

M2Dev FS - GNU/Linux avancé (35h) {.unnumbered .unlisted}

I'd just like to interject for a moment. What you're referring to as **Linux**, is in fact, GNU/**Linux**, or as I've recently taken to calling it, GNU plus **Linux**. **Linux** is not an operating system unto itself, but rather another free component of a fully functioning GNU system made useful by the GNU corelibs, shell utilities and vital system components comprising a full OS as defined by POSIX.

MyDigitalSchool - Rennes

Novembre 2025

Durée : 28h

Paul Schuhmacher

Version : 1

[Me contacter par email](#)

Objectifs

- Vous formez en tant qu'utilisateur·ice **avancé·e** et **autonome** de (GNU/)Linux;
- Être à l'aise sur un serveur en CLI;
- Connaître et savoir utiliser les outils les plus puissants des systèmes Unix;
- Savoir **faire/réparer les configurations fondamentales**, notamment pour le web;
- Orienté développeur·se et **pratique**;
- Savoir **automatiser ses tâches, développer ses outils** pour répondre à ses besoins.

"Power user", pas kernel developer ou ses modules

Plan

Module 01 : Introduction

- Une brève histoire d'UNIX et de GNU/Linux;
- Caractéristiques générales de l'OS;
- Les distributions GNU/Linux aujourd'hui.

Module 02 : Installation, sous GNU/Linux (Debian) et premiers pas

Faire ses premiers pas sur un environnement GNU/Linux *vanilla*

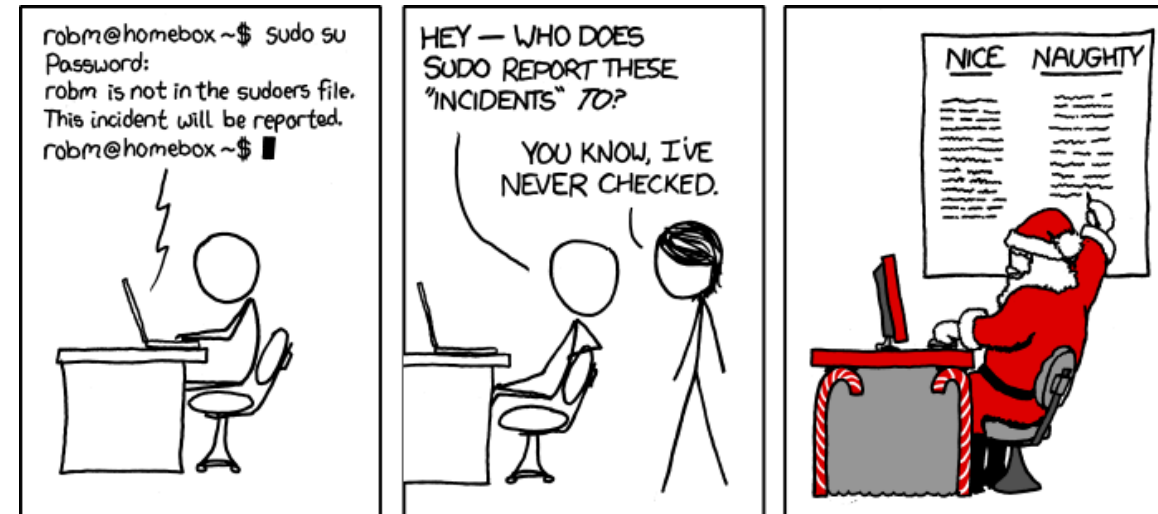
- Mise en place de notre environnement de travail avec une VM;
- Architecture générale d'une distribution GNU/Linux (noyau, interface, périphériques, **système de fichiers**, serveur graphique);
- Installer une distribution sans GUI (**Debian**);
- Gestionnaire de paquets (**aptitude**);
- **Premiers pas** : configuration, exploration de quelques commandes;
- **Exercices**.

Module 03 : Fichiers, utilisateurs, groupes et permissions

{.marp-bg-img}

- Les **fichiers**;
- Monter un système de fichiers;
- Gestion des **utilisateurs** et des **groupes**;
- Administrateur et l'usage de `sudo` ;
- Gestion des **permissions** (sécurité);
- Les processus et permissions;
- **Exercices**.

