

# Distributions

Ici toutes les spécificité par rapport au distributions

- Note sur les Distributions
  - Pop!\_OS
  - Rescuezilla
  - KDE Neon
  - Ubuntu
  - Fedora
  - Debian
  - Batocera
  - SteamOS (SteamDeck)
  - Solus OS
  - Manjaro
  - Secureboot
- Note sur les environnements
  - KDE

# Note sur les Distributions

# Pop!\_OS

`ubuntu-restricted-extras` installé de base sous Ubuntu permet d'avoir tous les codecs pour lire les vidéos.

## Affichage 24h sous GDM.

Pour avoir l'heure sous format 24h dans gdm3 (écran de connexion) il suffit de rajouter sous :

`[org/gnome/desktop/interface]`

La ligne :

`clock-format='24h'`

Dans le fichier qui se trouve dans : **`/etc/gdm3/greeter.dconf-defaults`**

## Mise à niveau sur version beta.

Pour mettre une Pop!-OS à niveau en version bêta, il suffit de taper la ligne de commande suivante :

`sudo pop-upgrade release upgrade -f`

## Changement du mot de passe de la partition chiffrée.

Commençons par lister les partitions de tous les disques installés :

`lsblk -f`

Le résultat peut être différent en fonction de la configuration du disque et de la table de partition. Vous pouvez voir la partition racine au point de montage '/'. À partir de là, nous pouvons utiliser cette sortie pour exécuter cette commande :

`sudo cryptsetup luksDump /dev/sda3`

En remplaçant '/dev/sda3' par l'emplacement de la partition racine sur votre système. Avec la sortie de cette commande, nous pouvons voir les sept emplacements supplémentaires que nous avons pour les mots de passe pour décrypter le lecteur. Les emplacements 1-7 sont les emplacements ouverts et l'emplacement 0 est l'emplacement actuel que le système a défini.

## Définir un mot de passe supplémentaire.

En suivant le schéma de partition de la commande précédente, nous pouvons former la commande suivante pour ajouter une nouvelle clé à l'emplacement de clé ouvert :

```
sudo cryptsetup luksAddKey /dev/sda3
```

Cette commande demandera le mot de passe de cryptage actuel avant que le nouveau mot de passe puisse être ajouté. Confirmez le mot de passe

Exécutons à nouveau cette commande pour confirmer que le mot de passe supplémentaire est défini :

```
sudo cryptsetup luksDump /dev/sda3
```

Vous devriez voir que l'emplacement de clé 1 est maintenant activé, ce qui confirme que le nouveau mot de passe est défini.

### Source

Pour révoquer une clé contenue dans un "slot" :

```
sudo cryptsetup luksKillSlot /dev/sda3 <numero_de_slot>
```

### Source

## Utiliser l'empreinte digitale pour à la place du mot de passe.

Pour pouvoir utiliser l'empreinte digitale (fingerprint) ailleurs que pour juste entrer dans la session, par exemple lors d'une commande **sudo** ou lors d'une installation de programme via le magasin. Il suffit de taper cette commande dans un terminal :

```
sudo pam-auth-update
```

Ensuite sélectionner **Fingerprint** et OK.

# Rescuezilla

Rescuezilla est une distribution identique à Clonezilla mais en interface graphique, il intègre également des outils tel que Gparted et Gnome-disque pour pouvoir faire des manipulations.

## Téléchargement

[Github](#)

Version d'Ubuntu

## Installation de Blivet-Gui dans Rescuezilla

Il peut être utile de pouvoir utiliser Blivet-Gui dans Rescuezilla pour pouvoir manipuler les partitions chiffrées.

Il suffit de lancer la commande suivante dans un terminal depuis Rescuezilla :

```
wget https://gitlab.com/julien_68/blivet-gui_rescuezilla/-/raw/main/installation_blivet-gui.sh -O- | tr -d '\r' >
installation_blivet-gui-rescuezilla.sh && sh installation_blivet-gui-rescuezilla.sh && exit
```

## Installation de Boot-Repair

Boot repair peut être utile pour réparer un démarrage.

Installation et exécution :

```
apt-get update && apt-get install software-properties-common python3-software-properties -y && add-apt-
repository -y ppa:yannubuntu/boot-repair && apt update && apt install -y boot-repair ; boot-repair
```

Attention, Les icônes du bureau ont tendance à disparaître, pas de souci, ce n'est qu'un live.

## Désactiver la mise en veille de l'écran

Pour désactiver la mise en veille de l'écran, voici les 2 commandes à lancer via le terminal XFCE :

```
xset s off
```

```
xset -dpms
```

## Source

# KDE Neon

## Écran noir fond vidéo

Pour résoudre le fond d'écran noir lors de la mise en place d'un fond d'écran vidéo via un plugin, il faut installer les paquets suivants puis redémarrer :

```
sudo apt-get install gstreamer1.0-plugins-bad gstreamer1.0-plugins-base gstreamer1.0-plugins-good  
gstreamer1.0-libav
```

## Restaurer le son de démarrage.

Il suffit de lancer ce script dans un terminal :

```
wget -P ~/.config/autostart/ https://gitlab.com/Julien.68/sound_on_startup_kde/-/raw/main/paplay.desktop
```

## Régler le problème de Discover

Pour régler le problème de paquet des dépôts d'Ubuntu introuvable sur Discover,  
Il suffit d'installer le paquet **discover**

```
sudo apt-get install discover
```

## Récupérer les photos de l'image du jour

Les photos des fonds d'écran se trouvent dans le dossier `~/.cache/plasma_engine_potd/`

# Ubuntu

## Installation de Firefox en DEB plutôt qu'en SNAP.

J'ai fait un petit script baser sur un tuto de numétopia, pour l'utiliser, il suffit de lancer la commande suivante dans un terminal :

```
wget -P /tmp https://gitlab.com/julien_68/firefox-deb-no-snap/-/raw/main/script.sh && sh /tmp/script.sh
```

[source](#)

## Problèmes réseau après sortie de veille

Après la sortie de veille de ma Kubuntu, j'ai remarqué que je n'avais plus de réseau, cela vient d'un problème lors de la sortie de veille des périphériques PCI.

