Système

- Chiffrement d'une partition (LUKS)
- Protection de GRUB par mot de passe
- Réinitialisation du mot de passe Root
- Initramfs
- SUDO
- Ajouter un nouveau service dans systemd
- Changement de langue du clavier
- MOTD (Message Of The Day)
- Ajout d'un service dans systemd

Chiffrement d'une partition (LUKS)

Même s'il est totalement recommandé de le faire sur une nouvelle partition afin d'éviter tout problème de perte de données, cryptsetup peut être utilisé sur une partition fraîchement crée ou une déjà existante.

Ce tuto est fait en étant déjà root, si vous ne l'êtes pas rajouter le mot clé "sudo" devant vos commandes en cas de manque de droit

> La partition que nous allons chiffrer n'est pas essentiel au démarrage, cela veut dire qu'elle n'est pas montée automatiquement et aucun mot de passe ne sera demandé lors du démarrage, pour la montée automatiquement voir du côté de /etc/fstab

Installer le paquet cryptsetup si vous ne l'avez pas avec la commande :

```
apt install cryptsetup -y
```

Les disques linux se nomment très souvent comme /dev/sda avec comme partitions (/dev/sda1, /dev/sda2 etc...)

Ces noms peuvent varier selon le type de disque vous avec, pour du nvme ça sera par exemple /dev/nvme0n1p1 etc..

Considérons que nous souhaitons chiffrer la partitions /dev/sde1, utiliser la commande :

```
```

> cryptsetup luksFormat --cipher aes-xts-plain64 --key-size 256 --hash sha256 --verify-passphrase /dev/sde1

Un Warning vous avertie que les données sur cette partitions seront écrasées, soyez sur de ne plus avoir besoin de ces dernières, ou avoir fait une sauvegarde au préalable.

> Faites attention à ne pas chiffrer les partitions de votre premier disque (/dev/sda) qui contient les fichiers de votre système, soyez certain de la partition que vous souhaitez verrouiller.

```
ATTENTION: Le périphérique /dev/sde1 contient déjà une signature pour un superblock « crypto_LUKS
».

WARNING!
========
Cette action écrasera définitivement les données sur /dev/sde1.

Are you sure? (Type 'yes' in capital letters):
```

Tapez "YES" en majuscule. Ensuite saisissez votre phrase secrète.

```
Cette action écrasera définitivement les données sur /dev/sde1.

Are you sure? (Type 'yes' in capital letters): YES
Saisissez la phrase secrète pour /dev/sde1 :
Vérifiez la phrase secrète :
root@beta:~#
```

Nous allons maintenant déchiffrer notre partition qu'on vient de chiffrer et donner un nom à notre mappage pour la partition LUKS ouverte. Faites la commande :

> cryptsetup luksOpen /dev/sde1 private_partition

Entrez la phrase secrète pour valider l'opération.

```
root@beta:~# cryptsetup luksOpen /dev/sde1 private_partition
Saisissez la phrase secrète pour /dev/sde1 :
root@beta:~#
```

Nous allons mettre notre partition chiffrée au format ext4 avec la commande :

> mkfs.ext4 /dev/mapper/private_partition

Il ne nous reste plus qu’à monter notre partition chiffrée sur le répertoire /mnt/private_partition par exemple.

Faites ces commandes là :

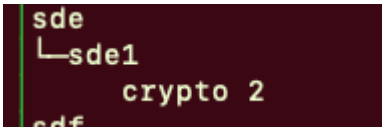
> mkdir /mnt/private_partition
> mount /dev/mapper/private_partition /mnt/private_partition

Pour vérrouiller à nouveau la partition, il faut démonter le système de fichier avant, ensuite refermer notre partition.

Faites ces commandes là :

> umount /mnt/private_partition
> cryptsetup luksClose private_partition

Avec l'option '-f' de lsblk vous pourrez voir les partitions chiffrées de votre machine, qui apparaissent avec le mot clé crypto_LUKS ou crypto 2



Protection de GRUB par mot de passe

> Nous allons utiliser l'utilisateur root pour protéger grub. L'utilisation de root n'est pas obligatoire, mais l'utilisateur doit forcément faire partie des sudoers (groupe sudo).

> Tout le tutoriel est fait en étant root, si ce n'est pas votre cas utilisez le mot clé "sudo" en cas de manque de droit

Exécuter cette commande qui permettra de rajouter notre utilisateur dans le fichier de config

```
echo 'set superusers="root"' >> /etc/grub.d/40_custom
```

Ensuite nous allons créer notre mot de passe avec cette commande.

```
grub-mkpasswd-pbkdf2
```

Copier le mot de passe générer en entier, exmple :

```
grub.pbkdf2.sha512.10000.C77CE445861F04EE33346E09B480A8F55732068F05
E3A3695B1C5ED4D2111A95CA7597FF6FC66B91020AF82D73FEA8F7075BCB0E7
A81275CA54F59B71B41A075.71DBB5FB9F8EFAC1A460B9DE6D42F2D596F0A76
19C0C09D34733F0661B3671133C068C2998301BECC091F9C40D6DE8E15F6CF
3C28452480D35757E5C1E100828
```

Éditez ensuite le fichier /etc/grub.d/40_custom

Rajouter à la ligne "password_pbkdf2 root" suivi du mot de passe, votre fichier devra ressembler à ça :

```
#!/bin/sh
exec tail -n +3 $0
# This file provides an easy way to add custom menu entries.  Simply type the
# menu entries you want to add after this comment.  Be careful not to change
# the 'exec tail' line above.
set superusers="root"
password_pbkdf2 root grub.pbkdf2.sha512.10000.A3306FD2653D36F9391CF865E0BAA53B4716EB37B58A0E6EA1C91FD35BF6635DF5EAFAB900013FD89D7BA45446B4E52A518E1E4448921EE1
9C17071A81B631FF.5C1229E0045689606D8A932ADCD6BE38F68AB74C4B464279D01D7372115D6CA02F746A4A4EEBA4F6142FA621734084FC6977EEBD17EB2E8C06F658E4A1D47375
~
```

> Ajustez si nécessaire si vous n'avez pas utiliser l'utilisateur root.

