Linux Debian - Administration

m2iformation.fr

Sommaire

Jour 1

Sommaire

Jour 2

Sommaire

Jour 3

- Description du processus de démarrage
- Gestionnaires de boot GRUB
- Gestion des unités service et cible (target) de systemd
- Gestion des services SysVinit
- Dépannage en mode Rescue ou Emergency
- Définition des processus, des threads et de l'ordonnancement
- Analyse de l'activité système (top, htop, pstree, ps...)
- Gestion des signaux (kill)
- Tâches avant et arrière plans (jobs, bg, fg...)
- Planification de tâches

Sommaire

Jour 4

- Compression et archivage
  - Compresser et décompresser un fichiers (gzip, bzip2, lzma, lzw)
  - Gestion d'une archive
  - Sans compression (tar, pax)
  - Avec compression (tar, pax)
- Présentation des dépôts Debian
  - Gestion des paquetages DEB (aptitude, apt, dpkg...)
  - Installation d'une application depuis une archive
  - Compilation et installation à partir de sources
- Gestion des sources
  - Etude avancée de la commande APT
  - Apt-get
  - Apt-file
  - Apt-rdepend.
- Gestion des journaux avec rsyslog et/ou systemd-journald

Sommaire

Jour 5

- Les fondamentaux du réseau
  - Configuration du client réseau
  - Démarrage et arrêt du firewall
  - Outils de diagnostic
- Gestion du service de temps
  - Réglages de la date et de l'heure
  - Paramétrer le client NTP
- Administration à distance
  - Configuration du serveur et du client OpenSSH
  - Copie et transfert de fichiers sécurisés via SSH

UTOPIOS®
TRANSFORMATION DIGITALE INTELLIGENTE

Jour 1

Présentation de l'organisation Debian

1. **Histoire et Origine de Debian**
   Debian a été créé par Ian Murdock en 1993 comme une distribution Linux non commerciale et ouverte, avec un accent particulier sur la communauté. Il visait à combiner les meilleurs aspects des distributions précédentes tout en maintenant un processus de développement ouvert et collaboratif. Le nom "Debian" provient de la combinaison du prénom de son fondateur, Ian, et de celui de sa petite amie de l'époque, Debra.

2. **Structure Organisationnelle de Debian**
   Debian est organisé de manière unique pour une distribution de logiciels. Contrairement à certaines autres distributions, elle n'est pas pilotée par une entreprise mais par des développeurs bénévoles à travers le monde. Ces bénévoles sont regroupés sous plusieurs entités :
   - **Les développeurs Debian (Debian Developers)** : Ils sont officiellement responsables des paquets logiciels dans Debian. Chaque développeur a des droits pour maintenir certains paquets, proposer des modifications et voter sur des questions importantes. Il existe un processus rigoureux pour devenir développeur officiel, ce qui inclut une révision des compétences techniques et de la compréhension des principes Debian.

Présentation de l'organisation Debian

- **Le Project Leader Debian (DPL)** : Élu chaque année par les développeurs Debian, le DPL est le représentant officiel de Debian. Le leader n'a pas d'autorité absolue, mais il est chargé de faciliter le développement et la communication au sein de la communauté.
- **Les équipes spécialisées (Debian Teams)** : Debian est divisé en plusieurs équipes pour différents aspects du système, comme la sécurité, l'installation, la documentation, etc. Chaque équipe se concentre sur une partie spécifique de la distribution.
- **Le comité technique (Technical Committee)** : Il s'agit d'une entité qui aide à résoudre les désaccords techniques entre les développeurs Debian lorsque ces derniers ne peuvent pas parvenir à une solution consensuelle.

Présentation de l'organisation Debian

3. **Le contrat social Debian**
   Debian se distingue par son engagement envers ses utilisateurs et ses contributeurs, qui est formalisé dans le "Contrat Social Debian". Ce document est un ensemble de cinq principes qui guident le développement de Debian :
   - **Debian restera 100% libre** : Debian s'engage à ne distribuer que des logiciels conformes aux principes du logiciel libre. Cela implique que tout le code source est disponible et que chacun a le droit d'étudier, de modifier et de redistribuer les logiciels.
   - **Nous ne cacherons pas les problèmes** : Les développeurs Debian s'engagent à toujours communiquer ouvertement et honnêtement sur les bugs et les problèmes de sécurité.
   - **Nos priorités sont nos utilisateurs et les logiciels libres** : Debian se concentre sur la fourniture de la meilleure expérience pour ses utilisateurs tout en maintenant l'intégrité du logiciel libre.
   - **Les travaux qui ne respectent pas nos standards de liberté peuvent être soutenus** : Debian peut fournir des outils pour permettre à ses utilisateurs de travailler avec des logiciels non libres si nécessaire, mais ces logiciels ne sont pas inclus dans les distributions officielles.
   - **Les composants non libres de notre système d'exploitation seront clairement séparés** : Le logiciel non libre est clairement séparé dans les dépôts “contrib” et “non-free” pour que les utilisateurs puissent choisir ce qu'ils installent.

Présentation de l'organisation Debian

4. **La structure des dépôts Debian**
   Les logiciels de Debian sont stockés dans différents dépôts selon leur niveau de liberté :
   - **Main** : Ce dépôt contient uniquement des logiciels qui respectent complètement les directives de logiciels libres de Debian (DFSG - Debian Free Software Guidelines).
   - **Contrib** : Ce dépôt contient des logiciels libres, mais qui dépendent de composants non libres pour fonctionner.
   - **Non-Free** : Ce dépôt contient des logiciels qui ne respectent pas les DFSG, mais qui peuvent être utiles aux utilisateurs.
5. **La gouvernance communautaire**
   La communauté Debian est basée sur le volontariat et suit un modèle démocratique. Les décisions importantes sont souvent prises par consensus, et lorsqu'il y a désaccord, un vote peut être organisé. Tous les développeurs ont le droit de vote, et les décisions sont prises en tenant compte des opinions majoritaires.
6. **La diversité des architectures supportées**
   Debian est réputé pour son support de nombreuses architectures matérielles. Au-delà des systèmes courants (x86, AMD64), Debian prend également en charge des architectures plus spécialisées comme ARM, MIPS, et RISC-V. Cela permet à Debian d'être utilisé sur une large gamme d'appareils, des ordinateurs personnels aux serveurs et systèmes embarqués.

Présentation de l'organisation Debian

7. **Le rôle de la sécurité dans Debian**
   Debian possède une équipe dédiée à la sécurité qui est responsable de la gestion des correctifs de sécurité pour les versions stables. Les mises à jour de sécurité sont publiées via les canaux de mise à jour Debian, garantissant que les systèmes utilisant Debian stable sont protégés contre les vulnérabilités.

8. **Les distributions dérivées de Debian**
   Debian est la base de nombreuses autres distributions Linux, notamment Ubuntu, Knoppix, et Raspbian. Ces distributions dérivées bénéficient du travail des développeurs Debian, tout en apportant leurs propres personnalisations pour répondre à des besoins spécifiques

Différentes versions de Debian : Stable, Testing, Unstable (Sid)

Debian se distingue par son modèle de publication unique basé sur trois branches principales : **Stable**, **Testing**, et **Unstable** (Sid). Chacune de ces branches sert un objectif spécifique dans le cycle de développement et de publication de Debian.

1. **Debian Stable**
   Debian Stable est la version la plus largement utilisée et recommandée pour les utilisateurs finaux, en particulier ceux qui recherchent un environnement fiable et sécurisé pour des serveurs ou des postes de travail.
   - **Objectif** : Fournir une distribution stable, bien testée et sécurisée.
   - **Fréquence de publication** : Debian Stable est publiée environ tous les 2 à 3 ans. Chaque nouvelle version stable reçoit un nom de code basé sur un personnage de la série de films "Toy Story". Par exemple, la version stable actuelle pourrait être nommée “Debian Bullseye” (Debian 11) ou “Debian Bookworm” (Debian 12).

Différentes versions de Debian : Stable, Testing, Unstable (Sid)

- **Caractéristiques principales** :
  - **Solidité et stabilité** : La priorité est donnée à la fiabilité des logiciels. Une fois qu'une version entre en phase Stable, elle reçoit uniquement des mises à jour de sécurité et des corrections de bugs critiques. Aucune nouvelle fonctionnalité n'est ajoutée après la publication.
  - **Sécurité** : Les correctifs de sécurité sont gérés par l'équipe Debian Security. Cela garantit que les systèmes utilisant Debian Stable restent protégés contre les vulnérabilités.
  - **Paquets anciens mais stables** : Les logiciels inclus dans Debian Stable sont généralement plus anciens car ils ont subi un long processus de test. Cela garantit une compatibilité et une stabilité éprouvées, mais cela signifie également que certaines applications peuvent être en retard par rapport à leurs dernières versions.

Différentes versions de Debian : Stable, Testing, Unstable (Sid)

2. **Debian Testing**
   Debian Testing est la branche où les futures versions stables de Debian sont préparées. Elle est un compromis entre la stabilité de la branche Stable et la nouveauté des logiciels présents dans la branche Unstable.
   - **Objectif** : Préparer la prochaine version Stable en testant de nouvelles fonctionnalités et en assurant la stabilité.
   - **Fréquence de mise à jour** : Debian Testing reçoit des mises à jour régulières, souvent plusieurs fois par semaine, lorsque les paquets ont passé avec succès une phase initiale dans la branche Unstable.
   - **Caractéristiques principales** :
     - **Mises à jour fréquentes** : Les paquets logiciels sont plus récents que dans Debian Stable, car ils sont introduits en Testing après avoir passé un certain temps en Unstable. Cela permet d'avoir des logiciels plus modernes, tout en conservant une certaine stabilité.
     - **Processus de gel (Freeze)** : Avant la sortie d'une nouvelle version Stable, Debian Testing entre dans une période de gel où seuls les correctifs de bugs sont autorisés. Cela permet de stabiliser la distribution avant la prochaine version Stable.
     - **Paquets plus récents** : Les logiciels dans Debian Testing sont souvent plus récents que ceux de la branche Stable, mais ils peuvent encore contenir des bugs, bien que moins nombreux qu'en Unstable.

Différentes versions de Debian : Stable, Testing, Unstable (Sid)

3. **Debian Unstable (Sid)**
   Debian Unstable, également connue sous le nom de "Sid", est la branche de développement active de Debian. C'est ici que tous les nouveaux paquets sont introduits avant d'être testés pour leur stabilité.
   - **Objectif** : Tester et développer de nouveaux paquets pour Debian.
   - **Fréquence de mise à jour** : Les mises à jour dans Debian Sid sont constantes et peuvent arriver quotidiennement. C'est la version la plus "vivante" de Debian, mais aussi la plus sujette aux bugs.
   - **Caractéristiques principales** :
     - **Logiciels les plus récents** : Sid est la première branche où de nouvelles versions de logiciels sont introduites. Cela signifie que les utilisateurs de Sid obtiennent les derniers logiciels presque immédiatement après leur sortie.
     - **Moins de tests** : Les paquets dans Unstable n'ont pas encore subi les tests rigoureux des autres branches. Il peut y avoir des bugs, des problèmes de dépendances ou d'instabilité générale.
     - **Peut être instable** : Comme son nom l'indique, Debian Unstable n'est pas conçue pour une utilisation dans des environnements critiques ou de production. Les utilisateurs peuvent rencontrer des erreurs majeures ou des défaillances système en raison de l'introduction de paquets non encore stabilisés.

Différentes versions de Debian : Stable, Testing, Unstable (Sid)

4. **Le processus de migration des paquets**
   Voici comment un paquet migre à travers les différentes branches :
   - **Ajout dans Unstable (Sid)** : Les nouveaux paquets ou les nouvelles versions des paquets existants sont d'abord introduits dans Unstable. Ces paquets doivent satisfaire à certaines exigences minimales en termes de qualité.
   - **Transition vers Testing** : Si un paquet reste dans Unstable sans bugs critiques pendant 10 à 15 jours, il peut migrer vers Testing. Cependant, cette migration peut être retardée si des dépendances importantes n'ont pas encore migré ou si des problèmes sont découverts.
   - **Publication dans Stable** : Lorsqu'une version de Testing est jugée suffisamment stable, elle est "gelée". Pendant la période de gel, seuls les correctifs de sécurité et les corrections de bugs critiques sont appliqués. Après cette phase de gel, une nouvelle version stable de Debian est publiée.

Préparation à l'installation de Debian

La phase de préparation à l'installation de Debian est une étape essentielle qui garantit que le système sera installé de manière optimale en fonction des besoins de l'utilisateur ou de l'infrastructure. Cela inclut la vérification du matériel, le choix de la méthode d'installation, la gestion du partitionnement des disques, et la sélection des composants logiciels nécessaires.

1. **Choix de la version adaptée de Debian**
   La première décision à prendre avant d'installer Debian concerne le choix de la version la plus adaptée à l'utilisation prévue du système. Debian propose trois principales branches.

Le choix de la version doit être fait en fonction de la stabilité requise pour le système. Pour un serveur en production, Debian Stable est souvent privilégiée. Pour un environnement de développement ou de test, Debian Testing ou Sid peut être plus adaptée.

Préparation à l'installation de Debian

2. **Exigences matérielles**
   Avant d'installer Debian, il est crucial de s'assurer que le matériel de la machine est compatible avec la distribution. Debian est célèbre pour sa compatibilité avec une large gamme d'architectures matérielles, allant des systèmes les plus courants (x86_64, ARM) aux architectures spécialisées (MIPS, RISC-V).

- **Processeur** : Debian supporte les architectures 32 bits (i386) et 64 bits (amd64). Il est recommandé d'utiliser la version 64 bits pour les machines récentes, car elle permet de mieux exploiter les capacités du matériel.
- **Mémoire vive (RAM)** : La quantité de mémoire nécessaire dépend du type d'installation. Une installation minimale avec une interface en ligne de commande nécessite environ 512 Mo de RAM, tandis qu'une installation avec un environnement graphique complet nécessite au moins 2 Go de RAM pour un fonctionnement fluide.
- **Espace disque** : L'espace disque nécessaire varie en fonction des paquets à installer. Une installation de base sans environnement graphique peut utiliser environ 2 Go de disque, tandis qu'une installation avec un environnement de bureau complet et de nombreux paquets logiciels peut nécessiter au moins 10 à 15 Go.

Debian offre un support matériel très large, mais certains composants propriétaires (comme les cartes graphiques Nvidia ou certains chipsets Wi-Fi) peuvent nécessiter l'installation de pilotes propriétaires ou l'activation de dépôts non libres.

Préparation à l'installation de Debian

3. **Choix de la méthode d'installation**
   Il existe plusieurs façons d'installer Debian, et le choix de la méthode dépendra de l'accès réseau, des préférences de l'utilisateur et de l'infrastructure disponible.

- **Netinstall (Installation par réseau)** : Cette méthode utilise une petite image ISO qui télécharge les paquets nécessaires à partir d'un serveur Debian au cours de l'installation. C'est l'option idéale si vous avez une connexion Internet rapide et stable. Elle permet de ne télécharger que les paquets nécessaires, ce qui peut réduire la taille initiale de l'image et éviter les installations de paquets inutiles.
- **Images complètes (DVD ou CD)** : Ces images contiennent une grande partie, voire la totalité, des paquets disponibles dans Debian. Elles sont pratiques pour une installation hors ligne ou sur des machines sans connexion Internet stable. Une installation complète peut nécessiter plusieurs DVD ou un disque USB de grande capacité.
- **Preseed (Installation automatisée)** : Debian permet des installations automatisées grâce à des fichiers preseed. Cela permet de définir à l'avance les choix à faire pendant l'installation, comme le partitionnement des disques ou la configuration réseau, rendant le processus d'installation non interactif. Cette méthode est souvent utilisée pour des déploiements en masse dans les centres de données ou pour des configurations répétitives.

Préparation à l'installation de Debian

4. **Partitionnement des disques**
   Le partitionnement est une étape clé de l'installation de Debian, car il permet d'organiser et d'optimiser l'utilisation de l'espace disque.

- **Partitionnement automatique** : Lors de l'installation, Debian propose un partitionnement automatique qui est généralement adapté pour les nouveaux utilisateurs. Il permet de créer automatiquement les partitions nécessaires, comme la partition racine (`/`), la partition utilisateur (`/home`), et une partition de swap (utilisée pour la gestion de la mémoire virtuelle).
- **Partitionnement manuel** : Les utilisateurs avancés peuvent préférer créer manuellement leurs partitions pour mieux gérer l'espace disque en fonction de leurs besoins spécifiques. Par exemple, une partition séparée pour `/var` ou `/tmp` peut améliorer la gestion des journaux ou des fichiers temporaires. Le partitionnement manuel permet également d'utiliser des systèmes de fichiers spécifiques, comme `ext4`, `xfs`, ou `btrfs`.
- **Gestion du swap** : Le swap est une partition ou un fichier qui agit comme une extension de la mémoire vive (RAM). Le swap est important pour la gestion des processus gourmands en mémoire. En général, pour un système avec peu de RAM (moins de 2 Go), il est recommandé d'avoir une partition de swap au moins égale à la taille de la RAM. Pour des machines avec beaucoup de RAM (plus de 8 Go), le swap est moins crucial, mais il peut tout de même être utile pour la mise en veille prolongée (hibernation).

Préparation à l'installation de Debian

5. **Sélection des composants logiciels**
   Debian offre une grande flexibilité dans le choix des logiciels à installer pendant le processus d'installation.

- **Environnement de bureau** : Debian prend en charge plusieurs environnements de bureau, comme GNOME, KDE, XFCE, LXDE, et d'autres. L'utilisateur peut choisir celui qui correspond le mieux à ses besoins. GNOME est souvent proposé par défaut, mais des environnements plus légers comme XFCE peuvent être plus adaptés pour des machines aux ressources limitées.
- **Installation minimale ou complète** : Il est possible de choisir une installation minimale, qui n'installe que le système de base, sans environnement graphique ni services supplémentaires. Cela peut être idéal pour des serveurs ou des systèmes embarqués. À l'inverse, une installation complète permet d'installer un large éventail d'applications et de services par défaut.

Préparation à l'installation de Debian

6. **Configurations réseau et accès à Internet**
   Pour certaines méthodes d'installation (comme la netinstall), une connexion réseau est essentielle. Debian peut configurer automatiquement les interfaces réseau via DHCP, ou bien vous pouvez définir manuellement les paramètres de votre réseau (IP, DNS, passerelle).