[xkcd : incident](#)



Module 04 : Commandes, programmes fondamentaux et services

- Les différents types de commandes dans le shell;
- Éditeurs en ligne de commande (vi, vim, emacs, nano), les bases;
- Manipulation des fichiers (man, cp, mv, cd, **grep**, **cut**, find, scp, **tar**, find);
- Savoir utiliser la documentation des programmes (**man**);
- Introduction à **awk** et **sed** ;
- Le **gestionnaire de services** **systemd** :
 - Gérer les services,
 - Créer ses propres services;
- **Exercices.**

Module 05 : Programmation shell

- Les shells;
- **Programmer en sh et bash** (variables, structures de contrôle, pattern matching, bonnes pratiques, etc.);
- Ecrire ses fonctions shell;
- Le pipe UNIX (|) et l'usage des **redirections de flux**;
- Configurer le shell (`.bashrc`), **alias** et variables globales (`export`);
- La **culture** des scripts et l'**automatisation des tâches** et créer des outils sur mesure;
- Utiliser `crontab` pour créer des tâches **programmées**;
- **Exercices.**

Module 06 : Monitorer sa machine

- Que faut-il monitorer et pourquoi monitorer ?
- `top` , `htop` et *tips*;
- **Monitoring** CPU, mémoire, réseau;
- Les **logs** systèmes (protocole `syslog`).
- Règles de **pare-feu** et UFW;

Module 07 : Sécuriser la machine 1/2

- Gestion des utilisateurs;
- Sécuriser et configurer l'accès `ssh` avec une clef (sans mot de passe);
- Contrôler l'utilisation du mode `sudo` ;
- Services sensibles exposés d'un serveur;
- Utiliser un scanner de vulnérabilité;
- Tester son serveur avec `nmap` ;
- Inventaire des attaques courantes (MITM, DDoS, Trojans, Worms, Spam/Flood, etc.).

Module 07 : Sécuriser la machine 2/2

- Configurer le pare-feu avec UFW;
- Bannir des IPs avec `fail2ban` (TP);
- Mettre en place une jail SSH et web (TP);
- Scan des vulnérabilités avec `Lynis`;
- Installer et configurer `CrowdSec` (TP);
- Gestion de la montée en charge (*availability*) : tests de charge et outils (TP);
- Sauvegarder un serveur Linux (**gestion des backups**).

Module 08 : Réseau, configuration serveurs web et reverse proxy

- Configuration d'un réseau de machines, configuration réseau (IP, DNS);

Réseau si on a le temps. C'est intéressant mais ce ne sera pas la priorité

- **Définition d'un reverse proxy;**
- Tour d'horizon des serveurs web du marché (Apache, Nginx, Caddy, Traefik, IIS, etc...);
- Pratique.

Module 08 : Mise en place d'un serveur proxy, certificat TLS et d'une jail

- Comprendre l'intérêt du *reverse proxy* dans la sécurité d'un système;
- Installer un serveur *reverse proxy* (TP);
- Installer une jail `fail2ban` sur le serveur web (TP);
- Mettre en place un monitoring du serveur web (TP);
- Déployer un certificat `Let's Encrypt` avec `cerbot` et le lier au serveur web, rappel protocole TLS (TP);
- **Exercices.**

Évaluations

- **(DM) : TP constitué de deux exercices** à faire en dehors des heures de cours (on les commencera en cours en prenant une journée ou 1/2 journée en fonction de notre avancement);
- Chaque exercice contribue pour la moitié de la note.

De l'usage de l'IA dans ce cours

- Essayer **d'abord en vous servant de la documentation (`man`)**, de ressources. Sinon vous n'allez **rien apprendre**;
- **Faire des erreurs**, pour **fixer la connaissance**;
- **Commentez** vos scripts, gardez-les comme documentation (dépôt);
- **Valider votre compréhension** de tout ce que vous **produisez**.

De l'usage de l'IA dans ce cours

- Utiliser l'IA :
 - Explorer les **concepts**;
 - Comme assistant de documentation (programmes utiles, options utiles) si bloqué;
 - Une fois votre travail proposé, **correction, amélioration** de vos scripts;
 - **Restreindre le scope tan que cela est possible** (programmes connus, méthodes claires, programmes cibles);
 - **Proposition d'exercices** sur un concept, commande, etc.

De l'usage de l'IA dans la vraie vie

- Rédaction de scripts compris, **révisés** et **testés** ;
- Tester dans une VM ou container isolé ;
- L'IA ne remplace pas le **feedback du shell** : permissions, erreurs, comportement inattendu;
- **Ne pas exécuter un script généré par une IA sans le comprendre;**
- **Ne Jamais exécuter aveuglément une commande copiée depuis l'IA**, surtout avec des privilèges administrateur.

"Apprendre" GNU/Linux : les setups

- Avoir une machine pour expérimenter :
 - Idéal : **Machine dédiée** (ancienne machine par ex, d'occasion),
 - Machines Virtuelles en local,
 - VPS avec **administration complète** (vous êtes `root`).

