

Linux – Administration

lundi 2 janvier 2023 14:01

Formateur : Loup FORMENT

Partie 1 : Gestion des processus

gestion des processus

Un processus est un programme exécuté par le processeur

Un processus a toujours un 'parent'
Donc des processus lancent des processus

le papa de tous les processus -> '/sbin/init'

```
[darksasuke@T801 ~]$ ls /sbin/init  
/sbin/init  
[darksasuke@T801 ~]$ file /sbin/init  
/sbin/init: symbolic link to ../../lib/systemd/systemd  
[darksasuke@T801 ~]$ |
```

voir la liste de tous les processus lancés

\$> top/htop

\$> ps -aux



\$> ps -u [user]

\$> pstree

htop/top n'est pas installé de base : il est nécessaire d'installer le paquet pour l'utiliser

```
vel@vel-VirtualBox:~$ sudo apt install htop  
[sudo] Mot de passe de vel :  
Lecture des listes de paquets... Fait  
Construction de l'arbre des dépendances... Fait  
Lecture des informations d'état... Fait  
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :  
    libflashrom1 libftdi1-2  
Veuillez utiliser « sudo apt autoremove » pour les supprimer.  
Paquets suggérés :  
    lm-sensors  
Les NOUVEAUX paquets suivants seront installés :
```

```
0[|||] 9.3% 3[|||] 2.6% 6[|||] 5.9% 9[|||] 6.6%  
1[|||] 3.4% 4[|||] 16.3% 7[|||] 2.0% 10[|||] 6.7%  
2[|||] 8.4% 5[|||] 7.2% 8[|||] 7.4% 11[|||] 7.9%  
Mem[|||||] 4.35G/15.1G Tasks: 54, 482 thr, 199 kthr; 3 running  
Swap[|||||] 0K/0K Load average: 0.42 0.76 0.78  
Uptime: 05:33:20
```

Possibilité de trier avec différents paramètres (% CPU / RAM / nom de programme etc...) ressemble fortement

Commande "PS" :

 que fait la commande ps sous linux ?

 La commande `ps` (process status) sous Linux permet de lister les processus en cours d'exécution sur le système. Elle affiche un certain nombre d'informations sur chaque processus, telles que l'ID du processus, le nom du processus, l'utilisateur qui a lancé le processus, etc. Vous pouvez utiliser différents arguments avec la commande `ps` pour afficher des informations supplémentaires ou filtrer les processus affichés. Par exemple, l'option `--aux` affiche tous les processus, y compris les processus de l'utilisateur et les processus système, avec des informations détaillées.

Exemple d'utilisation :

- **ps -aux** : affiche tous les process en cours

```
● vel@vel-VirtualBox:~$ ps -aux
USER      PID %CPU %MEM   VSZ   RSS TTY      STAT START   TIME COMMAND
root        1  0.2  0.3 166748 11760 ?
root        2  0.0  0.0     0   0 ?          S  14:32  0:00 [kthreadd]
root        3  0.0  0.0     0   0 ?          I* 14:32  0:00 [rcu_gp]
root        4  0.0  0.0     0   0 ?          I* 14:32  0:00 [rcu_par_gp]
root        5  0.0  0.0     0   0 ?          I* 14:32  0:00 [netns]
root        6  0.0  0.0     0   0 ?          I   14:32  0:00 [kworker/0:0-events]
root        7  0.0  0.0     0   0 ?          I* 14:32  0:00 [kworker/0:0H-events_highpri]
root        9  0.0  0.0     0   0 ?          I* 14:32  0:00 [kworker/0:1H-kblockd]
root       10  0.0  0.0     0   0 ?          I* 14:32  0:00 [mm_percpu_wq]
root       11  0.0  0.0     0   0 ?          S  14:32  0:00 [rcu_tasks_rude_]
root       12  0.0  0.0     0   0 ?          S  14:32  0:00 [rcu_tasks_trace]
root       13  0.0  0.0     0   0 ?          S  14:32  0:00 [ksoftirqd/0]
root       14  0.0  0.0     0   0 ?          I   14:32  0:00 [rcu_sched]
root       15  0.0  0.0     0   0 ?          S  14:32  0:00 [migration/0]
root       16  0.0  0.0     0   0 ?          S  14:32  0:00 [idle_inject/0]
root       18  0.0  0.0     0   0 ?          S  14:32  0:00 [cpuhp/0]
root       19  0.0  0.0     0   0 ?          S  14:32  0:00 [cpuhp/1]
root       20  0.0  0.0     0   0 ?          S  14:32  0:00 [idle_inject/1]
root       21  0.0  0.0     0   0 ?          S  14:32  0:00 [migration/1]
root       22  0.0  0.0     0   0 ?          S  14:32  0:00 [ksoftirqd/1]
root       24  0.0  0.0     0   0 ?          I* 14:32  0:00 [kworker/1:0H-events_highpri]
root       25  0.0  0.0     0   0 ?          S  14:32  0:00 [cpuhp/2]
root       26  0.0  0.0     0   0 ?          S  14:32  0:00 [idle_inject/2]
root       27  0.0  0.0     0   0 ?          S  14:32  0:00 [migration/2]
root       28  0.0  0.0     0   0 ?          S  14:32  0:00 [ksoftirqd/2]
root       30  0.0  0.0     0   0 ?          I* 14:32  0:00 [kworker/2:0H-events_highpri]
root       31  0.0  0.0     0   0 ?          S  14:32  0:00 [cpuhp/3]
root       32  0.0  0.0     0   0 ?          S  14:32  0:00 [idle_inject/3]
```