- **Interfaces réseau** : Il est important de vérifier si les cartes réseau de la machine sont correctement reconnues par Debian, en particulier pour les interfaces Wi-Fi ou les cartes réseau plus exotiques. Si un pilote est manquant, il peut être nécessaire de le fournir au système pendant l'installation.
- **Mise en place des dépôts logiciels** : Pendant l'installation, Debian configure automatiquement les dépôts logiciels (repositories) à partir desquels les paquets peuvent être téléchargés et installés. Ces dépôts peuvent inclure des logiciels libres (dépôt **main**) ou des logiciels non libres et propriétaires (dépôts **contrib** et **non-free**) en fonction des besoins.

Installation Preseed

La méthode **preseed** dans Debian est une technique permettant d'automatiser l'installation du système d'exploitation. Elle est particulièrement utile lorsque vous devez installer Debian sur plusieurs machines de manière répétitive, ou dans un environnement où l'interaction manuelle doit être limitée (par exemple, dans un centre de données ou lors d'un déploiement en masse).

1. Présentation de la méthode Preseed

La méthode **preseed** permet de configurer automatiquement Debian à l'aide d'un fichier de préconfiguration. Ce fichier, appelé **fichier preseed**, contient toutes les réponses aux questions posées durant le processus d'installation. Cela permet d'effectuer une installation sans aucune intervention humaine, rendant l'installation beaucoup plus rapide et uniforme.

2. Fonctionnement du Preseed

Le fichier preseed est un simple fichier texte qui suit une syntaxe spécifique. Il est lu par l'installateur Debian avant ou pendant le processus d'installation. Ce fichier définit les paramètres pour chaque étape clé de l'installation : langue, partitionnement, choix des paquets, configuration réseau, etc.

Installation Preseed

3. Les types d'informations dans un fichier Preseed

1. **Configuration de la langue et du clavier** :
   - Vous pouvez pré-configurer la langue du système, la région et la disposition du clavier. Cela permet d'éviter de passer par les écrans de sélection au démarrage de l'installation.

```
d-i debian-installer/language string fr
d-i debian-installer/country string FR
d-i keyboard-configuration/layoutcode string fr
```

2. **Paramètres de réseau** :
   - L'installation Debian requiert des informations pour configurer l'interface réseau (IP, DNS, passerelle). Ces paramètres peuvent être définis à l'avance pour éviter une configuration manuelle pendant l'installation.
   - Exemple :

```
d-i netcfg/choose_interface select auto
d-i netcfg/get_hostname string serveur-debian
d-i netcfg/get_domain string exemple.com
```

Installation Preseed

3. **Partitionnement automatique** :
    - Le partitionnement des disques est souvent une étape sensible. Avec le preseed, vous pouvez définir un schéma de partitionnement automatique (comme LVM ou non, avec ou sans chiffrement, etc.). Cela peut inclure des partitions spécifiques ou utiliser tout le disque.
    - Exemple :

```
d-i partman-auto/method string lvm
d-i partman-lvm/device_remove_lvm boolean true
d-i partman-auto/choose_recipe select atomic
d-i partman/confirm boolean true
```

Installation Preseed

4. **Installation de paquets supplémentaires** :
   - Vous pouvez spécifier les paquets logiciels à installer, y compris les environnements de bureau, des utilitaires spécifiques, ou même des paquets personnalisés.
   - Exemple :

```
tasksel tasksel/first multiselect standard, ssh-server, print-server
d-i pkgsel/include string openssh-server build-essential
```

5. **Configuration du compte root et des utilisateurs** :
   - Les étapes de création de l'utilisateur root ou des comptes utilisateurs peuvent être automatisées. Cela inclut la configuration des mots de passe ou la désactivation du compte root si nécessaire.
   - Exemple :

```
d-i passwd/root-login boolean false
d-i passwd/user-fullname string "Utilisateur Debian"
d-i passwd/username string utilisateur
d-i passwd/user-password-crypted password $6$G...$...
```

Installation Preseed

6. **Sélection de la zone horaire et du fuseau horaire** :
   - Vous pouvez configurer automatiquement le fuseau horaire en fonction du pays sélectionné ou le définir manuellement.
   - Exemple :

```
d-i time/zone string Europe/Paris
```

7. **Configurer des scripts post-installation** :
   - Vous pouvez également automatiser des tâches post-installation à l'aide de scripts. Cela peut être utile pour installer des configurations ou des applications spécifiques après l'installation du système de base.
   - Exemple :

```
d-i preseed/late_command string in-target apt-get -y install git
```

Installation Preseed

Avantages de la méthode Preseed

1. **Gain de temps** : L'automatisation permet de réduire le temps d'installation, en particulier lors de déploiements en masse ou répétitifs. Vous n'avez pas à intervenir pour répondre aux questions pendant le processus.
2. **Consistance des installations** : En utilisant un fichier preseed standardisé, toutes les installations sont identiques. Cela garantit que chaque machine a la même configuration de base, minimisant les erreurs humaines et assurant une uniformité sur les déploiements.
3. **Flexibilité** : Preseed permet de préconfigurer presque tous les aspects de l'installation. Vous pouvez personnaliser les installations en fonction des besoins spécifiques des utilisateurs, des environnements ou des infrastructures.
4. **Installation non surveillée** : Avec preseed, il est possible de réaliser une installation sans interaction humaine, utile pour des installations sur des serveurs distants ou en dehors des heures de travail.

Installation Preseed

Méthodes d'utilisation du fichier Preseed

Le fichier preseed peut être utilisé de différentes manières pour automatiser l'installation de Debian :

1. **Depuis une clé USB ou un CD** : Vous pouvez ajouter le fichier preseed à une clé USB ou un CD d'installation de Debian. Il suffit de le placer dans la structure de fichiers du support d'installation et de spécifier à l'installateur où trouver ce fichier au démarrage.
2. **Installation via le réseau** : Le fichier preseed peut également être chargé via une URL pendant l'installation, ce qui permet de le stocker sur un serveur web. Cette méthode est idéale pour des déploiements centralisés.
    - Exemple de commande de démarrage :

```
auto url=http://monserveur/preseed.cfg
```

3. **Utilisation avec PXE** : Lors des installations réseau en utilisant PXE (Preboot Execution Environment), le fichier preseed peut être utilisé pour automatiser entièrement le processus d'installation sur des machines sans support physique local.

Installation Preseed

Exemple complet de fichier Preseed

```
# Langue et clavier
d-i debian-installer/language string fr
d-i debian-installer/country string FR
d-i keyboard-configuration/layoutcode string fr

# Réseau
d-i netcfg/choose_interface select auto
d-i netcfg/get_hostname string serveur-debian
d-i netcfg/get_domain string exemple.com

# Partitionnement
d-i partman-auto/method string lvm
d-i partman-auto/choose_recipe select atomic
d-i partman/confirm boolean true

# Utilisateurs
d-i passwd/root-login boolean false
d-i passwd/user-fullname string "Utilisateur Debian"
d-i passwd/username string utilisateur
d-i passwd/user-password-crypted password $6$G...$...

# Zone horaire
d-i time/zone string Europe/Paris
```

Installation Preseed

Limites de la méthode Preseed

1. **Complexité** : Pour un utilisateur novice, la création d'un fichier preseed peut être complexe, car il faut comprendre chaque étape de l'installation et les options disponibles.
2. **Maintenance** : Lorsque Debian évolue (nouvelles versions, changements dans l'installateur), le fichier preseed peut nécessiter des mises à jour pour rester compatible.
3. **Sécurité** : Les informations sensibles, comme les mots de passe en clair, peuvent être visibles dans le fichier preseed. Il est donc important de les chiffrer (par exemple, les mots de passe cryptés) ou de stocker le fichier dans un environnement sécurisé.

Résoudre les problèmes rencontrés lors de l'installation

1. Problèmes de détection matérielle

- **Non-détection des disques** : Parfois, l'installateur Debian ou Linux en général ne détecte pas les disques durs, en particulier si vous utilisez un contrôleur RAID matériel ou une configuration de disque non standard.
  - **Solution** : Vous devrez peut-être charger des pilotes spécifiques à votre matériel. Ces pilotes peuvent être intégrés via l'installateur en fournissant un fichier de pilote à partir d'un autre support (clé USB, CD, etc.), ou en passant des paramètres au noyau au démarrage, comme `nodmraid` pour désactiver la détection automatique des RAID logiciels.
- **Problèmes de carte réseau** : Si votre carte réseau n'est pas reconnue, en particulier les cartes Wi-Fi qui nécessitent des firmwares propriétaires, l'installation peut échouer à la phase de configuration réseau.
  - **Solution** : Vérifiez que votre carte réseau est compatible avec Linux. Vous pouvez souvent résoudre ce problème en téléchargeant et en installant le firmware manuellement depuis une clé USB. Vous pouvez également vérifier les logs via `dmesg` pour voir quel module manque.
- **Problèmes graphiques** : Certains pilotes de cartes graphiques (en particulier Nvidia ou ATI/AMD) peuvent ne pas être détectés correctement par l'installateur, rendant l'interface graphique inutilisable après l'installation.
  - **Solution** : Utilisez l'installation en mode "nomodeset" pour contourner les problèmes graphiques liés au démarrage du noyau. Après l'installation, installez les pilotes propriétaires ou alternatifs via les dépôts non libres de Debian.

Résoudre les problèmes rencontrés lors de l'installation

2. Problèmes de partitionnement

- **Erreur lors du partitionnement automatique** : Si vous utilisez l'option de partitionnement automatique et que vous avez déjà une configuration de partition complexe ou des partitions existantes non reconnues, l'installateur peut échouer.
  - **Solution** : Utilisez l'option de partitionnement manuel dans l'installateur. Cela vous permettra de créer des partitions personnalisées et de choisir le système de fichiers approprié (ext4, btrfs, etc.).
- **Espace disque insuffisant** : Si le système ne trouve pas assez d'espace disque ou si la partition d'échange (swap) n'est pas correctement créée, l'installation peut échouer.
  - **Solution** : Vérifiez l'espace disque disponible et assurez-vous que les partitions sont bien créées. Utilisez l'outil `partman` pour ajuster manuellement la taille des partitions.

Résoudre les problèmes rencontrés lors de l'installation

3. Problèmes de configuration réseau

- **Échec de la configuration DHCP** : Si l'installateur ne parvient pas à obtenir une adresse IP via DHCP, cela peut empêcher l'installation du système de base via le réseau.
  - **Solution** : Configurez l'adresse IP manuellement en fournissant les informations réseau (IP, passerelle, DNS) pendant l'installation, ou vérifiez que le serveur DHCP sur le réseau fonctionne correctement.

4. Problèmes de paquets ou de dépôt

- **Dépôts non accessibles** : Si l'installateur ne parvient pas à se connecter aux dépôts en ligne pour télécharger les paquets, cela peut être dû à une mauvaise configuration réseau ou à des dépôts non disponibles.
  - **Solution** : Assurez-vous que la connexion Internet est fonctionnelle et que les dépôts sont disponibles. Vous pouvez utiliser un miroir local si le serveur Debian principal est inaccessible.
- **Problèmes de dépendances de paquets** : Si certains paquets ne sont pas installés correctement à cause de dépendances manquantes, l'installation peut échouer partiellement.
  - **Solution** : Après l'installation, vous pouvez essayer de résoudre les problèmes de dépendances en exécutant `apt-get -f install` pour forcer l'installation des paquets manquants.

Résoudre les problèmes rencontrés lors de l'installation

5. Problèmes de démarrage

- **Le système ne démarre pas après l'installation** : Si GRUB (le chargeur de démarrage) n'est pas installé correctement, ou si des partitions critiques ne sont pas accessibles, le système peut ne pas démarrer après l'installation.
  - **Solution** : Utilisez un Live CD ou USB pour accéder à votre système et réinstaller GRUB manuellement via `grub-install` et `update-grub`.

Mise à jour du système post-installation

Après l'installation de Debian ou de tout autre système Linux, il est essentiel de mettre à jour le système pour corriger les vulnérabilités de sécurité, installer des correctifs, et améliorer la compatibilité matérielle.

1. Mise à jour du cache des paquets

Avant de mettre à jour le système, il est nécessaire de mettre à jour le cache des paquets pour obtenir les dernières informations sur les versions disponibles des logiciels.

```
sudo apt update
```

- Cette commande permet à **APT** (l'outil de gestion des paquets de Debian) de synchroniser sa liste de paquets avec les dépôts configurés.

2. Mise à jour complète du système

Une fois la liste des paquets à jour, vous pouvez procéder à la mise à jour complète de votre système.

```
sudo apt upgrade
```

- `apt upgrade` télécharge et installe les mises à jour disponibles pour les paquets déjà installés.
- Cela met à jour tous les logiciels installés, corrige les failles de sécurité, et applique des correctifs pour améliorer la stabilité du système.

Mise à jour du système post-installation

3. Mise à jour avec gestion des dépendances et des paquets obsolètes

Dans certains cas, des mises à jour peuvent nécessiter l'installation de nouveaux paquets ou la suppression de paquets obsolètes. Pour ce faire, utilisez la commande suivante :

```
sudo apt full-upgrade
```

- `full-upgrade` gère les changements de dépendances entre les versions des paquets. Cela peut impliquer l'installation de nouveaux paquets ou la suppression de paquets devenus inutiles ou obsolètes.

4. Nettoyage des paquets

Après une mise à jour, il est recommandé de nettoyer les anciens paquets ou fichiers téléchargés, qui ne sont plus nécessaires au système.

```
sudo apt autoremove
sudo apt clean
```

- `autoremove` supprime les paquets installés automatiquement mais qui ne sont plus nécessaires.
- `clean` supprime les fichiers d’installation des paquets déjà téléchargés pour libérer de l’espace disque.

Mise à jour du système post-installation

5. Mise à jour du noyau

Il est important de vérifier si une nouvelle version du noyau Linux est disponible, car les nouvelles versions peuvent inclure des correctifs de sécurité critiques et des améliorations matérielles.

```
sudo apt install linux-image-generic
```

- Cela installera la dernière version du noyau pour votre système.

6. Configurer les mises à jour automatiques (facultatif)

Vous pouvez configurer Debian pour effectuer des mises à jour de sécurité automatiques. Cela peut être particulièrement utile sur des serveurs ou des systèmes critiques où vous voulez minimiser les risques de failles non corrigées.

1. Installez le paquet **unattended-upgrades** :

```
sudo apt install unattended-upgrades
```

2. Activez les mises à jour automatiques :

```
sudo dpkg-reconfigure unattended-upgrades
```

TP 1 : Installation Automatisée de Debian avec Preseed et Post-installation

Ce TP vous apprendra à automatiser l'installation de Debian à l'aide d'un fichier **preseed** tout en incluant une étape de **post-installation**. Vous allez configurer le fichier preseed pour installer Debian sur une machine virtuelle et exécuter des scripts après l'installation pour finaliser la configuration du système.

- Une machine virtuelle dans **VirtualBox** ou un autre hyperviseur.
- **Image ISO Debian Netinstall**
- Un éditeur de texte pour créer le fichier **preseed.cfg**.
- Une connexion Internet pour la machine virtuelle.

1. **Création de la machine virtuelle :**
    - Créez une machine virtuelle dans VirtualBox avec au moins 2 Go de RAM et 10 Go de disque dur virtuel.
    - Montez l'image ISO Debian Netinstall dans la machine virtuelle.

TP 1 : Installation Automatisée de Debian avec Preseed et Post-installation

2. **Rédaction du fichier preseed :**
   - Créez un fichier **preseed.cfg** qui permet de configurer les éléments suivants :
     - **Langue** : Français
     - **Pays** : France
     - **Disposition du clavier** : Français (AZERTY)
     - **Nom de la machine** : `debian-vm`
     - **Nom de l'utilisateur** : `utilisateur`
     - **Mot de passe utilisateur** : `motdepass`
     - **Partitionnement** : Automatique avec LVM, en utilisant tout le disque.
     - **Environnement de bureau** : GNOME
     - **Installation de paquets supplémentaires** : Serveur SSH, Git
     - **Post-installation** :
       - Installez les mises à jour du système via `apt`.
       - Créez un script post-install.

TP 1 : Installation Automatisée de Debian avec Preseed et Post-installation

3. **Exécution d'un script de post-installation :**
   - Utilisez la commande **preseed/late_command** pour exécuter le script de post-installation qui effectue les actions suivantes :
     - Mise à jour du système.
     - Création du fichier `/etc/motd` avec un message personnalisé : "Bienvenue sur Debian".
4. **Intégration du fichier preseed dans l'installation**
5. **Installation automatisée :**
   - Démarrez la machine virtuelle et lancez l'installation automatisée de Debian avec le fichier preseed configuré.
   - Vérifiez que tous les paramètres définis dans le fichier preseed sont appliqués correctement : création de l'utilisateur, partitionnement, installation des paquets, et exécution des scripts de post-installation.

Gestion des utilisateurs et des groupes locaux

La gestion des utilisateurs et des groupes est un aspect fondamental de l'administration sous Debian, car elle permet de contrôler qui peut accéder au système et les ressources disponibles pour chaque utilisateur ou groupe. L'administrateur doit être capable de créer, modifier, et supprimer des utilisateurs ainsi que des groupes, tout en gérant leurs permissions.

1. Création d'un utilisateur

Pour ajouter un utilisateur sous Debian, on utilise la commande `adduser`.

```
sudo adduser alice
```

Cette commande va vous demander de fournir des informations comme le mot de passe et d'autres détails (nom complet, numéro de téléphone, etc.). Vous pouvez également utiliser la commande `useradd` qui est plus basique et demande moins d'interactions

```
sudo useradd -m bob
```

- L'option `-m` crée un répertoire personnel pour l'utilisateur.