"Apprendre" GNU/Linux : *hacker culture*

- **Choisir une distribution cible** (pour apprendre les fondamentaux), distribution **stable**, bien **documentée**. Par exemple : [Arch](#), [Fedora](#), [Debian](#), etc.
- **Bidouiller**, *hacker*, **casser des trucs**, dans un environnement dédié
- *User (user space) vs developer (kernel space)*;

"Apprendre" GNU/Linux, mais quoi ?

De manière générale les *concepts* derrière les OS et les systèmes UNIX. Sujets à aborder :

Les OS de manière générale, concepts fondamentaux:

- Virtualisation du CPU ;
- Fondamentaux : abstraction processus ;
- Virtualisation de la mémoire ;
- *Concurrency*, time sharing, multi-threading ;
- Persistance.

"Apprendre" GNU/Linux

Système GNU/Linux, un OS:

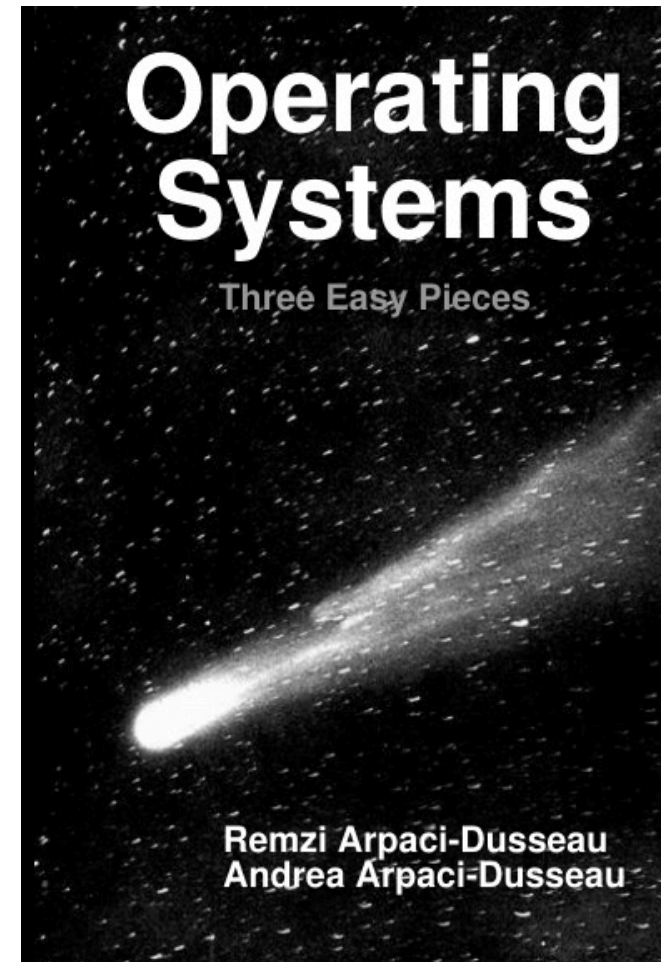
- Système de fichiers ;
- Processus et services ;
- Réseau ;
- Monitoring ;
- Sécurité ;
- Le **noyau** (kernel);
- Appel Systèmes (*syscalls*), Interface principale entre *userland* et kernel;
- Les **modules** du noyau ;
- **Programmer en C**, interface du *kernel* avec *userland* est en C.

Livres recommandés : *Comet OS Book* (fondamentaux)

{.marp-bg-img}

Operating Systems, three easy pieces, ou le *Comet OS Book*, de Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau (University of Wisconsin-Madison), publié par l'université du Wisconsin, 2008, continuellement mis à jour. [Accessible en ligne](#). Voir notamment **les chapitres 4 et 5** sur les processus. (LP++)

Étudier/Apprendre les fondamentaux. Concepts universels sur les OS, processus, threads, gestion mémoire, concurrency, etc.