Pour résoudre ce problème, il suffit de rajouter cette option dans GRUB:

**sky2.disable\_msi=1** ou **pci=noms,ioaer**

```
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash sky2.disable_msi=1"
```

ou

```
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash pci=noms,ioaer"
```

[source](#)

## Complément langue.

Pour ajouter le complément de langue, il suffit de lancer la commande suivante :

```
sudo apt install $(check-language-support)
```

## Gestion de performance du CPU.

L'utilitaire **cpu-frequtils** permet de gérer le profile du CPU.

Cette documentation est très utile : [doc Ubuntu](#)



# Kubuntu

## Mise à niveau

Pour faire la mise à niveau, il suffit de lancer cette commande :

```
pkexec do-release-upgrade -m desktop -f DistUpgradeViewKDE
```

## Pavé numérique au démarrage.

Pour avoir le pavé numérique activer dans SDDM (le gestionnaire de session) il suffit de rajouter ces infos dans le fichier **sddm.conf** qui se trouve dans **/etc**, si le fichier n'existe pas, il faut le créer.

```
[General]
Numlock=on
```

Voici un moyen de le mettre en une seule fois via la ligne de commande suivante :

```
echo -e "[General]\nNumlock=on" | sudo tee -a /etc/sddm.conf
```

## Restaurer le son de démarrage.

Il suffit de lancer ce script dans un terminal :

```
wget -P ~/.config/autostart/ https://gitlab.com/Julien.68/sound_on_startup_kde/-/raw/main/paplay.desktop
```

## Thème sombre fenêtre root.

Pour mettre les fenêtres que l'on ouvre en root dans le même thème, il suffit de lancer la configuration système en root.

```
sudo -i systemsettings5
```

Il suffit ensuite de mettre le même thème.

## Récupérer les photos de l'image du jour

Les photos des fonds d'écran se trouvent dans le dossier `~/.cache/plasma_engine_potd/`

## Raccourci clavier.

Pour pouvoir utiliser la touche méta seule pour un raccourci clavier KDE il suffit de faire la combinaison de touche `[ALT][F1]`.

## Problèmes

### APT-KEY deprecation warning

Cette erreur s'affiche généralement lorsque l'on recharge les sources.

Erreur :

```
W: https://linux.teamviewer.com/deb/dists/stable/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
```

Pour corriger le problème, il faut que la clé soit stockée dans un fichier individuel dans

`/etc/apt/trusted.gpg.d`.

Nous allons lister les clés :

```
sudo apt-key list
```

On va prendre exemple avec Teamviewer :

```
pub rsa4096 2020-01-29 [SC]
8CAE 012E BFAC 38B1 7A93 7CD8 C5E2 2450 0C12 89C0
uid [ unknown] TeamViewer Germany GmbH (TeamViewer Linux 2020) <support@teamviewer.com>
```

On prend les 8 derniers caractères de la clé, ici `0C12 89C0` et on enlève l'espace.

On exécute la commande suivante avec les 8 caractères précédemment relevés et on personnalise le nom du fichier :

```
sudo apt-key export 0C1289C0 | sudo gpg --dearmor -o /etc/apt/trusted.gpg.d/teamviewer.gpg
```

### Source

Pour ajouter correctement une clé, voici la commande à utiliser et adapter :

```
wget -qO- https://myrepo.example/myrepo.asc | sudo tee /etc/apt/trusted.gpg.d/myrepo.asc
```

# Fedora

Après installation, il faut installer plusieurs choses pour avoir une meilleure expérience.

Voici le site où les trouver : [RPM Configuration](#).

La commande pour tout installer d'un coup :

```
sudo dnf install https://mirrors.rpmfusion.org/free/fedora/rpmfusion-free-release-$(rpm -E %fedora).noarch.rpm  
https://mirrors.rpmfusion.org/nonfree/fedora/rpmfusion-nonfree-release-$(rpm -E %fedora).noarch.rpm -y &&  
sudo dnf groupupdate core -y && sudo dnf groupupdate multimedia --setopt="install_weak_deps=False" --  
exclude=PackageKit-gstreamer-plugin -y && sudo dnf groupupdate sound-and-video -y && sudo flatpak remote-  
add --if-not-exists flathub https://flathub.org/repo/flathub.flatpakrepo && flatpak install flathub  
com.github.tchx84.Flatseal -y && echo Ok
```

# Debian

Je suis en phase de test sur Debian Testing.

## Téléchargement de l'ISO

L'ISO est téléchargeable à cette ici [Debian net-install](#)

Il y a également la version non-free [Debian non-free net-install](#)

## Sources

J'ai modifié les sources pour que ce soit une distribution 'rolling' c'est-à-dire qu'il n'y a plus de mise à niveau à effectuer.

Les modifications sont le nom de code de version, remplacer par testing, car testing restera toujours testing et ajout d'une source manquante.

**/etc/apt/sources.list**

```
deb http://deb.debian.org/debian/ testing main contrib non-free non-free-firmware
deb-src http://deb.debian.org/debian/ testing main contrib non-free non-free-firmware

deb http://security.debian.org/debian-security testing-security main contrib non-free
deb-src http://security.debian.org/debian-security testing-security main contrib non-free

deb http://deb.debian.org/debian testing-updates main contrib non-free
deb-src http://deb.debian.org/debian testing-updates main contrib non-free
```

## Dépôt Unstable.

J'ai ajouté le dépôt unstable parce qu'il y a certains paquets qui ne sont pas disponibles dans le testing.

Rajout dans **/etc/apt/sources.list**

```
# dépôt instable
```

```
deb http://deb.debian.org/debian/ unstable main contrib non-free non-free-firmware
deb-src http://deb.debian.org/debian/ unstable main contrib non-free non-free-firmware
```

Réglage d'une priorité basse pour utiliser les dépôts testing en priorité :  
créer le fichier dans **/etc/apt/preferences.d/50-unstable** et mettre cette entrée :

```
Package: *
Pin: release n=sid
Pin-priority:100
```

Source

## Dépôt Backports

Le dépôt backports permet d'avoir des paquets plus récents sur la branche stable.

