

Présentation détaillée de la formation : Découverte et administration de Linux

Maîtrisez les fondamentaux et outils avancés de
Linux – Gaultier Large



Qui suis-je?

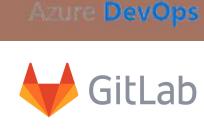
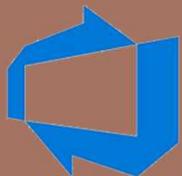
Gaultier Large

CEO d'Eduxim

Dev et architecte logiciel depuis 15 ans

Formateur depuis 12 ans

Coté tech



Précédents jobs



CGI



Evaluation

Objectif

Évaluer ta **compréhension**, ton **autonomie** et ta **capacité à explorer et administrer un système Linux**.

3 Compétences évaluées

- 1.Explorer le système Linux** → structure, commandes de base
- 2.Gérer utilisateurs, droits et processus** → sécurité, organisation
- 3.Automatiser et administrer un service** → scripting, services réseau

Méthode d'évaluation

Étape

 **Pendant les TPs**

Ce que tu fais

Tu réalises des missions concrètes

Outil

Terminal / scripts

 **Auto-évaluation**

Tu coches ton niveau et expliques tes choix

Eduxim / Notion

 **Validation**

Le formateur valide ton niveau avec tes preuves

Captures / oral

Niveaux de maîtrise

Niveau

 **Non acquis**

 **En cours**

 **Acquis**

 **Maîtrisé**

Description courte

Tu ne maîtrises pas encore la notion.

Tu comprends, mais fais encore des erreurs.

Tu sais faire et expliquer de manière fiable.

Tu vas plus loin que le cours (automatisation, optimisation...).

À retenir

-  Ce n'est pas la mémoire qui compte, mais ta compréhension.
-  Apprends à chercher, tester, expliquer.
-  L'auto-évaluation fait partie de ton apprentissage.
-  Laisse toujours une trace claire de ton travail.



Les preuves de ta montée en compétence

Tes **traces de progression** doivent être **visibles et centralisées**.

À conserver dans :

- Un **repo Git** personnel (fichiers .md, scripts, captures)
- ou un **Notion public** (page claire et structurée)

Tu dois renseigner le lien dans Eduxim.

Si aucune trace n'est déposée : **note = 0**, même si le travail a été fait.

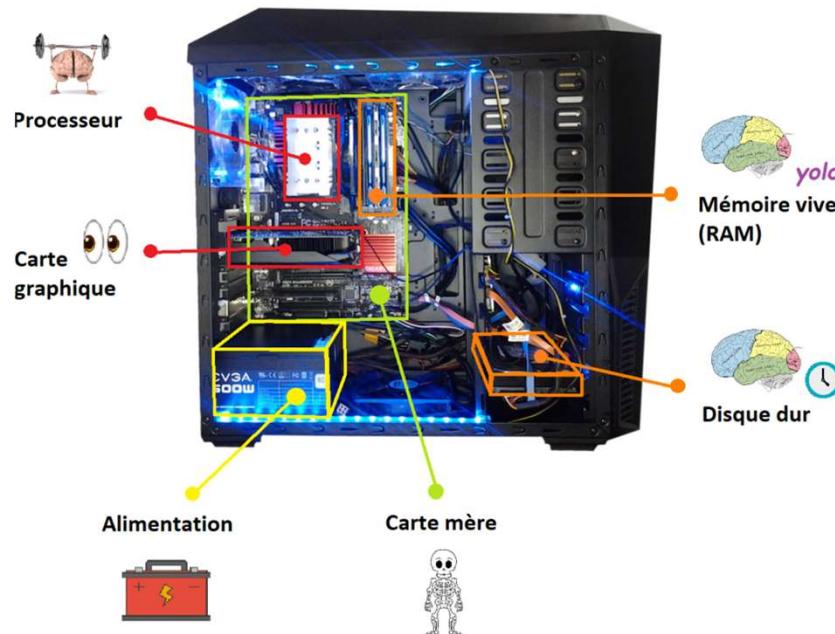
C'est ta responsabilité de documenter ton apprentissage, comme un vrai admin système.

Programme de la formation Linux

- Introduction à Linux et ses fondamentaux
- Architecture et composants de Linux
- Installation, virtualisation et prise en main
- Exploration du système de fichiers et gestion des droits
- Gestion des utilisateurs, groupes et processus
- Scripting, automatisation et édition de fichiers
- Réseau, services et supervision
- Clôture, évaluation et perspectives

Qu'est ce qu'un
système
d'exploitation

Comprendre un PC: les bases



Rôle du processeur

Le processeur exécute des calculs rapides et dirige les opérations principales du PC, garantissant le bon fonctionnement global.