- **ps -au** : affiche les process en cours pour l'utilisateur qui a tapé la commande

```
● vel@vel-VirtualBox:~$ ps -au
USER      PID %CPU %MEM    VSZ RSS TTY      STAT START   TIME COMMAND
vel        2107  0.0  0.1 19800 5388 pts/0    Ss 14:33  0:00 /usr/bin/bash --init-file /home/vel/.vscode-server/bin/e8a3071ea4344d9d48ef8a4df2c097372b0c51
vel        10786  0.0  0.2 171228 6512 tty2    Ss+ 14:37  0:00 /usr/libexec/gdm-wayland-session env GNOME_SHELL_SESSION_MODE=ubuntu /usr/bin/gnome-session -
vel        10789  0.0  0.5 231868 15644 tty2    S1+ 14:37  0:00 /usr/libexec/gnome-session-binary --session=ubuntu
vel        12489  0.0  0.1 21416 3556 pts/0    R+ 14:46  0:00 ps -au
```

- **Pstree** : affiche les processus parent et enfant

```
● vel@vel-VirtualBox:~$ pstree
systemd—ModemManager—2*[{ModemManager}]
           |—NetworkManager—2*[{NetworkManager}]
           |—accounts-daemon—2*[{accounts-daemon}]
           |—acpid
           |—avahi-daemon—avahi-daemon
           |—colord—2*[{colord}]
           |—cron
           |—cups-browsed—2*[{cups-browsed}]
           |—cupsd
           |—dbus-daemon
           |—fwupd—4*[{fwupd}]
           |—gdm3—gdm-session-wor—gdm-wayland-ses—gnome-session-b—2*[{gnome-session-b}]
           |           |—2*[{gdm-wayland-ses}]
           |           |—2*[{gdm-session-wor}]
           |           |—2*[{gdm3}]
           |—gnome-keyring-d—3*[{gnome-keyring-d}]
           |—irqbalance—{irqbalance}
           |—2*[{kerneloops}]
           |—networkd-dispat
           |—packagekitd—2*[{packagekitd}]
           |—polkitd—2*[{polkitd}]
           |—power-profiles—2*[{power-profiles-}]
           |—rsyslogd—3*[{rsyslogd}]
           |—rtkit-daemon—2*[{rtkit-daemon}]
           |—snapd—14*[{snapd}]
           |—sshd—sshd—sshd—bash—sh—node—node—bash—pstree
           |           |—sh—cpuUsage.sh—sleep
           |           |—node—7*[{node}]
           |           |—node—python3—{python3}
           |           |—node—11*[{node}]
           |           |—node—12*[{node}]
           |           |—sleep
           |—switcheroo-cont—2*[{switcheroo-cont}]
```

tty :

Quel est la commande "tty" sous linux ?