Gestion des utilisateurs et des groupes locaux

2. Suppression d'un utilisateur

Pour supprimer un utilisateur, on utilise la commande `deluser`.

```
sudo deluser alice
```

Si vous souhaitez également supprimer son répertoire personnel, vous pouvez utiliser l'option `--remove-home` :

```
sudo deluser --remove-home bob
```

3. Gestion des groupes

Les groupes permettent de regrouper plusieurs utilisateurs pour leur attribuer des permissions communes. Pour créer un groupe, utilisez la commande `addgroup` :

```
sudo addgroup developpeurs
```

Pour ajouter un utilisateur à un groupe existant :

```
sudo usermod -aG developpeurs alice
```

L'option `-aG` permet d'ajouter (`-a`) l'utilisateur au groupe (`-G`) sans supprimer les autres groupes auxquels il appartient.

Gestion des utilisateurs et des groupes locaux

4. Vérifier les groupes d'un utilisateur

Pour voir les groupes auxquels un utilisateur appartient, vous pouvez utiliser la commande `groups` :

```
groups alice
```

5. Gestion des fichiers `/etc/passwd`, `/etc/shadow` et `/etc/group`

Ces fichiers sont essentiels dans la gestion des utilisateurs et groupes sous Debian.

- **/etc/passwd** : Contient des informations de base sur chaque utilisateur (nom, identifiant utilisateur, etc.)
- **/etc/shadow** : Contient les mots de passe chiffrés des utilisateurs et les politiques de mot de passe.
- **/etc/group** : Contient les informations sur les groupes du système.

Profils et environnements

Chaque utilisateur sous Linux dispose d'un environnement de travail personnalisé, qui inclut les variables d'environnement, les configurations de shell, et les chemins d'accès spécifiques. Ces environnements sont configurés via différents fichiers de profil.

1. Variables d'environnement

Les variables d'environnement stockent des informations comme le chemin d'exécution des programmes ou les préférences d'un utilisateur.

Exemple d'affichage des variables d'environnement :

```
printenv
```

Pour définir une variable d'environnement, vous pouvez modifier des fichiers comme `~/.bashrc` ou `/etc/profile`.

```
export PATH=$PATH:/usr/local/bin
```

Cela ajoute un nouveau chemin au répertoire `PATH`.

Profils et environnements

2. Fichiers de configuration du shell

Sous Debian, il existe plusieurs fichiers où vous pouvez configurer l'environnement utilisateur :

- **/etc/profile** : Définit des configurations globales pour tous les utilisateurs.
- **~/.bashrc** : Définit des configurations spécifiques à chaque utilisateur. Il est exécuté à chaque fois que l'utilisateur ouvre un terminal.
- **~/.bash_profile** : Utilisé principalement pour les sessions de connexion. Il est souvent utilisé pour charger `~/.bashrc`.

Exemple : Pour ajouter une alias dans le fichier `~/.bashrc` :

```
echo "alias ll='ls -la'" >> ~/.bashrc
```

Cette commande crée un alias `ll` pour afficher le contenu d'un répertoire en format détaillé.

Profils et environnements

3. Changement d'environnement

Si vous voulez temporairement changer une variable d'environnement ou une configuration, vous pouvez le faire directement dans le terminal.

Exemple :

```
export EDITOR=nano
```

Cela change temporairement l'éditeur par défaut en `nano` pour la session en cours.

4. Définir des profils spécifiques

Vous pouvez également définir des profils spécifiques pour des utilisateurs dans leurs fichiers `~/.bash_profile` ou `~/.bashrc`. Par exemple, si vous souhaitez qu'un utilisateur ait un éditeur par défaut spécifique à chaque connexion, vous pouvez ajouter ceci dans `~/.bash_profile` :

```
export EDITOR=vim
```

TP2 : Gestion des utilisateurs, groupes et sécurité sous Debian

Objectifs :

1. Créer et gérer des utilisateurs et groupes.
2. Assurer la sécurité des mots de passe avec SHA-512.
3. Configurer les environnements utilisateurs via `.bashrc` et `.bash_profile`.
4. Analyser la sécurité des fichiers critiques.

- **Partie 1 : Création et gestion des utilisateurs et des groupes**

1. **Créer les utilisateurs suivants :**
   - `jean` (ID utilisateur personnalisé : 1501).
   - `marie` (ID utilisateur personnalisé : 1502).
2. **Créer les groupes suivants :**
   - `admin` (ajoutez `jean` à ce groupe).
   - `support` (ajoutez `marie` à ce groupe).

TP2 : Gestion des utilisateurs, groupes et sécurité sous Debian

3. **Vérifier les informations dans `/etc/passwd` et `/etc/group` :**
    - Confirmez que les utilisateurs et groupes sont correctement créés.

- **Partie 2 : Sécurisation avec SHA-512**

1. **Configurer PAM pour SHA-512**
2. **Changer le mot de passe de `jean` et vérifier dans `/etc/shadow` :**

TP2 : Gestion des utilisateurs, groupes et sécurité sous Debian

- **Partie 3 : Configuration des environnements utilisateurs**

1. **Ajouter une alias pour `jean` dans `~/.bashrc` :**
   - `alias ll='ls -la'`.
2. **Ajouter une variable d'environnement pour `marie` dans `~/.bash_profile` :**
   - `export EDITOR=nano`.
3. **Vérifier les changements en vous connectant en tant que chaque utilisateur.**

- **Partie 4 : Analyse des fichiers critiques**

1. **Examiner et analyser le contenu de `/etc/passwd`, `/etc/shadow` et `/etc/group` :**
   - Expliquer les risques liés à leur mauvaise gestion.
   - Proposer des bonnes pratiques pour sécuriser ces fichiers.

UTOPIOS®
TRANSFORMATION DIGITALE INTELLIGENTE

Jour 2

Présentation des différents systèmes de fichiers

1. ext4 (Fourth Extended Filesystem)

- **Description** : C'est l'un des systèmes de fichiers les plus utilisés sur Linux, en particulier sur les distributions comme Debian. Il est une amélioration des versions précédentes (ext2 et ext3), offrant de meilleures performances, une gestion optimisée de la fragmentation, et une plus grande capacité de stockage.
- **Caractéristiques** :
  - Taille maximale d'un fichier : 16 To (Téraoctets)
  - Taille maximale d'un système de fichiers : 1 Eo (Exaoctet)
  - Journalisation : Gère un journal pour réduire les risques de corruption des données en cas de panne.
  - Récupération rapide des données après un crash.
  - Prise en charge des attributs étendus et des quotas de disque.
- **Cas d'utilisation** : Système de fichiers par défaut dans de nombreuses distributions Linux, idéal pour les serveurs et postes de travail.

Présentation des différents systèmes de fichiers

2. Btrfs (B-tree File System)

- **Description** : Un système de fichiers moderne conçu pour remplacer ext4 à long terme. Il se concentre sur la gestion des grandes quantités de données, la fiabilité, et les fonctionnalités avancées comme la prise en charge de la **gestion des instantanés** (snapshots) et de la **compression** des données.
- **Caractéristiques** :
  - Gestion des snapshots et de la déduplication de données.
  - Taille maximale d'un fichier : 16 Eo.
  - Taille maximale d'un système de fichiers : 16 Eo.
  - Journalisation intégrée.
  - Possibilité d'effectuer des mises à jour atomiques grâce aux snapshots.
- **Cas d'utilisation** : Idéal pour des serveurs où des snapshots fréquents et la gestion de grands volumes de données sont requis. Souvent utilisé sur des systèmes de stockage complexes ou en cloud.

Présentation des différents systèmes de fichiers

3. XFS

- **Description** : Un autre système de fichiers performant, conçu initialement pour les serveurs SGI (Silicon Graphics). XFS est particulièrement adapté aux grandes bases de données et systèmes de fichiers nécessitant une très haute performance.
- **Caractéristiques** :
  - Taille maximale d'un fichier : 8 Eo.
  - Taille maximale d'un système de fichiers : 8 Eo.
  - Excellente gestion de fichiers de grande taille et des I/O (opérations d'entrée/sortie) intensives.
  - Journalisation à haut débit.
  - Optimisé pour la gestion des fichiers fragmentés.
- **Cas d'utilisation** : Utilisé dans les environnements de production de haute performance, comme les serveurs de base de données ou les systèmes où de très grandes quantités de données doivent être stockées et traitées rapidement.

Présentation des différents systèmes de fichiers

4. ZFS (Zettabyte File System)

- **Description** : Un système de fichiers avancé développé par Sun Microsystems, maintenant sous licence CDDL. ZFS combine des fonctionnalités de gestion de volumes avec un système de fichiers, ce qui le rend particulièrement intéressant pour les systèmes nécessitant des fonctionnalités comme la **gestion de volume**, la **correction automatique d'erreurs**, et une **intégrité des données** de haute qualité.
- **Caractéristiques** :
  - Gestion de très grandes quantités de stockage (pouvant atteindre des zettaoctets).
  - Intégrité des données grâce à des contrôles d'intégrité (checksums) automatiques.
  - Compression et déduplication des données.
  - Gestion des snapshots.
  - Résilience face à la corruption des données grâce à la réparation automatique.
- **Cas d'utilisation** : Très utilisé dans les systèmes de stockage massifs ou les infrastructures cloud, ainsi que pour des environnements où la sécurité et l'intégrité des données sont critiques.

Présentation des différents systèmes de fichiers

5. ReiserFS

- **Description** : Un système de fichiers conçu pour une gestion efficace des petits fichiers et des répertoires contenant de nombreux fichiers. Il est moins utilisé aujourd'hui mais reste une alternative à ext4 dans certains cas spécifiques.
- **Caractéristiques** :
  - Journalisation intégrée.
  - Efficacité accrue pour les petits fichiers.
  - Très bonne gestion de la fragmentation.
- **Cas d'utilisation** : Bien que moins populaire, il a été utilisé dans des systèmes où de nombreux petits fichiers devaient être stockés et manipulés efficacement.

Présentation des différents systèmes de fichiers

6. VFAT

- **Description** : Un système de fichiers basé sur la FAT (File Allocation Table), utilisé principalement pour la compatibilité avec les systèmes Windows ou pour les supports de stockage amovibles (clés USB, cartes SD, etc.).
- **Caractéristiques** :
  - Taille maximale d'un fichier : 4 Go.
  - Absence de journalisation.
  - Compatibilité avec Windows, ce qui le rend très utile pour les systèmes de fichiers partagés entre Linux et Windows.
- **Cas d'utilisation** : Utilisé pour des clés USB ou des cartes mémoire pour assurer la compatibilité entre Linux, Windows et d'autres systèmes.

Présentation des différents systèmes de fichiers

7. NTFS (New Technology File System)

- **Description** : Système de fichiers par défaut des systèmes Windows. Sous Debian, NTFS est principalement utilisé pour le montage de disques ou de partitions Windows en lecture/écriture, grâce à des outils comme `ntfs-3g`.
- **Caractéristiques** :
  - Journalisation.
  - Support des attributs étendus.
  - Support des fichiers volumineux (plus de 4 Go).
- **Cas d'utilisation** : Partage de fichiers entre Linux et Windows, notamment pour accéder aux partitions de disques durs formatées en NTFS.

Présentation des différents systèmes de fichiers

Lorsqu'il s'agit de gérer les systèmes de fichiers sur Debian, plusieurs outils sont disponibles pour formater, vérifier l'intégrité, et monter des partitions.

1. Formater une partition avec `mkfs`

Le formatage d'une partition est la première étape après la création d'une partition. Cela permet de préparer la partition en y installant un système de fichiers, afin qu'elle puisse stocker des données de manière organisée.

Outil : `mkfs` (make filesystem)

L'outil **`mkfs`** est utilisé pour créer un système de fichiers sur une partition. Il prend en charge plusieurs types de systèmes de fichiers, comme **ext4**, **XFS**, **Btrfs**, etc.

Pour formater une partition avec un système de fichiers **ext4**.

```
sudo mkfs.ext4 /dev/sda1
```

options:

- **`-t <type>`** : Spécifie le type de système de fichiers. Nous pouvons choisir ext4, XFS, Btrfs, ou tout autre système pris en charge.
- **`-L <label>`** : Attribue un label à la partition, une étiquette qui permet de l'identifier plus facilement.

Présentation des différents systèmes de fichiers

2. Vérifier l'intégrité du système de fichiers avec `fsck`

L'outil **`fsck`** (file system check) est utilisé pour vérifier et, si nécessaire, réparer l'intégrité d'un système de fichiers. Des problèmes peuvent survenir sur une partition en raison de coupures de courant, de plantages du système ou d'autres problèmes, et **`fsck`** peut détecter et corriger les erreurs dans ces cas.
Pour vérifier l'intégrité du système de fichiers sur la partition **/dev/sda1**.

```
sudo fsck /dev/sda1
```

- Il effectue un contrôle approfondi de la structure du système de fichiers.
- Si des erreurs sont détectées, **`fsck`** tentera de les corriger automatiquement, sauf si nous avons spécifié des options particulières.

Options:

- **`-y`** : Cette option indique à **`fsck`** de réparer automatiquement toutes les erreurs détectées, sans demander confirmation à chaque fois.
- **`-f`** : Forcer la vérification, même si le système de fichiers semble propre.
- **`-n`** : Ne pas faire de modifications, simplement signaler les erreurs trouvées.

Il est important d'utiliser **`fsck`** régulièrement, notamment après un plantage ou une coupure de courant, pour garantir que le système de fichiers reste cohérent et que les données ne sont pas corrompues.

Présentation des différents systèmes de fichiers

3. Monter un système de fichiers avec `mount`

Une fois que nous avons créé et formaté une partition, nous devons la **monter** pour qu'elle devienne accessible dans l'arborescence des fichiers. Le montage d'une partition signifie l'associer à un point de montage, c'est-à-dire un répertoire, afin que le système d'exploitation puisse y accéder et y stocker des fichiers.
L'outil `mount` permet de monter une partition sur un répertoire du système de fichiers.
Pour monter la partition **/dev/sda1** sur le répertoire **/mnt/data**.

```
sudo mount /dev/sda1 /mnt/data
```

Options:

- `-t <type>` : Spécifie le type de système de fichiers à monter, si nous avons besoin de le spécifier explicitement.
- `-o <options>` : Utilise des options spécifiques pour le montage, comme `rw` (lecture/écriture) ou `ro` (lecture seule).

Présentation des différents systèmes de fichiers

4. Monter automatiquement au démarrage via `/etc/fstab`

Pour que le montage soit automatique à chaque démarrage, nous devons ajouter une entrée dans le fichier `/etc/fstab`. Ce fichier contient les informations nécessaires pour monter automatiquement les partitions.

1. Ouvrir le fichier `/etc/fstab` :

```
sudo nano /etc/fstab
```

2. Ajouter une ligne comme celle-ci pour monter **/dev/sda1** sur **/mnt/data** avec un système de fichiers ext4 :

```
/dev/sda1  /mnt/data  ext4  defaults  0  2
```

La gestion des **partitions traditionnelles** sur un disque dur consiste à organiser l'espace de stockage de manière à diviser le disque en plusieurs segments indépendants, appelés **partitions**. Chaque partition peut avoir son propre système de fichiers et être utilisée à différentes fins, comme l'installation d'un système d'exploitation, le stockage de données, ou la gestion de fichiers temporaires.

Gestion des partitions traditionnelles (partitions primaires et étendues)

1. Partitions primaires

Les **partitions primaires** sont les partitions de base créées sur un disque dur. Sur un disque au format **MBR (Master Boot Record)**, nous ne pouvons avoir que **quatre partitions primaires maximum**. Une partition primaire peut contenir des données, un système de fichiers, ou être utilisée pour démarrer un système d'exploitation.

- **Limitation à quatre** : Un disque dur formaté en MBR ne peut pas avoir plus de quatre partitions primaires.
- **Système d'exploitation** : Au moins une des partitions primaires peut être marquée comme **bootable**, ce qui permet au système d'exploitation de démarrer depuis cette partition.
- **Simplicité** : Les partitions primaires sont simples et faciles à gérer. Elles sont idéales pour les systèmes où un petit nombre de partitions est suffisant.

Gestion des partitions traditionnelles (partitions primaires et étendues)

Pour créer une partition primaire, nous utilisons des outils comme `fdisk` ou `parted`.

1. Ouvrir `fdisk` pour gérer le disque dur (par exemple, **/dev/sda**) :

```
sudo fdisk /dev/sda
```

2. Créer une partition primaire :
   - Appuyer sur `n` pour créer une nouvelle partition.
   - Sélectionner `p` pour créer une partition primaire.
   - Suivre les instructions pour spécifier la taille de la partition.
3. Enregistrer les modifications et quitter `fdisk` :

```
w
```

4. Formater la partition nouvellement créée avec un système de fichiers, par exemple ext4 :

```
sudo mkfs.ext4 /dev/sda1
```

Gestion des partitions traditionnelles (partitions primaires et étendues)

2. Partitions étendues

Les **partitions étendues** existent pour dépasser la limitation des quatre partitions primaires. Comme un disque MBR ne peut contenir que quatre partitions primaires, si nous avons besoin de plus de quatre partitions, nous devons créer une partition étendue. Une partition étendue agit comme un conteneur, et à l'intérieur de cette partition, nous pouvons créer plusieurs **partitions logiques**.

- **Une seule partition étendue** : Il ne peut y avoir qu'une seule partition étendue par disque.
- **Conteneur pour partitions logiques** : La partition étendue ne contient pas directement de données, mais peut héberger un nombre illimité de partitions logiques (pratiquement limité par la taille du disque).
- **Évite la limitation des quatre partitions** : Une fois la partition étendue créée, nous pouvons y ajouter des partitions logiques supplémentaires.

Sur un disque où trois partitions primaires sont déjà créées, nous pouvons créer une partition étendue pour ajouter des partitions logiques supplémentaires. Par exemple, nous pourrions utiliser ces partitions logiques pour séparer des données, des fichiers de swap, ou des fichiers temporaires.

Gestion des partitions traditionnelles (partitions primaires et étendues)

1. Utiliser `fdisk` pour créer une partition étendue :
   - Ouvrir `fdisk` sur le disque (par exemple, **/dev/sda**) :

```
sudo fdisk /dev/sda
```

- Appuyer sur `n` pour créer une nouvelle partition.
- Sélectionner `e` pour créer une partition étendue.
- Suivre les instructions pour spécifier la taille.