Fonction de la mémoire RAM

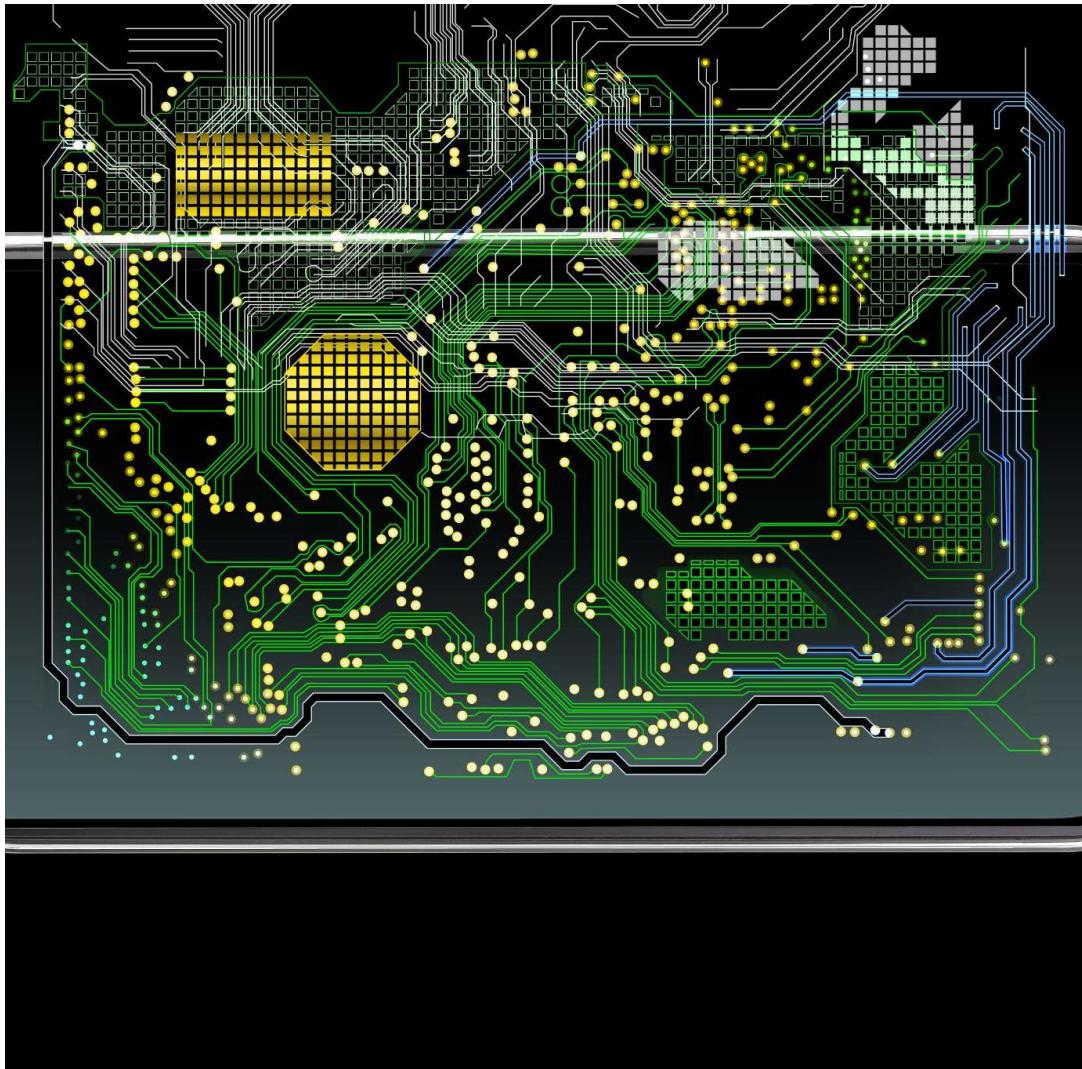
La RAM stocke temporairement les données en cours d'utilisation pour accélérer l'accès aux informations et améliorer la réactivité du PC.

Stockage sur disque dur

Le disque dur conserve durablement les fichiers, programmes et systèmes d'exploitation, permettant la sauvegarde des données importantes.

Carte graphique et affichage

La carte graphique gère l'affichage des images, vidéos et jeux, offrant une expérience visuelle optimale sur l'écran.



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

Gestion des ressources matérielles

Le système d'exploitation gère les composants matériels de l'ordinateur pour assurer un fonctionnement efficace et organisé.

Interface utilisateur-machine

Il fournit une interface conviviale permettant à l'utilisateur d'interagir facilement avec l'ordinateur et ses programmes.

Coordination des tâches et mémoire

Le système d'exploitation coordonne l'exécution des tâches, la gestion de la mémoire et des périphériques pour optimiser les performances.

Introduction à Linux et ses fondamentaux



Définition, historique et philosophie de Linux

Qu'est-ce que Linux ?

Linux est un système d'exploitation open source basé sur Unix. Il offre une alternative libre et collaborative aux systèmes propriétaires.

Historique rapide

Linux dérive de Unix, puis a intégré les composants GNU, donnant naissance à de nombreuses distributions variées.

Philosophie du libre

Linux est soutenu par la philosophie du logiciel libre, régie par la licence GPL qui garantit la liberté d'utilisation et de modification.