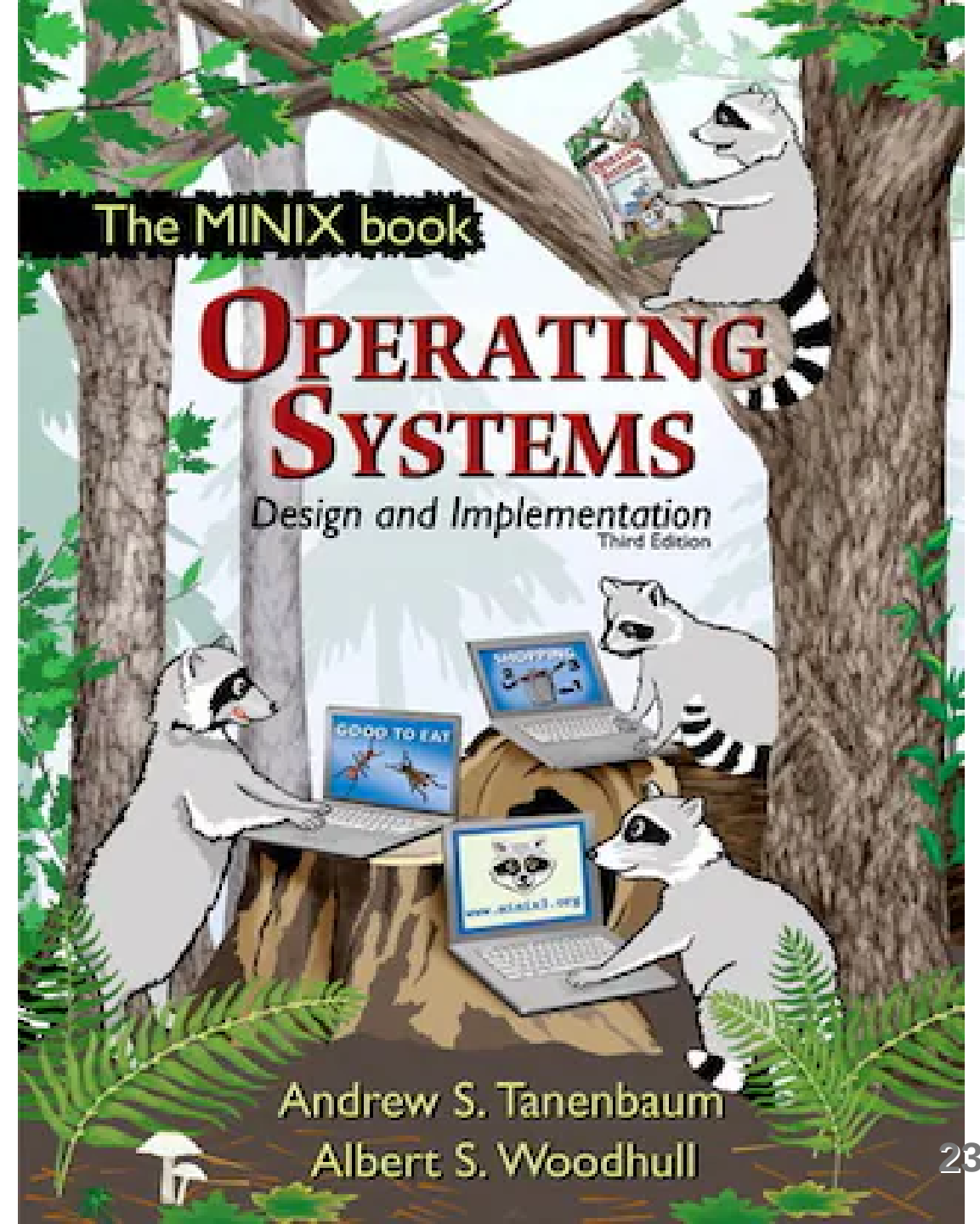


Livres recommandés : *Minix* (fondamentaux)

{.marp-bg-img}

[Operating Systems Design and Implementation, 3rd edition](#), d'Andrew S. Tanenbaum, publié chez Pearson, 2007. [Auteur de l'OS Minix](#), qui a inspiré Linus Torvalds pour créer le noyau Linux. **LE+++**

Fondamentaux. Faits pour les étudiant·es. Code source de [Minix](#) livré avec. Lien fort entre *pratique* et *théorie*.