Faites ensuite la commande suivante pour attribuer tout les droit à l'utilisateur :

> chmod 711 /etc/grub.d/40_custom

À l'aide de ces deux commandes enregistrez ensuite vos configurations avec cette commande :

> grub-mkconfig -o /boot/grub/grub.cfg
>
> update-grub

Redémarrez votre machine, et vous devrez vous identifier avec votre utilisateur et le mot de passe pour démarrer le système.



Pour désactiver le mot de passe, éditez le ficher /etc/grub.d/10_linux à la ligne 34 et rajouter l'instruction "--unrestricted" :

> nano +34 /etc/grub.d/10_linux

(la ligne pourra différencier selon les MAJ futurs, rechercher la manuellement le cas échéant, voir exemple de la ligne ci-dessous)

La ligne devra ressembler à ça :

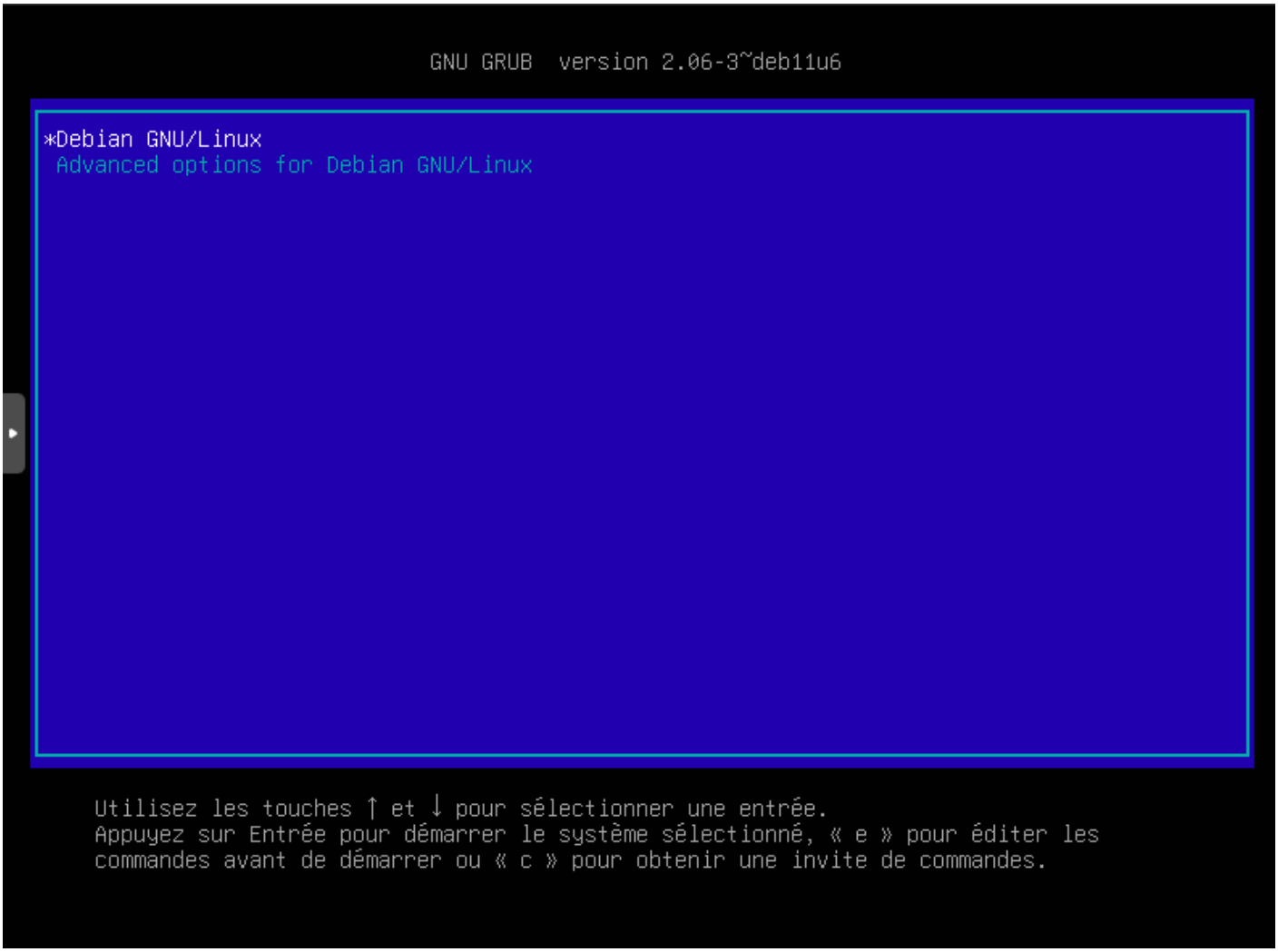


Enregistrez avec la commande : "***update-grub***" et le mot de passe sera désactivé.