Comparaison avec d'autres systèmes et usages actuels



Différences avec Windows et macOS

Linux se distingue par son modèle open source, sa personnalisation et sa sécurité accrue par rapport à Windows et macOS.

Usage dans les serveurs

Linux alimente environ 90 % des serveurs web, montrant sa domination dans l'infrastructure internet mondiale.

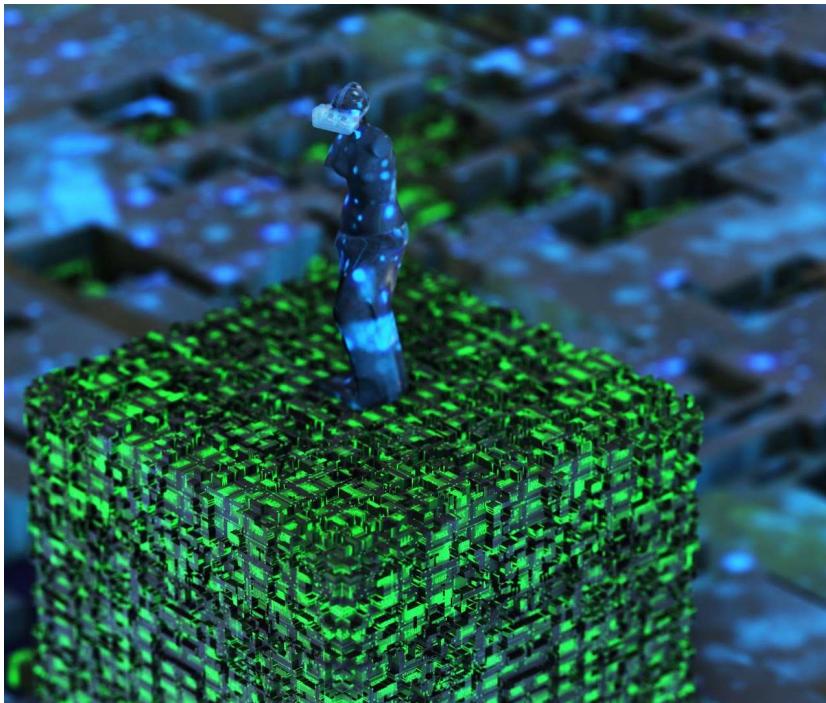
Présence dans Android

Le système d'exploitation Android est basé sur Linux, utilisé par des milliards de smartphones dans le monde.

Objets connectés et supercalculateurs

Linux est omniprésent dans les objets connectés et les supercalculateurs, soulignant sa flexibilité et puissance.

Avantages pour les professionnels et lien avec DevOps/Cloud



Stabilité et sécurité

Les professionnels privilégient des solutions stables et sécurisées pour garantir la fiabilité des systèmes en production.

Modularité flexible

La modularité permet une adaptation aisée des composants pour répondre aux besoins changeants des projets DevOps et Cloud.