Il faut d'abord rajouter le dépôt dans le `sources.list`

```
deb http://deb.debian.org/debian bookworm-backports main contrib non-free non-free-firmware
```

Pour l'installation du noyau, voici la syntaxe :

```
sudo apt -t bookworm-backports install linux-image-amd64
```

## Mettre un logo de chargement.

Il suffit d'installer le paquet `plymouth-themes` :

```
sudo apt-get install plymouth-themes
```

De rajouter l'option `splash` a la ligne **GRUB\_CMDLINE\_LINUX\_DEFAULT** dans `/etc/default/grub`  
Et enfin régénérer Grub avec la commande :

```
sudo update-grub
```

Source : [Linuxconfig.org](http://Linuxconfig.org)

## Problèmes

## Problèmes au premier démarrage avec GPU AMD.

J'ai eu un problème lors du premier boot car au final, il me manquait un paquet pour le GPU AMD. Il suffit lors de l'affichage du grub de rajouter l'option **nomodeset** sur la ligne **linux** puis démarrer. Une fois sur la session, il faut ouvrir un terminal puis modifier les sources (/etc/apt/sources.list) et à la suite des **main** derrière chaque source, il faut ajouter **contrib non-free** puis installer les paquets suivants : `firmware-linux-nonfree firmware-amd-graphics`. Une seule ligne de commande pour tout faire :

```
sudo sed -i 's/main/main contrib non-free/g' /etc/apt/sources.list
sudo apt-get install firmware-linux-nonfree firmware-amd-graphics -y
```

## Problème de lecture de fichier multimédia.

Il suffit de faire l'installation suivante :

```
sudo apt-get install sox twolame vorbis-tools lame faad libavcodec-extra
```

## Enlever le bip système.

Pour désactiver ce bip de la mort lors de l'arrêt ou redémarrage, il suffit de lancer cette commande :

```
sudo touch /etc/modprobe.d/blacklist.conf
echo blacklist pcspkr | sudo tee -a /etc/modprobe.d/blacklist.conf
sudo update-initramfs -u
sudo update-grub
```

Pour le désactiver dans le terminal lors de la tabulation, il faut décommenter l'option suivante dans **/etc/inputrc**:

```
set bell-style none
```

## Comment trouver Steam.

Il faut activer les dépôts 32 bits comme ceci :

```
sudo dpkg --add-architecture i386
```

## Utiliser shutdown sans sudo.

Pour cela, il suffit de rajouter la ligne suivante dans **/etc/sudoers**:

```
user hostname =NOPASSWD: /usr/bin/systemctl poweroff,/usr/bin/systemctl halt,/usr/bin/systemctl reboot
```

En remplaçant bien **user** par l'utilisateur et **hostname** par le nom de l'ordinateur.

Ensuite, il faut (sous ZSH) ajouter les chemins suivants pour la reconnaissance de la commande :

`~/.zshrc` Ajouter ces 2 chemins à la ligne **export PATH=\$PATH**

```
/usr/bin:/usr/sbin
```

## Rajouter les firmwares amdgpu manquants.

Les firmwares manquants se trouvent sur [kernel.org](https://kernel.org).

Un petit script pour les télécharger et placer au bon endroit, le paquet `git` est requis :

```
mkdir /tmp/amdgpu
cd /tmp/amdgpu
git clone https://git.kernel.org/pub/scm/linux/kernel/git/firmware/linux-firmware.git
cd ~
sudo cp /tmp/amdgpu/linux-firmware/amdgpu/* /lib/firmware/amdgpu/
sudo update-initramfs -u
```

## Branche testing lors de sortie officielle de la version suivante.

Quand on est sur la branche testing, lors de la sortie officielle de la version suivante, on obtient ce message lors du rechargement des dépôts :

```
E: Le dépôt « http://security.debian.org/debian-security InRelease » a modifié sa valeur
« Codename » de « bookworm-security » à « trixie-security »
N: Ceci doit être pleinement accepté avant que les mises à jour depuis ce dépôt puissent être appliquées.
Veuillez vous référer aux pages de manuel « apt-secure(8) » pour plus de détails.
E: Le dépôt « http://deb.debian.org/debian testing InRelease » a modifié sa valeur « Codename » de
« bookworm » à « trixie »
N: Ceci doit être pleinement accepté avant que les mises à jour depuis ce dépôt puissent être appliquées.
Veuillez vous référer aux pages de manuel « apt-secure(8) » pour plus de détails.
E: Le dépôt « http://deb.debian.org/debian testing-updates InRelease » a modifié sa valeur « Codename » de
« bookworm-updates » à « trixie-updates »
N: Ceci doit être pleinement accepté avant que les mises à jour depuis ce dépôt puissent être appliquées.
Veuillez vous référer aux pages de manuel « apt-secure(8) » pour plus de détails.
```

Solution

Lancer la commande suivante :

```
sudo apt-get --allow-releaseinfo-change update
```

## Connaître la version du noyau :

Voici la commande pour avoir la vraie version du noyau Linux :

```
$(echo $(dpkg -l | grep -i linux-image-amd64 | awk -F ' ' '{print $3}'))
```

## Installer des PPA sous Debian :

Un script fait par Xdavius permet de rajouter simplement un PPA sous Debian, il est disponible sur son [github](#).

Personnellement, j'utilise un alias qui va télécharger son script, installer le PPA et comme ça, la prochaine fois, il sera à jour.

```
wget https://raw.githubusercontent.com/Xdavius/debs/main/add-ppa-debian && chmod +x add-ppa-debian &&  
sudo ./add-ppa-debian
```

Mon alias est `ppa-deb`, donc la syntaxe est :

```
sudo ./add-ppa-debian ppa:user/ppa-name
```

Exemple pour : <https://launchpad.net/~oibaf/+archive/ubuntu/graphics-drivers>:

```
sudo ./add-ppa-debian ppa:oibaf/graphics-drivers
```

## Problème d'impression.

Pour résoudre l'erreur : *unable to open print file permission denied*

Il faut rajouter l'utilisateur au groupe `lp`.

```
sudo adduser $USER lp
```

Puis redémarrer l'ordinateur.