2. Créer des partitions logiques à l'intérieur de la partition étendue :
   - Après avoir créé la partition étendue, nous pouvons créer des partitions logiques de la même manière que des partitions primaires.
   - Lors de la création de nouvelles partitions, elles seront numérotées **/dev/sda5**, **/dev/sda6**, etc. (les partitions logiques commencent à partir de `5`).
3. Formater les partitions logiques créées avec un système de fichiers :

Gestion des partitions traditionnelles (partitions primaires et étendues)

3. MBR vs GPT

L'utilisation de **MBR** pour gérer les partitions est une approche traditionnelle, mais elle est de plus en plus remplacée par le format **GPT (GUID Partition Table)**, qui est plus moderne et offre davantage de flexibilité. Contrairement à MBR, GPT permet de créer un nombre illimité de partitions (pratiquement 128 partitions), tout en supportant des disques de très grande taille (au-delà de 2 To).

- **Pas de limitation à 4 partitions** : GPT permet de créer de nombreuses partitions sans avoir besoin de partitions étendues et logiques.
- **Support des disques de plus de 2 To** : Contrairement à MBR, qui est limité à 2 To par disque.
- **Meilleure protection des données** : GPT inclut plusieurs copies de la table des partitions, ce qui améliore la sécurité en cas de corruption.

Gestion des volumes logiques (LVM)

La **Gestion des Volumes Logiques (LVM)** est un système utilisé dans Linux pour fournir une gestion flexible et dynamique de l'espace disque. Contrairement aux partitions traditionnelles, LVM permet de regrouper plusieurs disques ou partitions en un seul espace de stockage (appelé **volume logique**), que nous pouvons redimensionner facilement sans interrompre les services.

1. **Volumes physiques (Physical Volumes, PV)** :
   - Un **volume physique** correspond à une partition ou un disque brut. Nous convertissons une partition ou un disque en volume physique pour qu'il puisse être utilisé par LVM. Par exemple, un disque ou une partition comme **/dev/sda1** ou **/dev/sdb** peut être un volume physique.
   - Ces volumes physiques sont ensuite regroupés en un groupe de volumes.
2. **Groupes de volumes (Volume Groups, VG)** :
   - Un **groupe de volumes** est un pool d'espace de stockage qui contient plusieurs volumes physiques. LVM regroupe ces volumes physiques pour créer un espace de stockage plus important et plus flexible.
   - Nous pouvons créer, agrandir ou réduire des volumes logiques à partir de ce groupe de volumes.

Gestion des volumes logiques (LVM)

3. **Volumes logiques (Logical Volumes, LV)** :
   - Un **volume logique** est l'espace alloué dans un groupe de volumes. Il fonctionne comme une partition classique, mais avec la flexibilité d'être agrandi ou réduit dynamiquement.
   - Un volume logique peut héberger des systèmes de fichiers (comme ext4, XFS, etc.) et peut être utilisé de la même manière que des partitions traditionnelles.

Gestion des volumes logiques (LVM)

LVM présente plusieurs avantages majeurs par rapport aux partitions traditionnelles :

- **Redimensionnement facile** : Nous pouvons augmenter ou réduire la taille des volumes logiques sans réinstaller le système ni affecter les données existantes.
- **Gestion dynamique** : Si nous manquons d'espace sur un volume logique, nous pouvons ajouter de nouveaux disques ou partitions au groupe de volumes et agrandir les volumes logiques.
- **Snapshots** : LVM permet la création de **snapshots**, des copies instantanées d’un volume logique à un moment donné, ce qui est utile pour les sauvegardes ou les tests.
- **Gestion de l'espace disque étendu** : LVM permet de regrouper plusieurs disques ou partitions physiques en un seul volume, ce qui facilite la gestion d'un grand nombre de disques et l'extension du stockage au fil du temps.

Gestion des volumes logiques (LVM)

Pour configurer LVM, nous devons suivre quelques étapes pour convertir des disques ou partitions en volumes physiques, créer un groupe de volumes, puis créer et gérer des volumes logiques.

1. Convertir un disque ou une partition en volume physique (PV)

Avant de créer des groupes de volumes et des volumes logiques, nous devons préparer les partitions ou disques pour qu'ils puissent être utilisés par LVM.

1. **Afficher les disques disponibles** :

```
lsblk
```

2. **Créer un volume physique** avec `pvcreate` :
   Par exemple, si nous voulons utiliser **/dev/sdb1** comme volume physique pour LVM :

```
sudo pvcreate /dev/sdb1
```

3. **Vérifier la création du volume physique** :
   Utilisons la commande `pvdisplay` pour vérifier :

```
sudo pvdisplay
```

Gestion des volumes logiques (LVM)

2. Créer un groupe de volumes (VG)

Une fois les volumes physiques créés, nous devons les regrouper dans un **groupe de volumes**. C'est dans ce groupe que nous allons créer les volumes logiques.

1. **Créer un groupe de volumes** avec `vgcreate` :
   Nous pouvons regrouper un ou plusieurs volumes physiques dans un groupe de volumes. Par exemple, pour créer un groupe de volumes appelé **vg_data** à partir de **/dev/sdb1** :

```
sudo vgcreate vg_data /dev/sdb1
```

2. **Vérifier le groupe de volumes** :
   Utilisons `vgdisplay` pour afficher les informations sur le groupe de volumes :

```
sudo vgdisplay
```

Gestion des volumes logiques (LVM)

3. Créer des volumes logiques (LV)

Une fois le groupe de volumes créé, nous pouvons créer un **volume logique** qui sera utilisé pour stocker les données, exactement comme une partition traditionnelle.

1. **Créer un volume logique** avec `lvcreate` :
   Par exemple, pour créer un volume logique de 10 Go appelé **lv_data** dans le groupe de volumes **vg_data** :

```
sudo lvcreate -L 10G -n lv_data vg_data
```

2. **Vérifier le volume logique** :
   Utilisons `lvdisplay` pour afficher les informations sur les volumes logiques :

```
sudo lvdisplay
```

Gestion des volumes logiques (LVM)

3. **Formater le volume logique** avec un système de fichiers :
   Une fois le volume logique créé, nous devons le formater pour pouvoir y stocker des fichiers. Par exemple, pour formater le volume logique **lv_data** en **ext4** :

```
sudo mkfs.ext4 /dev/vg_data/lv_data
```

4. **Monter le volume logique** :
   Enfin, nous devons monter le volume logique pour qu'il soit accessible depuis le système de fichiers :

```
sudo mount /dev/vg_data/lv_data /mnt/data
```

Gestion des volumes logiques (LVM)

4. Redimensionner un volume logique

Un des avantages majeurs de LVM est la capacité à redimensionner les volumes logiques à la volée. Nous pouvons augmenter la taille d'un volume logique si nous manquons d'espace.

1. **Étendre un volume logique** avec `lvextend` :
   Si nous voulons ajouter 5 Go supplémentaires au volume logique **lv_data** :

```
sudo lvextend -L +5G /dev/vg_data/lv_data
```

2. **Redimensionner le système de fichiers** pour qu'il utilise l'espace supplémentaire :
   Pour **ext4**, nous utilisons `resize2fs` :

```
sudo resize2fs /dev/vg_data/lv_data
```

Pour **XFS**, nous utilisons `xfs_growfs` :

```
sudo xfs_growfs /mnt/data
```

Gestion des volumes logiques (LVM)

5. Créer un snapshot avec LVM

LVM permet de créer des **snapshots**, des copies instantanées d'un volume logique. Ces snapshots sont utiles pour des sauvegardes ou des tests avant d'effectuer des changements importants.

1. **Créer un snapshot** avec `lvcreate` :
   Par exemple, pour créer un snapshot du volume logique **lv_data** dans **vg_data** avec une taille de 5 Go :

```
sudo lvcreate --size 5G --snapshot --name lv_data_snap /dev/vg_data/lv_data
```

2. **Monter le snapshot** pour l'utiliser :

```
sudo mount /dev/vg_data/lv_data_snap /mnt/snapshot
```

Gestion des volumes logiques (LVM)

6. Supprimer un volume logique ou un groupe de volumes

Si nous voulons supprimer un volume logique ou un groupe de volumes, nous devons démonter le volume logique, puis utiliser les commandes de suppression.

1. **Démonter le volume logique** :

```
sudo umount /mnt/data
```

2. **Supprimer le volume logique** avec `lvremove` :

```
sudo lvremove /dev/vg_data/lv_data
```

3. **Supprimer le groupe de volumes** avec `vgremove` (si plus aucun volume logique n'est présent dans le groupe) :

```
sudo vgremove vg_data
```

4. **Supprimer le volume physique** avec `pvremove` :

```
sudo pvremove /dev/sdb1
```

TP 3 Gestion des volumes logiques (LVM)

Objectif :

Dans cet exercice, les stagiaires utiliseront **LVM (Logical Volume Manager)** pour gérer dynamiquement l'espace de stockage dans un environnement **VirtualBox**. Ils vont créer un volume logique, le formater avec un système de fichiers, puis configurer son montage permanent via `fstab` et **systemd**.

Prérequis :

- Environnement VirtualBox fonctionnel avec une machine virtuelle Debian ou une autre distribution Linux.
- Compréhension de base des partitions, systèmes de fichiers, et du gestionnaire de volumes logiques (LVM).
- Une machine virtuelle avec une partition ou un disque supplémentaire non alloué.

1. **Ajouter un disque supplémentaire à la machine virtuelle**
2. **Créer un Volume Physique (PV)**
3. **Créer un Groupe de Volumes (VG)**
4. **Créer un Volume Logique (LV)**
5. **Formater le Volume Logique**
6. **Monter le Volume Logique**

TP 3 Gestion des volumes logiques (LVM)

7. **Configurer le Montage Permanent avec `fstab`**
8. **Configurer le Montage Permanent avec `systemd`**
9. **Vérifications**

Gestion des droits standards (chown, chmod, umask)

1. `chown` : Changer le propriétaire et le groupe des fichiers

Le propriétaire d'un fichier ou d'un dossier détermine qui peut le modifier ou y accéder. Dans Debian, chaque fichier a un propriétaire et un groupe. Le propriétaire est l'utilisateur qui a créé le fichier, et le groupe est souvent celui auquel appartient cet utilisateur.

Syntaxe :

```
chown [nouveau_propriétaire]:[nouveau_groupe] fichier
```

- `nouveau_propriétaire` : Le nom de l'utilisateur à qui vous voulez assigner le fichier.
- `nouveau_groupe` : Le nom du groupe auquel vous voulez assigner le fichier. Vous pouvez ignorer cette partie si vous ne souhaitez pas changer le groupe.
- `fichier` : Le fichier ou le répertoire cible.

Gestion des droits standards (chown, chmod, umask)

2. `chmod` : Modifier les droits d'accès aux fichiers

`chmod` permet de définir les permissions d'accès pour les fichiers et dossiers. Il y a trois types de permissions :

- **r** : Lecture (Read)
- **w** : Écriture (Write)
- **x** : Exécution (Execute)

Ces permissions sont attribuées à trois catégories :

- **u** : Utilisateur propriétaire (user)
- **g** : Groupe propriétaire (group)
- **o** : Autres (others)

Syntaxe :

```
chmod [options] mode fichier
```

Il existe deux façons principales d'utiliser `chmod` :

- **Symbolique** : Utiliser des lettres pour ajouter, enlever, ou définir les permissions (comme `u+r` pour ajouter la permission de lecture à l'utilisateur).
- **Numérique** : Utiliser des chiffres pour représenter les permissions (comme `755`).

Gestion des droits standards (chown, chmod, umask)

3. `umask` : Définir les permissions par défaut

Lorsque vous créez un nouveau fichier ou répertoire, les permissions par défaut sont définies par le masque d'utilisateur, ou `umask`. Ce masque soustrait certaines permissions à celles qui sont normalement attribuées à un fichier ou un répertoire nouvellement créé.

Syntaxe :

```
umask [valeur]
```

La valeur de `umask` est soustraite des permissions maximales possibles pour déterminer les permissions par défaut. Les permissions maximales sont :

- **666** pour les fichiers (lecture et écriture pour tous),
- **777** pour les répertoires (lecture, écriture, exécution pour tous).

Listes de contrôle d'accès (ACL)

Les **Listes de Contrôle d'Accès (ACL)** permettent de définir des permissions plus granulaires que celles offertes par les mécanismes traditionnels de gestion des permissions avec `chown` et `chmod`. Grâce aux ACL, vous pouvez accorder des permissions spécifiques à des utilisateurs ou des groupes particuliers pour des fichiers ou répertoires, sans être limité aux trois catégories classiques (utilisateur propriétaire, groupe propriétaire, et autres).

1. Qu'est-ce qu'une ACL ?

Une ACL permet de définir des permissions spécifiques pour plusieurs utilisateurs ou groupes, indépendamment des permissions définies via `chmod`. Cela offre plus de flexibilité dans la gestion des droits d'accès.

2. Activer les ACL sur un système de fichiers

Avant de pouvoir utiliser les ACL, il faut s'assurer que le système de fichiers prend en charge cette fonctionnalité. Généralement, les systèmes de fichiers comme **ext4** supportent les ACL, mais vous devez parfois les activer manuellement.

3. Utiliser les ACL

Une fois ACL activé, vous pouvez commencer à gérer les permissions supplémentaires sur vos fichiers et répertoires.

Listes de contrôle d'accès (ACL)

1. Affichage des ACL

Pour voir les ACL définies sur un fichier ou un répertoire, utilisez la commande `getfacl` :

```
getfacl nom_du_fichier
```

2. Définir des ACL avec `setfacl`

La commande `setfacl` permet d'ajouter, modifier ou supprimer des permissions ACL pour un fichier ou un répertoire.

```
setfacl -m u:[utilisateur]:[permissions] nom_du_fichier
```

- `u:[utilisateur]` : Définit l'utilisateur à qui vous donnez des permissions spécifiques.
- `[permissions]` : Peut inclure `r` (lecture), `w` (écriture), `x` (exécution).

3. Supprimer une ACL

Pour supprimer une ACL attribuée à un utilisateur, vous pouvez utiliser la commande suivante :

```
setfacl -x u:bob projet.txt
```

4. **Définir des ACL par défaut**
   Les ACL par défaut s'appliquent automatiquement à tous les nouveaux fichiers et répertoires créés à l'intérieur d'un répertoire particulier. Pour définir une ACL par défaut, utilisez l'option `-d` :

Listes de contrôle d'accès (ACL)

4. Visualisation des permissions avec `ls`

Lorsque vous utilisez des ACL, le résultat de la commande `ls -l` change légèrement. Vous verrez un `+` à la fin des permissions si des ACL sont en place :

```
ls -l projet.txt
```

Résultat :

```
-rw-rw-r--+ 1 alice developpeurs 0 Oct 20 10:10 projet.txt
```

Le `+` indique que des ACL supplémentaires sont définies pour ce fichier.

TP 4 : Droits Classiques et Création d'un Répertoire Collaboratif

Objectif :

Dans ce TP, vous allez manipuler les droits d'accès standards des fichiers et des répertoires sous Debian. Vous allez créer un répertoire collaboratif où plusieurs utilisateurs pourront travailler ensemble, tout en contrôlant les permissions d'accès de manière précise.

1. **Manipulation des droits classiques sur des fichiers et répertoires :**
    1. Créez un fichier appelé `document.txt` dans votre répertoire personnel.
    2. Modifiez les permissions de ce fichier pour que :
        - Le propriétaire ait les droits de lecture et d'écriture.
        - Le groupe ait les droits de lecture uniquement.
        - Les autres utilisateurs n'aient aucun droit d'accès.
    3. Vérifiez que les permissions ont bien été appliquées avec la commande `ls -l`.
    4. Changez le propriétaire du fichier `document.txt` à un autre utilisateur présent sur le système.
    5. Modifiez à nouveau les permissions pour que le propriétaire ait tous les droits (lecture, écriture, et exécution), et que le groupe et les autres utilisateurs n'aient que des droits de lecture.

TP 4 : Droits Classiques et Création d'un Répertoire Collaboratif

2. **Création d'un répertoire collaboratif :**
    1. Créez un répertoire appelé `projet_collaboratif` dans votre répertoire personnel.
    2. Assurez-vous que ce répertoire est accessible pour :
        - Le propriétaire avec des droits complets (lecture, écriture, exécution).
        - Le groupe avec des droits complets (lecture, écriture, exécution).
        - Les autres utilisateurs avec des droits de lecture uniquement.
    3. Ajoutez plusieurs utilisateurs à un groupe spécifique qui sera en charge de collaborer dans ce répertoire.
    4. Modifiez les permissions pour que tous les membres de ce groupe puissent :
        - Lire, écrire, et exécuter dans le répertoire.
        - Les fichiers créés dans ce répertoire doivent automatiquement appartenir à ce groupe et suivre les mêmes permissions.
    5. Testez les permissions en créant des fichiers à l'intérieur du répertoire `projet_collaboratif` avec différents utilisateurs appartenant au groupe collaboratif, puis vérifiez les permissions de chaque fichier.
    6. Modifiez les permissions pour qu'un fichier spécifique dans ce répertoire ne soit modifiable que par le propriétaire, tout en restant accessible en lecture pour les autres membres du groupe.

TP 4 : Droits Classiques et Création d'un Répertoire Collaboratif

3. **Gestion des erreurs et des permissions :**
    1. Vérifiez les permissions des fichiers et répertoires à chaque étape avec la commande `ls -l`.
    2. Testez ce qui se passe si un utilisateur non membre du groupe essaie d'ajouter un fichier dans le répertoire collaboratif. Quelles sont les erreurs rencontrées ?
    3. Corrigez ces erreurs en modifiant les permissions du répertoire ou des fichiers selon les besoins du projet collaboratif.

UTOPIOS®
TRANSFORMATION DIGITALE INTELLIGENTE

Jour 3

Description du processus de démarrage

1. **Description du processus de démarrage (Boot Process)**

Le processus de démarrage dans Linux est un ensemble d'étapes cruciales qui commencent dès que l'ordinateur est allumé et se terminent par le chargement du système d'exploitation. Ce processus peut être décomposé en plusieurs étapes :

a) BIOS/UEFI

- Lorsque vous allumez la machine, le **BIOS** (Basic Input/Output System) ou **UEFI** (Unified Extensible Firmware Interface) s'initialise.
- Le BIOS/UEFI effectue une série de tests matériels appelés **POST** (Power-On Self-Test) pour vérifier l'intégrité de l'équipement.
- Ensuite, il localise un périphérique amorçable, généralement le disque dur, qui contient le **secteur de démarrage principal** (MBR) ou **GPT**.