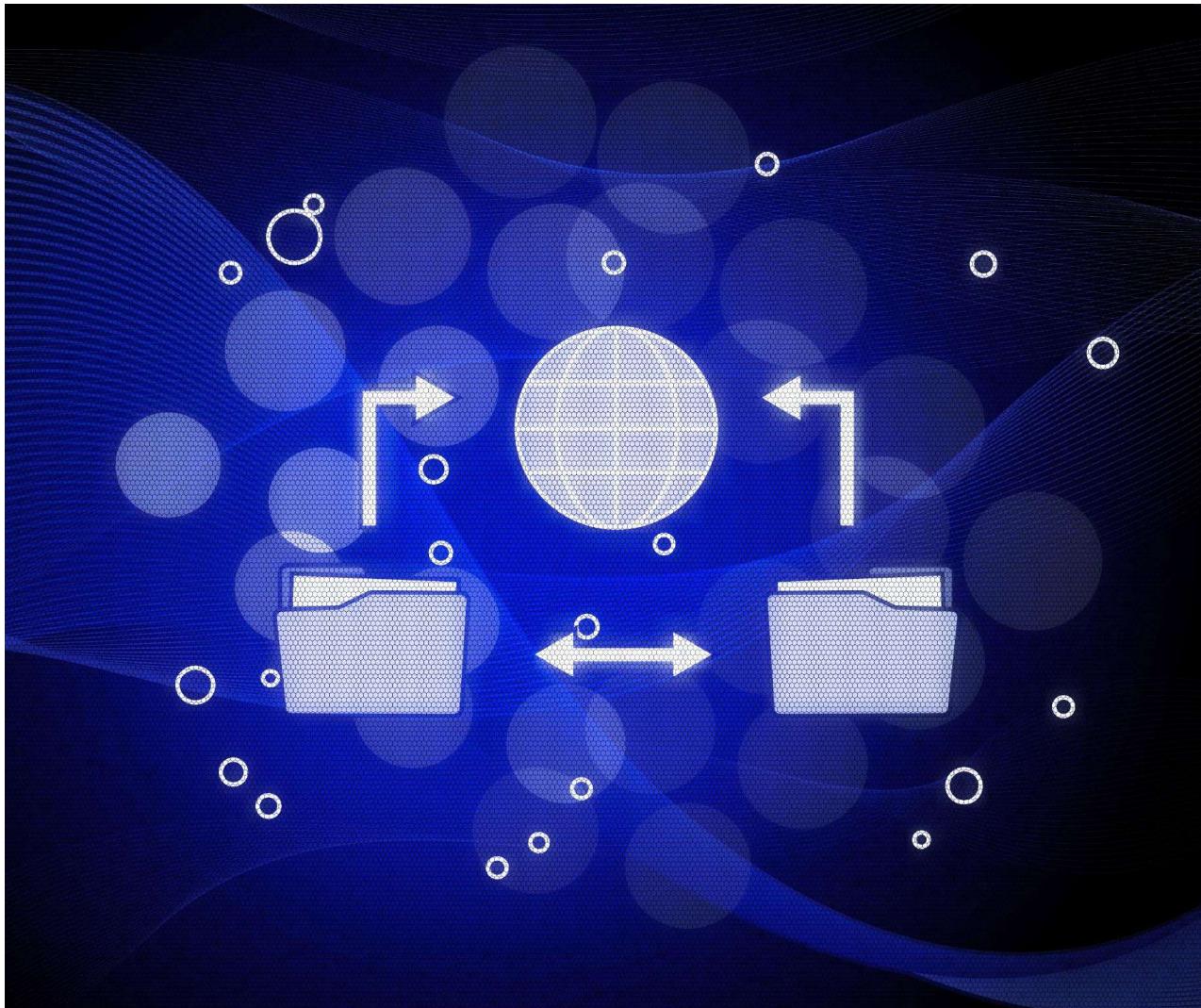
Communauté active

Une communauté dynamique favorise le partage des connaissances et l'évolution rapide des technologies utilisées.

Lien avec DevOps et Cloud

Les outils et pratiques intégrés facilitent la gestion, le déploiement et l'automatisation dans les environnements DevOps et Cloud.

Architecture et composants de Linux



Composants d'un OS Linux et principales distributions

Composants essentiels de Linux

Un OS Linux comprend le noyau, le shell, le système de fichiers, les services et l'interface graphique.

Diversité des distributions

Les distributions populaires incluent Ubuntu, Debian, Fedora, Arch et CentOS, chacune avec ses spécificités.

Structure commune des systèmes

Malgré la diversité, toutes les distributions Linux partagent une arborescence de fichiers commune.



Cette photo par Auteur inconnu est soumise à la licence [CC BY-NC-ND](#)

Le noyau, chef d'orchestre

Les **instruments**, ce sont les **périphériques** (disque, carte réseau, clavier...).

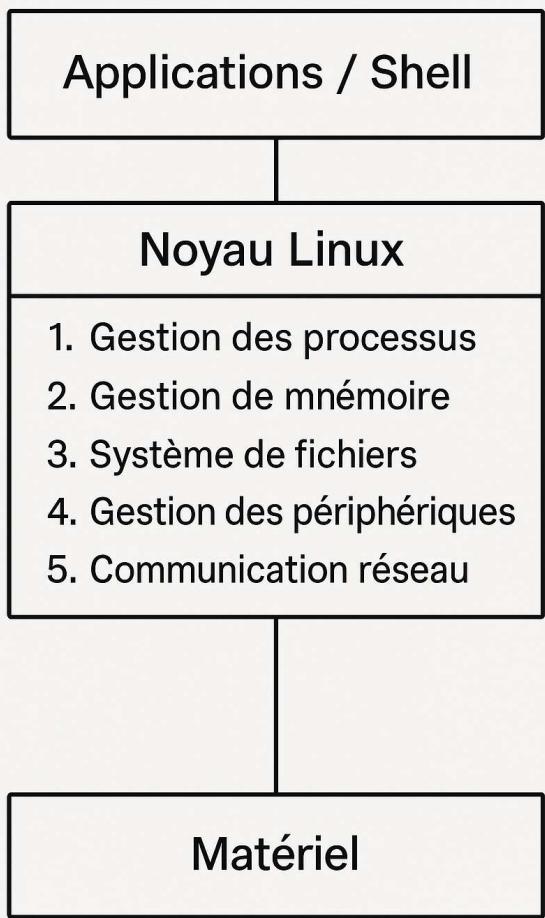
Les **musiciens**, ce sont les **programmes** (navigateur, terminal, serveur web...).

Le **chef d'orchestre**, c'est le **noyau Linux**.

👉 Le noyau s'assure que :

chaque programme a le bon accès au matériel ;
personne ne joue trop fort (gestion des ressources) ;
tout reste synchronisé (ordonnancement des processus).

👉 **Le noyau est la couche intermédiaire** entre le matériel et les logiciels.