Source.

# Problème pip3 :

J'ai eu un problème en voulant installer WoeUSB-ng, l'erreur est la suivante :

```
sudo pip3 install WoeUSB-ng
error: externally-managed-environment
```

× This environment is externally managed  
↳ To install Python packages system-wide, try apt install python3-xyz, where xyz is the package you are trying to install.

If you wish to install a non-Debian-packaged Python package, create a virtual environment using `python3 -m venv path/to/venv`. Then use `path/to/venv/bin/python` and `path/to/venv/bin/pip`. Make sure you have `python3-full` installed.

If you wish to install a non-Debian packaged Python application, it may be easiest to use `pipx install xyz`, which will manage a virtual environment for you. Make sure you have `pipx` installed.

See `/usr/share/doc/python3.11/README.venv` for more information.

note: If you believe this is a mistake, please contact your Python installation or OS distribution provider. You can override this, at the risk of breaking your Python installation or OS, by passing `--break-system-packages`.

hint: See PEP 668 for the detailed specification.

Pour résoudre le problème, j'ai rajouté l'option `--break-system-packages` :

```
sudo pip3 install WoeUSB-ng --break-system-packages
```

Pour le résoudre complètement :

1. Suppression du fichier `EXTERNALLY-MANAGED` qui se trouve dans le dossier `/usr/lib/python3.x/`.
2. Création du fichier `pip.conf` dans `~/.config/pip/` et ajout des lignes suivantes :

```
[global]
break-system-packages = true
```

Source

# Installation PipeWire.

Voici les paquet à installer pour pipewire sous Debian 12

```
sudo apt-get install pipewire pipewire-audio-client-libraries pipewire-alsa pipewire-jack pipewire-audio  
wireplumber pipewire-pulse pipewire-alsa libspa-0.2-bluetooth
```

Ensuite in faut activer WirePlumber dans systemd (à exécuter en simple utilisateur).

```
systemctl --user --now enable wireplumber.service
```

Si cela a bien fonctionné, la sortie de la commande suivante devrait afficher "Server Name: PulseAudio (on PipeWire 0.3.XX)"

```
LANG=C pactl info | grep '^Server Name'
```

Source.

# Changer de serveur DNS en ligne de commande.

Pour changer de serveur DNS sous Debian, il faut installer `resolvconf`

```
sudo apt-get update  
sudo apt-get install resolvconf
```

Vérifier que le service est lancé avec la commande :

```
systemctl status resolvconf.service
```

Si ce n'est pas le cas :

```
sudo systemctl start resolvconf.service  
sudo systemctl enable resolvconf.service  
sudo systemctl status resolvconf.service
```

Éditer le fichier head :

```
sudo nano /etc/resolvconf/resolv.conf.d/head
```

Préciser `nameserver` suivi de son IP :

```
# exemple pour cloudflare  
nameserver 1.1.1.1  
nameserver 1.0.0.1
```

Ensuite, il faut mettre à jour et appliquer les changements avec les 2 commandes suivantes :

```
sudo resolvconf --enable-updates  
sudo resolvconf -u
```

Source

# Batocera

Batocera.linux est une distribution de jeu rétro-source open source entièrement gratuite qui peut être copiée sur une clé USB ou une carte SD dans le but de transformer n'importe quel ordinateur / nano en console de jeu pendant un jeu ou de façon permanente. Batocera.linux ne nécessite aucune modification sur votre ordinateur. Notez que vous devez posséder les jeux auxquels vous jouez afin de vous conformer à la loi.

<https://www.youtube.com/embed/3Pi1wLjShkQ>

<https://batocera.org>

## Ajout de Flatpak

Pour ajouter des flatpak, il suffit d'aller dans le gestionnaire de fichier, menu application pour flatpak-config.

Une fois installé, il faut lancer la commande suivante :

```
batocera-flatpak-update
```

Les logiciels se trouvent dans le menu PORTS.

## Switch

### Rajouter la Switch

#### Prérequis

Télécharger les BIOS [Switch2.rar](#)

#### Installation

Lancer dans le terminal la commande :

```
curl -L switch.batocera.pro | bash
```

Ensuite, écraser les fichiers dans Batocera, avec les fichiers de l'archive précédemment télécharger.

## Rajouter Mii Switch

Lien du fichier à récupérer [Firmware and Keys 16.0.3.rar](#)

## Depuis un autre OS

Le contenu du dossier Firmware est à mettre dans :

```
/SHARE/system/configs/yuzu/nand/system/Contents/registered/
```

Le contenu de Keys dans :

```
/SHARE/bios/switch/
```

## Depuis Batocera

Le contenu du dossier Firmware est à mettre dans :

```
/userdata/system/configs/yuzu/nand/system/Contents/registered/
```

Le contenu de Keys dans :

```
/userdata/bios/switch/
```

# SteamOS (SteamDeck)

## Mode jeux

### Decky

Decky est un programme qui permet de rajouter des plugins divers et varier en mode jeux.

Mes plugins :

TunnelDeck permet d'utiliser les VPN qui ont été configurés en mode bureau.

Recorder permet de faire des enregistrements vidéo de l'écran, il a également le mode replay.

Autoflatpaks permet de mettre à jour les flatpak depuis le mode jeux.

AnimationChanger permet de changer l'animation de démarrage, ainsi que la mise en veille.

## Utilisation avancée.

### Créer un mot de passe utilisateur.

De base, il n'y a pas de mot de passe pour l'utilisateur `deck` de créer, cela protège des connexions SSH.