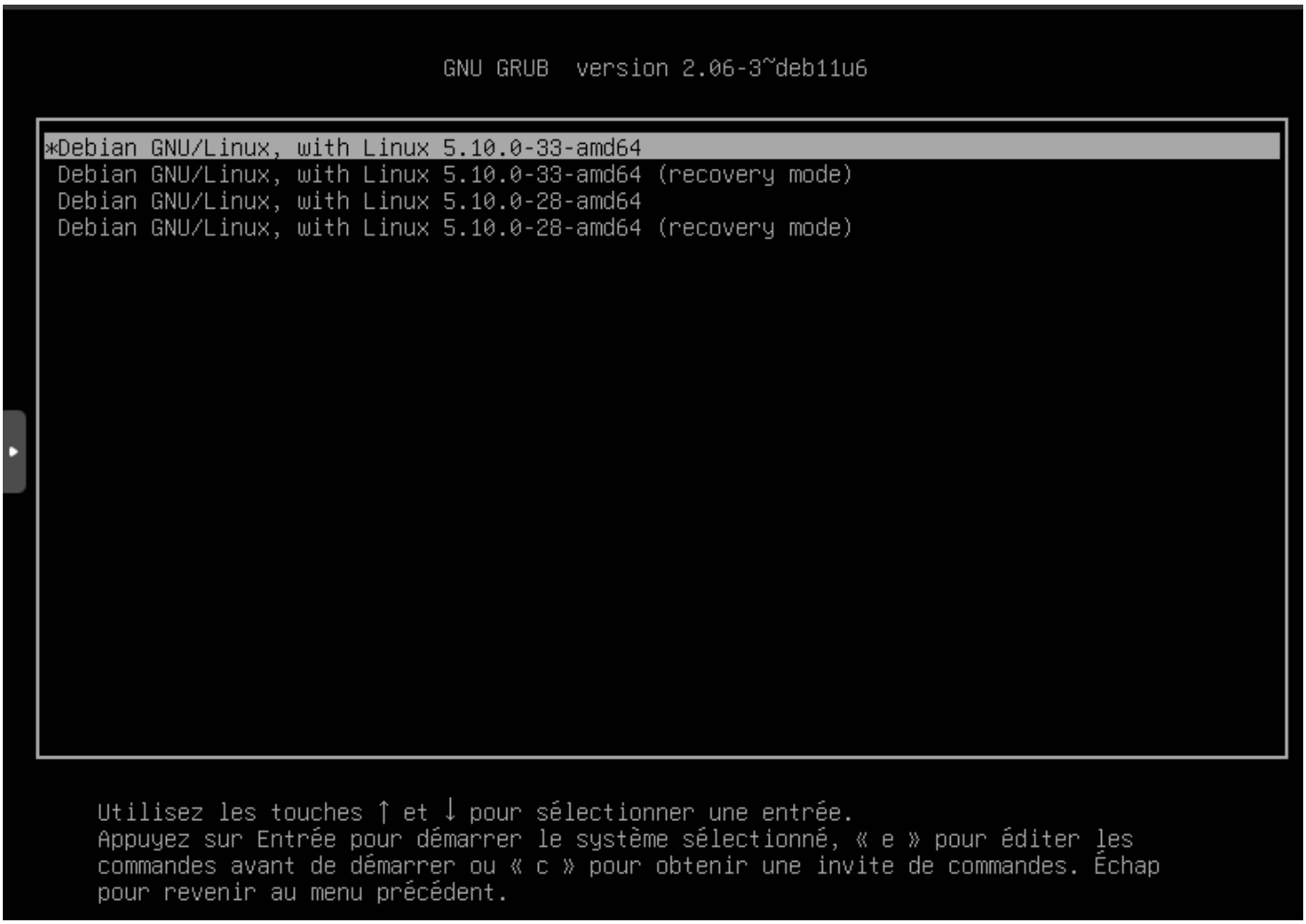
Réinitialisation du mot de passe Root

Méthode 1 :

Accéder au option avancé de GRUB au démarrage en sélectionnant la seconde ligne affichée ci-dessous



Choisissez ici la deuxième option pour démarrer en mode rescue



> Si vous avez aussi besoin du mot de passe root pour accéder au mode rescue, passer à la méthode 2

