
- Plus ciblé par les malwares.
- Usages : bureautique, jeux, logiciels propriétaires.
- Personnalisation limitée.

GNU/Linux

- Logiciel libre et open source.
- Ligne de commande (CLI) centrale, GUI disponible.
- Conçu multi-utilisateurs dès le départ.
- Usages : serveurs, réseaux, embarqué, développement.
- Très personnalisable (distributions, environnements).

De nombreux systèmes d'exploitation

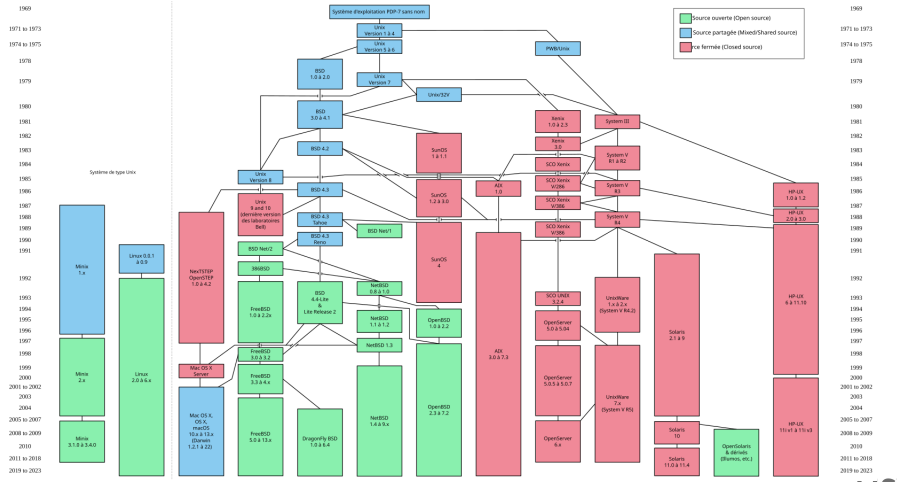
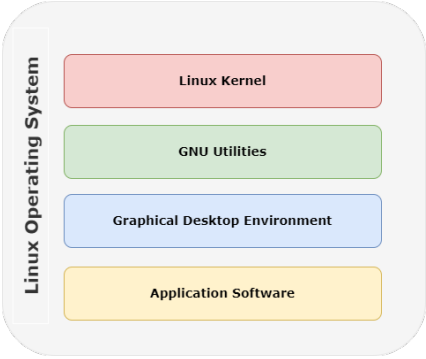


Figure – Source wikipedia

GNU/Linux

Un système d'exploitation modulaire composé de plusieurs éléments :

- Le noyau **Linux**
- Logiciels **GNU**
- Le système graphique **Xorg**
- Gestionnaires de bureau : **GNOME**, **KDE**, **XFCE**, **LXDE**
- Diverses bibliothèques logicielles



Distributions

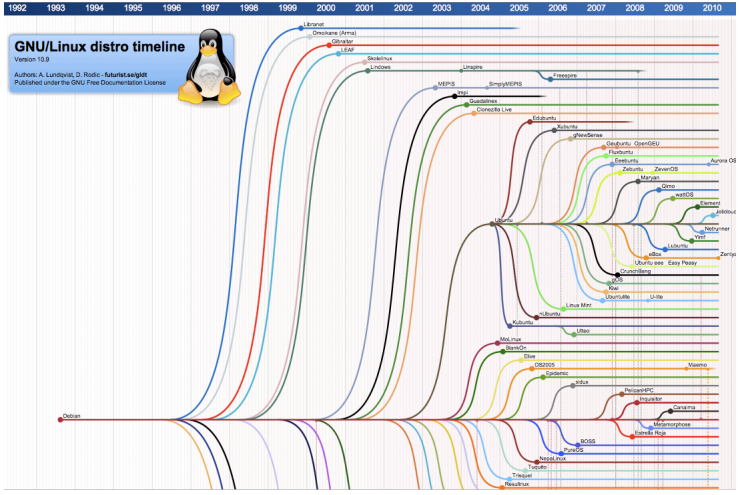


Figure – Source : wikipedia

Debian 13 Trixie

Figure – Source : wikipedia

Distributions GNU/Linux utilisées à l'IUT

- **Debian** Stable, largement utilisée sur serveurs et postes pédagogiques.
- **CentOS** Dérivée de Red Hat, utilisée pour l'administration système et réseau.
- **Kali Linux** Distribution spécialisée en sécurité et tests d'intrusion.

