

0. UEFI
1. GRUB
2. Noyau Linux
3. systemd
4. | -services
5. | -login
 - graphique
 - > X11 ou Wayland
 - > Env de
 - Bureau
 - Gnome/KDE/XFCE

Je veux installer le programme xxx dans une version récente ?

- il est dispo et à jour dans un dépôt de mon éditeur (red hat, rocky, debian, ubuntu)
cool !
 - il est dispo et à jour dans un dépôt du producteur d'origine (upstream) préparé pour votre distribution (ex: Docker, PostgreSQL, ...)
 - sinon ? absent des dépôts ou dans une vieille version
 - > rare sauf pour qq logiciels "clients lourds" ou produits propriétaires (ex : signal, airtame, ...)
1. télécharger les sources et compiler et installer
 - ./configure && make && make install
 - (maintenance et mises à jour à faire !)
 2. utiliser un format générique (pas deb, pas rpm) de "paquets" :
 - * snap (mis en avant par canonical)
 - * flatpak (de plus en plus populaire)

Résumé stockage:

disques sont /dev/sda /dev/sdb etc.

fichier "spéciaux" de type "bloc" (par opposition à caractère)

```
ls -l /dev/sda
```

comment partitionner un disque ?

simple pas très ergonomique : fdisk, parted

script : sfdisk

plus ergonomique : cfdisk

graphique : gparted

attention ! : cfdisk /dev/sda

Commençons par ajouter un nouveau disque dur
à notre machine virtuelle :

Virtual Box ne permet d'ajouter des disque
sur une vm allumée :

sudo poweroff

Configuration|Stockage

Contrôleur SATA

+ (ajoute un disque dur)

Créer... valeurs par défaut... Choose

Démarrer votre VM Debian

lsblk

-> on voit sdb ?

ls -l /dev/sdb

On va créer une partition unique sur le disque dur sdb avec cfdisk

```
$ sudo cfdisk /dev/sdb
```

```
-> type de "label" (table) : GPT 'GUID Partition Table'
```

```
-> Nouveau, laisser le défaut (occupe tout l'espace et type Linux)
```

```
-> Ecrire et taper "oui"
```

```
-> Quitter
```

```
$ lsblk
```

```
-> on doit voir sdb1
```

sdn

sdb1 existe, il faut le formater
état de l'art des filesystems "natif" Linux :
ext4, xfs

```
sudo mkfs.ext4 /dev/sdb1
```

ou

```
sudo apt install xfsprogs
```

```
sudo mkfs.xfs /dev/sdb1
```

en préparation : btrfs (pas officiellement
stable) cf. le wiki de btrfs pour voir le niveau
d'avancement

envisageable : ZFS (Zetabyte FS, SUN)

- * zfs utilise 50% de la RAM

il faut maintenant définir l'association entre cette partition formatée en XFS et un répertoire de notre choix

ex: choisissons : /srv/data
\$ sudo mkdir /srv/data

dans le fichier /etc/fstab on a les définition des points de montage : un par ligne

colonnes :

- spécifie le périph : /dev/sdb1 [périphérique, mais ce n'est pas "stable"] ou (mieux) l'UUID (identifiant numérique unique du filesystem)

UUID=....

- les autres colonnes : répertoire, le type du fs, les options, la sauvegarde avec dump (0), les tests de cohérence (1 pour la racine et 2 pour les autres)

on trouve l'uuid de notre système de fichier :
avec blkid ou lsblk --fs, copiez la valeur
de l'uid du fs sdb1
ajouter à la fin de /etc/fstab
(sudo nano /etc/fstab)

UUID=.votre valeur, sans guillemets... à la suite
(sur la même ligne, séparé par un ou + d'espace):

```
...          /srv/data    xfs    defaults    0    2
```

enregistrez, on déclenche le montage :

```
sudo mount -a
```

```
lsblk
```

```
sudo mkdir /srv/data/jpierre (votre id)
```

```
sudo chown jpierre /srv/data/jpierre
```

```
touch /srv/data/jpierre/test
```

Pour résumer :

Les espaces de stockage (partition ou des volumes logiques formatés selon un certain système de fichier, ext4, xfs, btrfs, etc.) sont associés ("montés") à des répertoire dans le fichier fstab où il sont identifiés par un nom de périphérique (pas fiable, sauf pour les volumes logiques) ou un UUID (id. num unique)

