CODING FACTORY
by ESIEE[tech]

# Architecture Linux

Félicien BOURY
Maxence VILLET SCHOUMAKER

# Sommaire

# Introduction à Linux et son architecture

- ## User Story 11

### Description

En tant qu'étudiant, je veux identifier les composants principaux de Linux (kernel, shell, système de fichiers), afin de comprendre comment le système fonctionne dans son ensemble.

### Définition

- **Kernel :** Noyau du système d'exploitation Linux, gérant les ressources matérielles et les communications entre le matériel et les logiciels.
- **Shell :** Interface en ligne de commande permettant à l'utilisateur d'interagir avec le système en exécutant des commandes.
- **Système de fichiers :** Structure organisationnelle qui gère le stockage, la récupération et la gestion des données sur le disque.

- ## User Story 12

### Description

En tant qu'équipe, nous voulons comparer deux distributions Linux, pour expliquer leurs points communs et différences à la classe.

### Debian vs Arch

#### Points communs

- Les deux distributions reposent sur le noyau Linux et partagent une philosophie open source.
- Elles utilisent des paquets pour gérer les logiciels, bien que leurs systèmes de gestion diffèrent (APT pour Debian, Pacman pour Arch).
- Toutes deux offrent une grande flexibilité et sont adaptées à des utilisateurs avancés.

#### Différences

- Debian est réputée pour sa stabilité, avec des versions bien testées et une approche plus conservatrice. Elle est idéale pour les serveurs et les environnements professionnels.
- Arch, en revanche, suit un modèle de publication rolling release, offrant des logiciels à la pointe de la technologie, mais au prix d'une complexité accrue. Elle est souvent préférée par les utilisateurs qui aiment personnaliser leur système dès l'installation.

## - User Story 13

### Description

En tant qu'utilisateur novice, je veux naviguer dans l'arborescence Linux avec des commandes de base (`ls`, `cd`, `pwd`), afin de me familiariser avec le système.

- `ls` : Cette commande me permet de lister les fichiers et dossiers présents dans le répertoire courant. C'est un premier pas pour visualiser le contenu d'un dossier.
- `cd` : Grâce à cette commande, je peux me déplacer d'un répertoire à un autre, en explorant les différents niveaux de l'arborescence. Par exemple, cd Documents m'emmène dans le dossier "Documents".
- `pwd` : Cette commande m'indique le chemin absolu du répertoire dans lequel je me trouve, ce qui est utile pour ne pas me perdre dans la structure des fichiers.

En maîtrisant ces commandes, je commence à comprendre comment Linux organise ses fichiers et je gagne en confiance pour explorer le système plus en profondeur.

- User Story 14

### Description

En tant qu'étudiant, je veux comprendre le rôle du noyau Linux dans la gestion des ressources matérielles, afin de mieux appréhender son importance dans le système.

Le noyau agit comme un intermédiaire essentiel entre le matériel (processeur, mémoire, disques, périphériques) et les logiciels. Voici ses principales fonctions :

- Gestion du processeur : Il alloue le temps CPU aux différents processus, assurant une exécution fluide et équilibrée des tâches.
- Gestion de la mémoire : Il contrôle l'accès à la RAM, optimise son utilisation et gère la mémoire virtuelle pour éviter les saturations.
- Gestion des périphériques : Il pilote les interactions avec les composants matériels (clavier, souris, disques, réseau) via des pilotes (drivers).
- Sécurité et isolation : Il garantit que les processus n'interfèrent pas entre eux et protège l'accès aux ressources critiques.

En comprenant ces mécanismes, je réalise à quel point le noyau est indispensable au bon fonctionnement du système. Il est le cœur invisible qui permet à Linux d'être à la fois puissant, stable et polyvalent.

- User Story 15

### Description

En tant qu'équipe, nous voulons expliquer les différences entre un shell graphique (GUI) et un shell en ligne de commande (CLI), pour savoir quand utiliser chacun.

#### Shell graphique (GUI)

- Interface visuelle : Le GUI propose des fenêtres, des icônes et des menus, ce qui le rend intuitif et accessible aux utilisateurs novices.
- Utilisation : Idéal pour les tâches courantes (navigation web, gestion de fichiers, bureautique) et pour ceux qui préfèrent une interaction visuelle.
- Avantages : Facilité d'utilisation, prise en main rapide, pas besoin de mémoriser des commandes.
- Exemples : GNOME, KDE, Windows Explorer.

### Shell en ligne de commande (CLI)

- Interface textuelle : Le CLI fonctionne via des commandes saisies au clavier, offrant un contrôle précis et direct sur le système.
- Utilisation : Essentiel pour l'administration système, l'automatisation de tâches, ou le travail sur des serveurs distants.
- Avantages : Puissance, flexibilité, capacité à exécuter des scripts et à gérer des systèmes sans interface graphique.
- Exemples : Bash, Zsh, PowerShell.

### Quand utiliser chacun ?

- GUI : Pour les utilisateurs débutants ou les tâches quotidiennes nécessitant une interaction visuelle.
- CLI : Pour les utilisateurs avancés, l'administration système, ou les environnements où les ressources sont limitées (serveurs, machines distantes).

## User Story 16

### Description

En tant qu'utilisateur, je veux tester différentes commandes pour interroger le système (uname, whoami, hostname), afin de me familiariser avec la CLI.

Voici trois commandes essentielles pour commencer :

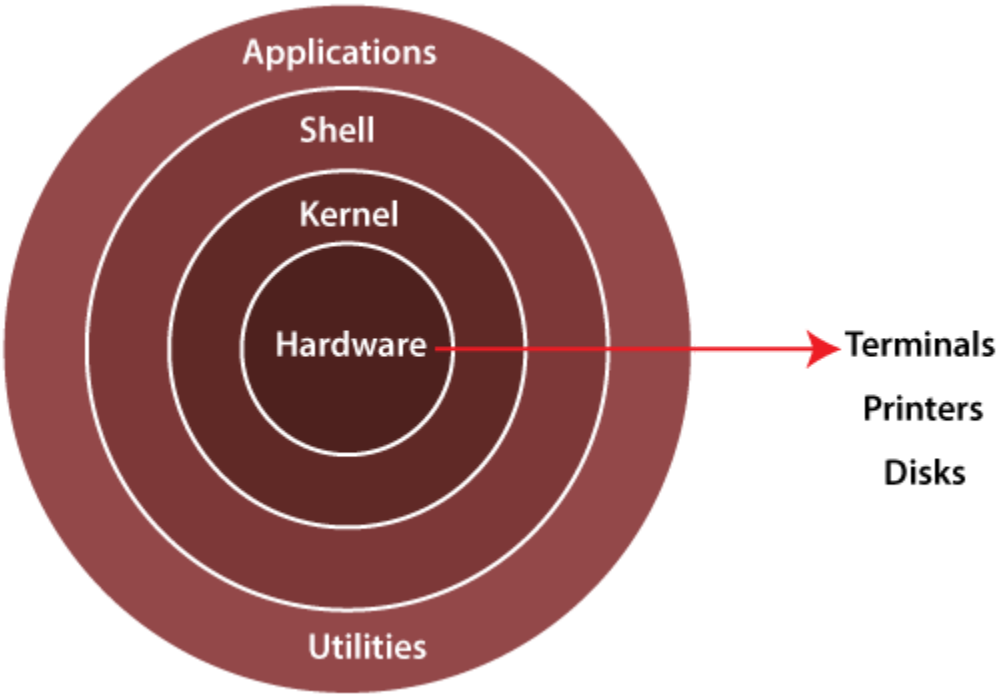
- `uname` : Cette commande affiche des informations sur le système d'exploitation. Par exemple, `uname -a` donne des détails complets, comme le nom du noyau, la version et l'architecture du système.
- `whoami` : Simple mais utile, cette commande indique le nom de l'utilisateur actuellement connecté. C'est pratique pour vérifier sous quel compte on travaille.
- `hostname` : Elle affiche le nom de la machine sur le réseau. Cela permet d'identifier rapidement le système sur lequel on se trouve, surtout dans un environnement avec plusieurs machines.

En utilisant ces commandes, je me familiarise avec la ligne de commande (CLI) et j'apprends à interagir directement avec le système. Cela me donne une base solide pour explorer des fonctionnalités plus avancées et mieux comprendre l'environnement Linux.

- User Story 17

Description

En tant qu'étudiant, je veux créer une représentation visuelle de l'architecture Linux (diagramme), pour expliquer clairement les interactions entre ses composants.