Les 5 rôles clés du noyau

1 - Gestion des processus

Le noyau orchestre la création, l'exécution et l'ordonnancement des processus sur le CPU pour optimiser les performances.

2 - Gestion de la mémoire

Il alloue la RAM, gère la pagination et le swap pour assurer l'efficacité de la mémoire.

3 - Contrôle des fichiers

Le noyau régule l'accès aux fichiers et les permissions pour garantir la sécurité et l'intégrité des données.

4 – 5 Gestion des périphériques et des réseaux

Il pilote la communication avec les périphériques matériels et gère les interfaces réseau avec les protocoles comme TCP/IP.

Les distribution linux



1. Un même noyau, plusieurs “habillages”

- Toutes les distributions partagent le même cœur : le noyau Linux.
- Mais autour de ce noyau, chaque distribution ajoute :
 - des **outils système** (gestion des paquets, services, etc.)
 - une **philosophie d'utilisation** (stabilité, performance, simplicité, liberté...)
 - une **interface ou un environnement graphique différent**
 - et une **communauté** qui maintient et met à jour le tout

 **Analogie :**

Le noyau = le moteur

La distribution = la voiture complète (carrosserie, options, interface...)

2. Ce qui change d'une distribution à l'autre

Élément	Exemple	Ce que ça change
Gestionnaire de paquets	apt (Debian/Ubuntu), dnf (Fedora), pacman (Arch)	Méthode d'installation et de mise à jour
Environnement de bureau	GNOME, KDE, XFCE	Apparence, ergonomie, performances
Objectif / philosophie	Ubuntu = simplicité, Debian = stabilité, Arch = personnalisation	Public visé
Cycle de mise à jour	Rolling release (Arch) vs LTS (Ubuntu, Debian)	Fréquence et risque de bugs
Support & communauté	Canonical (Ubuntu) vs communauté libre	Niveau d'assistance, documentation

3. Les grandes familles

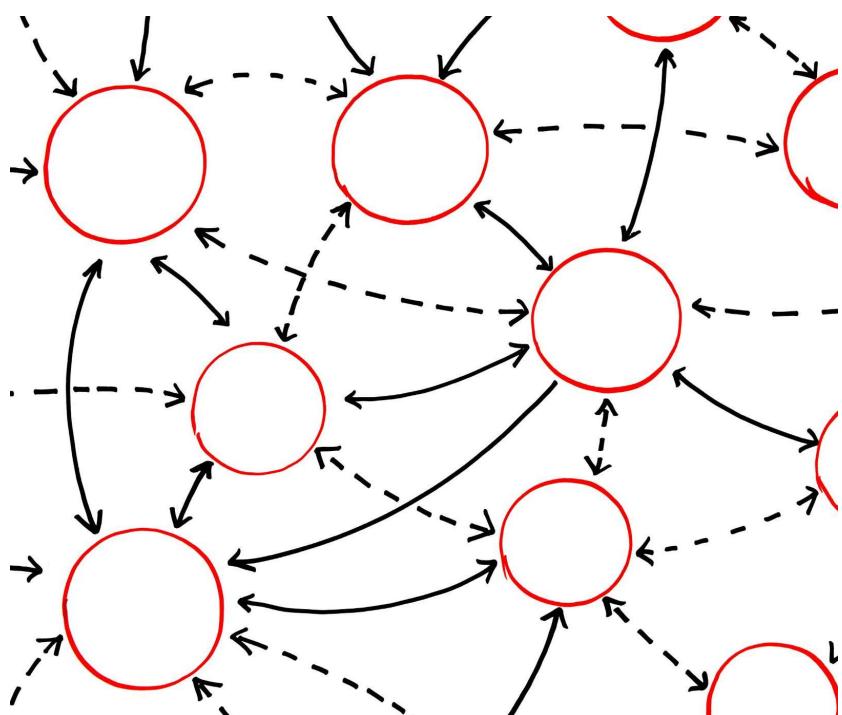


Quelques chiffres

Catégorie	Nombre approximatif	Exemples
● Actives & populaires	~300 à 600	Ubuntu, Debian, Fedora, Arch, openSUSE, Kali, Mint...
● En développement ou niches	~200	Garuda, Endless OS, Nitrux, ElementaryOS...
● Abandonnées / historiques	+400	Mandriva, Knoppix, CrunchBang, etc.

Installation, virtualisation et prise en main

Concepts de virtualisation et mise en pratique avec VirtualBox



Importance de la virtualisation

La virtualisation permet une utilisation flexible et efficace des ressources informatiques en séparant matériel et machines virtuelles.

Concept d'hyperviseur

L'hyperviseur gère plusieurs machines virtuelles sur un même matériel physique en allouant les ressources de façon dynamique.

Avantages pédagogiques

La virtualisation facilite l'apprentissage en permettant des expérimentations sécurisées et un environnement isolé.

Utilisation de VirtualBox

VirtualBox offre une interface intuitive pour créer des VM, gérer les snapshots et partager des dossiers facilement.