La commande `tty` (teletype) sous Linux permet de déterminer le terminal (ou console) auquel l'utilisateur est connecté. Lorsque vous ouvrez un terminal ou une console sous Linux, vous êtes en mesure d'interagir avec le système en utilisant une interface en mode texte. Le terminal est un programme qui exécute des commandes en mode texte et affiche les résultats sur l'écran.

La commande `tty` permet d'afficher le nom du terminal auquel vous êtes connecté. Par exemple, si vous êtes connecté au terminal principal (également appelé console ou `/dev/tty1`), la commande `tty` affichera `/dev/tty1`. Si vous êtes connecté à un terminal virtuel (par exemple, `/dev/tty2`), la commande `tty` affichera `/dev/tty2`. Vous pouvez utiliser la commande `tty` pour vérifier sur quel terminal vous êtes connecté ou pour vous assurer que vous êtes bien connecté à un terminal lorsque vous travaillez en mode texte.

```
● vel@vel-VirtualBox:~$ tty
/dev/pts/0
```

Commande **kill** : permet d'arrêter un processus

commande kill -9



La commande `kill` sous Linux permet de terminer un processus en cours d'exécution en envoyant un signal au processus. Le numéro du signal à envoyer est spécifié en tant qu'argument avec la commande `kill`. Le signal le plus communément utilisé est le signal d'interruption (SIGINT), qui peut être envoyé en utilisant la combinaison de touches Ctrl + C ou en utilisant la commande `kill` avec l'argument `-2`.

Le signal `-9` (SIGKILL) est un signal "brutal" qui force le processus à se terminer immédiatement, sans lui laisser le temps de se terminer proprement. Ce signal est généralement utilisé lorsqu'un processus ne répond plus et ne peut pas être interrompu de manière normale. Cependant, il convient de noter que le signal `SIGKILL` ne peut pas être ignoré par le processus et que, par conséquent, il peut entraîner des pertes de données ou des erreurs si le processus est en train de traiter des données importantes.

Pour utiliser la commande `kill`, vous devez connaître l'ID du processus que vous souhaitez terminer. Vous pouvez obtenir l'ID du processus en utilisant la commande `ps`. Par exemple, pour envoyer le signal `SIGKILL` au processus avec l'ID 12345, vous pouvez utiliser la commande suivante :

Copy code

```
kill -9 12345
```

Commande **Ctrl + c** : Permet de stopper un processus



commande ctrl + c sous linux



La combinaison de touches Ctrl + C sous Linux est utilisée pour interrompre (ou "tuer") un processus en cours d'exécution. Lorsque vous appuyez sur Ctrl + C, un signal d'interruption (SIGINT) est envoyé au processus actif. Le processus peut choisir de réagir à ce signal en se terminant proprement ou en ignorant le signal et en continuant à s'exécuter.

Vous pouvez utiliser Ctrl + C pour interrompre n'importe quel processus en cours d'exécution dans le terminal, qu'il s'agisse d'un programme que vous avez lancé vous-même ou d'un programme lancé par le système. Par exemple, si vous avez lancé un programme qui s'exécute en boucle et que vous souhaitez l'arrêter, vous pouvez appuyer sur Ctrl + C pour l'interrompre. Si vous utilisez Ctrl + C avec un processus système, il se peut que le processus ne soit pas interrompu et continue à s'exécuter. Dans ce cas, vous devrez peut-être utiliser d'autres méthodes pour arrêter le processus.

PROCESSUS EN ARRIERE PLAN
ils sont appelés jobs
(plus utilisés dans des scripts)

lancer une commande en arriere plan

\$> [CMD] &

voir la liste des commandes lancées en arriere plan:
\$> jobs

les jobs ont une id
on peut les kill en utilisant un '%':

\$> kill %[id]

tuer le dernier job lancé

```
$> kill %+
```

envoyer un programme en arrière plan

```
ctrl+z
```

renvoyer un programme au premier plan

```
$> fg %[id]
```

Processus en arrière-plan :

Commande **Sleep** :

CH commande sleep &

 La commande `sleep` sous Linux est utilisée pour mettre en pause l'exécution d'un script ou d'une commande pendant un certain nombre de secondes. Elle est souvent utilisée dans les scripts pour faire une pause entre chaque étape du script, afin de donner le temps au système de se mettre à jour ou de réaliser d'autres tâches.