- User Story 18

Description

En tant qu'équipe, nous voulons comparer les avantages et inconvénients de Linux par rapport à d'autres systèmes d'exploitation (Windows,macOS), afin d'argumenter sur ses points forts.

**Avantages de Linux :**

- **Gratuité et Open Source** : Contrairement à Windows et macOS, Linux est souvent gratuit et offre un code source modifiable.
- **Sécurité** : Moins vulnérable aux malwares grâce à une architecture robuste et une communauté active pour les correctifs.
- **Personnalisation** : Hautement configurable avec différentes distributions adaptées à divers besoins (Ubuntu, Arch, etc.).
- **Performance** : Léger et efficace, idéal pour les anciens matériels ou les serveurs.
- **Communauté active** : Une vaste base d'utilisateurs et de développeurs prête à aider.

**Inconvénients de Linux :**

- **Courbe d'apprentissage** : Moins intuitif pour les débutants, en particulier pour ceux habitués à Windows ou macOS.
- **Compatibilité logicielle** : Certaines applications populaires (comme Adobe Photoshop) ne sont pas nativement disponibles.
- **Support matériel** : Les pilotes matériels, notamment pour des périphériques spécifiques, peuvent manquer ou nécessiter des ajustements.

En résumé, Linux excelle en termes de flexibilité et de sécurité, mais peut poser des défis pour les nouveaux utilisateurs ou ceux ayant des besoins logiciels spécifiques.

# Gestion des fichiers et systèmes de fichiers

- User Story 21

Description

En tant qu'utilisateur, je veux créer et organiser des répertoires et fichiers, pour structurer mes données efficacement.

Pour structurer efficacement vos données sur Linux, voici quelques commandes essentielles

1. Créer un répertoire

   Utilisez mkdir pour créer un nouveau répertoire.

   *Exemple :*

   ```
   mkdir MonProjet
   ```

2. Naviguer entre les répertoires

   Utilisez cd pour vous déplacer dans un répertoire.

   *Exemple :*

   ```
   cd MonProjet
   ```

3. Créer un fichier vide

   Utilisez touch pour créer un fichier vide.

*Exemple :*

```
touch fichier.txt
```

4. Lister le contenu d'un répertoire :

   Utilisez ls pour afficher les fichiers et répertoires dans le répertoire courant.

   *Exemple :*

   ```
   ls
   ```

5. Organiser avec des sous-répertoires

   Vous pouvez combiner ces commandes pour créer une structure complexe.

   *exemple :*

   ```
   mkdir -p MonProjet/{Docs,Images,Scripts}
   ```
   Cela crée un répertoire MonProjet avec trois sous-répertoires : Docs, Images, et Scripts.

- User Story 22

Description

En tant qu'équipe, nous voulons démontrer comment monter un système de fichiers externe (clé USB ou disque virtuel), afin de comprendre son intégration dans Linux.

Pour monter un système de fichiers externe dans Linux (comme une clé USB ou un disque virtuel) :

**Brancher le périphérique** : Connectez la clé USB ou le disque à votre machine.

**Identifier le périphérique** : Utilisez la commande `lsblk` ou `fdisk -l` pour trouver le nom du périphérique (par ex., `/dev/sdb1`).

**Créer un point de montage** : Créez un dossier pour monter le système de fichiers, par exemple : `sudo mkdir /mnt/usb`.

**Monter le périphérique** : Montez-le avec la commande : `sudo mount /dev/sdb1 /mnt/usb`

**Accéder aux fichiers** : Le contenu est accessible dans le dossier `/mnt/usb`.

**Démonter le périphérique** : Une fois terminé, démontez-le pour éviter toute perte de données : `sudo umount /mnt/usb`

Cela illustre comment Linux intègre les périphériques de stockage externe via son système de fichiers hiérarchique.

- User Story 23

Description

En tant qu'étudiant, je veux apprendre les commandes pour gérer les fichiers (`cp`, `mv`, `rm`), pour manipuler les données en ligne de commande

1. Déplacer ou renommer des fichiers/répertoires

   Utilisez mv pour déplacer ou renommer un fichier/répertoire.

   *Exemple (déplacer) :*

```
mv fichier.txt /chemin/vers/dossier/
```

   *Exemple (renommer) :*

```
mv ancien_nom.txt nouveau_nom.txt
```

2. Copier des fichiers/répertoires

   Utilisez cp pour copier un fichier ou un répertoire.

   *Exemple :*

   ```
   cp fichier.txt /chemin/vers/dossier/
   ```

   *Pour copier un répertoire et son contenu, ajoutez l'option -r :*

   ```
   cp -r MonProjet /chemin/vers/dossier/
   ```

3. Supprimer des fichiers/répertoires

   Utilisez rm pour supprimer un fichier.

   *Exemple :*

   ```
   rm fichier.txt
   ```

   *Pour supprimer un répertoire et son contenu, utilisez rm -r :*

   ```
   rm -r MonProjet`
   ```

- User Story 24

Description

En tant qu'utilisateur, je veux explorer les différences entre les systèmes de fichiers courants (ext4, NTFS, FAT32), pour comprendre leur rôle dans Linux.

1. Utilisateur

- **Rôle :** L'utilisateur est la personne qui souhaite explorer et comprendre les différences entre les systèmes de fichiers.
- **Objectif :** L'utilisateur veut acquérir des connaissances sur les systèmes de fichiers pour mieux comprendre leur fonctionnement dans Linux.

2. Systèmes de fichiers

- ext4 :
  - Description : Un système de fichiers couramment utilisé sous Linux. Il est connu pour sa performance, sa stabilité et sa capacité à gérer de grands volumes de données.
  - Utilisation : Principalement utilisé dans les environnements Linux.
  - Caractéristiques : Journalisation, support des grandes tailles de fichiers et de partitions, gestion des permissions de fichiers.
- NTFS :
  - Description : Un système de fichiers développé par Microsoft, principalement utilisé dans les systèmes d'exploitation Windows.
  - Utilisation : Utilisé dans les environnements Windows, mais peut être lu et parfois écrit sous Linux.
  - Caractéristiques : Support des grandes tailles de fichiers et de partitions, journalisation, compression native, chiffrement.
- FAT32 :
  - Description : Un système de fichiers plus ancien, compatible avec de nombreux systèmes d'exploitation, y compris Windows, Linux, et macOS.
  - Utilisation : Souvent utilisé pour les clés USB, les cartes mémoire, et autres dispositifs de stockage amovibles.
  - Caractéristiques : Simplicité, compatibilité étendue, mais limitations en termes de taille de fichiers (4 Go maximum) et de partitions.

3. Contexte d'utilisation dans Linux

- **ext4 :** Le système de fichiers natif et préféré pour les installations Linux en raison de ses performances et de ses fonctionnalités avancées.
- **NTFS :** Souvent utilisé pour accéder à des partitions Windows depuis Linux, mais avec des limitations en écriture dans certains cas.
- **FAT32 :** Utilisé pour des dispositifs de stockage amovibles qui doivent être accessibles sur plusieurs systèmes d'exploitation.

4. Objectif de compréhension

- **Rôle des systèmes de fichiers :** L'utilisateur cherche à comprendre comment ces systèmes de fichiers fonctionnent, leurs avantages, leurs inconvénients, et leur pertinence dans un environnement Linux.

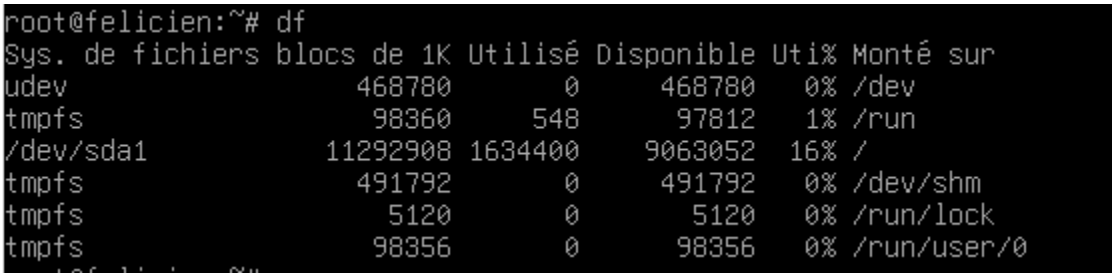
5. Résumé des différences

- **ext4 :** Optimisé pour Linux, performances élevées, gestion avancée des fichiers.
- **NTFS :** Optimisé pour Windows, compatible avec Linux, fonctionnalités avancées comme le chiffrement.
- **FAT32 :** Universellement compatible, mais limité en termes de taille de fichiers et de fonctionnalités.