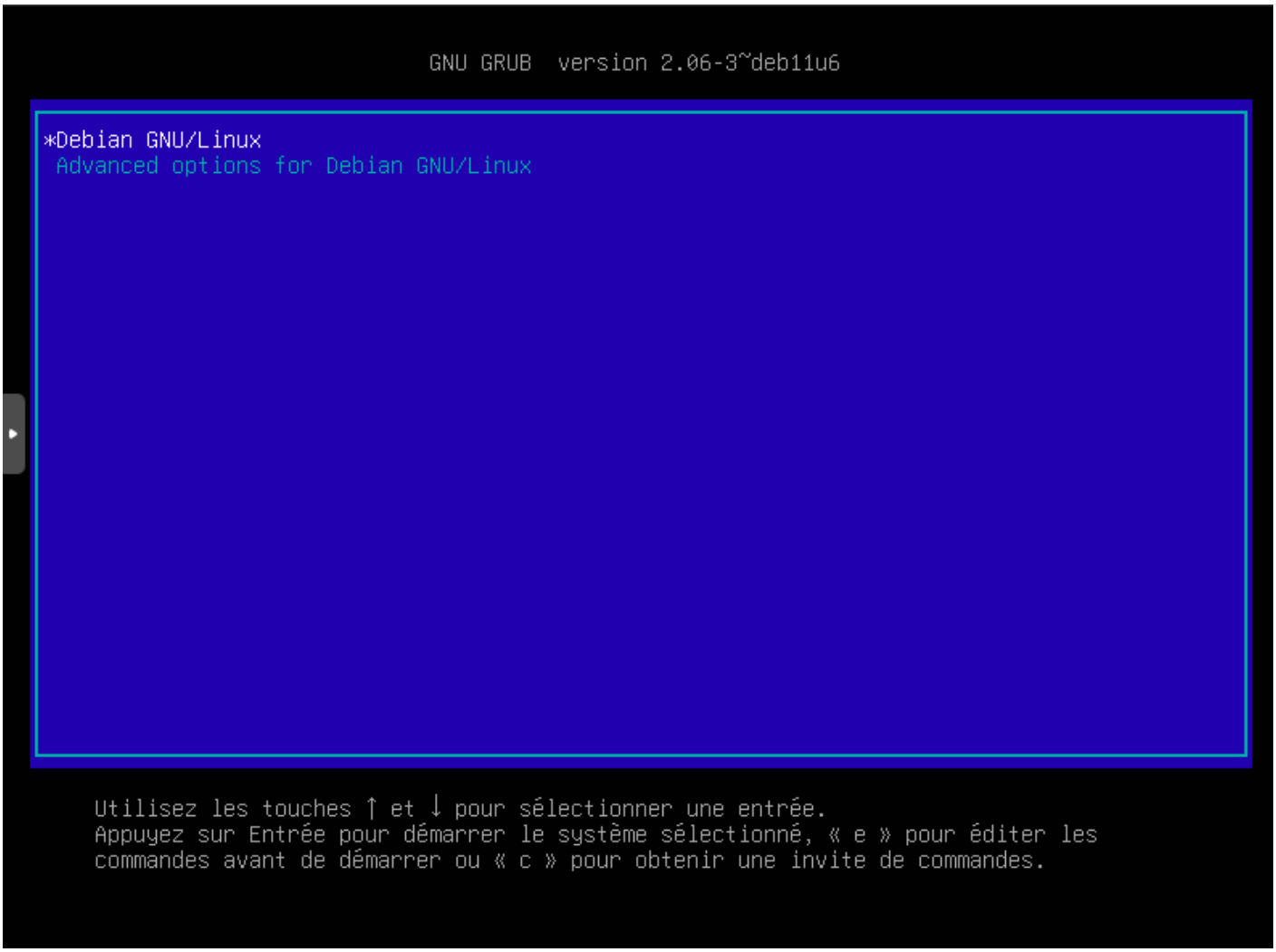
Une fois que vous avez accéder au prompt en mode root saisissez cette commande pour réinitialiser le mot de passe root :

> passwd root

Choisissez un nouveau mot de passe, et redémarrer votre machine à l'aide de la commande "reboot"

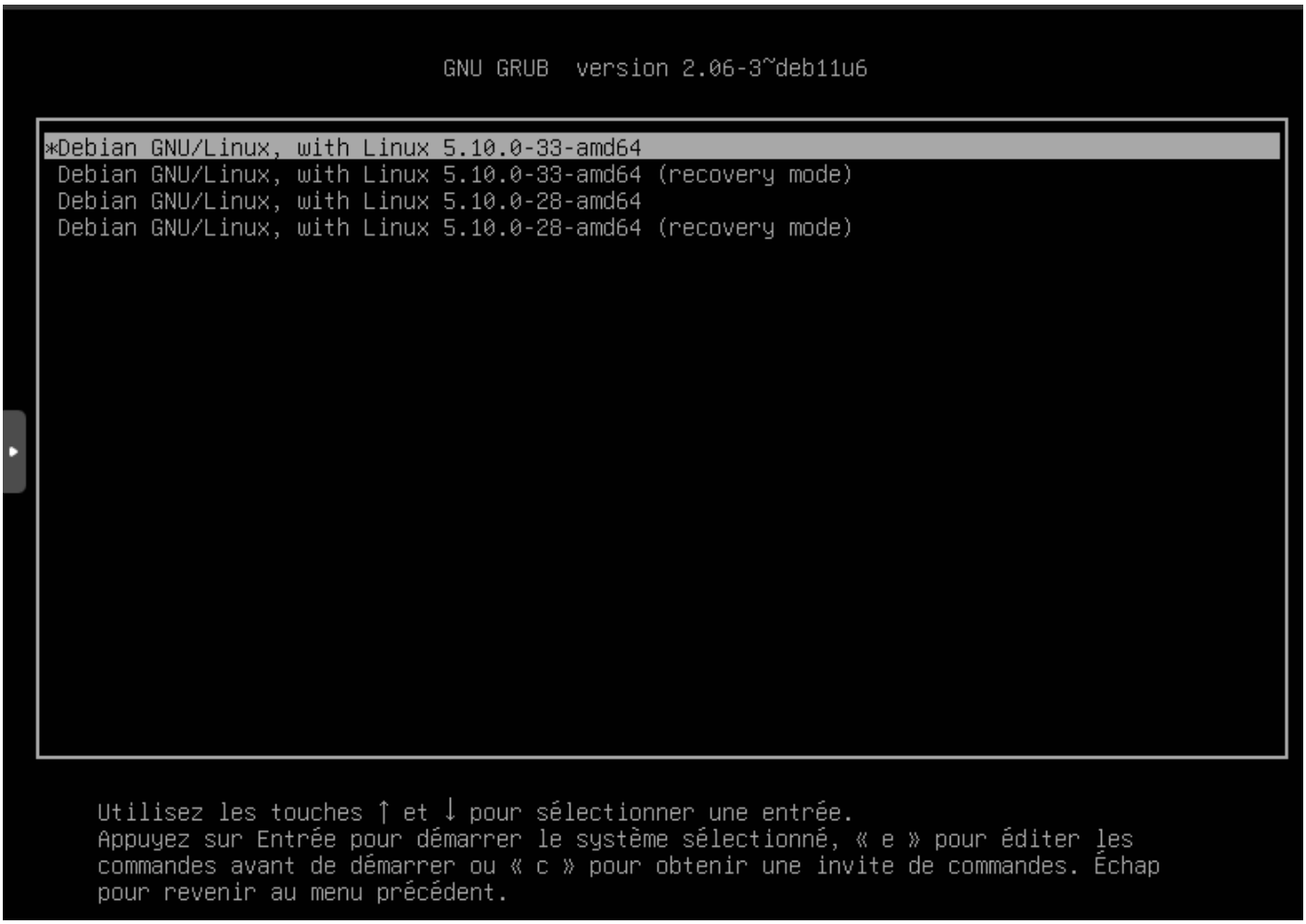
Méthode 2 :