L'utilisateur `deck` fait également partie du groupe `sudo`, pour créer un mot de passe, il suffit de lancer la commande suivante :

```
passwd
```

# Désactiver ou activer les modifications système.

De base la SteamDeck est verrouillé au niveau des modifications système, voici la commande pour activer ou désactiver les modifications système :

```
sudo steamos-readonly disable  
sudo steamos-readonly enable
```

## Activer les dépôts pacman.

Pour activer les dépôts Pacman sur la SteamDeck, il faut lancer les commandes suivantes :

```
sudo rm -r /etc/pacman.d/gnupg  
sudo pacman-key --init  
sudo pacman-key --populate
```

# Solus OS

## Installation des outils de compilations

```
sudo eopkg install -c system.devel linux-current-headers git
```

## Historique et restauration du logiciel dans Solus

Solus propose une fonctionnalité permettant d'afficher votre historique des modifications logicielles et de rétablir les versions précédentes de votre système.

Vous pouvez utiliser cette fonctionnalité si vous rencontrez des problèmes après l'installation de packages ou de mises à jour du système et que vous devez revenir à un état de fonctionnement.

## Historique des modifications logicielles

- Pour voir votre historique des modifications logicielles, ouvrez un terminal et exécutez :

```
eopkg history
```

Le terminal affiche une liste de toutes les modifications logicielles apportées à votre système.

sortie de l'historique eopkg

## Restauration

Important



L'annulation des modifications logicielles fonctionne si :

- Le référentiel Solus possède la version du package dont vous avez besoin, ou
- Vous disposez d'une copie locale du package.

1. Désactivez tout référentiel local dont vous disposez.

Les référentiels locaux peuvent provoquer des erreurs lors des restaurations.

2. Ouvrez un terminal.

3. Afficher l'historique des modifications logicielles :

```
eopkg history
```

4. Notez le numéro de la modification logicielle la plus récente.

5. Revenez à l'état souhaité :

```
sudo eopkg history -t NUMBER
```

Remplacer `NUMBER` avec un nombre inférieur à la dernière modification logicielle.

Par exemple, si vous souhaitez annuler l'opération 100, utilisez 99 comme nombre.

Après être revenu à l'état souhaité, vérifiez l'état de votre système. Vous devrez peut-être redémarrer votre système pour voir certaines des modifications.

Source

# Réglage de la luminosité du clavier sur Macbook.

J'ai installé Solus KDE sur un Macbook de 2012, mais le raccourci clavier pour faire varier la luminosité du clavier ne fonctionnait pas.

Voici comment j'ai fait pour palier à ce problème.

Créer un script dans `/usr/bin` qui permettra d'augmenter ou diminuer la luminosité.

```
sudo nano /usr/bin/keyboard-backlight
```

Écrire le contenu suivant à l'intérieur :

```
#!/usr/bin/bash  
VALUE=$(cat /sys/class/leds/smc::kbd_backlight/brightness)
```

```
INCREMENT=32
TOTAL=unset

case $1 in
up)
TOTAL=`expr $VALUE + $INCREMENT`
;;
down)
TOTAL=`expr $VALUE - $INCREMENT`
;;
full)
TOTAL=255
;;
off)
TOTAL=0
;;
esac

if [ $TOTAL == unset ]; then
echo "Please specify up, down, full, or off"
exit 1
fi

if [ $TOTAL -gt 255 ]; then TOTAL=255; fi
if [ $TOTAL -lt 0 ]; then TOTAL=0; fi
echo $TOTAL > /sys/class/leds/smc::kbd_backlight/brightness
```

Regler les permissions sur 755

```
sudo chmod 755 /usr/bin/keyboard-backlight
```

Entré dans visudo

```
sudo visudo
```

Rajouter la ligne suivante en modifiant `ici_utilisateur` par le nom de l'utilisateur

```
ici_utilisateur ALL=NOPASSWD:/usr/bin/keyboard-backlight up, /usr/bin/keyboard-backlight down
```

Cela permet de pouvoir lancer les commande pour augmenter ou diminuer la luminosité en `sudo` sans mots de passe.

Il reste plus qu'à rajouter deux raccourcis clavier avec les commandes suivantes :

Pour augmenter

```
sudo keyboard-backlight up
```

Pour diminuer

```
sudo keyboard-backlight down
```

Source

# Manjaro

## Pour Jouer aux jeux avec Wine.

```
sudo pacman --needed -S alsa-plugins lib32-alsa-plugins alsa-lib lib32-alsa-lib \
freetype2 lib32-freetype2 \
libxft lib32-libxft \
flex lib32-flex \
fluidsynth lib32-fluidsynth \
libxrandr lib32-libxrandr xorg-xrandr \
libldap lib32-libldap \
mpg123 lib32-mpg123 \
libxcomposite lib32-libxcomposite \
libxi lib32-libxi \
libxinerama lib32-libxinerama \
libxss lib32-libxss \
openal lib32-openal \
pipewire-pulse lib32-pipewire pipewire-jack lib32-pipewire-jack \
krb5 lib32-krb5 \
gnutls lib32-gnutls \
giflib lib32-giflib \
gst-libav gst-plugin-pipewire gst-plugins-ugly \
gst-plugins-bad gst-plugins-bad-libs \
gst-plugins-base-libs lib32-gst-plugins-base-libs gst-plugins-base lib32-gst-plugins-base \
gst-plugins-good lib32-gst-plugins-good \
gstreamer lib32-gstreamer \
libpng lib32-libpng \
v4l-utils lib32-v4l-utils \
vulkan-icd-loader lib32-vulkan-icd-loader \
libgpg-error lib32-libgpg-error \
libjpeg-turbo lib32-libjpeg-turbo \
libgcrypt lib32-libgcrypt \
ncurses lib32-ncurses \
ocl-icd lib32-ocl-icd \
```

```
libxslt lib32-libxslt \  
libxcrypt-compat lib32-libxcrypt-compat \  
libva lib32-libva \  
sqlite lib32-sqlite \  
gtk3 lib32-gtk3 \  
libpulse lib32-libpulse \  
openssl-1.1 lib32-openssl-1.1 \  
libnm lib32-libnm \  
gamemode lib32-gamemode \  
cabextract
```