- User Story 25

Description

En tant qu'étudiant, je veux utiliser la commande `df` pour visualiser l'espace disque disponible et comprendre son affichage.




Cette commande affiche l'espace disque utilisé et disponible pour chaque système de fichiers.

## Explication des colonnes :

- **Sys. de fichiers** : Nom du système de fichiers ou de la partition (par ex. `/dev/sda1`).
- **blocs de 1K** : Taille totale de la partition en blocs de 1 kilooctet.
- **Utilisé** : Espace déjà utilisé.

- **Disponible** : Espace restant disponible.
- **Uti%** : Pourcentage d'utilisation de la partition.
- **Monté sur** : Point de montage, où le système de fichiers est attaché dans l'arborescence Linux.

## User Story 26

### Description

En tant qu'équipe, nous voulons configurer des alias pour des commandes de gestion de fichiers, afin de rendre nos tâches plus efficaces.

Les alias sont des raccourcis que vous pouvez configurer dans votre terminal pour exécuter des commandes longues ou répétitives avec une simple abréviation. Cela permet de gagner du temps et de rendre vos tâches de gestion de fichiers plus efficaces. Voici comment configurer des alias pour vos besoins quotidiens :

- **Ouvrir le fichier de configuration du shell**
  Selon le shell que vous utilisez (Bash, Zsh, etc.), ouvrez le fichier approprié avec un éditeur de texte. Par exemple :

  Pour Bash : `nano ~/.bashrc`

  Pour Zsh : `nano ~/.zshrc`

- **Créer un alias**
  Ajoutez une ligne dans le fichier pour définir un alias. La syntaxe est la suivante :

  ```
  alias nom_alias='commande'
  ```

  Par exemple :

  Pour une copie rapide de fichiers :

  ```
  alias cpv='cp -v' # copie en mode verbeux
  ```

  Pour lister les fichiers avec des détails :

  ```
  alias ll='ls -l'
  ```

Pour supprimer des fichiers avec confirmation :

```
alias rm='rm -i'
```

- **Enregistrer et recharger le fichier de configuration**
  Une fois les alias ajoutés, enregistrez le fichier (Ctrl + O puis Entrée sous Nano) et rechargez-le dans votre session actuelle avec :

  ```
  source ~/.bashrc
  ```

  (ou remplacez `~/.bashrc` par le fichier de configuration de votre shell, si nécessaire).

- **Tester vos alias**
  Tapez simplement le nom de l'alias dans le terminal pour vérifier qu'il fonctionne. Par exemple, `ll` devrait exécuter la commande `ls -l`.

En configurant des alias adaptés à vos besoins, vous simplifierez vos tâches répétitives et optimiserez votre gestion de fichiers au quotidien.

- User Story 27

Description

En tant qu'utilisateur, je veux apprendre à archiver et compresser des fichiers (tar, gzip), pour réduire leur taille et faciliter leur partage.

Pour archiver et compresser des fichiers sous Linux, on utilise généralement les outils **tar** et **gzip** ensemble. Voici les étapes clés :

## Archiver avec `tar` :

L'outil **tar** permet de regrouper plusieurs fichiers et dossiers en un seul fichier d'archive sans compression. Par exemple, pour archiver un dossier nommé `dossier` :
`tar -cvf archive.tar dossier`

- `-c` : crée une nouvelle archive.
- `-v` : affiche les détails des fichiers traités.
- `-f` : spécifie le nom de l'archive.

### Compresser avec `gzip` :

On peut ensuite compresser l'archive avec **gzip** pour réduire sa taille :
`gzip archive.tar`
Cela produit un fichier compressé nommé `archive.tar.gz`.

### Méthode combinée :

Une méthode simplifiée permet de combiner les deux opérations en une seule commande :
`tar -cvzf archive.tar.gz dossier`

- `-z` : ajoute la compression avec **gzip**.

### Extraction :

Pour décompresser et extraire l'archive, utilisez :
`tar -xvzf archive.tar.gz`

- `-x` : extrait les fichiers.

Ces commandes rendent le partage des fichiers plus simple grâce à une taille réduite et à un regroupement efficace.

- User Story 28

Description

En tant qu'étudiant, je veux démonter un système de fichiers monté précédemment, pour comprendre comment gérer des disques amovibles de manière sécurisée.

Pour démonter un système de fichiers monté précédemment sous Linux, il faut utiliser la commande **umount**. Cette opération est essentielle pour gérer les disques amovibles de

manière sécurisée, en s’assurant que toutes les données en cours d’écriture sur le périphérique ont été enregistrées.

Voici les étapes principales :

- Identifiez le point de montage ou le périphérique à démonter, par exemple `/mnt/usb` ou `/dev/sdb1`.
- Utilisez la commande suivante pour démonter le système de fichiers :
  umount /mnt/usb
  ou
  umount /dev/sdb1

Si le système de fichiers est occupé (par exemple, si un fichier est ouvert ou un répertoire est en cours d'utilisation), la commande échouera. Dans ce cas, identifiez les processus en cours avec :
lsof +D /mnt/usb
ou
fuser -v /mnt/usb

Une fois les processus terminés ou libérés, répétez la commande pour démonter le système.

Enfin, une fois démonté, vous pouvez retirer le disque amovible en toute sécurité.

## Gestion des fichiers et systèmes de fichiers

### User Story 31

#### Description

En tant qu'administrateur système, je veux créer des utilisateurs et des groupes avec des droits spécifiques, afin de sécuriser l’accès à mon système.

Pour créer des utilisateurs et des groupes avec des droits spécifiques sous Linux, il est essentiel de suivre une approche structurée pour sécuriser l'accès au système. Voici les étapes principales :

- **Créer un groupe** : Utilisez la commande groupadd suivie du nom du groupe. Par exemple, groupadd projet crée un groupe nommé "projet".

- **Créer un utilisateur** : Utilisez la commande useradd pour créer un utilisateur et l'associer à un groupe. Par exemple, useradd -m -g projet utilisateur crée un utilisateur nommé "utilisateur" avec un répertoire personnel et l'assigne au groupe "projet".
- **Définir un mot de passe** : Utilisez passwd suivi du nom de l'utilisateur pour définir un mot de passe sécurisé pour l'utilisateur.
- **Attribuer des droits spécifiques** : Les permissions sur les fichiers et répertoires peuvent être configurées à l'aide des commandes chmod et chown. Par exemple, chown utilisateur:projet fichier permet de définir "utilisateur" comme propriétaire d'un fichier et "projet" comme groupe associé. Ensuite, chmod 770 fichier donne des droits de lecture, écriture et exécution au propriétaire et au groupe, tout en interdisant l'accès aux autres.
- **Gestion des sudoers** : Pour accorder des droits administratifs à un utilisateur, ajoutez-le au groupe sudo (ou wheel sur certaines distributions) avec la commande usermod -aG sudo utilisateur.

Ces étapes permettent de structurer les accès et de protéger le système en limitant les droits des utilisateurs à leurs besoins spécifiques.

- User Story 32

Description

En tant qu'utilisateur, je veux comprendre les concepts de permissions Linux (`chmod`, `chown`, `ls -l`), pour gérer l'accès aux fichiers de manière sécurisée.

Les permissions Linux sont essentielles pour gérer l'accès aux fichiers et dossiers de manière sécurisée. Chaque fichier ou dossier possède trois types de permissions : lecture (r), écriture (w) et exécution (x), qui s'appliquent à trois catégories d'utilisateurs : le propriétaire, le groupe et les autres.

Pour visualiser les permissions, la commande ls -l affiche une liste détaillée des fichiers avec leurs droits. Par exemple, une ligne commençant par -rw-r--r-- indique que le propriétaire a les droits de lecture et d'écriture, le groupe a seulement les droits de lecture, et les autres aussi.

La commande chmod permet de modifier les permissions. Par exemple, chmod 750 fichier attribue des droits de lecture, écriture et exécution au propriétaire, des droits de lecture et exécution au groupe, et aucun droit aux autres. Les valeurs numériques sont basées sur la somme des droits : lecture vaut 4, écriture 2 et exécution 1.

La commande chown permet de changer le propriétaire et/ou le groupe associé à un fichier. Par exemple, chown utilisateur:groupe fichier attribue "utilisateur" comme propriétaire et "groupe" comme groupe.