Description du processus de démarrage

b) Bootloader (GRUB)

- Une fois que le BIOS ou UEFI a localisé un disque contenant un secteur d'amorçage valide, il passe le contrôle au bootloader.
- Sur Debian 12, le gestionnaire de boot le plus couramment utilisé est **GRUB** (GRand Unified Bootloader).
- GRUB lit son fichier de configuration et propose un menu permettant à l'utilisateur de sélectionner le noyau Linux ou un autre système d'exploitation à démarrer (si plusieurs systèmes sont installés).
- Le fichier de configuration de GRUB est généralement situé dans `/boot/grub/grub.cfg`. Il contient des informations telles que les noyaux disponibles et les options de démarrage (par exemple, le mode recovery).

c) Chargement du noyau (Kernel)

- Une fois qu'un noyau a été sélectionné (par défaut ou par l'utilisateur), GRUB charge ce noyau en mémoire.
- Le noyau Linux s'initialise, puis charge les modules nécessaires pour interagir avec les matériels de la machine (drivers pour disque, réseau, etc.).

d) Initramfs

- **Initramfs** (initial RAM filesystem) est une image temporaire montée en mémoire par le noyau. Elle contient les fichiers et modules nécessaires pour permettre au noyau de monter le système de fichiers racine (root filesystem).
- L'image initramfs est également chargée par GRUB au démarrage.

Description du processus de démarrage

e) Systemd

- Après le montage du système de fichiers racine, le processus **init** (le premier processus à démarrer sous Linux) prend le relais. Sur Debian 12, c'est **systemd** qui est responsable de gérer ce processus.
- Systemd orchestre le démarrage de tous les services et démons nécessaires pour faire fonctionner le système (comme les interfaces réseau, les services SSH, etc.).
- Systemd lit une série d'unités (unit files) pour lancer les services et configure les cibles (targets) qui déterminent l'état final du système (multi-user, graphical, etc.).

Gestionnaires de boot : GRUB

a) Rôle de GRUB

- GRUB est un gestionnaire de démarrage multi-plateforme utilisé pour charger le noyau Linux et d'autres systèmes d'exploitation installés sur un système.
- Il peut amorcer des noyaux Linux ou des systèmes comme Windows, et permet à l'utilisateur de choisir entre plusieurs options de démarrage (comme un noyau spécifique ou le mode de récupération).

b) Structure de GRUB

- GRUB est divisé en plusieurs parties :
  - **Stage 1** : C'est le tout premier code exécuté à partir du secteur d'amorçage (MBR ou GPT). Il est responsable de charger la deuxième étape.
  - **Stage 2** : Cette étape charge le système de fichiers et affiche le menu GRUB.
  - **Modules** : GRUB a également des modules qui peuvent être utilisés pour ajouter des fonctionnalités supplémentaires (comme le support réseau pour un démarrage PXE).

Gestionnaires de boot : GRUB

c) Configuration de GRUB

- Le fichier de configuration principal de GRUB se trouve dans `/boot/grub/grub.cfg`. Il ne doit pas être modifié directement, car il est automatiquement généré à partir du fichier `/etc/default/grub` et des scripts dans `/etc/grub.d/`.
- Pour mettre à jour la configuration de GRUB, vous devez modifier le fichier `/etc/default/grub` et exécuter la commande `update-grub`. Cela régénérera le fichier de configuration final.

d) Personnalisation de GRUB

- Vous pouvez personnaliser GRUB pour changer le délai avant qu'il ne sélectionne automatiquement une option par défaut, ajouter des paramètres au noyau (comme des options de dépannage), ou même ajouter des fonds d'écran au menu de démarrage.
- Exemple : pour réduire le délai du menu GRUB, vous pouvez modifier la ligne `GRUB_TIMEOUT=5` dans `/etc/default/grub` et exécuter `update-grub`.

e) Utilisation en mode dépannage

- GRUB peut être utilisé pour démarrer en **mode single-user** ou en **mode de récupération** pour dépanner un système cassé.
- Vous pouvez également modifier temporairement les paramètres du noyau au démarrage depuis l'interface GRUB pour par exemple désactiver un module problématique ou ajouter un mode verbose.

Gestionnaires de boot : GRUB

f) Gestion du dual boot avec GRUB

- Si vous avez plusieurs systèmes d’exploitation (comme Debian et Windows), GRUB détecte automatiquement les autres systèmes et les affiche dans son menu de démarrage.
- L'utilisateur peut configurer GRUB pour démarrer automatiquement un système spécifique ou définir un délai d'attente avant le démarrage automatique.

Exercice - GRUB

Objectif :

Ajouter une image de fond personnalisée dans le menu GRUB pour améliorer l'apparence du gestionnaire de démarrage sur Debian 12.

En tant qu'administrateur système, vous souhaitez personnaliser l'apparence de GRUB sur votre machine Debian 12 en ajoutant une image de fond dans le menu de démarrage.

1. Choisir une image de fond :

- Sélectionnez une image (au format PNG) qui sera utilisée comme fond d'écran pour le menu GRUB.

2. Copier l'image dans le bon répertoire :

- Placez l'image choisie dans le répertoire adéquat sur votre système pour que GRUB puisse y accéder.

3. Modifier le fichier de configuration de GRUB :

- Configurez GRUB pour qu'il utilise cette image comme fond dans son menu d'amorçage.

4. Tester et vérifier l'affichage :

- Appliquez les changements et redémarrez votre machine pour vérifier si l'image apparaît correctement dans le menu de démarrage.

Gestion des unités service et cible (target) dans systemd

Systemd est le gestionnaire de démarrage et des services par défaut sur Debian 12. Il remplace l'ancien système **SysVinit** et permet de gérer les services et d'autres aspects du système de manière plus flexible et efficace. Les unités de systemd (services, cibles, etc.) jouent un rôle central dans l'initiation et la gestion des composants du système.

1. Qu'est-ce qu'une unité (unit) dans systemd ?

Une **unité** dans systemd est un fichier de configuration qui décrit une ressource ou un service à gérer. Il peut s'agir d'un service, d'un périphérique, d'un montage de disque, d'un socket, ou même d'une cible regroupant plusieurs unités.

Les unités sont stockées dans des fichiers ayant l'extension **.service**, **.target**, **.socket**, etc., et sont situées dans les répertoires suivants :

- `/etc/systemd/system/` : pour les unités personnalisées ou spécifiques à l'utilisateur.
- `/lib/systemd/system/` : pour les unités installées par les paquets logiciels.
- `/usr/lib/systemd/system/` : pour les configurations par défaut du système.

2. Unités de service (.service)

Les unités **service** gèrent les services démarrés, arrêtés, redémarrés et surveillés par systemd. Chaque unité de service contient une définition du service, ses dépendances et ses actions spécifiques.

Gestion des unités service et cible (target) dans systemd

Structure d'une unité de service

Voici un exemple d'unité de service `apache2.service` :

```
[Unit]
Description=The Apache HTTP Server
After=network.target

[Service]
ExecStart=/usr/sbin/apachectl start
ExecStop=/usr/sbin/apachectl stop
ExecReload=/usr/sbin/apachectl graceful
Type=forking

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

Gestion des unités service et cible (target) dans systemd

Commandes pour gérer les services

- **Démarrer un service** :

```
sudo systemctl start apache2
```

- **Arrêter un service** :

```
sudo systemctl stop apache2
```

- **Redémarrer un service** :

```
sudo systemctl restart apache2
```

- **Vérifier le statut d'un service** :

```
sudo systemctl status apache2
```

- **Activer un service au démarrage** :

```
sudo systemctl enable apache2
```

- **Désactiver un service au démarrage** :

```
sudo systemctl disable apache2
```

Gestion des unités service et cible (target) dans systemd

3. Unités cible (target)

Les **cibles** (targets) sont des unités spéciales qui regroupent d'autres unités afin de coordonner le démarrage de groupes de services ou d'unités. Les targets ne démarrent pas de services eux-mêmes, mais organisent le démarrage des services en fonction de l'état voulu du système.

Principales cibles dans systemd

- **default.target** : C'est la cible par défaut qui est atteinte au démarrage du système. Sur un système en mode texte, elle est souvent liée à `multi-user.target`.
- **graphical.target** : Cible utilisée pour démarrer le système avec une interface graphique.
- **multi-user.target** : Cible pour un système multi-utilisateur en mode texte, souvent utilisé pour les serveurs.
- **rescue.target** : Cible pour le mode de récupération (single-user mode) pour effectuer des réparations.
- **shutdown.target** : Cible utilisée pour éteindre proprement le système.

Utilisation des cibles

Les cibles permettent de spécifier des états systèmes plus larges, comme démarrer tous les services nécessaires pour l'interface graphique, ou préparer le système pour un shutdown.

Gestion des unités service et cible (target) dans systemd

4. Relation entre services et cibles

Les services peuvent être associés à des cibles via la section `[Install]` des fichiers de service. Par exemple, un service peut être démarré automatiquement avec la cible `multi-user.target` si son fichier d'unité contient la directive `WantedBy=multi-user.target`.

De cette manière, **systemd** coordonne le démarrage, l'arrêt et la gestion de différents services et groupes de services en fonction des besoins de l'administrateur système.

Exercice

Objectif :

Comprendre et manipuler les unités de service et les cibles (targets) dans systemd sur Debian 12. Cet exercice vous permettra de créer, modifier et gérer des services, ainsi que d'utiliser les cibles pour gérer les groupes de services au démarrage.

En tant qu'administrateur système, vous devez gérer divers services sur un serveur Debian 12. Certains services doivent démarrer automatiquement, d'autres doivent être désactivés, et vous devez également modifier le comportement des cibles pour répondre aux besoins spécifiques du système.

1. Créer une unité de service personnalisée :

- Créez un service systemd personnalisé qui exécute un script simple au démarrage.
- Le script doit afficher un message dans les logs système.

2. Activer et démarrer le service :

- Activez ce service pour qu'il démarre automatiquement au démarrage du système.
- Vérifiez ensuite que le service fonctionne correctement en consultant les journaux.

Exercice

3. Désactiver un service existant :

- Identifiez un service actuellement actif sur le système qui n'est pas nécessaire pour cet exercice.
- Désactivez et arrêtez ce service pour qu'il ne démarre plus automatiquement.

4. Changer la cible par défaut de systemd :

- Modifiez la cible par défaut pour passer du mode graphique au mode multi-utilisateur en mode texte uniquement.
- Redémarrez le système pour vérifier que le changement est effectif.

5. Créer une nouvelle cible personnalisée :

- Créez une cible systemd personnalisée qui inclut votre service créé à l'étape 1.
- Démarrez cette cible manuellement et assurez-vous que votre service se lance correctement avec cette nouvelle cible.

Dépannage en mode Rescue ou Emergency

Le **mode Rescue** et le **mode Emergency** sont deux options importantes de **systemd** pour le dépannage du système sur Debian 12. Ces modes sont utilisés lorsque le système ne peut pas démarrer normalement ou lorsque des problèmes critiques doivent être résolus (comme la corruption du système de fichiers, une mauvaise configuration, ou des erreurs empêchant les services de fonctionner correctement).

1. Mode Rescue (rescue.target)

Le **mode Rescue** est similaire au mode single-user dans les systèmes basés sur **SysVinit**. Il permet d'obtenir une session root avec le système de fichiers monté en lecture-écriture mais ne démarre que les services essentiels, comme le montage des partitions.

Utilisation typique :

- Problèmes liés à des services critiques empêchant le système de démarrer correctement.
- Configuration réseau cassée.
- Problèmes de mot de passe root.

Dépannage en mode Rescue ou Emergency

Étapes pour accéder au mode Rescue :

1. **Redémarrer et accéder à GRUB** :
   - Démarrez ou redémarrez le système.
   - Au menu GRUB, appuyez sur la touche `e` pour éditer les options de démarrage.
2. **Modifier les options de démarrage** :
   - Localisez la ligne qui commence par `linux`, qui contient le chemin vers le noyau à démarrer.
   - Ajoutez `systemd.unit=rescue.target` à la fin de cette ligne. Cela indiquera à systemd de démarrer en mode rescue.

```
linux /boot/vmlinuz-5.10.0-8-amd64 root=/dev/sda1 ro quiet systemd.unit=rescue.target
```

3. **Démarrer en mode Rescue** :
   - Appuyez sur `Ctrl + X` ou `F10` pour démarrer avec les nouvelles options.
   - Vous serez redirigé vers un shell avec des privilèges root après avoir fourni le mot de passe root.

Dépannage en mode Rescue ou Emergency

4. **Résoudre les problèmes** :
   - En mode rescue, seules les partitions nécessaires sont montées. Vous pouvez effectuer des opérations comme :
     - **Modifier les fichiers de configuration** :

```
nano /etc/fstab  # Exemple pour corriger une erreur dans les montages
```

     - **Vérifier les systèmes de fichiers** (si nécessaire) :

```
fsck /dev/sda1
```

5. **Redémarrer le système** une fois les correctifs apportés :

```
reboot
```

Dépannage en mode Rescue ou Emergency

2. Mode Emergency (emergency.target)

Le **mode Emergency** est encore plus minimal que le mode Rescue. Il ne monte pas les systèmes de fichiers automatiquement, et seul le shell root est disponible. Ce mode est utilisé lorsque les systèmes de fichiers ne peuvent pas être montés, ou lorsqu'une erreur critique empêche le montage des partitions ou le lancement des services de base.

Utilisation typique :

- Systèmes de fichiers corrompus ou non montables.
- Problèmes critiques de montage dans `/etc/fstab`.
- Problèmes de démarrage du noyau.

Dépannage en mode Rescue ou Emergency

Étapes pour accéder au mode Emergency :

1. **Redémarrer et accéder à GRUB** :
   - Comme pour le mode Rescue, redémarrez et accédez au menu GRUB en appuyant sur `e`.
2. **Modifier les options de démarrage** :
   - Localisez la ligne commençant par `linux`.
   - Ajoutez `systemd.unit=emergency.target` à la fin pour démarrer directement en mode Emergency.

     ```
     linux /boot/vmlinuz-5.10.0-8-amd64 root=/dev/sda1 ro quiet systemd.unit=emergency.target
     ```
3. **Démarrer en mode Emergency** :
   - Appuyez sur `Ctrl + X` ou `F10`.
   - Vous accéderez à un shell minimal sans montage des partitions. Le système attendra le mot de passe root avant de vous accorder l'accès.

Dépannage en mode Rescue ou Emergency

4. **Effectuer des actions de dépannage** :
   - Comme aucun système de fichiers n'est monté par défaut, vous devrez peut-être les monter manuellement :

```
mount -o rw,remount /  # Remonter la partition root en lecture/écriture
mount /dev/sda1 /mnt   # Monter des partitions supplémentaires
```

   - **Vérifier et réparer les systèmes de fichiers** si nécessaire :

```
fsck /dev/sda1
```

5. **Redémarrer le système** après avoir résolu le problème :

```
reboot
```

Dépannage en mode Rescue ou Emergency

3. Comparaison entre mode Rescue et mode Emergency

Caractéristique	Mode Rescue	Mode Emergency
Accès réseau	Non (seulement les services critiques démarrent)	Non (aucun service démarré)
Systèmes de fichiers	Les systèmes de fichiers critiques sont montés	Aucun système de fichiers n'est monté
Démarrage de services	Quelques services essentiels sont démarrés	Aucun service n'est démarré
Utilisation	Problèmes liés aux services ou à des configurations	Problèmes critiques liés au noyau ou aux systèmes de fichiers

Dépannage en mode Rescue ou Emergency

4. Cas pratiques de dépannage en mode Rescue et Emergency

Exemple 1 : Problème de fichier `/etc/fstab` corrompu

Si le fichier `/etc/fstab` contient une erreur, le système peut échouer à démarrer en mode normal, bloquant sur le montage des partitions. Le mode Emergency vous permettra de corriger cette erreur.

1. Démarrez en mode Emergency.
2. Montez le système de fichiers root en lecture/écriture :

```
mount -o remount,rw /
```

3. Corrigez le fichier `/etc/fstab` avec un éditeur de texte.
4. Redémarrez le système.

Dépannage en mode Rescue ou Emergency

Exemple 2 : Problème avec un service réseau empêchant le démarrage

Si un service réseau est mal configuré et empêche le démarrage du système, le mode Rescue permet d'intervenir.

1. Démarrez en mode Rescue.
2. Désactivez le service problématique :

```
systemctl disable nom_du_service
```

3. Redémarrez et vérifiez que le système démarre correctement.

Définition des processus, des threads et de l'ordonnancement

1. **Définition des processus**
   - **Qu'est-ce qu'un processus ?**
     - Un processus est une instance d'un programme en exécution. Il inclut des éléments comme le code, les données en cours, et les ressources système nécessaires pour son exécution. Dans Debian 12, chaque processus a un **PID (Process ID)** unique qui permet de le gérer au niveau du système.
   - **Création de processus :**
     - Un processus peut être créé via la commande `fork()` pour générer un processus fils. Chaque processus a un processus parent.
     - Commandes utiles :
       - `ps -aux` : Affiche la liste des processus.
       - `top` ou `htop` : Outils interactifs pour surveiller les processus en temps réel.
   - **Gestion des processus :**
     - **Exécuter un processus en arrière-plan** : Utilisation du `&` à la fin d'une commande.
     - **Tuer un processus** : Utilisation de `kill` suivi du PID (`kill -9 <PID>` pour forcer la fermeture).
     - **Changer les priorités** : Utilisation des commandes `nice` et `renice`.

Définition des processus, des threads et de l'ordonnancement

2. **Définition des threads**
   - **Qu'est-ce qu'un thread ?**
     - Un thread est la plus petite unité d'exécution dans un processus. Un processus peut contenir plusieurs threads partageant les mêmes ressources (mémoire, fichiers ouverts, etc.). Sous Linux, Debian 12 gère les threads via le modèle **N:M**, où N threads sont mappés sur M noyaux.
   - **Différence processus vs thread :**
     - Un processus a son propre espace mémoire, tandis que les threads partagent la mémoire du processus parent.
     - Les threads sont plus légers et permettent un traitement parallèle plus rapide au sein du même processus.
   - **Commande liée :**
     - `ps -eLf` : Liste tous les threads du système.

Définition des processus, des threads et de l'ordonnancement

3. **Ordonnancement**
   - **Qu'est-ce que l'ordonnancement ?**
     - L'ordonnancement est le processus par lequel le noyau Linux attribue du temps CPU aux processus et aux threads. L'algorithme d'ordonnancement détermine quel processus ou thread aura accès au CPU à un moment donné.
   - **Types d'ordonnancement :**
     - **Ordonnancement préemptif** : Le noyau peut interrompre un processus en cours d'exécution pour donner la priorité à un autre, plus urgent.
     - **Ordonnancement non préemptif** : Un processus ne peut pas être interrompu tant qu'il n'a pas libéré volontairement le CPU.
   - **Classes d'ordonnancement dans Linux (Debian 12) :**
     - **SCHED_OTHER** : Classe par défaut pour les processus normaux.
     - **SCHED_FIFO** et **SCHED_RR** : Classes pour les processus temps réel, avec des priorités plus strictes.

top : Surveillance en temps réel des processus

- **Description :**
  La commande `top` affiche une liste des processus actifs en temps réel. Elle fournit des informations sur l'utilisation du CPU, de la mémoire, et d'autres statistiques importantes du système.
- **Exemple :**