## Spécifique AMD (GPU)

```
sudo pacman --needed -S mesa lib32-mesa vulkan-radeon lib32-vulkan-radeon vulkan-icd-loader lib32-vulkan-icd-loader
```

## Configurations supplémentaires

```
sudo sed -i "s@#DefaultLimitNOFILE=1024:524288@DefaultLimitNOFILE=1048576@g"  
/etc/systemd/system.conf  
sudo sed -i "s@#DefaultLimitNOFILE=@DefaultLimitNOFILE=1048576@g" /etc/systemd/user.conf
```

## Augmenter le vm.max\_map\_count

```
echo vm.max_map_count=2147483642 >> /etc/sysctl.d/99-sysctl.conf
```

## Contrôle

```
cat /proc/sys/vm/max_map_count
```

## Source

# Secureboot

Voici une page qui regroupe la gestion du secureboot pour les distributions.

## SolusOS

<https://help.getsol.us/docs/user/quick-start/installation/secure-boot/>

## CachyOS

[https://wiki.cachyos.org/configuration/secure\\_boot\\_setup/](https://wiki.cachyos.org/configuration/secure_boot_setup/)

## LinuxMint

<https://forums.linuxmint.com/viewtopic.php?p=2331461&sid=e4acb5bb9c44ece8314322cc6221d100#p2331461>

# Toutes les distributions qui utilisent Mokutils.

Vidéo => <https://youtu.be/yvoTlvJ0kJ0>

Wiki Adrien => <https://www.linuxtricks.fr/wiki/fedora-signer-les-modules-noyau-tiers-kmod-pour-le-secure-boot>

# Introduction

Fedora Linux utilise des signatures numériques approuvées par l'UEFI Secure Boot. Les binaires du chargeur de démarrage (comme GRUB) et le noyau de Fedora sont signés avec des certificats reconnus par les clés de la plateforme UEFI préinstallées sur la plupart des ordinateurs. Par conséquent, il n'est pas nécessaire de désactiver le Secure Boot pour installer et utiliser Fedora Linux.

Cependant, les modules fournis sous forme d'akmods, livrés dans RPM Fusion par exemple tels que NVidia, Broadcom, VirtualBox, ... sont compilés sur la machine et ne sont pas signés. Ils ne sont pas chargés si le Secure Boot est actif (oui, Secure Boot empêche de charger des modules non signés, c'est son rôle).

Plutôt que de désactiver ce mécanisme de sécurité, nous allons voir comment générer notre certificat et nos clés de signature et les importer dans l'UEFI de la machine.

Tous les modules (de type akmod) seront ensuite automatiquement signés lors de la compilation sans intervention manuelle.

Pour info, akmod signifie Auto-Kernel Module. Akmod compile les modules du noyau au démarrage si un nouveau noyau est détecté et aucun module précompilé n'est disponible. Cela assure que les modules sont toujours disponibles, même après une mise à jour du noyau.

## Prérequis

Toutes les manipulations seront réalisées avec les droits root.

Dans un premier temps, on installe les outils nécessaires pour la gestion des clés machines, et des modules akmods :

```
dnf install kmodtool akmods mokutil openssl
```

Une fois fait, vous pouvez vérifier le statut du SecureBoot avec la commande :

```
mokutil --sb-state
```

Sur ma machine, il est actif :

# Exemple avec un module non signé

Cet exemple est pour la démonstration dans ce tutoriel, ne l'installez pas si vous n'en avez pas le besoin !

Dans cet exemple, j'installe juste le module VirtualBox (qui n'est donc pas signé) :

```
dnf install akmod-VirtualBox
```

Je compile dans la foulée le kmod :

```
akmods
```

Lorsque j'essaie de charger le module :

```
modprobe vboxdrv
```

J'ai une erreur, car le module n'est pas signé et reconnu par ma machine :

```
modprobe: ERROR: could not insert 'vboxdrv': Key was rejected by service
```

## Génération des certificats et clés

Dans un premier temps, on va générer un certificat et une paire de clés pour signer les modules :

```
kmodgenca -a
```

L'option **-a** permet de générer avec des valeurs par défaut sans poser de questions.

Deux fichiers sont créés :

**/etc/pki/akmods/private/private\_key.priv** : la clé privée pour signer les modules

**/etc/pki/akmods/certs/public\_key.der** : la clé publique pour vérifier la signature

Ce sont en réalité des liens symboliques vers des noms différents suivant la machine, comme on peut le voir sur ma machine :

```
/etc/pki/akmods/certs/public_key.der -> /etc/pki/akmods/certs/superlinux-788348939.der
```

```
/etc/pki/akmods/private/private_key.priv -> /etc/pki/akmods/private/superlinux-788348939.priv
```



# Enregistrer les certificats et clés dans l'UEFI

## Lançons l'import dans l'UEFI

On va ensuite enregistrer les éléments précédemment générés dans l'UEFI et le module Secure Boot.

Cela permettra au noyau Linux de faire confiance aux pilotes signés avec notre clé :

```
mokutil --import /etc/pki/akmods/certs/public_key.der
```

On va devoir entrer un mot de passe.

Il n'a pas besoin d'être très long ou complexe. On en aura juste besoin pour importer la clé dans le secure boot.

Attention, on sera en qwerty pour la saisie, donc on fera simple !

Une fois fait, on va redémarrer le système :

```
reboot
```

## Validons l'import côté UEFI

Lorsque l'UEFI s'initialise, on est intercepté pour ajouter la clé en attente.

secureboot-mok1

On valide par Entrée

Ensuite :

secureboot-mok2

Sélectionner **Enroll MOK**

secureboot-mok3

Puis **Continue**

secureboot-mok4

Puis **Yes**

secureboot-mok5

On saisit le mot de passe renseigné précédemment (rappel on est en qwerty)

secureboot-mok6

Puis **Reboot**

# Installer ou réinstaller des modules

## Installer de nouveaux modules

Maintenant, tous les modules "akmods" seront compilés et signés automatiquement avec notre clé, que SecureBoot connaît.