TP 0 – Installation de VirtualBox et mise en place d'Ubuntu Linux

Voir le sujet dans notion

Ligne de commande

Le terminal, c'est quoi ?

```
devtmpfs: mounted
Freeing unused kernel memory: 160K
This architecture does not have kernel memory protection.
Run /sbin/init as init process
Run /etc/init as init process
Run /bin/init as init process
process '/bin/busybox' started with executable stack
init started: BusyBox v1.28.4 (2018-06-01 07:15:55 UTC)
random: fast init done
ethoc 92000000.ethoc eth0: Link is Up - 10Mbps/Full - flow control off
udhcpc: started, v1.28.4
Setting IP address 0.0.0.0 on eth0
udhcpc: sending discover
udhcpc: sending select for 10.5.143.96
udhcpc: lease of 10.5.143.96 obtained, lease time 900
Setting IP address 10.5.143.96 on eth0
Deleting routers
route: SIOCDELRT: No such process
Adding router 10.5.0.1
Recreating /etc/resolv.conf
  Adding DNS server 10.5.0.1
  Adding DNS server 8.8.8.8
  Adding DNS server 8.8.4.4
Console Linux / login:
```

Le terminal = une interface en ligne de commande pour dialoguer avec le système.

Tout ce que tu fais en clics dans une interface graphique, tu peux le faire ici en texte.

Avantages :

Plus rapide et précis

Reproductible (scripts)

Accessible à distance (SSH)

La structure de base d'une commande

```
commande [options] [arguments]
```

- **Exemple :**

```
ls -l /home  
cp fichier.txt dossier/  
rm -r ancien_dossier
```

Comment me déplacer dans Linux ?

Action	Commande	Exemple
Lister le contenu	ls	ls -l
Changer de dossier	cd	cd /home/user
Revenir en arrière	cd ..	
Emplacement actuel	pwd	
Créer un dossier	mkdir	mkdir test

Gestion des fichiers

Action	Commande	Exemple
Copier	cp	cp a.txt b.txt
Déplacer / renommer	mv	mv a.txt dossier/
Supprimer	rm	rm fichier.txt
Créer un fichier	touch	touch notes.txt
Lire un fichier	cat, less	cat notes.txt

Gestion des droits et utilisateurs

Action	Commande	Exemple
Voir les permissions	ls -l	
Changer les droits	chmod	chmod 755 script.sh
Changer le propriétaire	chown	chown user:group fichier
Savoir qui je suis	whoami	
Devenir super-utilisateur	sudo	sudo apt update

Rechercher et gérer les processus

Objectif

Rechercher un fichier

Chercher un mot dans un fichier

Voir les processus

Tuer un processus

Commande

find

grep

ps, top, htop

kill

Exemple

find / -name *.conf

grep "root" /etc/passwd

kill 1234



TP 1 – Premiers pas dans le terminal Linux

Voir le sujet dans notion

Exploration du système Linux

Contexte du TP

Tu viens d'arriver dans une entreprise dont tous les postes tournent sous Linux.

L'administrateur système est parti sans laisser de documentation.

Ta mission, en binôme : **retrouver et comprendre les fichiers clés du système.**

Aucune commande ne t'est donnée – à toi d'explorer, de chercher, et de déduire leur rôle.

Objectifs du TP

Explorer le système de fichiers Linux
à partir de la racine /

Identifier les grands dossiers et
comprendre leur rôle

Lire le contenu de fichiers texte
système

Utiliser les commandes de base (ls,
cd, cat, less, pwd...)

Expliquer tes découvertes à la classe



Tout commence à
la racine /

```
└── bin/  
└── etc/  
└── home/  
└── var/  
└── dev/
```

Les outils à ta disposition

Objectif	Commande utile
Se déplacer	cd, pwd
Voir ce qu'il y a	ls, ls -l
Lire un fichier	cat, less
Rechercher	find, grep
Comprendre	man

 TP 2 – Exploration du système Linux

Voir le sujet dans notion

Gestion des utilisateurs, groupes et autorisation

Pourquoi gérer les droits ?

Sur un système Linux, **tout est fichier**.

Et chaque fichier doit avoir :

- un **propriétaire** (qui peut tout faire),

- un **groupe** (qui peut partager certains accès),

- et des **autorisations précises** (lecture, écriture, exécution).

Les utilisateurs et les groupes

Élément	Description	Exemple
 Utilisateur	Personne ou service ayant un compte sur la machine	gaultier, www-data, root
 Groupe	Ensemble d'utilisateurs avec des droits communs	students, admins, sudo
 root	Le super-utilisateur : a <i>tous les droits</i>	À manier avec précaution 

Créer des utilisateurs et des groupes

Action	Commande	Exemple
Créer un utilisateur	<code>sudo useradd <nom></code>	<code>sudo useradd alice</code>
Créer un utilisateur + dossier perso	<code>sudo useradd -m <nom></code>	<code>sudo useradd -m alice</code>
Définir le mot de passe	<code>sudo passwd <nom></code>	<code>sudo passwd alice</code>
Créer un groupe	<code>sudo groupadd <nom></code>	<code>sudo groupadd profs</code>
Ajouter un utilisateur dans un groupe	<code>sudo usermod -aG <groupe> <user></code>	<code>sudo usermod -aG profs alice</code>

Où sont stockés les utilisateurs et groupes ?