Redémarrer le système et accéder au menu de démarrage grub



Choisissez la seconde option pour avoir les paramètres avancés

Appuyer ensuite sur la touche 'e' pour éditer les commandes exécutées avant le démarrage du système



Identifier la ligne qui commence par "linux" ou "linux16" et ajouter ceci en fin de ligne :

> init=/bin/bash

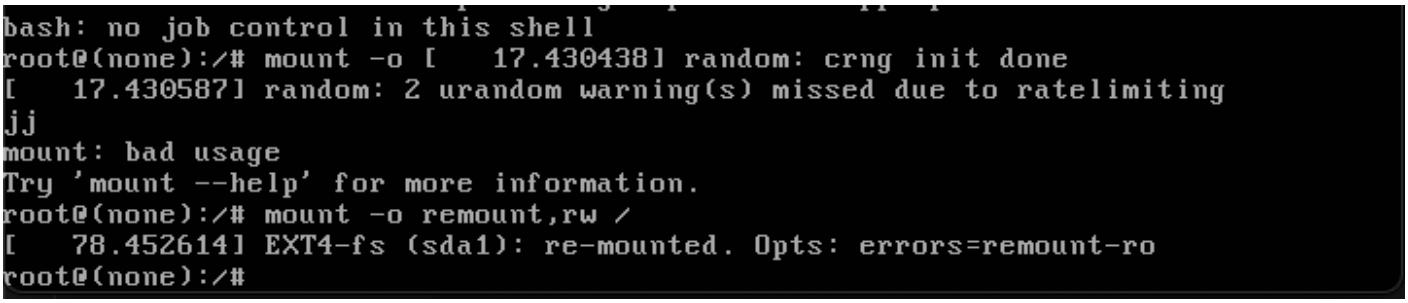


Faites ctrl+X pour sortir

Si vous avez des warnings affichés faite Entrez pour les faire disparaître et avoir le prompt

Rentrez cette commande :

> mount -o remount,rw /



Modifier votre mot de passe root à l'aide de la commande passwd :

> passwd root

```
root@(none):/# passwd root
New password:
Retype new password:
passwd: password updated successfully
root@(none):/#
```

Ensuite tapez cette commande pour créez un fichier .autorelabel pour forcer SELinux à relabelliser au prochain démarrage :

> touch /.autorelabel

Et enfin cette commande pour redémarrer le système :

> exec /sbin/init

Si le mot de pass root est demander saisissez le nouveau mot de passe créer pour accéder au prompt

```
You are in rescue mode. After logging in, type "journalctl -xb" to view
system logs, "systemctl reboot" to reboot, "systemctl default" or "exit"
to boot into default mode.
Donnez le mot de passe du superutilisateur pour la maintenance
(ou appuyez sur Ctrl et D pour continuer) :
root@beta:~# _
```

Ensuite vous pouvez rentrer la commande "reboot" pour redémarrer normalement votre système

Initramfs

Résolution du problème initramfs

Causes principales. Corruption du système de fichiers.

Blocs défectueux sur le disque dur.

Mises à jour système incomplètes.

Configuration incorrecte du chargeur de démarrage.

Étapes de résolution

1. Accéder à l'invite initramfs Lorsque votre système présente l'erreur, vous verrez une invite (initramfs).
2. Identifier vos partitions Utilisez la commande suivante pour afficher les partitions disponibles :

```
blkid
```

Notez la partition racine (généralement de type ext4).

3. Vérifier et réparer le système de fichiers Exécutez la commande suivante pour réparer le système de fichiers (remplacez /dev/sdaX par votre partition) :

```
fsck -f /dev/sdaX -y
```

Le paramètre -y répond automatiquement "oui" à toutes les questions de réparation.

4. Redémarrer le système Après la réparation, tapez :

```
reboot
```

Si cette commande ne fonctionne pas, essayez :

```
exit
```

Méthode avancée (si le problème persiste)

1. Démarrer sur un Live CD/USB Utilisez un Live CD/USB comme GParted Live.
2. Monter votre partition racine Créez un point de montage et montez votre partition racine :

```
mkdir /mnt/temp
```

`mount /dev/sdaX /mnt/temp`

`mount /dev/sdaY /mnt/temp/boot` (Si vous avez une partition /boot séparée.)

3. Monter les systèmes virtuels Montez les systèmes nécessaires pour accéder à l'environnement chroot :

`mount --bind /dev /mnt/temp/dev`

`mount --bind /dev/pts /mnt/temp/dev/pts`

`mount --bind /proc /mnt/temp/proc`

`mount --bind /sys /mnt/temp/sys`

4. Accéder à l'environnement chroot Entrez dans l'environnement chroot :

`chroot /mnt/temp`

5. Mettre à jour initramfs et GRUB Mettez à jour initramfs et GRUB pour corriger les problèmes :

`update-initramfs -u`

`update-grub`

6. Redémarrer le système Quittez l'environnement chroot et redémarrez :

`exit`

`reboot`

Remarque importante Si cette erreur survient fréquemment, il est possible que votre disque dur soit défaillant. Il est recommandé de sauvegarder vos données et de remplacer le disque dur dès que possible.