```
top
```

- **Colonnes importantes** :
  - **PID** : Identifiant unique du processus.
  - **USER** : Nom de l'utilisateur propriétaire du processus.
  - **PR** : Priorité du processus.
  - **%CPU** : Pourcentage d'utilisation du CPU.
  - **%MEM** : Pourcentage d'utilisation de la mémoire.
  - **TIME+** : Temps total d'utilisation du CPU par le processus.

top : Surveillance en temps réel des processus

- **Utilisation avancée :**
- Pour trier les processus par utilisation de la mémoire :

```
top
(puis appuyer sur Shift + M)
```

- Pour tuer un processus directement depuis `top` :
  - Appuyer sur `k`, puis entrer le **PID** du processus à tuer.

htop : Version améliorée de top

- **Description :**
  `htop` est une version plus conviviale de `top`. Elle permet de naviguer de manière interactive dans la liste des processus avec des touches directionnelles et d'obtenir une vue plus visuelle de l'utilisation du CPU et de la mémoire.
- **Exemple :**

```
htop
```

  - **Avantages par rapport à `top` :**
    - Interface colorée et intuitive.
    - Utilisation de la souris pour sélectionner et interagir avec les processus.
    - Permet de voir facilement les différents cœurs du CPU et leur utilisation.
- **Utilisation avancée :**
  - Pour trier les processus par CPU ou mémoire, il suffit d'utiliser les touches de direction et de cliquer sur les en-têtes des colonnes.
  - Pour tuer un processus : Sélectionner le processus avec les flèches, puis appuyer sur `F9`.

pstree : Visualisation des processus en forme d'arbre

- **Description :**
  La commande `pstree` affiche les processus actifs sous forme d'arbre, ce qui permet de voir les relations parent-enfant entre eux. C'est un outil pratique pour comprendre la hiérarchie des processus.
- **Exemple :**

```
pstree
```

  - Cela affiche tous les processus en cours avec leur hiérarchie. Par exemple, si tu exécutes un programme à partir de ton terminal, il apparaîtra comme un enfant de ton processus shell (comme `bash`).
- **Utilisation avancée :**
  - Pour afficher l'arbre avec les PID :

```
pstree -p
```

  - Pour limiter l'affichage à un utilisateur spécifique :

```
pstree <nom_utilisateur>
```

ps : Instantané des processus

- **Description :**
  La commande `ps` affiche une liste instantanée des processus. Contrairement à `top` et `htop`, `ps` ne met pas à jour l'affichage en temps réel, mais fournit un instantané à l'instant de la commande.
- **Exemple :**

```
ps aux
```

- **Options courantes** :
  - `a` : Affiche tous les processus d'autres utilisateurs.
  - `u` : Affiche les informations sous un format détaillé (utilisateur, CPU, mémoire, etc.).
  - `x` : Affiche les processus n'ayant pas de terminal associé (par exemple, les processus en arrière-plan).

ps : Instantané des processus

- **Exemple de sortie** :

```
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.0  0.1  16124  2328 ?        Ss   09:44   0:03 /sbin/init
user        1931  0.0  0.2  56724  4144 pts/0    Ss   09:45   0:00 bash
user        2112  0.0  0.5  94876  9232 pts/0    R+   10:00   0:00 ps aux
```

- **Colonnes importantes** :
  - **PID** : Identifiant du processus.
  - **USER** : Utilisateur propriétaire du processus.
  - **%CPU** : Pourcentage d'utilisation du CPU.
  - **%MEM** : Pourcentage d'utilisation de la mémoire.
  - **COMMAND** : Commande exécutée par le processus.

ps : Instantané des processus

- **Utilisation avancée** :
- Pour lister les processus d'un utilisateur spécifique :

```
ps -u <nom_utilisateur>
```

- Pour afficher les threads en cours d'exécution :

```
ps -eLf
```

- Pour trouver un processus spécifique par nom :

```
ps aux | grep <nom_du_processus>
```

Utilisation combinée pour une analyse approfondie

- **Trouver des processus et tuer :**
  Tu peux combiner plusieurs commandes pour effectuer des analyses et prendre des actions. Par exemple, pour identifier un processus consommateur de mémoire et le tuer :

```
ps aux --sort=-%mem | head
kill -9 <PID>
```

- **Vérification de l'arbre des processus avec détails :**
  Pour voir la hiérarchie d'un processus spécifique, par exemple, avec son PID, utilise `pstree` :

```
pstree -p <PID>
```

Gestion des signaux (kill)

Les signaux sont des notifications envoyées aux processus pour leur indiquer qu'ils doivent effectuer une action spécifique. Chaque signal a un nom et un numéro unique, et voici un aperçu détaillé de certains signaux courants ainsi que leur rôle exact.

1. SIGTERM (Signal 15) : Ce signal demande poliment à un processus de s'arrêter. C'est le signal de terminaison par défaut. Le processus reçoit une demande de terminaison, ce qui lui permet d'exécuter une procédure d'arrêt propre (comme enregistrer des données ou libérer de la mémoire).

- Valeur : 15
- Usage : Utilisé lorsque l'on souhaite arrêter un processus sans forcer, en lui donnant le temps de faire ses opérations de nettoyage.

2. SIGKILL (Signal 9) : Ce signal force l'arrêt immédiat d'un processus, sans lui laisser l'opportunité d'effectuer une procédure de nettoyage. Cela peut être nécessaire si le processus ne répond pas ou est bloqué.

- Valeur : 9
- Usage : Utilisé lorsque le processus ne répond pas au SIGTERM ou s'il est bloqué.

Gestion des signaux (kill)

3. SIGHUP (Signal 1) : Ce signal indique à un processus qu'il doit recharger sa configuration. Il est souvent utilisé pour forcer les services à relire leurs fichiers de configuration sans redémarrer.

- Valeur : 1
- Usage : Très utile pour recharger les paramètres d'un serveur sans redémarrage complet.

4. SIGINT (Signal 2) : Ce signal interrompt le processus en cours, en simulant un appui sur CTRL+C dans le terminal. Il est souvent utilisé pour arrêter les processus en avant-plan.

- Valeur : 2
- Usage : Souvent utilisé pour arrêter des scripts en cours d'exécution dans le terminal.

5. SIGSTOP (Signal 19) et SIGCONT (Signal 18): Ces signaux permettent de suspendre (stop) et de reprendre (continue) un processus.

- Valeur de SIGSTOP : 19 (il arrête immédiatement le processus, sans possibilité de l'intercepter pour sauvegarder son état).
- Valeur de SIGCONT : 18 (permet de reprendre l'exécution d'un processus suspendu).

Tâches avant et arrière plans (jobs, bg, fg...)

1. Avant-Plan vs Arrière-Plan

- Avant-Plan (Foreground) : Lorsque vous exécutez une commande normalement dans le terminal, elle s'exécute en avant-plan. Vous ne pouvez rien faire d'autre dans le terminal jusqu'à ce que la commande se termine.
- Arrière-Plan (Background) : Une commande peut être exécutée en arrière-plan pour libérer le terminal pour d'autres tâches. Cela est utile pour des processus longs ou continus.
- Pour lancer une commande en arrière-plan, on ajoute un & à la fin de la commande. Par exemple :

2. Commandes Clés pour la Gestion des Tâches

jobs : Afficher les Tâches en Cours

- La commande jobs affiche la liste des tâches en cours dans le terminal actuel, avec leur statut (en cours, suspendue, etc.).
- Chaque tâche est associée à un numéro de tâche (différent du PID) représenté par %n (par exemple, %1, %2).

bg : Envoyer une Tâche en Arrière-Plan

- bg (background) envoie une tâche suspendue en arrière-plan, permettant à celle-ci de continuer à s'exécuter sans bloquer le terminal.
- Pour envoyer une tâche suspendue en arrière-plan, on utilise bg suivi du numéro de la tâche (%n).

Tâches avant et arrière plans (jobs, bg, fg…)

fg : Ramener une Tâche en Avant-Plan

- fg (foreground) ramène une tâche en arrière-plan à l'avant-plan, permettant d'interagir directement avec elle.
- On utilise fg suivi du numéro de tâche (%n) pour sélectionner la tâche souhaitée.

3. Suspension et Reprise des Tâches avec CTRL+Z, bg, et fg

- Suspension (CTRL+Z) : Lorsqu'une commande s'exécute en avant-plan, vous pouvez la suspendre temporairement en appuyant sur CTRL+Z. Cela arrête le processus sans le terminer.
- Reprise en Arrière-Plan (bg) : Après avoir suspendu une tâche, vous pouvez la reprendre en arrière-plan avec bg.
- Reprise en Avant-Plan (fg) : Pour reprendre une tâche suspendue en avant-plan, utilisez fg.

4. Utiliser disown pour Détacher une Tâche du Terminal

Une tâche lancée en arrière-plan est généralement liée au terminal d'où elle est lancée. Si ce terminal est fermé, la tâche s'arrête aussi. disown permet de "détacher" une tâche pour qu'elle continue de s'exécuter même si le terminal est fermé.

Planification de tâches

La planification des tâches sous Linux (comme Debian 12) est une compétence essentielle pour automatiser l'exécution de commandes ou de scripts à des moments précis. Cette planification repose principalement sur deux outils : cron pour les tâches récurrentes et at pour les tâches ponctuelles.

1. Planification Récurrente avec `cron

Le service cron permet de planifier des tâches récurrentes, comme l'exécution quotidienne d'un script de sauvegarde ou une vérification hebdomadaire des mises à jour. Les tâches cron sont définies dans le fichier crontab, qui spécifie quand et quelle commande exécuter.

- Une entrée crontab suit cette structure : minute heure jour mois jour_de_la_semaine commande
- Chaque champ correspond à un intervalle de temps :
  - minute : Minute exacte (de 0 à 59)
  - heure : Heure exacte (de 0 à 23)
  - jour : Jour du mois (de 1 à 31)
  - mois : Mois (de 1 à 12 ou noms abrégés : jan, feb, etc.)
  - jour_de_la_semaine : Jour de la semaine (de 0 à 6, où 0 est dimanche, ou noms abrégés : mon, tue, etc.)
- Pour éditer le fichier crontab de l'utilisateur actuel, utilisez : crontab -e

Planification de tâches

2. Planification Ponctuelle avec at

La commande at est utilisée pour exécuter une tâche une seule fois, à un moment précis. Contrairement à cron, at ne nécessite pas de configuration récurrente et est idéal pour les tâches uniques.

Utilisation de Base de at

1. Lancer at pour une commande à un moment donné : at HH:MM

Planification de tâches

3. Contrôle des Journaux de cron et at

Pour vérifier si une tâche s'est bien exécutée ou si elle a rencontré une erreur, consultez les logs de cron et at.

- Vérifier les logs de cron : cat /var/log/syslog | grep cron
- Vérifier les erreurs :
  Vous pouvez rediriger les erreurs de cron vers un fichier spécifique en utilisant 2> dans la commande : 0 3 * * * /path/to/script.sh 2>> /path/to/error.log

UTOPIOS®
TRANSFORMATION DIGITALE INTELLIGENTE

Jour 4

Compresser et décompresser un fichiers (gzip, bzip2, lzma, lzw)

1. gzip: gzip est un utilitaire de compression largement utilisé dans les systèmes Linux. Il réduit la taille des fichiers en supprimant les redondances. La compression avec gzip se termine par l'extension .gz.

2. bzip2: bzip2 offre une meilleure compression que gzip, mais peut être plus lent. Les fichiers compressés avec bzip2 utilisent l'extension .bz2.

3. lzma: lzma est un algorithme de compression offrant de très bons ratios de compression, mais peut être plus lent et consommateur en ressources. Les fichiers compressés avec lzma sont suffixés en .lzma.

Gestion des Archives

tar et pax permettent de regrouper plusieurs fichiers en une seule archive sans compression.

1. tar: tar est largement utilisé pour créer des archives, que l'on appelle souvent des "fichiers tarball". Par défaut, tar n'applique pas de compression.

2. pax: pax est un autre outil d'archivage, moins couramment utilisé mais standardisé.

Archivage avec Compression

- En combinant tar avec des outils de compression, nous pouvons créer des archives compressées.
- tar avec gzip
- tar avec bzip2
- tar avec lzma

Sujet d'Exercice : Gestion de fichiers de logs pour une application

Votre tâche est de préparer et gérer une série de fichiers de logs générés par une application pour l'archivage et la compression. Ces logs sont stockés dans plusieurs dossiers, et l'objectif est de les organiser de manière efficace.

Objectifs :

1. Compresser les fichiers de log en utilisant divers outils de compression (gzip, bzip2, lzma).
2. Créer des archives pour regrouper ces fichiers, avec et sans compression.
3. Automatiser la gestion des archives pour une utilisation future.

Scénario :

Votre entreprise souhaite conserver des logs d'application de manière organisée, en réduisant l'espace disque occupé et en préparant ces logs pour des audits périodiques. Pour cela, suivez les étapes ci-dessous.

Sujet d'Exercice : Gestion de fichiers de logs pour une application

1. Création et compression de fichiers
   - Dans votre répertoire personnel, créez un dossier logs_archive et ajoutez-y trois sous-dossiers : app1_logs, app2_logs, et app3_logs.
   - Dans chaque sous-dossier, créez 5 fichiers de log de test (par exemple log1.txt, log2.txt, etc.) et remplissez-les avec du contenu aléatoire en utilisant la commande suivante :

