Ubuntu

Ubuntu est l'une des distributions Linux les plus populaires et accessibles, conçue pour offrir une expérience conviviale tout en étant puissante. Lancée en 2004 par la société Canonical, Ubuntu repose sur les bases solides de Debian tout en mettant l'accent sur la facilité d'utilisation, la sécurité et la stabilité. Le nom "Ubuntu" provient d'un ancien concept africain signifiant "humanité envers les autres", reflétant l'engagement de cette distribution à créer une communauté inclusive et ouverte.

Ubuntu se distingue par son interface utilisateur intuitive, propulsée par l'environnement de bureau GNOME, et sa vaste bibliothèque de logiciels facilement installables via son gestionnaire de paquets. Il est conçu pour convenir aussi bien aux débutants qu'aux utilisateurs expérimentés, avec des mises à jour régulières qui intègrent les dernières innovations tout en garantissant une stabilité à long terme. Les versions LTS (Long Term Support) d'Ubuntu offrent un support étendu, ce qui en fait un choix idéal pour les environnements de production et les utilisateurs qui recherchent une plateforme fiable.

Que ce soit pour un usage personnel, des projets de développement ou pour des serveurs, Ubuntu simplifie l'expérience Linux tout en maintenant la puissance et la flexibilité attendues d'un système open source. Sa large adoption par des entreprises, des institutions académiques et des particuliers témoigne de sa capacité à répondre à une grande variété de besoins technologiques.

Commandes et Config

Les **commandes Linux** constituent l'essence même de l'interaction avec le système d'exploitation à travers un terminal ou une interface en ligne de commande (CLI). Elles permettent d'exécuter une multitude de tâches, allant de la navigation dans le système de fichiers à la gestion des processus, en passant par la configuration réseau et la gestion des utilisateurs.

Le système Linux repose fortement sur la philosophie UNIX, où chaque commande effectue une tâche spécifique. Ces commandes peuvent être combinées en utilisant des pipes et des redirections pour créer des scripts puissants et automatiser des processus complexes. Les utilisateurs de Linux apprécient la polyvalence et la puissance des commandes, qui offrent un contrôle total sur le système.

Qu'il s'agisse de simples commandes comme ls pour lister les fichiers ou de commandes plus complexes comme grep pour rechercher du texte à travers plusieurs fichiers, maîtriser les commandes Linux est une compétence précieuse pour les administrateurs système, les développeurs et les utilisateurs expérimentés.

Commandes basique

Quelques commandes essentielles pour débuter :

- **`ls`** : Affiche la liste des fichiers et répertoires dans un répertoire donné. Exemples :
  - `ls` : Liste les fichiers dans le répertoire courant.
  - `ls -l` : Liste les fichiers avec des détails comme les permissions, la taille et la date de modification.
- **`cd`** : Change le répertoire courant. Exemples :
  - `cd /home/user/Documents` : Se déplace vers le répertoire `Documents`.
  - `cd ..` : Revient au répertoire parent.
- **`cp`** **et** **`mv`** : Copie ou déplace des fichiers/répertoires.
  - `cp file1.txt /home/user/` : Copie `file1.txt` vers le répertoire `/home/user/`.
  - `mv file1.txt /home/user/` : Déplace `file1.txt` vers le répertoire `/home/user/`.
- **`rm`** : Supprime des fichiers ou des répertoires. Attention à l'utilisation car cette commande ne place pas les fichiers dans une corbeille.
  - `rm file1.txt` : Supprime le fichier `file1.txt`.
  - `rm -r /home/user/dir` : Supprime récursivement le répertoire `dir` et son contenu.
- **`chmod`** **et** **`chown`** : Modifie les permissions et le propriétaire d'un fichier ou d'un répertoire.
  - `chmod 755 script.sh` : Donne les permissions de lecture, écriture et exécution au propriétaire, et les permissions de lecture et exécution aux autres.
  - `chown user:user file1.txt` : Change le propriétaire et le groupe de `file1.txt`.
- **`ps`** **et** **`top`** : Affiche les processus en cours.
  - `ps aux` : Affiche tous les processus en cours avec des informations détaillées.
  - `top` : Affiche les processus actifs avec une mise à jour en temps réel.
- **`grep`** : Recherche des chaînes de caractères dans des fichiers.
  - `grep 'mot' fichier.txt` : Recherche toutes les occurrences de `mot` dans `fichier.txt`.
  - `grep -r 'mot' /etc/` : Recherche récursivement `mot` dans tous les fichiers du répertoire `/etc/`.
- **`find`** : Recherche des fichiers dans un répertoire donné.
  - `find /home/user/ -name "*.txt"` : Recherche tous les fichiers `.txt` dans le répertoire `/home/user/`.
- **`tar`** : Archive et compresse des fichiers.
  - `tar -cvf archive.tar /home/user/dir` : Crée une archive `tar` de `dir`.
  - `tar -xvf archive.tar` : Décompresse l'archive `tar`.
- **`ssh`** : Se connecte à un serveur distant en utilisant le protocole SSH.

    - `ssh user@ip_serveur` : Se connecte en SSH au serveur avec l'adresse `ip_serveur`.
- **`sudo`** : Exécute des commandes avec des privilèges administratifs.
    - `sudo apt update` : Exécute la commande de mise à jour des paquets avec les droits d'administration.