Type d'info	Fichier système	Contenu
Liste des utilisateurs	/etc/passwd	nom, UID, répertoire, shell
Mots de passe chiffrés	/etc/shadow	(accessible seulement à root)
Liste des groupes	/etc/group	nom du groupe, GID, membres

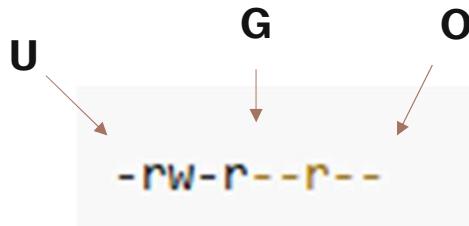
Supprimer ou modifier un utilisateur

Action	Commande	Exemple
Supprimer un utilisateur	<code>sudo userdel <nom></code>	<code>sudo userdel alice</code>
Supprimer aussi le dossier perso	<code>sudo userdel -r <nom></code>	<code>sudo userdel -r alice</code>
Supprimer un groupe	<code>sudo groupdel <nom></code>	<code>sudo groupdel profs</code>
Modifier le shell ou le dossier	<code>sudo usermod -s /bin/bash <nom></code>	<code>sudo usermod -s /bin/bash alice</code>

Les trois catégories de droits

Catégorie	Symbole	Description
User (u)		Le propriétaire du fichier
Group (g)		Les membres du groupe associé
Others (o)		Tous les autres utilisateurs

Les types de permissions



Lettre	Nom	Action possible sur un fichier	Sur un dossier
r	Read	Lire le contenu	Lister le contenu
w	Write	Modifier / supprimer	Créer ou supprimer des fichiers
x	Execute	Exécuter un programme	Entrer dans le dossier

Lire les permissions

-rwxr-Xr--

Partie

-

rwx

r-x

r--

Signification

type de fichier (ici, fichier classique)

propriétaire : lecture + écriture + exécution

groupe : lecture + exécution

autres : lecture seule

Modifier les droits

Objectif	Commande	Exemple
Changer les droits	chmod	chmod 755 script.sh
Changer le propriétaire	chown	chown user fichier
Changer le groupe	chgrp	chgrp profs fichier

r	w	x	Valeur
1	1	1	7
1	0	1	5
1	0	0	4

Gestion des processus

Qu'est-ce qu'un processus ?

- Un **processus** = un programme en cours d'exécution.
- Quand tu lances un logiciel, le système crée un processus pour lui :
 - il occupe de la mémoire,
 - utilise du CPU,
 - et vit jusqu'à sa fin ou sa fermeture.

L'arborescence des processus

- Chaque processus a un **PID (Process ID)** unique.
Certains ont un **parent (PPID)** – celui qui les a lancés.

```
systemd (PID 1)
├─ gdm
│  └─ gnome-shell
└─ bash
    ├─ nano
    └─ firefox
```

Observer les processus

Commande	Rôle	Exemple
ps	Liste les processus courants	ps aux
top	Vue dynamique en temps réel	top
htop	Version améliorée et colorée	htop
pidof	Donne le PID d'un programme	pidof bash

États et hiérarchie des processus

État	Signification	Exemple
R	Running (en cours)	processus actif
S	Sleeping	en attente d'un événement
Z	Zombie	processus terminé mais encore listé
T	Stopped	suspendu (Ctrl+Z)

Gérer les processus

Action	Commande	Exemple
Lancer en tâche de fond	commande &	sleep 100 &
Voir les tâches en fond	jobs	
Ramener en avant	fg %1	
Arrêter un processus	kill <PID>	kill 1234
Forcer l'arrêt	kill -9 <PID>	
Arrêter tous les processus d'un nom	pkill <nom>	pkill firefox

Processus, services et démons

Service

sshd

cron

cupsd

NetworkManager

Rôle

Connexions distantes

Tâches planifiées

Impression

Réseau



TP 3 – Gestion des utilisateurs, droits et processus

Voir le sujet dans notion

Scripting,
automatisation
et édition de
fichiers

Pourquoi écrire un script ?

- **Objectif :** donner du sens à l'automatisation.
 - 💬 Un **script** = une suite de commandes que ton ordinateur exécute à ta place.
 - 🤝 C'est comme écrire une recette que Linux suivra *sans te poser de questions*.
- **Exemples concrets :**
 - Lancer une sauvegarde tous les soirs
 - Nettoyer un dossier automatiquement
 - Générer un rapport système

Structure d'un script Bash

- Un script est un **fichier texte** contenant des commandes Bash.
Il s'exécute comme un petit programme.

```
#!/bin/bash
echo "Bonjour, je suis ton premier script !"
date
```

- `#!/bin/bash` → indique quel interpréteur exécute le script.
- Chaque ligne = une commande.
- Les `#` servent pour les commentaires.