Ces outils permettent de contrôler précisément qui peut accéder, modifier ou exécuter un fichier, contribuant ainsi à la sécurité du système.

- User Story 33

### Description

En tant qu'équipe, nous voulons présenter un scénario où des permissions mal configurées peuvent poser problème, afin de sensibiliser à l'importance de ce sujet.

Pour personnaliser l'environnement d'un utilisateur sous Linux, le fichier .bashrc, situé dans le répertoire personnel de l'utilisateur, joue un rôle central. Ce fichier est exécuté à chaque ouverture d'un terminal interactif et permet de configurer divers aspects de l'environnement.

On peut, par exemple, ajouter des alias pour simplifier l'exécution de commandes fréquentes. Par exemple, un alias comme alias ll='ls -l' permet de taper ll au lieu de ls -l pour afficher une liste détaillée des fichiers.

Il est également possible d'ajouter ou de modifier les variables d'environnement. Par exemple, en ajoutant export PATH=$PATH:/chemin/vers/un/dossier, on inclut un nouveau chemin dans la variable PATH pour rendre des programmes spécifiques accessibles depuis n'importe où.

Enfin, on peut personnaliser l'invite de commande (le prompt) en modifiant la variable PS1. Par exemple, PS1="\u@\h:\w$ " affiche le nom de l'utilisateur, le nom de la machine et le répertoire courant.

Après avoir modifié le fichier .bashrc, il est nécessaire d'exécuter source .bashrc pour appliquer immédiatement les changements. Ces ajustements rendent l'environnement de l'utilisateur plus pratique et adapté à ses besoins spécifiques.

- User Story 34

### Description

En tant qu'administrateur système, je veux personnaliser l'environnement d'un utilisateur en modifiant son fichier .bashrc, afin de simplifier ses interactions avec le système.

Pour personnaliser l'environnement d'un utilisateur sous Linux, le fichier .bashrc situé dans le répertoire personnel est un outil clé. Ce fichier est exécuté automatiquement à chaque ouverture d'un terminal interactif, permettant d'adapter l'environnement de l'utilisateur à ses besoins.

Vous pouvez y ajouter des alias pour simplifier des commandes fréquentes. Par exemple, un alias comme alias ll='ls -la' permet d'utiliser une commande abrégée, ici ll, pour afficher les fichiers dans un format détaillé.

Il est également possible d'ajouter ou de modifier des variables d'environnement. Par exemple, en ajoutant export PATH=$PATH:/chemin/vers/un/dossier, vous pouvez inclure des répertoires spécifiques dans la variable PATH, rendant des programmes ou scripts accessibles depuis n'importe où.

Pour rendre l'invite de commande plus informative, modifiez la variable PS1. Par exemple, une configuration comme PS1="\u@\h:\w$ " personnalise l'affichage du prompt pour inclure le nom de l'utilisateur, celui de la machine et le répertoire courant.

Après avoir enregistré les modifications dans .bashrc, appliquez-les immédiatement en exécutant source .bashrc. Ces ajustements simplifient les interactions de l'utilisateur avec le système en rendant les commandes plus accessibles et l'environnement plus intuitif.

- User Story 35

### Description

En tant qu'équipe, nous voulons simuler un scénario où un utilisateur essaie d'accéder à un fichier sans permissions suffisantes, pour analyser les erreurs affichées.

Pour simuler un scénario où un utilisateur tente d'accéder à un fichier sans permissions suffisantes, vous pouvez suivre ces étapes :

- Créez un fichier avec un utilisateur ou un compte administrateur et définissez des permissions restrictives à l'aide de chmod. Par exemple, retirez les droits d'accès pour les autres utilisateurs en utilisant chmod 700 nom_du_fichier.
- Changez de session ou passez à un autre utilisateur sans privilèges sur ce fichier, par exemple en utilisant `su nom_utilisateur` ou en vous connectant directement à un autre compte.
- Essayez d'accéder au fichier, que ce soit pour le lire, le modifier ou l'exécuter. Par exemple, tentez d'afficher son contenu avec une commande de lecture.

Lorsqu'il n'a pas les permissions nécessaires, l'utilisateur verra des messages d'erreur spécifiques. Par exemple, "Permission denied" s'affichera pour une tentative de lecture ou d'accès.

Ce scénario est utile pour comprendre les mécanismes de sécurité liés aux permissions sous Linux, ainsi que pour diagnostiquer et corriger des erreurs d'accès au système.

- ## User Story 36

### Description

En tant qu'utilisateur, je veux comprendre les permissions spéciales comme le SUID, SGID et Sticky Bit, pour les appliquer dans des cas spécifiques.

Les permissions spéciales comme le **SUID**, **SGID** et **Sticky Bit** sont utilisées dans des cas spécifiques pour contrôler l'accès et le comportement des fichiers ou répertoires dans un système Linux. Voici une explication de chaque permission et ses usages :

1. **SUID (Set User ID)** :
    - Lorsqu'un fichier exécutable a le SUID activé, il s'exécute avec les privilèges de son propriétaire, quel que soit l'utilisateur qui l'exécute.
    - Utilisation : Cela est souvent utilisé pour des programmes nécessitant des privilèges élevés, comme `passwd`, qui modifie des fichiers système sensibles.

- Activation : Attribuez le bit SUID avec `chmod u+s nom_fichier`. Dans une liste des permissions, un fichier avec SUID apparaît avec un `s` à la place du `x` dans les permissions utilisateur (par exemple, `-rwsr-xr-x`).

2. **SGID (Set Group ID)** :
   - Si le SGID est défini sur un fichier exécutable, il s'exécute avec les privilèges du groupe propriétaire, quelle que soit l'identité de l'utilisateur.
   - Lorsqu'il est défini sur un répertoire, tous les fichiers créés à l'intérieur héritent automatiquement du groupe du répertoire, plutôt que du groupe par défaut de l'utilisateur.
   - Utilisation : Utile pour les projets collaboratifs où plusieurs utilisateurs partagent des fichiers dans un même répertoire.
   - Activation : Utilisez `chmod g+s nom_fichier_ou_repertoire`. Dans une liste des permissions, le SGID apparaît avec un `s` à la place du `x` dans les permissions de groupe (par exemple, `drwxr-sr-x`).
3. **Sticky Bit** :
   - Lorsqu'un répertoire a le Sticky Bit, seuls le propriétaire du fichier, le propriétaire du répertoire, ou l'administrateur peuvent supprimer ou modifier les fichiers dans ce répertoire, même si d'autres utilisateurs ont les permissions d'écriture.
   - Utilisation : Souvent utilisé pour des répertoires partagés comme `/tmp`, afin d'éviter que des utilisateurs ne suppriment les fichiers des autres.
   - Activation : Utilisez `chmod +t nom_repertoire`. Dans une liste des permissions, le Sticky Bit apparaît avec un `t` à la place du dernier `x` (par exemple, `drwxrwxrwt`).

Ces permissions spéciales permettent de répondre à des besoins spécifiques en matière de sécurité et de gestion des accès dans un environnement multi-utilisateur.

## User Story 37

### Description

En tant qu'étudiant, je veux créer un script pour automatiser la gestion des groupes et des permissions, afin de gagner du temps lors de configurations répétées.

Pour automatiser la gestion des groupes et des permissions sur un système Linux, vous pouvez créer un script qui exécute les tâches répétitives de manière efficace. Voici comment procéder :

1. **Définir les tâches à automatiser** : Identifiez les actions fréquentes, comme créer des groupes, ajouter des utilisateurs à des groupes, ou définir des permissions sur des répertoires.
2. **Écrire le script** :
    - Utilisez un langage de script comme **bash** pour écrire un fichier contenant les commandes nécessaires.
    - Par exemple :
        - Pour créer un groupe : `groupadd nom_groupe`
        - Pour ajouter un utilisateur à un groupe : `usermod -aG nom_groupe nom_utilisateur`
        - Pour configurer des permissions : `chmod 770 /chemin/du/repertoire` et `chgrp nom_groupe /chemin/du/repertoire`
3. **Intégrer des paramètres dynamiques** :
    - Utilisez des variables pour personnaliser les actions selon les besoins.
    - Exemple : `groupadd $1` où `$1` est un argument passé au script lors de son exécution pour définir le nom du groupe.
4. **Ajouter des contrôles de validation** :
    - Vérifiez si un groupe ou un utilisateur existe déjà avant de les créer, pour éviter les erreurs.
    - Par exemple : `if grep -q "^nom_groupe:" /etc/group; then echo "Le groupe existe déjà"; else groupadd nom_groupe; fi`
5. **Tester et exécuter le script** :
    - Sauvegardez le script dans un fichier avec une extension comme `.sh`.
    - Rendez-le exécutable avec la commande `chmod +x nom_du_script.sh`.
    - Exécutez-le avec les permissions administratives nécessaires : `sudo ./nom_du_script.sh`.
6. **Exemple d'utilisation** :
    - Vous pouvez inclure des commandes comme celles-ci dans votre script pour gérer des configurations courantes :
        - Création d'un groupe : `groupadd developpeurs`
        - Ajout d'un utilisateur : `usermod -aG developpeurs alice`
        - Configuration d'un répertoire partagé : `chmod 770 /projets` et `chgrp developpeurs /projets`

Avec un script bien conçu, vous gagnerez du temps en automatisant des configurations répétitives tout en réduisant les erreurs humaines.