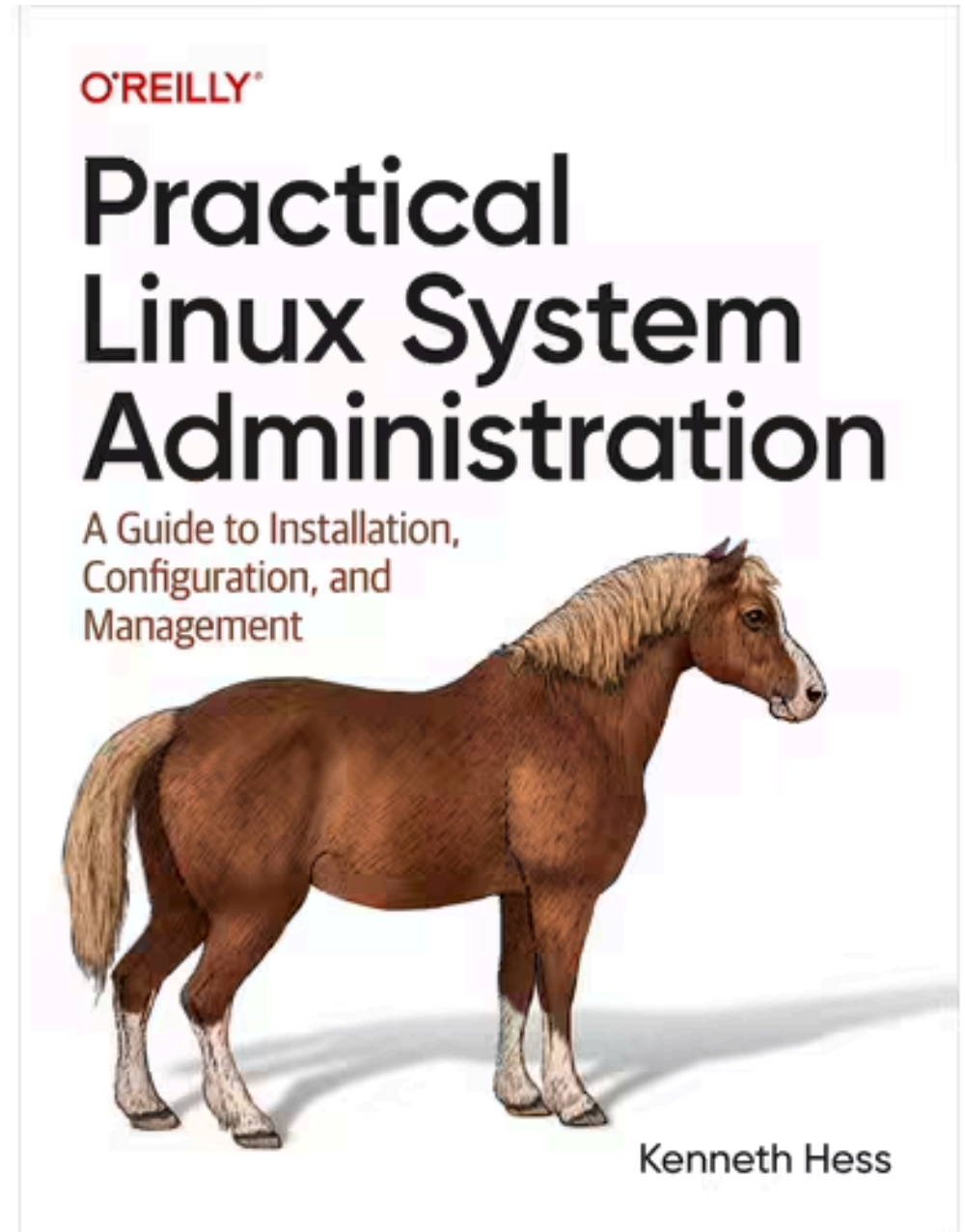


Livres recommandés : *GNU/Linux* (guide pratique)

{.marp-bg-img}

[Practical Linux System Administration](#), de Kenneth Hess,
O'Reilly, 2023.

Approche **pratique**. Guides sur les différentes procédures.
A disposer auprès de soi. Très complet, clair.



Livres recommandés : *GNU/Linux* (avancé)

{.marp-bg-img}

[The Linux Programming Interface](#), Michael Kerrisk, publié chez No Starch Press, 2010.

Orienté **développer pour l'OS. Avancé**. Explique l'API POSIX/Linux en C et montre comment **interagir avec le noyau**. Couvre également les notions fondamentales d'Unix

THE LINUX PROGRAMMING INTERFACE

A Linux and UNIX* System Programming Handbook

MICHAEL KERRISK



"Apprendre" GNU/Linux : Sur le web

- [The Art of Unix Programming](#), d'Eric S. Raymond, personne centrale du mouvement *open source* ("opposé" au free software)
- [Basics of the Unix Philosophy](#), extrait sur la philosophie et le design d'Unix

Vidéos (YouTube)

- [UNIX: Making Computers Easier To Use -- AT&T Archives film from 1982, Bell Laboratories](#)
- [Archives AT&T : Le système d'exploitation UNIX](#), reportage vidéo aux Bells Labs avec démonstration d'UNIX, de son usage au sein du laboratoire
- [Computerphile](#) (chaîne)
- [Deep Linux](#) (chaîne)
- [Core Dumped](#) (chaîne)

Liens utiles

[Dépôt associé au cours](#) (sources des démos, correction, biblio pour aller plus loin)