Créer et exécuter un script

Étape	Commande
1 Créer le fichier	<code>nano monscript.sh</code>
2 Rendre exécutable	<code>chmod +x monscript.sh</code>
3 Lancer le script	<code>./monscript.sh</code>

Variables et affichage

```
#!/bin/bash
nom="Gaultier"
echo "Bonjour $nom !"
```

- Les variables ne prennent pas d'espace autour du =
- Pour les utiliser → \$nom
- Les guillemets évitent les surprises (espaces, caractères spéciaux)

Conditions et boucles

```
if [ -f "/etc/passwd" ]; then
    echo "Le fichier existe"
else
    echo "Fichier introuvable"
fi
```

```
for fichier in *.log
do
    echo "Suppression de $fichier"
    rm $fichier
done
```

Erreurs, journalisation et automatisation

Objectif

Rediriger les erreurs

Enregistrer la sortie

Lancer en arrière-plan

Exécuter à intervalle régulier

Astuce / Commande

commande 2> erreurs.log

commande >> sortie.txt

./script.sh &

cron **ou** crontab -e

Les éditeurs de texte sous Linux



Pourquoi c'est important ?

- Sous Linux, *tout* est fichier texte :
les scripts, les configurations, les logs, les mots de passe...
- 👉 Donc savoir éditer un fichier = savoir administrer ton système.

Trois familles d'éditeurs

Type	Exemple	Avantage	Quand l'utiliser
 Éditeurs simples (ligne de commande)	nano	Facile, intuitif, parfait pour débuter	Modifier un fichier rapidement
 Éditeurs avancés (CLI)	vim, vi	Ultra puissants, macros, navigation rapide	Pour les utilisateurs expérimentés
 Éditeurs graphiques	gedit, code (VS Code)	Interface visuelle, multi-fichiers, extensions	Sur une session graphique complète

Les essentiels à retenir sur nano

Action	Raccourci
Enregistrer	Ctrl + O
Quitter	Ctrl + X
Aide	Ctrl + G
Couper / coller	Ctrl + K / Ctrl + U

Petit rappel pratique

- Ouvrir un fichier : nano monscript.sh
- Créer un nouveau fichier : nano nouveau.sh
- Sauvegarder et quitter : Ctrl + O, puis Ctrl + X



TP 4 – Crée un script utile pour ton système

Voir le sujet dans notion

Réseau, services et supervision

Qu'est-ce qu'un service réseau

-  **Un service réseau = un programme en tâche de fond (daemon)** qui **écoute sur un port** et **répond à des requêtes** d'autres machines.
- Exemple :

Service	Port	Rôle
ssh	22	Connexion distante
apache2 / nginx	80 / 443	Serveur web
smb	445	Partage de fichiers
mysql	3306	Base de données

Processus, démons et systemd

- Derrière chaque service → un **processus** géré par **systemd**.

Élément	Rôle
Processus	Programme en cours d'exécution
Daemon (service)	Processus lancé en arrière-plan
systemd	Le “chef d'orchestre” des services (PID 1)

Gérer un service

Action	Commande	Exemple
Démarrer	<code>sudo systemctl start <service></code>	<code>sudo systemctl start apache2</code>
Arrêter	<code>sudo systemctl stop <service></code>	<code>sudo systemctl stop ssh</code>
Redémarrer	<code>sudo systemctl restart <service></code>	
Vérifier l'état	<code>sudo systemctl status <service></code>	
Activer au démarrage	<code>sudo systemctl enable <service></code>	
Désactiver au démarrage	<code>sudo systemctl disable <service></code>	

Identifier les ports et connexions

Commande	Rôle	Exemple
ss -tulnp	Liste les ports ouverts et les processus associés (ancienne version)	ss -tulnp
netstat -tulnp		
curl / wget	Tester l'accès web	curl http://localhost
ping	Vérifier la connectivité réseau	ping 127.0.0.1

Diagnostiquer un service

- Un service peut **tomber, planter, ou refuser une connexion.**
Il faut savoir où chercher les infos 

Vérification

État général

Logs détaillés

Ports ouverts

Processus en cours

Fichiers de conf

Commande / Fichier

`systemctl status <service>`

`journalctl -u <service>`

`ss -tulnp`

``ps aux`

`/etc/<service>/`

Cycle de vie d'un service

Installation → Activation → Exécution → Supervision → Journalisation

- **Rôle de l'administrateur :**

- savoir **où** est installé le service,
- savoir **quand** il tourne,
- savoir **pourquoi** il s'arrête.



TP 5 – Mettre en place et diagnostiquer un service réseau

Voir le sujet dans notion

 TP 6 – Superviser et sauvegarder ton mini-serveur Linux

Voir le sujet dans notion