```
for i in {1..5}; do echo "Log data for appX" > appX_logs/log$i.txt; done
```

- Compressez chacun de ces fichiers en utilisant les outils suivants :
- gzip pour les fichiers dans app1_logs
- bzip2 pour les fichiers dans app2_logs
- lzma pour les fichiers dans app3_logs

2. Création d'archives sans compression
   - Créez une archive all_logs.tar qui regroupe les dossiers app1_logs, app2_logs, et app3_logs sans compression. Vérifiez son contenu pour vous assurer que tous les fichiers sont bien inclus.

Sujet d'Exercice : Gestion de fichiers de logs pour une application

3. Création d'archives avec compression
   - Créez les archives suivantes :
   - Une archive all_logs.tar.gz contenant tous les dossiers compressée avec gzip.
   - Une archive all_logs.tar.bz2 compressée avec bzip2.
   - Une archive all_logs.tar.lzma compressée avec lzma.
4. Extraction et vérification des archives
   - Créez un dossier extracted_logs et extrayez chaque archive (non compressée et compressée) dans ce dossier.
   - Vérifiez que tous les fichiers sont présents après extraction.
5. Automatisation de la gestion des archives
   - Écrivez un script backup_logs.sh qui :
   - Compresse les fichiers de chaque dossier de log selon les formats définis (gzip, bzip2, lzma).
   - Crée une archive compressée backup_logs_DATE.tar.gz avec la date du jour.
   - Sauvegarde cette archive dans le dossier /backup (vous pouvez créer ce dossier pour l'exercice).
   - Optionnel : Testez votre script en modifiant les fichiers et en recréant une archive compressée.

Présentation des dépôts Debian

Les dépôts Debian sont des collections de paquets logiciels organisés et hébergés pour une distribution Debian. Ces dépôts sont classés par sections (main, contrib, et non-free) et par versions (stable, testing, et unstable).

- Sections :
  - **Main** : Paquets entièrement libres, maintenus par la communauté Debian. Tous les logiciels nécessaires à un système Debian sont ici.
  - **Contrib** : Logiciels libres nécessitant des dépendances non libres (comme certains pilotes matériels).
  - **Non-free** : Logiciels sous des licences non libres, disponibles pour les utilisateurs mais sans les garanties de la section main.
- Versions :
  - **Stable** : La version stable est destinée aux environnements de production car elle est bien testée. Elle reçoit principalement des mises à jour de sécurité.
  - **Testing** : Contient des paquets qui seront inclus dans la prochaine version stable. Elle est plus à jour que stable, mais avec des risques de bugs résiduels.
  - **Unstable (Sid)** : Cette version est continuellement mise à jour avec les derniers logiciels, mais elle peut être instable et sujette aux bugs.

Ces dépôts sont gérés via le fichier /etc/apt/sources.list, qui contient les adresses des serveurs et les sections (par exemple, main, contrib) que le système utilise pour télécharger les logiciels.

Gestion des paquetages DEB

Le format de paquet DEB est la structure de paquets utilisée par Debian et ses dérivés. Différentes commandes permettent de gérer ces paquets :

- **APT (Advanced Package Tool)** : Interface en ligne de commande pour la gestion des paquets. Les commandes principales incluent :
  - apt update : Met à jour la liste des paquets.
  - apt upgrade : Installe les dernières versions des paquets.
  - apt install <package_name> : Installe un paquet spécifique.
  - apt remove <package_name> : Supprime un paquet.
- **Aptitude** : Un outil de gestion des paquets avec une interface en ligne de commande, similaire à APT mais offrant des options avancées pour gérer les dépendances.
  - dpkg : Commande de bas niveau pour installer et gérer les paquets DEB :
  - dpkg -i <package.deb> : Installe un paquet local.
  - dpkg -r <package_name> : Supprime un paquet.
  - dpkg -l : Liste les paquets installés.

Installation d’une application depuis une archive

Il est parfois nécessaire d’installer un logiciel en dehors des dépôts officiels, par exemple avec une archive tarball (souvent .tar.gz ou .tar.bz2). Voici les étapes typiques :

- Téléchargement et décompression :
  - Pour télécharger, on peut utiliser wget ou curl.
  - Pour décompresser : tar -xzf fichier.tar.gz ou tar -xjf fichier.tar.bz2, selon le type d’archive.
  - Installation : Selon le contenu de l’archive, il peut y avoir un script d’installation, souvent install.sh, ou un simple exécutable. Si c’est un script, exécutez-le avec ./install.sh. Parfois, il suffit de déplacer l’exécutable dans /usr/local/bin pour l’intégrer au système.

Cela peut être utile pour des logiciels plus spécifiques ou récents, non présents dans les dépôts Debian, mais attention aux problèmes de sécurité et de mises à jour manuelles.

Compilation et installation à partir de sources

La compilation permet d'installer des applications qui ne sont pas disponibles en paquets précompilés ou pour personnaliser certaines options.
Étapes générales :

1. **Installation des dépendances** : Avant de compiler, installez les bibliothèques nécessaires avec APT (par exemple, apt install build-essential pour les outils de base).
2. **Téléchargement des sources** : Les sources sont souvent compressées dans des fichiers .tar.gz ou .tar.bz2.
3. **Décompression** : Utilisez tar comme précédemment.
4. **Configuration** : La plupart des projets utilisent un script ./configure pour vérifier les dépendances et configurer les options d'installation.
5. **Compilation** : Une fois configurée, lancez la compilation avec make.
6. **Installation** : Avec sudo make install, les fichiers sont copiés vers les répertoires adéquats du système.

- Avantages et inconvénients :
  - Avantages : Plus de flexibilité pour personnaliser et optimiser les options d'installation.
  - Inconvénients : La compilation est plus complexe, prend du temps, et nécessite un suivi pour les mises à jour.

Exercice : Déployer et gérer une stack complète avec des outils variés

Contexte :

Vous devez préparer un serveur Debian pour héberger un environnement d'analyse de logs, incluant les éléments suivants :

- Installation et configuration d'un dépôt externe.
- Gestion de plusieurs paquets avec différents outils (apt, dpkg, aptitude).
- Installation d'une application depuis des sources (non disponible dans les dépôts) pour compléter l'environnement.

Étape 1 :

1. Ajouter et configurer un dépôt externe

- https://artifacts.elastic.co/GPG-KEY-elasticsearch
- https://artifacts.elastic.co/packages/8.x/apt

2. Installez ElasticSearch et assurez-vous qu'il fonctionne :

Exercice : Déployer et gérer une stack complète avec des outils variés

Étape 2 : Gestion des paquets via divers outils

1. Téléchargez et installez Logstash en tant que fichier .deb avec dpkg
2. Utilisez aptitude pour installer Kibana.
3. Démarrez Kibana et testez son fonctionnement

Étape 3 : Installation depuis des sources

1. Téléchargez les sources de Filebeat (collecteur de logs) https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-8.10.0-linux-x86_64.tar.gz
2. Configurez et installez Filebeat :
   - Copiez l'exécutable dans /usr/local/bin

Gestion des journaux avec `rsyslog` et `systemd-journald`

La gestion des journaux (logs) est un aspect crucial de l'administration système. Les journaux permettent de suivre les activités du système, des services, des utilisateurs, et de diagnostiquer les problèmes. Sur Debian, deux systèmes principaux gèrent les journaux : `rsyslog` et `systemd-journald`.

1. Présentation de `rsyslog`

- `rsyslog` (Rocket-fast System for Log Processing) est un démon classique de gestion des journaux sous Linux.
- Il collecte, analyse, et stocke les journaux système dans des fichiers locaux ou les transfère à des systèmes de gestion centralisés (via des protocoles comme syslog ou RELP).
- **Stockage flexible :** Par défaut, les journaux sont stockés dans `/var/log/`.
- **Filtrage et redirection :** Vous pouvez configurer `rsyslog` pour envoyer certains types de journaux vers des fichiers spécifiques ou des serveurs distants.
- **Compatibilité :** `rsyslog` est compatible avec les anciens formats syslog.

Structure des fichiers journaux avec `rsyslog` :

- Les fichiers journaux sont situés dans `/var/log/`. Quelques exemples :
  - `/var/log/syslog` : Contient les événements généraux du système.
  - `/var/log/auth.log` : Journaux liés à l'authentification.
  - `/var/log/kern.log` : Journaux du noyau.

Gestion des journaux avec `rsyslog` et `systemd-journald`

Configuration de `rsyslog` :

1. **Fichier principal de configuration :**
   - Situé dans `/etc/rsyslog.conf`.
   - Contient des directives pour le traitement des journaux.
2. **Exemple de règle simple :**
   - Pour enregistrer tous les journaux de niveau `error` dans un fichier personnalisé `/var/log/errors.log` :

```
*.err /var/log/errors.log
```

Gestion des journaux avec `rsyslog` et `systemd-journald`

3. **Activer l'envoi de journaux vers un serveur distant :**
   - Ajoutez cette ligne dans `/etc/rsyslog.conf` :

```
*.* @@remote-log-server:514
```

- **Explication :**
  - `*.*` : Capture tous les messages, tous niveaux de gravité.
  - `@@` : Indique un envoi sécurisé via TCP. Un seul `@` utilise UDP.

4. **Redémarrer `rsyslog` après modification :**

```
sudo systemctl restart rsyslog
```

Gestion des journaux avec `rsyslog` et `systemd-journald`

2. Présentation de `systemd-journald`

- **`systemd-journald`** est le gestionnaire de journaux intégré au système `systemd`. Il collecte les journaux des services `systemd`, des applications, et du noyau.
- **Stockage binaire :** Les journaux sont stockés dans un format binaire (pas des fichiers texte), ce qui permet des recherches rapides et des fonctionnalités avancées.
- **Consolidation :** Tous les journaux, y compris ceux du noyau, des services, et des utilisateurs, sont centralisés.
- **Outil de visualisation :** Utilise la commande `journalctl` pour afficher et interroger les journaux.

Gestion des journaux avec `rsyslog` et `systemd-journald`

Gestion des journaux avec `systemd-journald` :

1. **Affichage des journaux :**
   - Voir les journaux récents :

```
journalctl
```

- Filtrer par service (par exemple, `ssh`):

```
journalctl -u ssh.service
```

- Voir uniquement les erreurs :

```
journalctl -p err
```

2. **Fichiers de stockage des journaux :**
   - Par défaut, les journaux sont stockés dans :
     - **Volatile (temporaire) :** `/run/log/journal/`
     - **Persistant :** `/var/log/journal/` (si activé).

Gestion des journaux avec `rsyslog` et `systemd-journald`

3. **Activer le stockage persistant des journaux :**
   - Créez le répertoire pour les journaux persistants :

```
sudo mkdir -p /var/log/journal
```

   - Redémarrez le service :

```
sudo systemctl restart systemd-journald
```

4. **Configurer `journald` :**
   - Modifiez le fichier `/etc/systemd/journald.conf`.
   - Exemple : Limiter la taille maximale des journaux à 100 Mo :

```
SystemMaxUse=100M
```

Gestion des journaux avec `rsyslog` et `systemd-journald`

3. Comparaison `rsyslog` vs `systemd-journald`

Fonctionnalité	`rsyslog`	`systemd-journald`
Stockage	Fichiers texte dans `/var/log`	Format binaire
Filtrage	Basé sur des règles dans `rsyslog.conf`	Puissant avec `journalctl`
Envoi distant	Oui (syslog, RELP, etc.)	Nécessite un outil tiers
Persistant par défaut	Oui	Non
Performance	Adapté pour les gros volumes	Optimisé pour `systemd`

Gestion des journaux avec `rsyslog` et `systemd-journald`

4. Utilisation conjointe de `rsyslog` et `systemd-journald`

- Par défaut, sur les systèmes modernes (comme Debian 12), `journald` collecte les journaux et les transmet à `rsyslog`.
- Cela combine les avantages des deux systèmes :
  - `journald` pour le stockage temporaire et l'analyse rapide.
  - `rsyslog` pour le stockage persistant et l'envoi des journaux à un serveur distant.

Gestion des journaux avec `rsyslog` et `systemd-journald`

Cas pratiques :

1. **Afficher les erreurs système des dernières 24 heures avec `systemd-journald` :**

```
journalctl --since "24 hours ago" -p err
```

2. **Configurer `rsyslog` pour sauvegarder uniquement les journaux d'authentification :**
   - Modifiez `/etc/rsyslog.conf` pour inclure :

```
auth.* /var/log/auth-only.log
```

3. **Envoyer tous les journaux vers un serveur distant tout en les gardant locaux :**

- Avec `rsyslog`, configurez :

```
*.* /var/log/all-logs.log
*.* @@remote-server:514
```

4. **Activer la rotation des journaux :**
   - Utilisez `logrotate` pour gérer la taille et l'archivage des fichiers `/var/log/`.

logrotate

1. **Limiter la taille des fichiers de log :** Les fichiers de journaux dans `/var/log/` peuvent grossir rapidement et remplir le disque.
2. **Archiver les anciens journaux :** Sauvegarder les journaux anciens au cas où ils seraient nécessaires pour une analyse future.
3. **Supprimer les anciens journaux :** Nettoyer automatiquement les journaux trop vieux pour libérer de l'espace disque.
4. **Automatiser le processus :** `logrotate` s'exécute périodiquement (par exemple, une fois par jour) et applique les règles de rotation définies.

`logrotate` est un utilitaire Linux qui automatise la gestion des fichiers de journaux. Il est configuré à l'aide de fichiers contenant des règles (dans `/etc/logrotate.d/` et `/etc/logrotate.conf`).

logrotate

1. Les étapes de rotation

Lorsqu'un fichier de journal atteint un certain critère (par exemple, taille ou ancienneté) :

1. **Rotation :** `logrotate` renomme le fichier actuel (par exemple, de `syslog` à `syslog.1`).
2. **Archivage :** L'ancien fichier peut être compressé (en `.gz`) pour économiser de l'espace.
3. **Création d'un nouveau fichier :** Un fichier de log vide est créé pour continuer à recevoir les nouveaux journaux.
4. **Suppression :** Les journaux trop vieux ou dépassant un nombre défini sont supprimés.

logrotate

Configuration de `logrotate`

1. Fichier principal de configuration

- Le fichier principal est `/etc/logrotate.conf`. Il définit les règles globales et inclut les configurations spécifiques des journaux dans `/etc/logrotate.d/`.

Exemple de fichier `/etc/logrotate.conf` :

```
weekly                  # Rotation des logs une fois par semaine
rotate 4                # Conserver les 4 dernières archives
create                  # Créer un nouveau fichier après la rotation
compress                # Compresser les fichiers de log archivés
```

logrotate

2. Configurations spécifiques

- Les configurations spécifiques pour chaque service ou fichier de log se trouvent dans `/etc/logrotate.d/`.
- Exemple : fichier `/etc/logrotate.d/syslog` pour gérer `/var/log/syslog` :

```
/var/log/syslog {
    daily                # Rotation quotidienne
    rotate 7             # Conserver 7 jours de journaux
    compress             # Compresser les journaux archivés
    delaycompress        # Ne pas compresser immédiatement le dernier journal
    missingok            # Ne pas afficher d'erreur si le fichier est absent
    notifempty           # Ne rien faire si le fichier est vide
    create 0640 syslog adm  # Créer un nouveau fichier avec les permissions et le propriétaire spécifiés
    postrotate           # Commande exécutée après la rotation
        /usr/lib/rsyslog/rsyslog-rotate
    endscript
}
```

logrotate

Exemple pratique : Rotation des journaux pour un fichier personnalisé

1. **Créer un fichier de log fictif :**

```
sudo touch /var/log/myapp.log
```

2. **Configurer `logrotate` pour ce fichier :**
   - Créez un fichier `/etc/logrotate.d/myapp` :

```
/var/log/myapp.log {
    size 10M            # Rotation lorsque le fichier atteint 10 Mo
    rotate 5            # Conserver 5 anciennes copies
    compress            # Compresser les anciennes copies
    delaycompress       # Ne pas compresser immédiatement la copie la plus récente
    missingok           # Ne pas afficher d'erreur si le fichier est absent
    notifempty          # Ne rien faire si le fichier est vide
    create 0640 root root  # Créer un fichier vide avec les permissions spécifiées
    postrotate
        systemctl reload myapp.service # Recharger le service après la rotation
    endscript
}
```

logrotate

3. **Tester la configuration :**
   - Exécutez `logrotate` en mode test pour vérifier la configuration :

```
sudo logrotate -d /etc/logrotate.conf
```

   - **Explications :**
     - `-d` (debug) permet de voir ce que `logrotate` ferait sans effectuer les changements.
4. **Forcer la rotation :**
   - Pour tester la rotation immédiatement, utilisez :

```
sudo logrotate -f /etc/logrotate.conf
```

logrotate

Options courantes de configuration dans `logrotate`

Option	Description
`daily`	Rotation quotidienne des logs.
`weekly`	Rotation hebdomadaire des logs.
`size`	Effectuer la rotation quand le fichier atteint une certaine taille (ex. `10M`).
`rotate <n>`	Conserver uniquement les `<n>` dernières copies des logs.
`compress`	Compresser les fichiers archivés pour économiser de l'espace disque.
`delaycompress`	Ne pas compresser immédiatement le fichier le plus récent.
`missingok`	Ne pas afficher d'erreur si le fichier n'existe pas.
`notifempty`	Ne rien faire si le fichier est vide.
`create`	Créer un nouveau fichier après la rotation avec des permissions spécifiques.
`postrotate`	Exécuter une commande après la rotation (par exemple, recharger un service).

TP : Installation d'un Serveur Web et Gestion des Journaux

Objectif :

Installer un serveur web (Apache ou Nginx), configurer un journal dédié pour les logs, et mettre en place une rotation automatique des journaux pour une gestion optimale.

Étape 1 : Installation du serveur web

1. Installer le serveur web choisi (Apache ou Nginx).
2. Vérifier que le serveur est opérationnel avec sa configuration par défaut.

Étape 2 : Configuration des journaux

1. **Configurer un journal dédié :**
   - Modifier la configuration du serveur web pour rediriger les logs d'accès et d'erreur vers un fichier spécifique (exemple : `/var/log/webserver/access.log` et `/var/log/webserver/error.log`).
2. **Vérifier la génération des journaux :**
   - Générez des requêtes vers le serveur web (par exemple, en accédant à la page d'accueil) pour confirmer que les journaux sont bien enregistrés dans les fichiers dédiés.

TP : Installation d'un Serveur Web et Gestion des Journaux

Étape 3 : Rotation des journaux

1. **Créer une configuration de rotation :**
   - Dans `/etc/logrotate.d/`, créez un fichier de configuration spécifique pour le serveur web. Définissez les règles suivantes :
     - Rotation quotidienne ou hebdomadaire selon la taille des logs.
     - Conservation d'un nombre limité de versions (par exemple, les 5 dernières).
     - Compression automatique des anciens logs pour économiser de l'espace.
     - Rechargement du serveur web après chaque rotation pour recréer les fichiers de logs.
2. **Tester la configuration :**
   - Utilisez `logrotate` pour simuler et valider la rotation des journaux.
3. **Forcer une rotation :**
   - Lancez une rotation manuellement pour vérifier que les journaux sont archivés et compressés correctement.

UTOPIOS®
TRANSFORMATION DIGITALE INTELLIGENTE

Jour 5

Configuration du client réseau

La configuration du réseau est une étape cruciale pour tout administrateur système. Sur Debian 12, plusieurs méthodes sont disponibles pour configurer les interfaces réseau, notamment **ifupdown**, **NetworkManager** et **systemd-networkd**. Nous allons explorer ces méthodes en détail.

Configuration avec ifupdown

ifupdown utilise le fichier `/etc/network/interfaces` pour la configuration.

1. **Édition du fichier de configuration**

```
sudo nano /etc/network/interfaces
```

2. **Configuration d'une interface en DHCP**

```
auto enp0s3
iface enp0s3 inet dhcp
```

- **auto enp0s3** : Démarre automatiquement l'interface au démarrage.
- **iface enp0s3 inet dhcp** : Configure l'interface `enp0s3` pour utiliser DHCP.

Configuration du client réseau

3. **Configuration d'une adresse IP statique**

```
auto enp0s3
iface enp0s3 inet static
      address 192.168.1.100
      netmask 255.255.255.0
      gateway 192.168.1.1
      dns-nameservers 8.8.8.8 8.8.4.4
```

- **address** : Adresse IP statique assignée à l'interface.
- **netmask** : Masque de sous-réseau.
- **gateway** : Passerelle par défaut.
- **dns-nameservers** : Serveurs DNS à utiliser.

4. **Redémarrage du service réseau**

```
sudo systemctl restart networking
```

Remarque : Assurez-vous que le service `networking` est activé pour le démarrage automatique.

Configuration avec NetworkManager

NetworkManager est un service qui simplifie la gestion des connexions réseau, notamment pour les environnements de bureau.

1. **Installation de NetworkManager** (si non installé)

```
sudo apt install network-manager
```

2. **Utilisation de l'interface graphique**
   - Ouvrez **Paramètres réseau**.
   - Ajoutez ou modifiez une connexion.
   - Renseignez les détails tels que l'adresse IP, le masque de sous-réseau, la passerelle et les serveurs DNS.

Configuration avec NetworkManager

3. **Utilisation de nmcli (ligne de commande)**
   - **Afficher les connexions existantes**

```
nmcli connection show
```

   - **Ajouter une nouvelle connexion avec une IP statique**

```
nmcli connection add type ethernet ifname enp0s3 con-name "Connexion statique" ipv4.addresses 192.168.1.100/24 ipv4.gateway 192.168.1.1 ipv4.dns "8.8.8.8 8.8.4.4" ipv4.method manual
```

   - **Activer la connexion**

```
nmcli connection up "Connexion statique"
```

Configuration avec systemd-networkd

systemd-networkd est un service de gestion des interfaces réseau pour les systèmes sans interface graphique.

1. **Activer systemd-networkd**

```
sudo systemctl enable systemd-networkd
sudo systemctl start systemd-networkd
```

Configuration avec systemd-networkd

2. **Créer un fichier de configuration**
   - Créez un fichier `/etc/systemd/network/10-static-enp0s3.network` :

```
sudo nano /etc/systemd/network/10-static-enp0s3.network
```

- Contenu du fichier :

```
[Match]
Name=enp0s3

[Network]
Address=192.168.1.100/24
Gateway=192.168.1.1
DNS=8.8.8.8 8.8.4.4
```

3. **Redémarrer systemd-networkd**

```
sudo systemctl restart systemd-networkd
```

Vérification de la configuration réseau

- **Vérifier les interfaces actives**

```
ip addr show
```

- **Tester la connectivité**

```
ping -c 4 8.8.8.8
```

Démarrage et arrêt du firewall

La sécurité est une préoccupation majeure en administration système. Un pare-feu contrôle le trafic réseau entrant et sortant en fonction de règles de sécurité définies.

Options de pare-feu sous Debian 12

- **iptables/nftables** : Outils avancés pour la configuration du pare-feu au niveau du noyau.
- **ufw (Uncomplicated Firewall)** : Interface simplifiée pour iptables/nftables.
- **firewalld** : Gestion dynamique du pare-feu avec zones.

Installation de UFW

```
sudo apt update
sudo apt install ufw
```

Configuration de base

1. **Vérifier le statut du pare-feu**

```
sudo ufw status
```

Démarrage et arrêt du firewall

2. **Autoriser les connexions SSH**
    - Par défaut, ufw peut bloquer SSH. Il est donc crucial d'autoriser le port SSH (par défaut 22) pour éviter de perdre l'accès distant.