Pourquoi plusieurs distributions ?

- Découvrir différents environnements.
- Préparer aux contextes professionnels variés.
- Adapter l'OS aux besoins spécifiques (serveur, sécurité, enseignement).

Le terminal et le Shell Bash

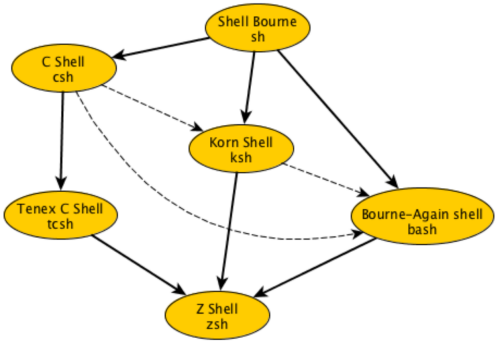
Terminal et Shell Bash

Objectifs du chapitre :

1. Comprendre le rôle du terminal et du Shell.
2. Structure d'une commande

Terminal et Shell

- **Terminal (console)** : fenêtre où l'on saisit des commandes.
- **Shell** : programme qui interprète ces commandes.
- Exemples de Shell :
  - Bash (le plus courant)
  - Bourne Shell (`sh`)
  - Zsh, ksh, tcsh, etc.

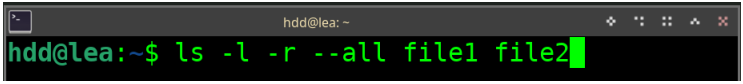


Terminal Bash

Prompt :

- signe d'invite (souvent $) pour un utilisateur normal
- signe d'invite (souvent #), pour un super utilisateur (administrateur), appelé couramment *root*

Commande et options



- Une commande se compose de plusieurs éléments :
  - **nom de la commande** : mot-clé exécuté par le shell. Exemple : `ls`
  - **options** : modifient le comportement de la commande.
    Exemple : `ls -r`, `ls --reverse`
  - **arguments** : précisent la cible de la commande Exemple : `ls Documents`
- **Types d'options :**
  - Options courtes : précédées d'un seul tiret -. Exemple : `ls -l -a` ≡ `ls -la`
  - Options longues : précédées de deux tirets --
    Exemple : `ls --all`, `ls --reverse`
  - Syntaxe BSD : pas de tirets (héritage UNIX).
    Exemple : `tar xzvf fichier.tar.gz`

Astuces d'utilisation

- La casse est importante : `-r` ≠ `-R`.
- Flèches ↑ ↓ : rappeler une commande tapée.
- Flèches ←→ : déplacer le curseur.
- `Tab` : auto-complétion.
- Une commande correcte peut ne rien afficher si tout s'est bien passé
- La plupart des commandes acceptent plusieurs options combinées

Le manuel et l'aide

- **man <commande>** : manuel complet d'une commande Exemple : `man ls`
- **man man** : manuel du manuel (apprendre à lire man)
- **whatis <commande>** : description courte d'une commande Exemple : `whatis ls`
- **apropos <mot>** : recherche par mot-clé dans les pages de manuel Exemple : `apropos directory`
- **info <commande>** : documentation plus détaillée (ex. `info coreutils`)
- **<commande> -h** ou **<commande> - -help** : aide rapide intégrée à la commande Exemple : `ls - -help`
- Rappel : **RTFM** = *Read The F***** Manual*