Vous pouvez copier directement ce contenu dans votre documentation en Markdown.

SUDO

La commande `sudo` (abréviation de *Super User DO*) est un outil essentiel sous Linux qui permet à un utilisateur autorisé d'exécuter des commandes avec des privilèges élevés, généralement ceux du superutilisateur (root). Voici un résumé complet mais concis pour votre wiki.

Fonctionnement de `sudo`

- **Objectif** : Permettre à des utilisateurs spécifiques d'exécuter des commandes nécessitant des privilèges administratifs sans se connecter directement en tant que root.
- **Sécurité** : `sudo` demande l'authentification avec le mot de passe de l'utilisateur courant (et non celui de root), limitant ainsi les risques liés au partage du mot de passe root.
- **Durée** : Une session `sudo` est valide pendant une période définie (15 minutes par défaut), après quoi le mot de passe doit être saisi à nouveau.

Syntaxe de base

sudo [commande]

Exemple :

`sudo apt update`

Cela exécute la commande `apt update` avec les privilèges administratifs.

Options utiles

- `sudo -u [utilisateur] [commande]` : Exécute une commande en tant qu'un autre utilisateur.
  - Exemple : `sudo -u john ls /home/john`
- `sudo -l` : Liste les commandes que l'utilisateur peut exécuter avec `sudo`.
- `sudo !!` : Réexécute la dernière commande avec `sudo`.
- `sudo -k` : Invalide les privilèges actifs et force une nouvelle authentification.
- `sudo -v` : Valide ou renouvelle les privilèges sans exécuter de commande.
- `sudo -s` : Lance un shell interactif avec les privilèges actuels.
- `sudo -i` : Simule une connexion initiale en tant que root.

Configuration via le fichier sudoers

La gestion des permissions se fait dans le fichier `/etc/sudoers`. Il est recommandé d'utiliser la commande sécurisée `visudo` pour éditer ce fichier afin d'éviter les erreurs de syntaxe.

Exemple d'entrée dans sudoers :

- > # Autoriser un utilisateur à tout faire sans mot de passe : john ALL=(ALL) NOPASSWD: ALL
- > # Autoriser un groupe pour certaines commandes : %developers ALL=(ALL) /usr/bin/apt-get, /usr/bin/dpkg
- > # Limiter un utilisateur à des commandes spécifiques en tant qu'un autre utilisateur : jane ALL=(webuser) /usr/bin/systemctl restart apache2
- > # Autoriser un utilisateur sur des hôtes spécifiques : tom SERVER1,SERVER2=(ALL) ALL
- > # Définir et utiliser un alias de commande : Cmnd_Alias SERVICES = /usr/bin/systemctl start, /usr/bin/systemctl stop, /usr/bin/systemctl restartoperator ALL=(root) SERVICES
- > # Autoriser un groupe à tout faire sauf certaines commandes : %admins ALL=(ALL) ALL, !/usr/bin/su, !/usr/bin/passwd
- > # Définir et utiliser un alias d'utilisateur : User_Alias WEBMASTERS = alice, bob, charlieWEBMASTERS ALL=(ALL) /usr/bin/systemctl restart

> nginx, /usr/bin/systemctl restart php-fpm

- > # Autoriser un utilisateur à tout faire, mais sans mot de passe pour certaines commandes : sarah ALL=(ALL) ALL, NOPASSWD: /sbin/reboot, /sbin/shutdown

D'autres commandes utiles :

Ajouter un utilisateur au groupe sudo :

`sudo usermod -aG sudo username`

Redémarrer un service :

`sudo systemctl restart apache2`

Ajouter un nouveau service dans systemd

C'est quoi systemd



Systemd est un outil nous permettant d'organiser nos tâches liée à un service, par exemple le démarrer le stopper regarder son status etc...

Vous pouvez lier n'importe quel script à systemd pour une meilleure prise en main.

Il faut un nouveau fichier avec **`sudo nano /etc/systemd/system/monscript.service`** et ensuite y mettre cette configuration ci dessous:

```
[Unit]
Description=Service pour script
After=network.target

[Service]
Type=simple
ExecStart=/usr/bin/python3 /chemin/vers/mon_script.py
Restart=on-failure
User=nom_utilisateur
WorkingDirectory=/chemin/vers
Environment=PYTHONUNBUFFERED=1
```

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

Par exemple un script qui renferme un bot qu tourne en continue, avec comme chemin */home/tintin/mon_bot.py*

Vous allez créer le fichier `/etc/systemd/system/mon_bot.service`

```
[Unit]
Description=Service pour mon bot
After=network.target

[Service]
Type=simple
ExecStart=/usr/bin/python3 /home/tintin/mon_bot.py
Restart=on-failure
User=tintin
WorkingDirectory=/home/tintin
Environment=PYTHONUNBUFFERED=1

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

Vous enregistrer et pouvez maintenant utliser votre bot, voici quelques commandes basiques commandes là:

Démarrage : `systemctl start mon_bot.service`
Status : `systemctl status mon_bot.service`
Stopper : `systemctl stop mon_bot.service`

Changement de langue du clavier

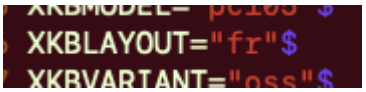
Changement de langue du clavier

> **Note :** ce changement s'applique au clavier **local** de la machine (console et session graphique). Il n'a aucun effet sur une session SSH : en SSH, c'est le clavier de votre poste client qui est utilisé.

Pour Ubuntu

Pour changer la disposition du clavier sur une machine Ubuntu, il suffit d'accéder au fichier de configuration :