- User Story 38

Description

En tant qu’équipe, nous voulons auditer les permissions sur une arborescence de répertoires, pour identifier les failles de sécurité potentielles.

Pour auditer les permissions sur une arborescence de répertoires et identifier les failles de sécurité potentielles, vous pouvez utiliser des outils et commandes permettant de lister les fichiers et leurs droits de manière détaillée.

Commencez par utiliser la commande ls -lR sur le répertoire racine de l’arborescence que vous souhaitez auditer. Cela affiche récursivement les fichiers et dossiers avec leurs permissions, les propriétaires et les groupes associés.

Ensuite, recherchez les fichiers ou dossiers avec des permissions trop larges, comme des droits d’écriture pour tous les utilisateurs, symbolisés par les lettres w dans les colonnes associées aux "autres". Vous pouvez filtrer ces résultats à l’aide de commandes comme grep ou en exportant la liste dans un fichier pour un examen approfondi.

Vérifiez également les fichiers appartenant à root ou à des utilisateurs inattendus, ainsi que ceux ayant le bit setuid ou setgid activé, qui peuvent poser des risques de sécurité.

Enfin, analysez les permissions globales sur les dossiers critiques, tels que les répertoires contenant des configurations sensibles ou des données confidentielles, pour vous assurer qu’ils sont correctement restreints. Cette approche permet d’identifier et de corriger rapidement les failles potentielles dans les permissions de fichiers et répertoires.

## Processus et gestion de services

- User Story 41

Description

En tant qu'utilisateur avancé, je veux surveiller les processus en cours avec les commandes ps et top, afin d’identifier ceux qui consomment le plus de ressources.

La commande ps permet d'afficher une liste instantanée des processus actifs. En la combinant avec des options, vous pouvez obtenir des informations détaillées, comme l'identifiant du processus (PID), l'utilisateur qui l'exécute, la quantité de mémoire et de CPU utilisée. Par exemple, en triant la sortie par utilisation de la mémoire ou du processeur, vous identifiez rapidement les processus les plus gourmands.

La commande top offre une vue interactive et en temps réel des processus. Elle affiche les processus triés par défaut selon leur utilisation du processeur. Vous pouvez observer des informations clés comme le pourcentage de CPU et de mémoire consommés, l'utilisateur qui a lancé le processus et le temps total d'exécution. Depuis l'interface de top, vous pouvez également tuer un processus ou ajuster son niveau de priorité.

Ces outils sont essentiels pour analyser les performances du système et repérer les processus qui pourraient ralentir l'ensemble des opérations.

- ## User Story 42

### Description

En tant qu'équipe, nous voulons démontrer comment démarrer,
arrêter, et redémarrer un service avec systemctl, pour comprendre la gestion des services sous Linux.

Pour gérer les services sous Linux, la commande systemctl est l'outil principal. Elle permet de démarrer, arrêter et redémarrer des services, offrant ainsi un contrôle efficace sur le fonctionnement du système.

Pour démarrer un service, utilisez systemctl suivi de start et du nom du service. Cela lance le service immédiatement si celui-ci est configuré correctement.

Pour arrêter un service, remplacez start par stop. Cette commande met fin à l'exécution du service, libérant ainsi les ressources qu'il utilisait.

Pour redémarrer un service, utilisez restart. Cette commande arrête puis relance le service, ce qui est utile pour appliquer des modifications ou résoudre des problèmes liés à son fonctionnement.

Enfin, pour vérifier l'état d'un service, utilisez status avec systemctl. Cela permet de voir si le service est actif, inactif ou en erreur, ainsi que des informations supplémentaires comme les journaux récents. Ces commandes sont essentielles pour administrer et dépanner les services sous Linux.

- User Story 43

### Description

En tant qu'étudiant, je veux créer un script Bash simple pour automatiser une tâche répétitive, afin d'appliquer les principes d'automatisation dans Linux.

Pour automatiser une tâche répétitive sous Linux, vous pouvez créer un script Bash simple. Un script Bash est un fichier texte contenant une série de commandes qui sont exécutées dans l'ordre, comme si elles étaient saisies manuellement dans un terminal.

Commencez par créer un fichier, par exemple script.sh, et ajoutez une première ligne appelée shebang, qui indique au système d'utiliser Bash pour exécuter le script. Ensuite, insérez les commandes nécessaires pour accomplir la tâche. Par exemple, pour sauvegarder un dossier, vous pouvez utiliser des commandes pour créer une archive, la compresser, et la copier dans un emplacement spécifique.

Ajoutez des variables pour rendre le script plus flexible. Par exemple, définissez des chemins ou des noms de fichiers dans des variables que vous pouvez facilement modifier sans toucher au reste du code.

Rendez le script exécutable en modifiant ses permissions avec chmod. Une fois prêt, exécutez-le simplement en l'appelant par son nom. Cela vous permet de gagner du temps et d'éviter les erreurs liées à la répétition manuelle de tâches complexes.

- User Story 44

Description

En tant qu'utilisateur avancé, je veux comprendre les différences entre les processus en avant-plan et en arrière-plan, afin de mieux les gérer avec
des commandes comme `bg` et `fg`.

Un processus en avant-plan s'exécute directement dans le terminal, occupant l'espace jusqu'à ce qu'il se termine ou soit interrompu. Par exemple, lorsque je lance une commande longue comme `tar -czf archive.tar.gz dossier/`, je ne peux pas interagir avec le terminal tant que la commande n'est pas terminée.
En revanche, un processus en arrière-plan me permet de continuer à utiliser le terminal pendant son exécution. Pour lancer un processus en arrière-plan, j'ajoute `&` à la fin de la commande, comme `tar -czf archive.tar.gz dossier/ &`. Le processus tourne alors en arrière-plan, et je reçois son PID (identifiant du processus) pour le gérer ultérieurement.
Si je veux mettre un processus en avant-plan en arrière-plan, je peux utiliser Ctrl+Z pour le suspendre, puis la commande bg pour le relancer en arrière-plan. Inversement, la commande fg permet de ramener un processus en avant-plan. Par exemple, si j'ai suspendu un processus avec Ctrl+Z, je peux taper fg pour le reprendre là où il s'était arrêté.
En maîtrisant ces commandes, je peux mieux gérer mes tâches, basculer entre les processus et optimiser mon temps de travail sur le terminal.

- User Story 45

Description

En tant qu'étudiant, je veux explorer les niveaux de priorité des processus (nice, renice), pour optimiser l'utilisation des ressources.

Pour cela, je veux explorer les niveaux de priorité des processus, notamment à travers les commandes nice et renice.
La commande nice me permet de lancer un processus avec un niveau de priorité spécifique. Par défaut, les processus ont une priorité de 0, mais en utilisant nice, je peux ajuster cette valeur entre -20 (priorité la plus élevée) et 19 (priorité la plus basse). Par exemple, si je veux lancer un script gourmand en ressources avec une priorité basse pour ne pas surcharger le système, je peux taper :

```
nice -n 19 ./mon_script.sh
```

Si un processus est déjà en cours d'exécution et que je veux modifier sa priorité, j'utilise la commande renice. Par exemple, si je découvre qu'un processus avec le PID 1234 consomme trop de ressources, je peux réduire sa priorité avec :

```
renice +10 -p 1234
```

En comprenant et en utilisant ces commandes, je peux mieux gérer les ressources de mon système, m'assurer que les tâches critiques ont la priorité nécessaire, et éviter que mon ordinateur ne ralentisse lors de l'exécution de programmes intensifs. Cela m'aide à travailler plus efficacement, même sur des machines aux capacités limitées.