```
sudo ufw allow ssh
```

- Si vous avez modifié le port SSH :

```
sudo ufw allow 2222/tcp
```

3. **Activer le pare-feu**

```
sudo ufw enable
```

- Confirmez l'activation lorsque demandé.

Démarrage et arrêt du firewall

4. **Définir des règles supplémentaires**
    - **Autoriser le trafic HTTP et HTTPS**

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

- **Bloquer une adresse IP spécifique**

```
sudo ufw deny from 203.0.113.0
```

- **Autoriser un port spécifique**

```
sudo ufw allow 8080/tcp
```

5. **Vérifier les règles actives**

```
sudo ufw status numbered
```

Démarrage et arrêt du firewall

6. **Supprimer une règle**
   - Identifiez le numéro de la règle avec `ufw status numbered`.
   - Supprimez la règle :

```
sudo ufw delete [numéro_de_règle]
```

Outils de diagnostic

Les outils de diagnostic réseau sont essentiels pour identifier et résoudre les problèmes de connectivité.

1. **ping**
    - **Description** : Vérifie la connectivité avec une autre machine en envoyant des paquets ICMP Echo Request.
    - **Utilisation de base**

```
ping 8.8.8.8
```

- **Options courantes**
    - **-c [nombre]** : Nombre de requêtes à envoyer.

```
ping -c 4 8.8.8.8
```

- **-i [intervalle]** : Intervalle entre les requêtes.

Outils de diagnostic

2. **traceroute**
    - **Description** : Affiche le chemin emprunté par les paquets pour atteindre une destination.
    - **Installation**

```
sudo apt install traceroute
```

- **Utilisation**

```
traceroute www.google.com
```

- **Options**
    - **-n** : N'affiche pas les noms d'hôte, uniquement les adresses IP.

Outils de diagnostic

3. **mtr (My Traceroute)**
    - **Description** : Combine les fonctionnalités de ping et traceroute.
    - **Installation**

```
sudo apt install mtr
```

- **Utilisation**

```
mtr www.google.com
```

Outils de diagnostic

4. **netstat et ss**
   - **Description** : Affichent les connexions réseau actives, les sockets et les ports ouverts.
   - **Utilisation de netstat**

```
netstat -tulpn
```

     - **-t** : Affiche les connexions TCP.
     - **-u** : Affiche les connexions UDP.
     - **-l** : Affiche les sockets en écoute.
     - **-p** : Affiche le PID et le nom du programme.
     - **-n** : Affiche les adresses numériques.
   - **Utilisation de ss**

```
ss -tulpn
```

   - **Exemple : Trouver les processus écoutant sur le port 80**

```
sudo ss -ltnp | grep ':80'
```

Outils de diagnostic

5. **nslookup et dig**
   - **Description** : Interrogent les serveurs DNS pour résoudre les noms de domaine.
   - **Utilisation de nslookup**

```
nslookup www.example.com
```

   - **Utilisation de dig**

```
dig www.example.com
```

     - **Obtenir les enregistrements MX**

```
dig MX example.com
```

     - **Interroger un serveur DNS spécifique**

```
dig @8.8.8.8 www.example.com
```

Outils de diagnostic

6. **ifconfig et ip**
   - **ifconfig** est en cours de dépréciation au profit de **ip**.
   - **Affichage des interfaces avec ifconfig**

```
sudo apt install net-tools  # Si non installé
ifconfig -a
```

   - **Affichage des interfaces avec ip**

```
ip addr show
```

     - **Afficher les routes**

```
ip route show
```

Outils de diagnostic

7. **nmap**
   - **Description** : Outil de scan de ports et d'audit de sécurité.
   - **Installation**

```
sudo apt install nmap
```

   - **Scan des ports ouverts sur une machine**

```
nmap 192.168.1.1
```

   - **Scan complet avec détection du système d'exploitation**

```
sudo nmap -A 192.168.1.1
```

8. **tcpdump**
   - **Description** : Capture et analyse le trafic réseau.
   - **Installation**

```
sudo apt install tcpdump
```

Réglages de la date et de l'heure

Un système avec une heure précise est essentiel pour les journaux système, les certificats SSL, les tâches planifiées et la cohérence dans les environnements distribués.

Affichage de la date et de l'heure

- **Commande date**

```
date
```

- **Utilisation de timedatectl**

```
timedatectl
```

  - Affiche l'état de la synchronisation NTP, le fuseau horaire, l'heure système et matérielle.

Modification manuelle de la date et de l'heure

- **Modifier la date**

```
sudo date --set="2023-10-01"
```

- **Modifier l'heure**

```
sudo date --set="12:00:00"
```

Configuration du fuseau horaire

1. **Lister les fuseaux horaires disponibles**

```
timedatectl list-timezones
```

2. **Définir le fuseau horaire**

```
sudo timedatectl set-timezone Europe/Paris
```

3. **Vérifier le fuseau horaire**

```
timedatectl
```

- **Forcer la synchronisation de l'horloge matérielle avec l'horloge système**

```
sudo hwclock --systohc
```

Paramétrer le client NTP

Le protocole NTP (Network Time Protocol) permet de synchroniser l'heure de votre système avec des serveurs de temps. Debian 12 recommande l'utilisation de **chrony** plutôt que **ntpd** pour la synchronisation NTP.

Installation de chrony

```
sudo apt update
sudo apt install chrony
```

Configuration de chrony

1. **Éditer le fichier de configuration**

```
sudo nano /etc/chrony/chrony.conf
```

2. **Configurer les serveurs NTP**
   - Les serveurs par défaut pointent vers le pool Debian :

```
pool 0.debian.pool.ntp.org iburst
pool 1.debian.pool.ntp.org iburst
pool 2.debian.pool.ntp.org iburst
pool 3.debian.pool.ntp.org iburst
```

Paramétrer le client NTP

3. **Ajouter des serveurs NTP locaux ou spécifiques**
   - Par exemple :

```
server ntp.example.com iburst
```

4. **Redémarrer le service chrony**

```
sudo systemctl restart chrony
```

Paramétrer le client NTP

Vérification de la synchronisation NTP

- **Afficher le statut de chrony**

```
chronyc tracking
```

- **Lister les sources NTP**

```
chronyc sources -v
```

- **Vérifier la synchronisation**
  - Si le décalage est faible et que l'état est **synchronisé**, la configuration est correcte.

Remarque : **chrony** est préféré pour les systèmes qui ne sont pas toujours connectés ou qui ont des changements fréquents de connexion.

Configuration du serveur et du client OpenSSH

SSH (Secure Shell) est un protocole réseau qui permet d'établir une connexion sécurisée entre deux hôtes. Il est essentiel pour l'administration à distance sécurisée des serveurs.

Installation du serveur SSH

```
sudo apt update
sudo apt install openssh-server
```

- **Vérifier le statut du service SSH**

  ```
  sudo systemctl status ssh
  ```

  - Le service doit être actif (en cours d'exécution).

Configuration du serveur SSH

1. **Sauvegarder le fichier de configuration original**

   ```
   sudo cp /etc/ssh/sshd_config /etc/ssh/sshd_config.bak
   ```

2. **Éditer le fichier de configuration**

   ```
   sudo nano /etc/ssh/sshd_config
   ```

Configuration du serveur et du client OpenSSH

3. **Paramètres de sécurité importants**
   - **Changer le port par défaut** (optionnel)

     ```
     Port 2222
     ```

     Remarque : Si vous changez le port, assurez-vous que le pare-feu autorise ce port.
   - **Désactiver l'accès root via SSH**

     ```
     PermitRootLogin no
     ```
   - **Limiter les utilisateurs autorisés**

     ```
     AllowUsers utilisateur1 utilisateur2
     ```
   - **Désactiver l'authentification par mot de passe** (si vous utilisez des clés SSH)

     ```
     PasswordAuthentication no
     ```

Configuration du serveur et du client OpenSSH

4. **Redémarrer le service SSH**

```
sudo systemctl restart ssh
```

Configuration du client SSH

- **Installation du client SSH** (généralement préinstallé)

```
sudo apt install openssh-client
```

Génération et utilisation des clés SSH

1. **Générer une paire de clés sur le client**

```
ssh-keygen -t rsa -b 4096 -C "votre_email@example.com"
```

   - **-t rsa** : Type de clé RSA.
   - **-b 4096** : Taille de la clé en bits.
   - **-C** : Commentaire (généralement votre email).

Configuration du serveur et du client OpenSSH

2. **Chemin par défaut pour les clés**
   - **Clé privée** : `~/.ssh/id_rsa`
   - **Clé publique** : `~/.ssh/id_rsa.pub`
3. **Copier la clé publique sur le serveur**
   - **Méthode automatisée avec ssh-copy-id**

```
ssh-copy-id -i ~/.ssh/id_rsa.pub utilisateur@serveur -p 2222
```

   - **Méthode manuelle**
     - Copier le contenu de `~/.ssh/id_rsa.pub` du client.
     - Sur le serveur, ajoutez-le à `~/.ssh/authorized_keys` :

```
cat id_rsa.pub >> ~/.ssh/authorized_keys
chmod 600 ~/.ssh/authorized_keys
```

Configuration du serveur et du client OpenSSH

4. **Connexion au serveur avec la clé**

```
ssh utilisateur@serveur -p 2222
```

   - **Options utiles**
     - **-v** : Mode verbeux pour le débogage.
     - **-i [chemin_clé_privée]** : Spécifier une clé privée différente.

Configuration du serveur et du client OpenSSH

Sécurisation supplémentaire avec fail2ban

- **Installation de fail2ban**

```
sudo apt install fail2ban
```

- **Configuration de base**
  - Créez un fichier de configuration local :

```
sudo nano /etc/fail2ban/jail.local
```

  - Exemple de configuration pour SSH :

```
[sshd]
enabled = true
port = 2222
filter = sshd
logpath = /var/log/auth.log
maxretry = 5
```

- **Redémarrer fail2ban**

Copie et transfert de fichiers sécurisés via SSH

SSH ne sert pas uniquement à établir des connexions à distance sécurisées, mais aussi à transférer des fichiers en toute sécurité.
**Utilisation de scp (Secure Copy)
**

- **Copier un fichier du client vers le serveur**

```
scp -P 2222 /chemin/local/fichier.txt utilisateur@serveur:/chemin/distant/
```

- **Copier un fichier du serveur vers le client**

```
scp -P 2222 utilisateur@serveur:/chemin/distant/fichier.txt /chemin/local/
```

- **Copier un répertoire entier (option -r)**

```
scp -P 2222 -r /chemin/local/dossier utilisateur@serveur:/chemin/distant/
```

- **Options utiles**
  - **-C** : Compression des données pendant le transfert.
  - **-v** : Mode verbeux pour le débogage.

Copie et transfert de fichiers sécurisés via SSH

Utilisation de sftp (SSH File Transfer Protocol)

1. **Se connecter au serveur SFTP**

```
sftp -P 2222 utilisateur@serveur
```

2. **Commandes SFTP de base**
   - **Naviguer dans les répertoires**
     - **lcd [chemin_local]** : Changer le répertoire local.
     - **cd [chemin_distant]** : Changer le répertoire distant.
     - **lpwd** : Afficher le répertoire local courant.
     - **pwd** : Afficher le répertoire distant courant.
   - **Lister les fichiers**
     - **ls** : Lister les fichiers distants.
     - **lls** : Lister les fichiers locaux.

Copie et transfert de fichiers sécurisés via SSH

- **Transférer des fichiers**
  - **put [fichier_local]** : Envoyer un fichier vers le serveur.
  - **get [fichier_distant]** : Télécharger un fichier depuis le serveur.
- **Transférer des répertoires**
  - **put -r [dossier_local]** : Envoyer un dossier.
  - **get -r [dossier_distant]** : Télécharger un dossier.
- **Supprimer des fichiers**
  - **rm [fichier_distant]** : Supprimer un fichier distant.
- **Quitter la session**

```
bye
```

Copie et transfert de fichiers sécurisés via SSH

Utilisation de rsync sur SSH

rsync est un outil puissant pour synchroniser des fichiers et des répertoires.

- **Synchroniser un répertoire local vers un serveur distant**

```
rsync -avz -e "ssh -p 2222" /chemin/local/ utilisateur@serveur:/chemin/distant/
```

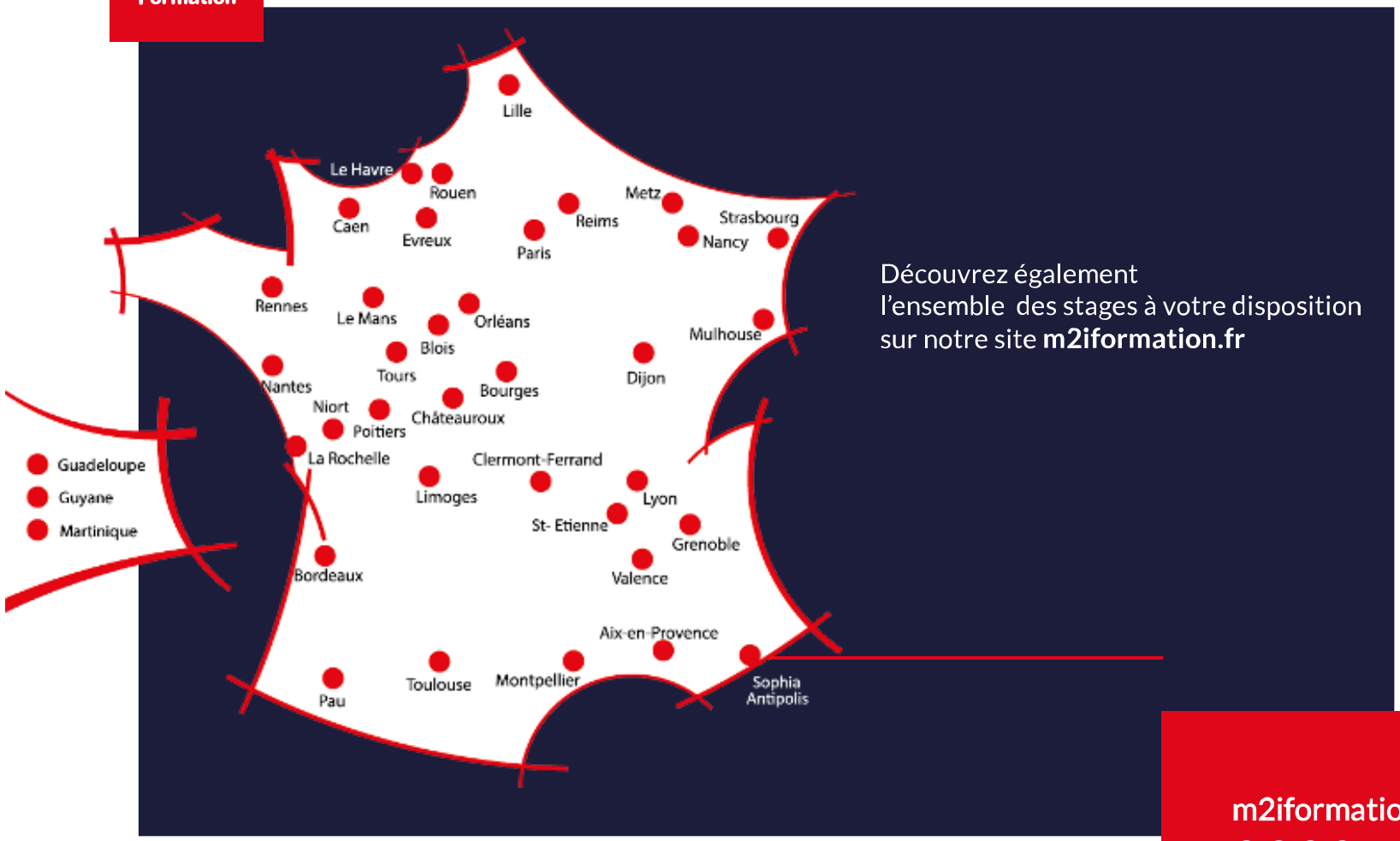
  - **-a** : Mode archive (préserve les permissions, les liens, etc.).
  - **-v** : Mode verbeux.
  - **-z** : Compression pendant le transfert.
  - **-e** : Spécifie le shell distant (ici avec le port SSH personnalisé).

- **Synchroniser un répertoire distant vers le local**

```
rsync -avz -e "ssh -p 2222" utilisateur@serveur:/chemin/distant/ /chemin/local/
```

Avantages de rsync

- **Efficacité** : Ne transfère que les différences entre les fichiers.
- **Reprise sur erreur** : Peut reprendre un transfert interrompu.
- **Options avancées** : Exclusion de fichiers, synchronisation bidirectionnelle, etc.



Découvrez également
l'ensemble des stages à votre disposition
sur notre site **m2iformation.fr**

m2iformation.fr