Réinstaller des modules existants

Si on avait déjà des modules installés, il suffit de forcer leur recompilation :

```
akmods --rebuild --force
```

Dans l'exemple de ce tuto avec le module VirtualBox :

```
Checking kmods exist for 6.9.8-200.fc40.x86_64      [ OK ]  
Building and installing VirtualBox-kmod             [ OK ]
```

On régénère aussi l'image initrd avec :

```
dracut --force
```

Et enfin on reboot :

```
reboot
```

# Vérifier que les modules sont chargés

Si notre module est signé correctement, alors il est chargé par le système.

On pourra le constater avec la fameuse commande lsmod :

```
lsmod | grep vbox
```

Ici, le module est bien chargé :

```
vboxdrv      704512  1
```

# Signer des modules installés manuellement

Si vous n'avez pas de modules installés sous forme de "akmod", il est possible de signer les modules "manuellement".

Je prends dans cet exemple VirtualBox, mais installé avec le .rpm du site Oracle.

Après avoir généré et importé les certificats et clés, on recherche le chemin du module :

```
modinfo vboxdrv
```

Cela me renvoie :

```
filename:    /lib/modules/6.9.8-200.fc40.x86_64/misc/vboxdrv.ko
version:     7.0.18 r162988 (0x00330004)
license:     GPL
description:  Oracle VM VirtualBox Support Driver
```

```
author:      Oracle and/or its affiliates
srcversion:  9CBC44140E50A5E89770210
depends:
retpoline:   Y
name:        vboxdrv
vermagic:    6.9.8-200.fc40.x86_64 SMP preempt mod_unload
parm:        disabled:Disable automatic module loading (int)
parm:        force_async_tsc:force the asynchronous TSC mode (int)
```

Il existe un script pour signer les modules, qui est installé grâce au paquet kernel-devel nommé **sign-file**.

Son chemin est le suivant :

```
/usr/src/kernels/$(uname -r)/scripts/sign-file
```

Pour signer notre module, on utilisera le script, avec les clés privées et publiques créées par la commande **kmodgenca** :

```
/usr/src/kernels/$(uname -r)/scripts/sign-file sha256 /etc/pki/akmods/private/private_key.priv /etc/pki/akmods/certs
```

Pour mon exemple :

```
/usr/src/kernels/$(uname -r)/scripts/sign-file sha256 /etc/pki/akmods/private/private_key.priv /etc/pki/akmods/certs
```

Évidemment, ce sont des actions manuelles à faire à chaque mise à jour du noyau et recompilation du module !

On peut charger le module avec succès et vérifier que le module est signé :

```
modinfo vboxdrv
```

```
filename:    /lib/modules/6.9.8-200.fc40.x86_64/misc/vboxdrv.ko
version:     7.0.18 r162988 (0x00330004)
license:     GPL
description: Oracle VM VirtualBox Support Driver
author:      Oracle and/or its affiliates
srcversion:  9CBC44140E50A5E89770210
depends:
retpoline:   Y
name:        vboxdrv
vermagic:    6.9.8-200.fc40.x86_64 SMP preempt mod_unload
sig_id:      PKCS#7
signer:      fedora-4164085003
sig_key:     4B:2F:C9:65:78:87:ED:2D:8B:C0:03:C1:87:54:1B:1A:6F:9F:B7:6B
sig_hashalgo: sha256
```

signature: 91:EE:D8:DB:64:16:CE:40:C8:0F:BC:D9:A3:0F:7D:A5:57:4D:06:C7:  
E9:9D:01:94:92:C3:AE:0F:72:BF:23:B5:73:2C:DC:B3:43:8B:2A:7F:  
07:EA:72:6A:F7:38:CC:F5:72:79:02:05:27:EE:C9:83:14:CB:10:5D:  
C2:2E:AD:A8:E9:F2:B4:02:7F:72:B8:75:F4:85:F0:14:0E:67:B1:CE:  
AF:E3:35:6A:58:03:74:9E:BE:BC:05:D8:39:18:53:9C:DD:9F:9E:F0:  
DD:00:B0:80:05:32:D9:2B:ED:96:8D:D2:36:E8:8A:03:EA:00:7E:5D:  
6E:3A:1F:05:B4:F7:41:65:6D:33:EA:53:88:89:16:89:5D:AC:AC:59:  
B5:78:74:0C:7A:00:F4:09:18:DE:96:2F:8B:B8:60:6E:90:57:77:2A:  
A2:97:9C:C2:EE:78:6C:7B:50:16:62:AC:43:71:70:E2:15:E5:CE:40:  
EF:DF:50:93:08:33:EB:9F:59:AD:33:D0:3E:97:94:4D:4B:50:C3:FE:  
18:14:09:D9:6C:7A:33:90:48:DC:75:2D:CF:03:58:41:02:BC:BD:2F:  
D7:36:42:AC:34:D0:26:65:60:12:02:DA:D3:37:92:A4:3B:9E:40:B0:  
52:0F:F1:53:E6:48:BB:01:70:A5:0F:D0:81:29:E1:78:6B:53:97:A3:  
BC:36:F2:5F:F2:DD:E6:4F:53:AC:CE:4A:59:B6:86:D7:FA:68:A2:DA:  
F9:66:5D:53:21:40:EB:79:D7:B9:A9:D8:F3:22:6B:E1:98:DD:95:1D:  
AB:AD:71:E8:4A:BE:B7:2F:67:DB:95:A4:05:BD:3E:C0:FB:42:0D:1F:  
B5:34:2F:1E:7E:43:25:15:1B:DC:AE:60:24:BB:33:0D:52:C7:56:C4:  
A2:76:6E:6D:FA:12:F5:BA:D2:E4:AD:4F:69:F0:B4:EB:E6:11:CB:17:  
47:4E:6B:DA:22:6F:82:E6:BB:5C:CE:3B:EC:9F:BF:91:BF:84:A3:FB:  
89:0B:0D:8F:6F:05:3E:B9:17:42:5D:11:4D:D2:59:24:3D:14:8F:A6:  
92:64:47:D0:57:F3:59:D7:07:AF:51:31:80:CE:9E:5A:11:52:F1:91:  
1B:0F:4F:4F:39:9B:82:BA:25:DE:46:B9:A4:19:90:8C:60:0A:44:4B:  
E7:CA:49:D8:2D:78:4D:A0:71:E3:03:56:BC:41:31:47:0D:41:17:30:  
D8:D0:AA:D4:EE:50:C6:DB:92:E9:B0:25:02:77:72:61:31:A6:AD:55:  
16:9A:B1:AB:37:0D:93:D8:08:C1:A3:3C:04:4A:B5:80:12:B2:41:53:  
E6:B5:CB:38:71:05:32:F7:06:D3:A9:B0