- ## User Story 46

### Description

En tant qu'équipe, nous voulons créer et tester un service personnalisé (fichier `.service`), pour mieux comprendre le fonctionnement de `systemd`.

Nous commençons par créer un fichier de configuration pour notre service, par exemple mon_service.service, dans le répertoire /etc/systemd/system/. Ce fichier contient des directives essentielles comme Description, `ExecStart` pour spécifier la commande à exécuter, et `WantedBy` pour définir le moment où le service doit être démarré. Voici un exemple simple :

```
[Unit]
Description=Mon service personnalisé

[Service]
ExecStart=/usr/bin/python3 /chemin/vers/mon_script.py
Restart=on-failure

[Install]
WantedBy=multi-user.target
```

Une fois le fichier créé, nous rechargeons la configuration de systemd avec la commande :

```
sudo systemctl daemon-reload
```

Ensuite, nous démarrons notre service pour le tester :

```
sudo systemctl start mon_service.service
```

Pour vérifier son statut et nous assurer qu'il fonctionne correctement, nous utilisons :

```
sudo systemctl status mon_service.service
```

Si tout fonctionne comme prévu, nous activons le service pour qu'il démarre automatiquement au boot du système :

```
sudo systemctl enable mon_service.service
```

En créant et testant ce service personnalisé, nous acquérons une meilleure compréhension de systemd, de son fonctionnement et de son rôle dans la gestion des services sous Linux. Cela nous permet également d'automatiser des tâches et d'améliorer l'efficacité de nos déploiements.

- User Story 47

Description

En tant qu'utilisateur, je veux utiliser les journaux système (`journalctl`) pour diagnostiquer les erreurs d'un service, afin de pratiquer le dépannage.

Lorsqu'un service ne fonctionne pas comme prévu, je commence par vérifier son statut avec :

```
sudo systemctl status mon_service.service
```

Si le statut indique une erreur ou un échec, j'utilise `journalctl` pour explorer les logs détaillés du service. Par exemple, pour afficher les logs spécifiques à `mon_service.service`, je tape :

```
sudo journalctl -u mon_service.service
```

Cette commande me montre les entrées de journal associées au service, ce qui me permet d'identifier les messages d'erreur ou les avertissements. Si les logs sont trop volumineux, je

peux filtrer par période (avec `--since` ou `--until`) ou par niveau de priorité (comme `-p err` pour afficher uniquement les erreurs).
Pour suivre les logs en temps réel pendant que je redémarre le service, j'utilise l'option `-f` :

```
sudo journalctl -u mon_service.service -f
```

En analysant ces logs, je peux comprendre la cause des erreurs, qu'il s'agisse d'un problème de configuration, d'une dépendance manquante ou d'une permission incorrecte. Cela me permet de résoudre les problèmes efficacement et de pratiquer le dépannage système.
Grâce à journalctl, je gagne en autonomie pour diagnostiquer et résoudre les problèmes liés aux services, ce qui renforce mes compétences en administration système.

- User Story 48

Description

En tant qu'étudiant, je veux configurer un processus pour qu'il démarre automatiquement au démarrage du système, afin de comprendre la gestion des services persistants.

Pour commencer, je crée un fichier de configuration de service dans le répertoire `/etc/systemd/system/`. Par exemple, je nomme mon fichier `mon_service.service`. Ce fichier contient les directives nécessaires pour définir le comportement du service, comme la commande à exécuter (`ExecStart`) et les dépendances éventuelles. Voici un exemple simple :

```
[Unit]
Description=Mon service personnalisé

[Service]
ExecStart=/chemin/vers/mon_script.sh
Restart=on-failure

[Install]
```

```
WantedBy=multi-user.target`
```

Une fois le fichier créé, je recharge la configuration de systemd pour prendre en compte le nouveau service :

```
sudo systemctl daemon-reload
```

Ensuite, je démarre le service manuellement pour vérifier qu'il fonctionne correctement :

```
sudo systemctl start mon_service.service
```

Je vérifie son statut pour m'assurer qu'il est actif et sans erreur :

```
sudo systemctl status mon_service.service
```

Si tout est en ordre, j'active le service pour qu'il démarre automatiquement au prochain démarrage du système :

```
sudo systemctl enable mon_service.service
```

En configurant ce service, je comprends mieux comment systemd gère les processus persistants et comment automatiser des tâches au démarrage du système. Cela renforce mes compétences en administration système et me prépare à gérer des environnements plus complexes à l'avenir.

# Projet final

- User Story 51

Description

En tant qu'équipe, nous voulons configurer un serveur de fichiers local sous Linux, afin de partager des données entre différents utilisateurs avec des permissions adaptées.

Pour commencer, nous choisissons d'utiliser Samba, un outil puissant qui permet de partager des fichiers entre des machines Linux, mais aussi avec des systèmes Windows ou macOS. Nous installons Samba sur notre serveur avec la commande :

```
sudo apt install samba
```

Ensuite, nous configurons Samba en éditant son fichier de configuration principal :

```
sudo nano /etc/samba/smb.conf
```

Dans ce fichier, nous définissons un partage personnalisé en ajoutant une section comme celle-ci :

```
[PartageEquipe]
 path = /srv/partage_equipe
 browseable = yes
 writabl = yes
 valid users = user1, user2
 create mask = 0660
 directory mask = 0770
```
p p mr

Nous créons ensuite le répertoire partagé et définissons les permissions appropriées :

```
sudo mkdir -p /srv/partage_equipe
sudo chown user1:group_equipe /srv/partage_equipe
sudo chmod 2770 /srv/partage_equipe
```

Nous ajoutons les utilisateurs autorisés à accéder au partage avec la commande :

```
sudo smbpasswd -a user1
```

Après avoir redémarré le service Samba pour appliquer les modifications :

```
sudo systemctl restart smbd
```

Nous testons l'accès au partage depuis d'autres machines pour nous assurer que tout fonctionne correctement. Enfin, nous vérifions que les permissions sont bien respectées et que seuls les utilisateurs autorisés peuvent accéder ou modifier les fichiers.
En configurant ce serveur de fichiers, nous améliorons notre collaboration tout en apprenant à gérer les permissions et les partages réseau sous Linux. Cela renforce nos compétences en administration système et nous prépare à des déploiements plus complexes à l'avenir.

- User Story 52

### Description

En tant qu'équipe, nous voulons documenter chaque étape de la configuration d'un environnement Linux, afin de créer un guide pour de futurs étudiants.

Nous commençons par définir les objectifs du guide : installer un système Linux de base, configurer les utilisateurs et les groupes, gérer les permissions, mettre en place des services essentiels (comme un serveur de fichiers ou un service web), et automatiser des tâches avec des scripts ou `systemd`.

Par exemple, lors de l'installation de Samba pour partager des fichiers, nous documentons :

L'installation du paquet :

```
sudo apt install samba
```

La configuration du fichier smb.conf :

```
[PartageEquipe]
 path = /srv/partage_equipe
 browseable = yes
 writable = yes
```

La gestion des permissions et des utilisateurs :

```
sudo chmod 2770 /srv/partage_equipe
sudo smbpasswd -a user1
```

- User Story 53

### Description

En tant qu'utilisateur, je veux mettre en place une infrastructure Linux de développement proche d'un environnement de production (Base de données et application).

Pour mettre en place une infrastructure Linux de développement proche d'un environnement de production sous Debian, voici les étapes à suivre pour installer et configurer MariaDB, PHP et Nginx.

#### 1. Mise à jour du système

Avant de commencer, assurez-vous que votre système est à jour :

```
sudo apt update && sudo apt upgrade -y
```

#### 2. Installation de MariaDB

MariaDB est un système de gestion de base de données relationnelle. Pour l'installer :

```
sudo apt install mariadb-server -y
```

Une fois installé, sécurisez votre installation MariaDB :

```
sudo mysql_secure_installation
```

Suivez les instructions pour définir un mot de passe root, supprimer les utilisateurs anonymes, désactiver la connexion root à distance et supprimer les bases de données de test.

#### 3. Installation de PHP

PHP est nécessaire pour exécuter des scripts côté serveur. Installez PHP ainsi que les modules couramment utilisés :

```
sudo apt install php-fpm php-mysql php-cli php-curl php-gd php-mbstring php-xml php-zip -y
```

Cela installe PHP-FPM (FastCGI Process Manager), qui est recommandé pour Nginx.