L'arborescence Linux

L'arborescence Linux

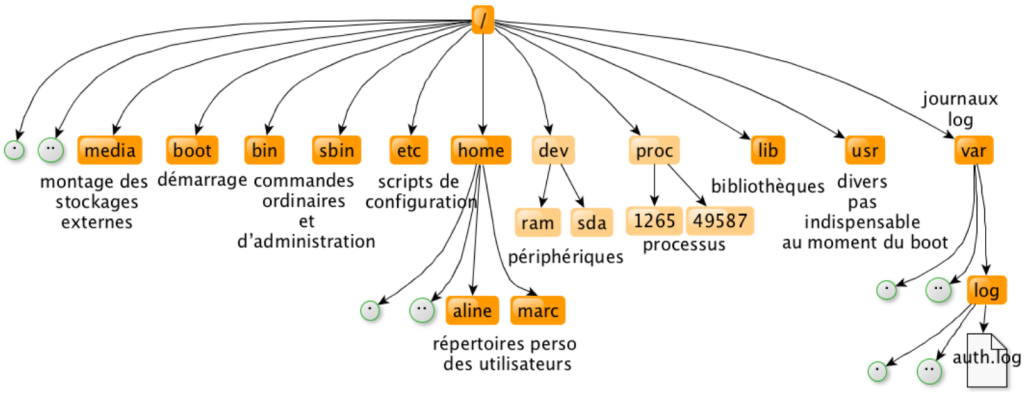
Objectifs du chapitre :

1. Comprendre l'organisation hiérarchique des fichiers.
2. Identifier les répertoires systèmes essentiels.
3. Savoir naviguer et se repérer dans l'arborescence.

Organisation hiérarchique

- Sous Linux, tout est organisé en **arborescence**.
- La racine est le répertoire `/`.
- Pas de lettre de disque (`C:`, `D:`) comme sous Windows.
- Tous les périphériques (disques, clés USB, CD-ROM, etc.) sont intégrés dans l'arborescence.
- Exemple : `/media/usb` → clé USB montée dans le système.

Organisation hiérarchique



Répertoires principaux

- `/` : racine de l'arborescence.
- `/home` : répertoires personnels des utilisateurs.
- `/root` : répertoire personnel de l'administrateur.
- `/bin` : commandes de base essentielles (ls, cp, mv. . .).
- `/sbin` : commandes systèmes pour l'administration.
- `/etc` : fichiers de configuration.
- `/var` : données variables (logs, mails, spool).
- `/usr` : applications et librairies utilisateur.
- `/tmp` : fichiers temporaires.
- `/dev` : périphériques (disques, imprimantes, etc.).
- `/proc` : informations sur les processus et le noyau (virtuel).

Raccourcis utiles

- `.` : répertoire courant
- `..` : répertoire parent
- ~ : répertoire personnel de l'utilisateur
- Exemple : `cd ..` → remonte d'un niveau `cd ~` → retourne dans `/home/utilisateur`

Commandes de base

Commandes de base sous Linux

Objectifs du chapitre :

1. Savoir naviguer dans l'arborescence.
2. Manipuler fichiers et répertoires.
3. Lire et consulter des fichiers.

Navigation dans l'arborescence

Commandes :

- `pwd` : afficher le répertoire courant
- `ls` : lister les fichiers
- `cd <dir>` : changer de répertoire
- `tree` : afficher l'arborescence

Options et raccourcis :

- `ls -a` : fichiers cachés
- `ls -l` : format détaillé
- `ls -al` : combinaison
- `cd ..` : répertoire parent
- `cd ~` : répertoire personnel
- `cd -` : précédent répertoire

Lecture et consultation de fichiers

- `cat <fichier>` : affiche le contenu d'un fichier
- `less <fichier>` : affiche le contenu page par page (q = quitter)
- `head <fichier>` : affiche les 10 premières lignes
- `tail <fichier>` : affiche les 10 dernières lignes
- `file <fichier>` : indique le type de fichier

Manipulation de fichiers et dossiers

Commandes :

- `touch <fichier>` : créer un fichier vide
- `cp origine destination` : copier
- `mv origine destination` : déplacer/renommer
- `rm <fichier>` : supprimer
- `mkdir <dossier>` : créer un dossier
- `rmdir <dossier>` : supprimer un dossier vide