parm: disabled:Disable automatic module loading (int)

parm: force\_async\_tsc:force the asynchronous TSC mode (int)

# Note sur les environnements

Note spécifique par environnement de bureau

# KDE

## Redémarrer plasma

Voici une commande pour redémarrer l'environnement plasma en cas de problèmes :

```
kquitapp5 plasmashell && kstart5 plasmashell > /dev/null 2>&1
```

## Raccourcis clavier

Pour utiliser uniquement la touche **SUPER**, il faut mettre en raccourcis **META** + **F7** ou **ALT** + **F1**.

## Pavé numérique au démarrage sur SDDM.

Pour avoir le pavé numérique activé dans SDDM (le gestionnaire de session) il suffit de rajouter ces infos dans le fichier **sddm.conf** qui se trouve dans **/etc**, si le fichier n'existe pas, il faut le créer.

```
[General]
Numlock=on
```

Voici un moyen de le mettre en une seule fois via la ligne de commande suivante

```
echo -e "[General]\nNumlock=on" | sudo tee -a /etc/sddm.conf
```

# Pavé numérique au démarrage de KDE sur wayland.

Pour avoir le pavé numérique activer au démarrage de la session KDE sur Wayland, il suffit de rajouter ces infos dans le fichier **kcminputrc** qui se trouve dans **~/.config**,

```
[Keyboard]
NumLock=0
```

Ensuite activer le service **fstrim**.

```
sudo systemctl enable fstrim.timer
```

Voici un moyen de le mettre en une seule fois via la ligne de commande suivante

```
echo -e "[Keyboard]\nNumLock=0" | tee -a ~/.config/kcminputrc
sudo systemctl enable fstrim.timer
```

[Source](#)

## Ouvrir Dolphin en ROOT (KDE5)

Il faut ajouter un fichier, par exemple `dolphin-root.desktop` dans `~/.local/share/kservices5/ServiceMenus/` avec le contenu suivant :

```
[Desktop Entry]
Type=Service
X-KDE-ServiceTypes=KonqPopupMenu/Plugin
MimeType=inode/directory;
Actions=ouvrirDolphinEnRoot;
X-KDE-AuthorizeAction=shell_access
[Desktop Action ouvrirDolphinEnRoot]
Name=Ouvrir Dolphin en root ici
TryExec=pkexec
Exec=pkexec env DISPLAY=$DISPLAY XAUTHORITY=$XAUTHORITY dolphin %U
Icon=folder-red
```



Voici une commande pour tout faire via le terminal :

```
mkdir -p ~/.local/share/kservices5/ServiceMenus/  
echo -e '[Desktop Entry]  
Type=Service  
X-KDE-ServiceTypes=KonqPopupMenu/Plugin  
MimeType=inode/directory;  
Actions=ouvrirDolphinEnRoot;  
X-KDE-AuthorizeAction=shell_access  
[Desktop Action ouvrirDolphinEnRoot]  
Name=Ouvrir Dolphin en root ici  
TryExec=pkexec  
Exec=pkexec env DISPLAY=$DISPLAY XAUTHORITY=$XAUTHORITY dolphin %U  
Icon=folder-red' > ~/.local/share/kservices5/ServiceMenus/dolphin-root.desktop
```

Source

# Ouvrir Dolphin en ROOT (KDE6)

Il faut ajouter un fichier, par exemple `dolphin-root.desktop` dans `/usr/share/kio/servicemenus/` avec le contenu suivant :

```
[Desktop Entry]  
Type=Service  
X-KDE-ServiceTypes=KonqPopupMenu/Plugin  
MimeType=inode/directory;  
Actions=ouvrirDolphinEnRoot;  
X-KDE-AuthorizeAction=shell_access  
[Desktop Action ouvrirDolphinEnRoot]  
Name=Ouvrir Dolphin en root ici  
TryExec=pkexec  
Exec=pkexec env DISPLAY=$DISPLAY XAUTHORITY=$XAUTHORITY dolphin %U  
Icon=folder-red
```

Voici une commande pour tout faire via le terminal :

```
echo -e '[Desktop Entry]  
Type=Service
```

```
X-KDE-ServiceTypes=KonqPopupMenu/Plugin
MimeType=inode/directory;
Actions=ouvrirDolphinEnRoot;
X-KDE-AuthorizeAction=shell_access
[Desktop Action ouvrirDolphinEnRoot]
Name=Ouvrir Dolphin en root ici
TryExec=pkexec
Exec=pkexec env DISPLAY=$DISPLAY XAUTHORITY=$XAUTHORITY dolphin %U
Icon=folder-red' | sudo tee -a /usr/share/kio/servicemenus/dolphin-root.desktop
```

# Composant graphique

## Panon

Un widget de visualisation audio pour KDE Plasma.



<https://github.com/flafflar/panon>

Dépendances pour Solus :

```
sudo eopkg install qt6-websockets python-docopt PyAudio numpy python-cffi python-websockets
```

## Choix du GPU (multi GPU)

Pour avoir le choix du GPU via le menu contextuel, il faut installer le paquet `switcheroo-control`

[Source](#)