#### 4. Installation de Nginx

Nginx est un serveur web performant. Pour l'installer :

```
sudo apt install nginx -y
```

Une fois installé, démarrez et activez Nginx :

```
sudo systemctl start nginx
sudo systemctl enable nginx
```

### 5. Configuration de Nginx pour PHP

Modifiez la configuration de Nginx pour qu'il puisse traiter les fichiers PHP. Ouvrez un fichier de configuration pour votre site (par exemple, `/etc/nginx/sites-available/mon-site`) :

```
sudo nano /etc/nginx/sites-available/mon-site`
```

Ajoutez la configuration suivante :

```
server {
  listen 80;
  server_name mon-site.local;

  root /var/www/mon-site;
  index index.php index.html index.htm;

  location / {
    try_files $uri $uri/ =404;
  }

  location ~ \.php$ {
    include snippets/fastcgi-php.conf;
    fastcgi_pass unix:/var/run/php/php-fpm.sock;
  }

  location ~ /\.ht {
    deny all;
  }
}
```

Activez le site en créant un lien symbolique :

```
sudo ln -s /etc/nginx/sites-available/mon-site
/etc/nginx/sites-enabled/
```

Testez la configuration Nginx pour vérifier qu'il n'y a pas d'erreurs :

```
sudo nginx -t
```

Si tout est correct, rechargez Nginx :

```
sudo systemctl reload nginx
```

### 6. Création d'un répertoire pour votre site

Créez un répertoire pour héberger les fichiers de votre site :

```
sudo mkdir -p /var/www/mon-site
```

Donnez les permissions appropriées :

```
sudo chown -R www-data:www-data /var/www/mon-site
sudo chmod -R 755 /var/www/mon-site
```

### 7. Test de l'installation

Créez un fichier index.php dans le répertoire de votre site pour tester PHP :

```
sudo nano /var/www/mon-site/index.php
```

Ajoutez le code suivant :

```
<?php
      phpinfo();
?>
```

Ouvrez votre navigateur et accédez à `http://mon-site.local`. Vous devriez voir la page d'information PHP, confirmant que tout fonctionne correctement.

### 8. Sécurisation et optimisation (optionnel)

- Sécuriser Nginx : Configurez un certificat SSL avec Let's Encrypt pour sécuriser votre site.
- Optimiser MariaDB : Ajustez les paramètres de MariaDB pour améliorer les performances en fonction de vos besoins.
- Configurer un pare-feu : Utilisez ufw pour restreindre l'accès aux ports nécessaires.

- User Story 54

Description

En tant qu'utilisateur, je veux tester les configurations mises en place par une autre équipe, afin de valider leur fonctionnement et de proposer des améliorations.

- User Story 55

Description

En tant qu'équipe, nous voulons configurer un système de sauvegarde automatique pour les fichiers critiques, afin de garantir leur sécurité.

Configurer un système de sauvegarde automatique pour vos fichiers critiques est une étape essentielle pour éviter la perte de données importantes. Voici un guide simple pour y parvenir :

1. **Identifiez les fichiers critiques** : Déterminez les fichiers ou dossiers qui nécessitent une sauvegarde régulière. Ils peuvent inclure des documents professionnels, des bases de données ou des fichiers multimédias essentiels.
2. **Choisissez un emplacement de sauvegarde** : Utilisez un disque dur externe, un serveur réseau ou un service cloud (comme Google Drive, OneDrive ou Dropbox). Assurez-vous que cet emplacement est accessible et sécurisé.
3. **Automatisez le processus** : Utilisez un outil ou un script adapté à votre système d'exploitation.
    - Sur Windows, vous pouvez utiliser le "Planificateur de tâches" pour exécuter un script. Un exemple de script en PowerShell pourrait ressembler à ceci : `Copy-Item -Path "C:\Chemin\Vers\Fichiers" -Destination "D:\Sauvegarde" -Recurse`
    - Sur macOS ou Linux, un script basé sur `rsync` peut être utilisé. Par exemple : `rsync -av --delete /chemin/vers/fichiers /chemin/de/sauvegarde`
4. Ajoutez ce script à une tâche cron sur Linux ou à une tâche planifiée sur macOS.
5. **Planifiez la fréquence des sauvegardes** : Décidez si les sauvegardes doivent être effectuées quotidiennement, hebdomadairement ou selon un autre intervalle. Configurez votre outil pour exécuter automatiquement le script à la fréquence souhaitée.
6. **Vérifiez vos sauvegardes** : Testez régulièrement le processus pour vous assurer que les fichiers sont copiés correctement et que les sauvegardes sont utilisables.

7. **Ajoutez des mesures de sécurité** : Chiffrez vos sauvegardes pour protéger les données sensibles et mettez en place une stratégie de gestion des versions pour conserver plusieurs copies en cas de besoin.

En suivant ces étapes, vous aurez un système fiable pour protéger vos fichiers critiques contre les pertes imprévues.

- User Story 56

Description

En tant qu'étudiant, je veux documenter les étapes de configuration d'un système multi-utilisateur, afin de produire une documentation claire pour de futurs projets.

Configurer un système multi-utilisateur sur une machine virtuelle VirtualBox exécutant Linux nécessite plusieurs étapes simples. Voici un guide pour y parvenir :

1. **Installer Linux sur la machine virtuelle** : Si ce n'est pas encore fait, créez une machine virtuelle dans VirtualBox et installez une distribution Linux (comme Ubuntu, Debian, ou CentOS). Suivez l'assistant d'installation pour configurer le système de base.
2. **Créer des utilisateurs** :
   - Connectez-vous à la machine virtuelle avec un compte administrateur (généralement `root` ou le premier utilisateur créé).
   - Utilisez la commande `sudo adduser nom_utilisateur` pour créer un nouvel utilisateur. Suivez les instructions pour définir un mot de passe et remplir les informations supplémentaires.
3. **Attribuer des privilèges aux utilisateurs** :
   - Si un utilisateur a besoin de droits administratifs, ajoutez-le au groupe `sudo` (ou équivalent selon la distribution) avec la commande `sudo usermod -aG sudo nom_utilisateur`.
   - Les utilisateurs sans privilèges administratifs auront des permissions limitées par défaut.
4. **Configurer les répertoires personnels** :
   - Lors de la création d'un utilisateur, un répertoire personnel (`/home/nom_utilisateur`) est automatiquement généré. Ce répertoire est uniquement accessible par l'utilisateur et l'administrateur.

   - Vous pouvez ajuster les permissions avec la commande `chmod` si nécessaire. Par exemple, `chmod 700 /home/nom_utilisateur` garantit que seuls l'utilisateur et `root` peuvent accéder au répertoire.
5. **Configurer des groupes pour le partage de fichiers** :
   - Pour permettre à plusieurs utilisateurs de collaborer, créez un groupe partagé avec `sudo groupadd nom_groupe`.
   - Ajoutez des utilisateurs à ce groupe avec `sudo usermod -aG nom_groupe nom_utilisateur`.
   - Définissez les permissions sur les répertoires partagés avec `chmod` et `chgrp` pour que les membres du groupe puissent accéder et modifier les fichiers.
6. **Tester les connexions utilisateurs** :
   - Utilisez `su - nom_utilisateur` pour basculer vers un autre utilisateur et vérifier que les permissions et les répertoires sont correctement configurés.
   - Les utilisateurs peuvent se connecter à la machine via SSH si le service SSH est configuré.
7. **Gérer les politiques de sécurité** :
   - Activez le pare-feu avec `ufw` ou une autre solution adaptée à votre distribution.
   - Vérifiez les paramètres de mot de passe avec des outils comme `passwd -l nom_utilisateur` pour verrouiller ou modifier les mots de passe.

Ces étapes permettent de configurer un environnement multi-utilisateur fonctionnel et sécurisé sur une machine virtuelle VirtualBox exécutant Linux.

- User Story 57

Description

En tant qu'utilisateur, je veux intégrer une tâche cron pour automatiser une tâche régulière (ex. nettoyage de répertoires temporaires), afin de mieux gérer le système.

Objectif

Créer, utiliser et supprimer une tâche cron pour automatiser une tâche régulière, comme le nettoyage de répertoires temporaires.

---

Étape 1 : Création d'une tâche cron

1. Ouvrir le crontab
   Ouvrez le fichier crontab pour l'utilisateur courant avec la commande suivante :

   ```
   crontab -e
   ```

2. Ajouter une nouvelle tâche

   Dans l'éditeur, ajoutez une ligne pour définir la tâche cron. La syntaxe est la suivante :

   ```
   * * * * * /chemin/vers/le/script.sh
   ```

3. Les 5 étoiles représentent respectivement :
   - Minute (0-59)
   - Heure (0-23)
   - Jour du mois (1-31)
   - Mois (1-12)
   - Jour de la semaine (0-7, où 0 et 7 = dimanche)
   - Remplacez `/chemin/vers/le/script.sh` par le chemin absolu de votre script ou commande.