```
sudo vim /etc/default/keyboard
```



Modifiez les lignes suivantes :

- **XKBLAYOUT="fr"** en **XKBLAYOUT="us"** (clavier anglais américain, ou `gb` pour le britannique ; `en` n'est pas un code valide)
- **XKBVARIANT="oss"** en **XKBVARIANT=""** (la variante `oss` est propre au clavier français)

Appliquez ensuite la configuration à la console (clavier uniquement) :

```
sudo setupcon -k
```

> Sans l'option `-k`, `setupcon` essaie aussi de changer la police de la console et peut afficher des erreurs `setfont: ERROR kdfontop.c:183 put_font_kdfontop: Unable to load such font with such kernel version`. Ces erreurs sont sans gravité (VM, SSH, console sans gestion des polices) et n'empêchent pas le changement de clavier.

Si la machine a une interface graphique, fermez puis rouvrez la session, ou lancez dans la session graphique :

```
setxkbmap us
```

Pour être sûr que tout est bien pris en compte, redémarrez la machine :

```
sudo reboot
```

Vérification :

```
cat /etc/default/keyboard
localectl status
```

`XKBLAYOUT` doit afficher `us`.

> Alternative interactive : `sudo dpkg-reconfigure keyboard-configuration` pose les questions et réécrit lui-même `/etc/default/keyboard`, donc il n'est pas nécessaire de modifier le fichier à la main au préalable.

Pour Linux (autres distributions)

Sur les distributions utilisant systemd (Debian, Fedora, RHEL / CentOS / Rocky, Arch, etc.), la commande `localectl` permet de changer le clavier sans éditer de fichier.

Afficher la configuration actuelle :

```
localectl status
```

Lister les dispositions disponibles :

```
localectl list-keymaps | grep -i us
```

Changer le clavier de la console (TTY), de façon persistante :

```
sudo localectl set-keymap us
```

Changer le clavier de l'environnement graphique (X11), de façon persistante :

```
sudo localectl set-x11-keymap us
```

Pour appliquer immédiatement, sans rien rendre persistant :

```
sudo loadkeys us    # console (TTY)
setxkbmap us        # session graphique
```

Vérifiez avec `localectl status` que `VC Keymap` et `X11 Layout` affichent bien `us`.

Comme pour Ubuntu, ces commandes n'ont aucun effet sur une session SSH.

MOTD (Message Of The Day)

Documentation : Gestion du MOTD sur Proxmox / Debian

Le **MOTD** (*Message Of The Day*) est le message qui s'affiche lors d'une connexion réussie en SSH ou via la console. Sur Proxmox (basé sur Debian), il existe deux manières de le gérer : statique ou dynamique.

---

1. Le MOTD Statique (Basique)

C'est la méthode la plus simple. Elle affiche un texte fixe à chaque connexion.

- **Fichier concerné :** `/etc/motd`
- **Fonctionnement :** Le contenu de ce fichier est simplement lu et affiché par le système.
- **Comment le modifier :**

```
nano /etc/motd
```

- **Note :** Sur les systèmes modernes, ce fichier est souvent vide car Debian privilégie la version dynamique.

---

2. Le MOTD Dynamique (Avancé)

Le MOTD dynamique permet d'afficher des informations en temps réel (température, utilisation CPU, messages aléatoires).

Fonctionnement technique

Le système utilise un utilitaire appelé `update-motd`. Lors de la connexion, il exécute dans l'ordre numérique tous les scripts situés dans : `/etc/update-motd.d/`

Les scripts par défaut sont généralement :

- `00-header` :
- `10-uname` :
- `90-footer` :

Créer son propre message dynamique

Pour ajouter votre propre logique (comme un message aléatoire) :

1. **Créer un nouveau script** (utilisez un numéro élevé comme `99` pour qu'il s'affiche à la fin) :

   ```
   nano /etc/update-motd.d/99-custom
   ```

2. **Exemple de script pour messages aléatoires :**

   ```
   #!/bin/bash

   # Liste des messages possibles
   messages=(
       "Bienvenue votre machine."
       "Rappel : Vérifie l'état des disques avec 'df -h'."
       "Astuce : On ne fait jamais rm -rf .* sans réfléchir !"
       "L'uptime actuel est de : $(uptime -p)"
   )

   # Tirage au sort
   RANDOM_MSG=${messages[$RANDOM % ${#messages[@]}]}

   # Affichage avec couleur (Cyan)
   echo -e "\e[1;36m[INFO] $RANDOM_MSG\e[0m"
   ```

3. **Donner les droits d'exécution :** C'est l'étape la plus importante. Si le script n'est pas exécutable, il sera ignoré.

   ```
   chmod +x /etc/update-motd.d/99-custom
   ```

---

3. Configuration de SSH pour le MOTD

Pour que le MOTD s'affiche correctement en SSH, le serveur doit être configuré pour cela.

- **Fichier :** `/etc/ssh/sshd_config`
- **Options requises :**
  - `PrintMotd yes` : Affiche le contenu de `/etc/motd` (statique).
  - `PrintLastLog yes` : Affiche la date de la dernière connexion (optionnel).

Après modification de la config SSH, redémarrez le service : `systemctl restart ssh`

---

4. Astuces de dépannage

Si votre message ne s'affiche pas :

1. **Tester manuellement :** Exécutez la commande suivante pour voir si vos scripts génèrent des erreurs :

   ```
   run-parts /etc/update-motd.d
   ```

2. **Vérifier le shell :** Assurez-vous que la première ligne de votre script est bien `#!/bin/bash` ou `#!/bin/sh`.
3. **Nettoyage :** Si vous avez des messages en double, vérifiez que vous n'avez pas à la fois un texte dans `/etc/motd` ET un script dans `/etc/update-motd.d/`.

Ajout d'un service dans systemd

Mettre un bot Python dans systemd

Ce guide explique comment exécuter un bot Python comme service **systemd** sur Linux, avec un environnement virtuel et un fichier de variables d'environnement. Le point clé est d'utiliser le binaire Python du virtualenv directement dans `ExecStart`, puis d'activer le service pour qu'il démarre automatiquement au boot.

Prérequis

Le projet doit idéalement contenir :

- `bot.py`
- un virtualenv, ici nommé `env`
- un fichier `.env` pour les variables sensibles comme le token Discord

Exemple d'arborescence :