Options utiles :

- `cp -r` : copie récursive (répertoires)
- `cp -v` : affiche les fichiers copiés
- `rm -r` : suppression récursive
- `rm -i` : demande confirmation
- `mkdir -p` : crée l'arborescence complète

Commandes pratiques supplémentaires

- `history` : affiche l'historique des commandes
- `echo <texte>` : affiche un texte
- `date` : affiche la date et l'heure
- `whoami` : affiche l'utilisateur courant
- `clear` : nettoie l'écran du terminal

La commande grep

La commande grep

Objectifs du chapitre :

1. Comprendre le rôle de `grep`.
2. Savoir rechercher du texte dans des fichiers.
3. Utiliser les options courantes et les regex simples.

Rechercher un motif avec `grep`

- `grep` signifie : **Global Regular Expression Print**.
- Permet de rechercher un **motif** (texte ou expression régulière).
- Fonctionne sur un ou plusieurs fichiers, ou sur la sortie d'une commande.
- Syntaxe générale :

```
grep [options] motif fichier
```

Exemples simples avec `grep`

- `grep "bonjour" texte.txt` → recherche "bonjour" dans le fichier.
- `grep "main" *.c` → recherche dans tous les fichiers C du dossier.

Option courantes :

- `-i` : ignorer la casse.
- `-n` : afficher le numéro de ligne.
- `-r` : recherche récursive.
- `-v` : inverser la recherche (exclure le motif).
- `-c` : afficher uniquement le nombre de lignes correspondantes.
- `-color=auto` : surligner le motif trouvé.

Les éditeurs de texte

Les éditeurs de texte sous Linux

Objectifs du chapitre :

1. Découvrir les principaux éditeurs de texte en ligne de commande.
2. Apprendre les bases de `nano`, `vi/vim`, et `emacs`.
3. Savoir lequel utiliser selon le contexte.

nano

- Éditeur simple et intuitif.
- Commandes affichées en bas de l'écran.
- Idéal pour débuter.
- Raccourcis utiles :
  - `Ctrl+O` : enregistrer le fichier.
  - `Ctrl+X` : quitter.
  - `Ctrl+W` : rechercher.

vi/vim

- Puissant, installé par défaut sur presque toutes les distributions.
- Deux modes principaux :
  - **Mode commande** : navigation, suppression, copier/coller.
  - **Mode insertion** : édition du texte (`i` pour insérer).
- Commandes essentielles :
  - `:w` → enregistrer.
  - `:q` → quitter.
  - `:wq` → enregistrer et quitter.
  - `dd` → supprimer une ligne.
  - `/mot` → rechercher un mot.

emacs

- Éditeur avancé, extrêmement personnalisable.
- Dispose d'un écosystème complet (éditeur, shell, client mail, etc.).
- Raccourcis fréquents :
  - `Ctrl+X Ctrl+S` → enregistrer.
  - `Ctrl+X Ctrl+C` → quitter.
  - `Ctrl+X Ctrl+F` → ouvrir un fichier.
  - `Ctrl+K` → couper une ligne.
- Plus complexe à maîtriser que `nano`.

Comparison des editeur de texte

Nano :

- Simple d'utilisation
- Facile à prendre en main
- Idéal pour débutants

Vi/Vim :

- Puissant
- Toujours disponible
- Nécessite apprentissage

Emacs :

- Très puissant
- Écosystème complet
- Personnalisable à l'extrême
- Idéal pour le développement
- Plutôt pour utilisateurs avancés

Quelques commandes à retenir

- ls
- cd
- pwd
- touch
- cp
- mv
- rm
- mkdir
- rmdir
- tree
- free

- find
- locate
- which
- man
- echo
- cat
- more
- less
- head
- tail
- uname

- wc
- cut
- sort
- uniq
- grep
- diff
- cmp
- chmod
- chown
- df
- uptime

- du
- tar
- gzip
- gunzip
- zip
- unzip
- tee
- history
- alias
- clear