   Exemple :

   Pour exécuter un script de nettoyage tous les jours à minuit :

   ```
   0 0 * * * /usr/local/bin/nettoyage_tmp.sh
   ```

   Sauvegardez le fichier et quittez l'éditeur. La tâche cron est maintenant active.

---

Étape 2 : Utilisation de la tâche cron

1. Vérifier l'exécution

   Pour vérifier que la tâche s'exécute correctement :
   - Consultez les logs cron (généralement dans `/var/log/syslog` ou `/var/log/cron`).
   - Ajoutez des logs dans votre script pour suivre son exécution.

2. Tester manuellement

Vous pouvez exécuter manuellement le script pour vérifier son bon fonctionnement :

```
/chemin/vers/le/script.sh
```

---

### Étape 3 : Suppression d'une tâche cron

Ouvrir le crontab
Ouvrez à nouveau le fichier crontab :

```
crontab -e
```

Localisez la ligne correspondant à la tâche que vous souhaitez supprimer et effacez-la.

Sauvegardez le fichier et quittez l'éditeur. La tâche cron est maintenant supprimée.

---

### Astuces supplémentaires

- Lister les tâches cron actuelles
  Pour afficher les tâches cron de l'utilisateur courant :

  ```
  crontab -l
  ```

- Utiliser des variables d'environnement
  Vous pouvez définir des variables d'environnement dans le crontab pour personnaliser l'exécution des tâches.
- Rediriger les sorties
  Pour capturer les sorties (`stdout` et `stderr`) de votre script, redirigez-les vers un fichier log:

  ```
  0 0 * * * /usr/local/bin/nettoyage_tmp.sh >>
  /var/log/nettoyage.log 2>&1
  ```

---

## Conclusion

En suivant ces étapes, vous pouvez facilement créer, utiliser et supprimer des tâches `cron` pour automatiser des tâches régulières sur votre système.

- User Story 58

### Description

En tant qu'équipe, nous voulons tester la configuration d'Pour tester la configuration d'un pare-feu basique, comme **ufw** ou **iptables**, voici les étapes à suivre :

1. **Vérifiez l'état du pare-feu** :
   - Assurez-vous que le pare-feu est activé. Pour **ufw**, utilisez la commande `sudo ufw status` pour vérifier les règles actives. Avec **iptables**, utilisez `sudo iptables -L` pour afficher les chaînes et leurs règles.
2. **Testez les règles en place** :
   - Identifiez les ports ou services autorisés ou bloqués par le pare-feu.
   - Par exemple, si le pare-feu autorise les connexions sur le port 22 (SSH), essayez de vous connecter à distance via SSH depuis une autre machine. Si la connexion est réussie, la règle fonctionne.
   - Si un port est bloqué (comme le port 80 pour HTTP), tentez d'y accéder depuis un navigateur ou avec un outil comme `curl`. Une connexion refusée ou un délai d'expiration confirme que la règle de blocage fonctionne.
3. **Simulez des connexions non autorisées** :
   - Utilisez des outils comme `telnet` ou `nmap` pour scanner les ports ouverts de votre machine. Par exemple, un scan devrait ne montrer que les ports explicitement autorisés par vos règles.
4. **Testez les règles spécifiques** :
   - Ajoutez temporairement une règle pour autoriser ou bloquer une adresse IP ou un sous-réseau, puis testez depuis une machine ayant cette adresse. Par exemple, bloquez une IP spécifique et essayez d'établir une connexion depuis cette machine pour vérifier que l'accès est refusé.
5. **Vérifiez les journaux** :
   - Activez la journalisation des événements dans **ufw** ou **iptables** pour suivre les connexions autorisées et bloquées. Pour **ufw**, activez la journalisation avec `sudo ufw logging on`, puis examinez les journaux avec `sudo less /var/log/ufw.log`.
   - Les entrées du journal montrent si les paquets sont acceptés ou rejetés conformément aux règles définies.
6. **Corrigez et renforcez les règles si nécessaire** :
   - Si vous détectez un comportement inattendu (comme un port ouvert qui devrait être bloqué), ajustez les règles avec des commandes appropriées. Par exemple, ajoutez une règle pour refuser tout trafic entrant sauf les services essentiels.

En testant méthodiquement chaque aspect de la configuration, vous pouvez valider que le pare-feu protège efficacement le système tout en permettant les connexions nécessaires.

un pare-feu
basique (ufw ou iptables), pour sécuriser le système.

- User Story 59

### Description

En tant qu'étudiant, je veux intégrer un outil de supervision comme `htop` ou `glances` dans mon projet, pour surveiller les performances en temps réel.

Intégrer un outil de supervision comme `htop` ou `glances` pour surveiller les performances système en temps réel dans le cadre d'un projet étudiant.

---

### Étapes d'intégration

1. Choix de l'outil
   - `htop` : Interface en terminal pour surveiller les processus, l'utilisation du CPU, de la mémoire, etc.
   - `glances` : Outil plus complet avec une interface web optionnelle, surveillant CPU, mémoire, disque, réseau, etc.
2. Installation
   - Pour htop :

     ```
     sudo apt-get install htop  # Sur Debian/Ubuntu
     ```

   - Pour glances :

     ```
     sudo apt-get install glances  # Sur Debian/Ubuntu
     ```
3. Utilisation

   Lancer dans le terminal :

   `htop`
   Utilisez les flèches pour naviguer et F10 pour quitter.

   `glances`

Pour l'interface web, utilisez :

```
glances -w
```

Accédez à `http://<adresse-ip>:61208` dans un navigateur.

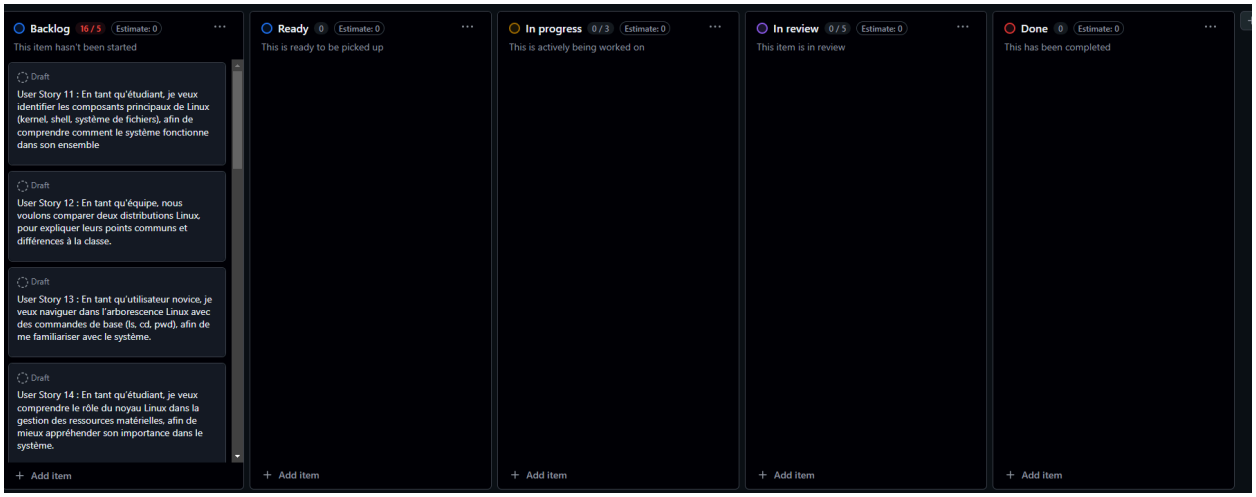
4. Intégration dans le projet
   - Ajoutez des commandes pour lancer `htop` ou `glances` dans vos scripts ou documentation.
   - Si nécessaire, capturez des métriques spécifiques via des commandes personnalisées (ex : `glances --export csv` pour exporter les données).

## Pour l'organisation et le suivi

- User Story 61

#### Description

En tant qu'équipe, nous voulons tenir un tableau Kanban de nos tâches, afin de suivre notre progression et rester organisés.




- User Story 62

#### Description

En tant qu'étudiant, je veux participer à des rétrospectives après chaque sprint, afin d'identifier les points à améliorer pour les prochains travaux.