```
/home/toto/discord-bot-application/
├── bot.py
├── .env
└── env/
```

Le virtualenv peut avoir n'importe quel nom, mais il faut ensuite pointer vers le bon chemin dans `ExecStart`.

Vérifier les chemins

Avant de créer le service, il faut vérifier que les chemins existent réellement :

```
ls -la /home/toto/discord-bot-application
ls -la /home/toto/discord-bot-application/env/bin/python
```

Si le second chemin existe, systemd pourra lancer le script avec le Python du virtualenv.

Créer le fichier .env

Le fichier `.env` sert à stocker les variables d'environnement du bot, comme le token Discord ou les IDs du serveur et du salon. Avec systemd, ce fichier peut être chargé via `EnvironmentFile=`.

Exemple :

```
DISCORD_BOT_TOKEN=TON_TOKEN_ICI
DISCORD_GUILD_ID=TON_SERVER_ID
DISCORD_CHANNEL_ID=TON_CHANNEL_ID
```

Le format attendu est `CLE=VALEUR`, sans `export`.

Pour limiter l'accès au fichier :

```
chmod 600 /home/toto/discord-bot-application/.env
```

Tester le bot manuellement

Avant de passer par systemd, il est préférable de confirmer que le script démarre correctement à la main depuis le virtualenv.

```
cd /home/toto/discord-bot-application
source env/bin/activate
python bot.py
```

Si le bot affiche un message comme `Bot prêt`, la partie Python est probablement correcte.

Créer le service systemd

Créer le fichier du service :

```
sudo nano /etc/systemd/system/discord-job-bot.service
```

Puis coller ce contenu :

```
[Unit]
Description=Discord Job Bot
After=network.target

[Service]
Type=simple
User=toto
Group=toto
WorkingDirectory=/home/toto/discord-bot-application
```

```
EnvironmentFile=/home/toto/discord-bot-application/.env
Environment=PYTHONUNBUFFERED=1
ExecStart=/home/toto/discord-bot-application/env/bin/python /home/toto/discord-bot-application/bot.py
Restart=always
RestartSec=5

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

Explication des lignes importantes

- `User` et `Group` : définissent l'utilisateur Linux qui exécutera le bot.
- `WorkingDirectory` : fixe le dossier courant du projet.
- `EnvironmentFile` : charge les variables du fichier `.env`.
- `Environment=PYTHONUNBUFFERED=1` : force l'affichage immédiat des logs Python dans journalctl.
- `ExecStart` : lance directement le script avec le Python du virtualenv `env`.
- `Restart=always` : redémarre automatiquement le bot s'il s'arrête.

Recharger systemd et démarrer le service

Après création ou modification du fichier `.service`, il faut recharger la configuration systemd, activer le service au démarrage, puis le lancer.

```
sudo systemctl daemon-reload
sudo systemctl enable discord-job-bot.service
sudo systemctl start discord-job-bot.service
```

Vérifier l'état du service

Pour vérifier que le service est bien actif :

```
sudo systemctl status discord-job-bot.service
```

Pour suivre les logs en direct :

```
journalctl -u discord-job-bot.service -f
```

`journalctl` est l'outil standard pour consulter les logs systemd et diagnostiquer les problèmes de démarrage.

Commandes utiles

Redémarrer le service après modification du code :

```
sudo systemctl restart discord-job-bot.service
```

Arrêter le service :

```
sudo systemctl stop discord-job-bot.service
```

Désactiver le lancement automatique au démarrage :

```
sudo systemctl disable discord-job-bot.service
```

Erreurs fréquentes

Les problèmes les plus fréquents sont :

- mauvais chemin dans `ExecStart`
- mauvais utilisateur dans `User`
- fichier `.env` absent ou mal formaté
- dépendances Python non installées dans le virtualenv
- permissions insuffisantes sur les fichiers du projet

Pour afficher les 50 dernières lignes de logs sans pagination :

```
journalctl -u discord-job-bot.service -n 50 --no-pager
```

Exemple final prêt à copier

Fichier service

```
[Unit]
Description=Discord Job Bot
After=network.target

[Service]
Type=simple
User=toto
Group=toto
WorkingDirectory=/home/toto/discord-bot-application
EnvironmentFile=/home/toto/discord-bot-application/.env
Environment=PYTHONUNBUFFERED=1
ExecStart=/home/toto/discord-bot-application/env/bin/python /home/toto/discord-bot-application/bot.py
```

```
Restart=always
RestartSec=5

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

Commandes d'activation

```
sudo systemctl daemon-reload
sudo systemctl enable discord-job-bot.service
sudo systemctl start discord-job-bot.service
sudo systemctl status discord-job-bot.service
journalctl -u discord-job-bot.service -f
```