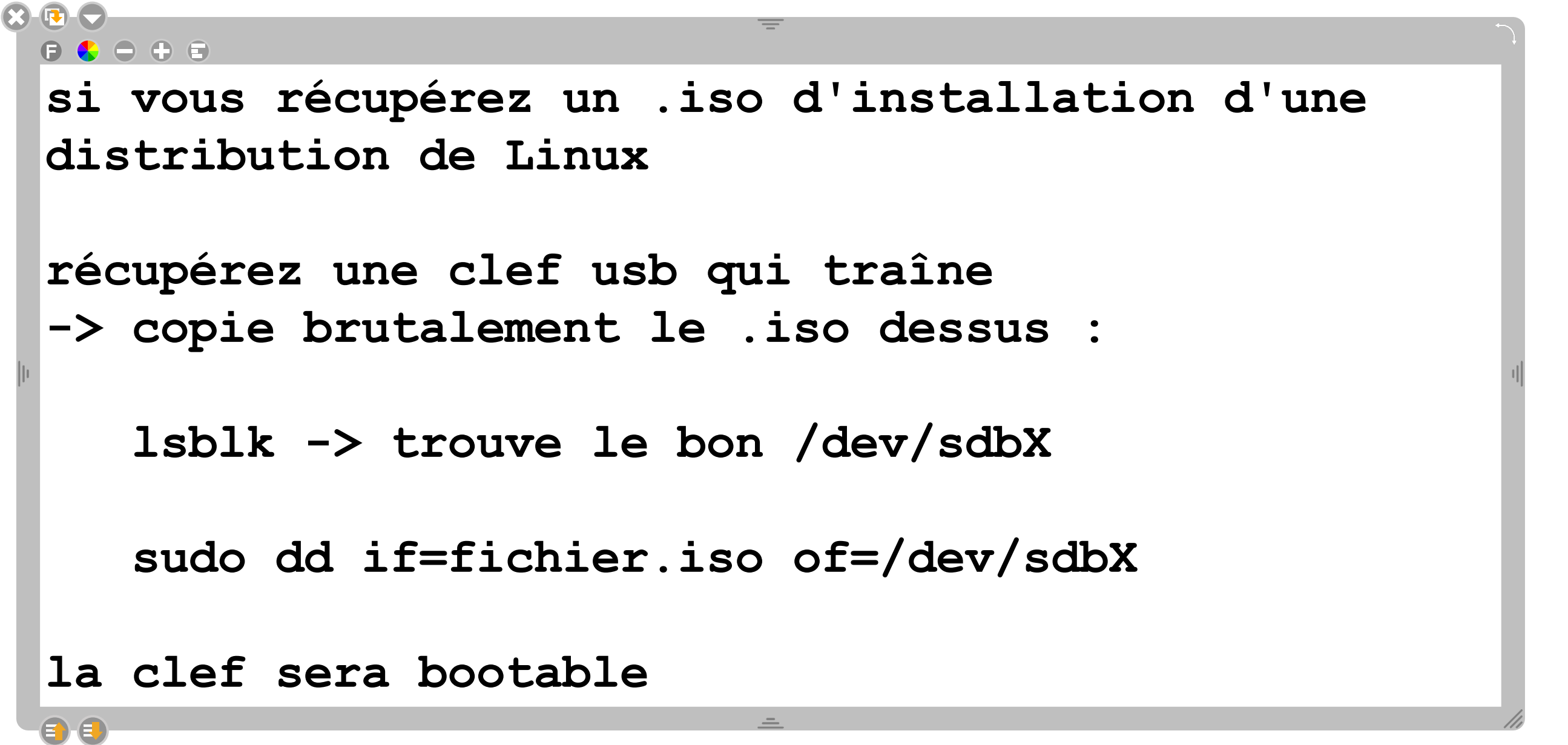
* à terme c'est systemd qui gèrera ça et fstab sera vide

* Périphérique amovible ?

- cdrom : mount /media/cdrom

outils de sauvegarde

- reposent sur des outils d'archivage de fichiers et de répertoire
 - tar (surtout) et cpio
- outils centralisé : backuppc (interface web)



si vous récupérez un .iso d'installation d'une distribution de Linux

récupérez une clef usb qui traîne

-> copie brutalement le .iso dessus :

lsblk -> trouve le bon /dev/sdbX

sudo dd if=fichier.iso of=/dev/sdbX

la clef sera bootable

man tar

action fichier_archive répertoire(s)
type de compression (z:gzip,j:bzip2,xz,...)

c (créer)

tar t (tester) v (verbeux) fichier.tar.XX

x (extraire)

ex : créer une archive compressé en bz2 de notre
répertoire /home

créer une archive :

sudo tar jcf /root/backuphome.tar.bz2 /home

tester l'archive :

sudo tar tvf /root/backuphome.tar.bz2