Configuration de base pour la gestion des commandes :

1. **Bashrc et Aliases :**
    - **`~/.bashrc`** : Le fichier de configuration des commandes du terminal Bash où tu peux définir des alias pour simplifier les commandes répétitives. Par exemple :
      bash
      ```
      alias ll='ls -alF'
      alias grep='grep --color=auto'
      ```
      Ces alias te permettent de taper `ll` au lieu de `ls -alF` pour voir les fichiers détaillés, ou de colorer automatiquement les résultats de `grep`.
2. **Gestionnaire de paquets :**
    - Selon ta distribution, tu utiliseras des commandes comme `apt` (Debian/Ubuntu) ou `pacman` (Arch) pour gérer l'installation, la mise à jour et la suppression de logiciels. Par exemple :
        - Sur Ubuntu : `sudo apt install package_name`
        - Sur Arch : `sudo pacman -S package_name`
3. **Scripting Bash :**
    - L’écriture de scripts Bash permet d'automatiser des tâches courantes. Par exemple, un script simple pour sauvegarder un répertoire pourrait ressembler à ceci :
      bash
      ```
      #!/bin/bash
      tar -czvf backup.tar.gz /home/user/dossier
      echo "Sauvegarde terminée."
      ```

UFW (firewall)

UFW (Uncomplicated Firewall)

UFW (Uncomplicated Firewall) est un outil de gestion de pare-feu conçu pour simplifier l'utilisation de `iptables`, le pare-feu sous-jacent de Linux. Créé pour être convivial, UFW permet aux utilisateurs, même ceux qui n'ont pas une grande expérience avec les systèmes de pare-feu, de configurer des règles de sécurité réseau rapidement et efficacement. Il est particulièrement populaire sur des distributions comme Ubuntu.

UFW gère l'accès aux ports réseau en autorisant ou bloquant le trafic entrant et sortant, offrant ainsi une couche de protection essentielle contre les attaques extérieures. Sa simplicité d'utilisation, combinée à sa puissance, en fait un outil idéal pour sécuriser les serveurs, les ordinateurs personnels ou les réseaux domestiques.

Avec une syntaxe intuitive et des commandes simples, UFW permet d'appliquer des règles de pare-feu courantes en quelques étapes seulement, tout en offrant la flexibilité nécessaire pour gérer des configurations réseau plus complexes.

Quelques commandes essentielles pour UFW :

- **Installation (si nécessaire) :**
  - Sur Ubuntu :

    ```
    sudo apt install ufw
    ```

  - Sur Arch Linux :

    ```
    sudo pacman -S ufw
    ```

- **Activer et désactiver UFW :**
  - Active le pare-feu UFW :

    ```
    sudo ufw enable
    ```

  - Désactive le pare-feu UFW :

```
sudo ufw disable
```

- **Vérifier l'état de UFW :**
  - Affiche l'état actuel du pare-feu et les règles actives :

```
sudo ufw status
```

  - Affiche l'état avec plus de détails (par exemple, si le pare-feu est actif ou inactif) :

```
sudo ufw status verbose
```

- **Autoriser et bloquer des connexions :**
  - Autorise les connexions entrantes sur le port 22 (par exemple, pour SSH) :

```
sudo ufw allow 22
```

  - Bloque les connexions entrantes sur le port 80 (HTTP) :

```
sudo ufw deny 80
```

  - Autorise le trafic entrant en provenance de l'adresse IP `192.168.1.100` :

```
sudo ufw allow from 192.168.1.100
```

  - Bloque tout trafic provenant de l'adresse IP `203.0.113.1`:

```
sudo ufw deny from 203.0.113.1
```

- **Autoriser ou bloquer des connexions pour des services spécifiques :**
  - Autorise les connexions SSH (équivalent à `sudo ufw allow 22`) :

```
sudo ufw allow ssh
```

  - Autorise le trafic HTTP (port 80) :

```
sudo ufw allow http
```

  - Autorise le trafic HTTPS (port 443) :

```
sudo ufw allow https
```

- **Supprimer des règles :**
  - Supprime la règle qui autorise les connexions sur le port 22 :

```
sudo ufw delete allow 22
```

  - Supprime la règle qui bloque les connexions sur le port 80 :

```
sudo ufw delete deny 80
```

- **Limiter les tentatives de connexion pour prévenir les attaques bruteforce :**
  - Limite les connexions SSH pour prévenir les tentatives répétées d'attaques bruteforce :

```
sudo ufw limit ssh
```

- **Gérer les règles par IP ou sous-réseau :**
  - Autorise les connexions SSH depuis un sous-réseau spécifique (`192.168.1.0/24`) :

```
sudo ufw allow from 192.168.1.0/24 to any port 22
```

- **Réinitialiser UFW :**
  - Réinitialise UFW à sa configuration par défaut, supprimant toutes les règles existantes :

```
sudo ufw reset
```

Configuration de base pour UFW :

1. **Activer UFW et autoriser les connexions SSH :**
   - Avant d'activer UFW sur un serveur distant, assure-toi de permettre les connexions SSH, sinon tu pourrais te bloquer toi-même :

```
sudo ufw allow ssh sudo ufw enable
```

2. **Configurer les règles de pare-feu pour un serveur web :**
   - Pour un serveur web typique, tu pourrais vouloir autoriser le trafic HTTP et HTTPS, tout en bloquant tout le reste :

```
sudo ufw allow http sudo ufw allow https sudo ufw enable
```

3. **Limiter les connexions SSH pour plus de sécurité :**
   - Pour protéger ton serveur des attaques bruteforce SSH :

```
sudo ufw limit ssh
```

4. **Autoriser les connexions depuis un réseau local seulement :**
   - Si tu veux autoriser les connexions à un service seulement depuis un réseau local spécifique :

```
sudo ufw allow from 192.168.1.0/24 to any port 3306
```

   - Cela autorise les connexions à MySQL (port 3306) uniquement depuis le sous-réseau `192.168.1.0/24`.
5. **Afficher et gérer les règles :**
   - Après avoir configuré tes règles, tu peux toujours vérifier l'état actuel avec :

```
sudo ufw status
```