Lorsque vous ajoutez un `&` à la fin de la commande `sleep`, vous indiquez au système de lancer la commande en arrière-plan. Cela signifie que la commande s'exécutera en parallèle du reste du script ou de la commande, au lieu de bloquer l'exécution jusqu'à ce que la commande `sleep` se termine.

Voici un exemple d'utilisation de la commande `sleep` avec l'option `&` :

```
sleep 30 &
```

Cette commande mettra en pause l'exécution pendant 30 secondes et lancera la commande en arrière-plan. Le script ou la commande continuera à s'exécuter pendant que la commande `sleep` s'exécute en arrière-plan.

Vous pouvez utiliser la commande `fg` (foreground) pour reprendre l'exécution d'une commande en arrière-plan. Par exemple, si vous avez lancé la commande `sleep 30 &` et que vous souhaitez reprendre l'exécution de la commande en arrière-plan, vous pouvez utiliser la commande `fg` suivie de l'ID de la commande. Par exemple :

```
fg %1
```

Cette commande reprendra l'exécution de la commande en arrière-plan avec l'ID 1.

Vérifier les processus en arrière-plan : commande **jobs**

CH commande jobs linux

 La commande `jobs` sous Linux permet de lister les processus en cours d'exécution en arrière-plan dans le terminal actif. Lorsque vous lancez un processus en arrière-plan en utilisant l'opérateur `&`, le processus est exécuté en parallèle du reste du script ou de la commande, mais il reste attaché au terminal actif. Vous pouvez utiliser la commande `jobs` pour afficher la liste de ces processus en arrière-plan et pour reprendre l'exécution de ces processus en utilisant la commande `fg` (foreground).

Voici un exemple d'utilisation de la commande `jobs` :

```
< sleep 30 &
```

```
[1] 1234
$ sleep 20 &
[2] 1235
$ jobs
[1]-  Running                  sleep 30 &
[2]+  Running                  sleep 20 &
```

Dans cet exemple, nous avons lancé deux commandes `sleep` en arrière-plan en utilisant l'opérateur `&`. La commande `jobs` affiche la liste de ces processus en arrière-plan, avec leur ID et leur statut. Le `+` signifie que le processus est le processus en arrière-plan actif et le `-` signifie que le processus est en arrière-plan, mais qu'il n'est pas actif. Vous pouvez utiliser la commande `fg` suivie de l'ID du processus pour reprendre l'exécution d'un processus en arrière-plan. Par exemple, pour reprendre l'exécution du processus avec l'ID 1, vous pouvez utiliser la commande suivante :

```
fg %1
```

```
[darksasuke@T801 ~]$ sleep 200 &
[1] 15875
[darksasuke@T801 ~]$ jobs
[1]+  En cours d'exécution      sleep 200 &
[darksasuke@T801 ~]$
```

Envoyer un programme en arrière-plan : CTRL-Z

Envoyer un programme en 1er plan : fg (foreground)

```
[darksasuke@T801 ~]$ jobs
[2]+  Stoppé                      sleep 300
[darksasuke@T801 ~]$ fg %+
sleep 300
```

Récupérer l'id d'un processus en particulier :

```
[darksasuke@T801 ~]$ ps -u darksasuke | grep "sleep" | grep -oE "
```

Commande pkill :

pkill -9

La commande `pkill` (process kill) sous Linux permet de terminer un ou plusieurs processus en cours d'exécution en utilisant leur nom ou leur ID. Elle est similaire à la commande `kill`, mais elle permet de terminer plusieurs processus en même temps et de filtrer les processus à terminer en utilisant des expressions régulières.

Lorsque vous utilisez `pkill` avec l'argument `-9`, vous indiquez au système d'envoyer le signal `SIGKILL` (numéro 9) aux processus ciblés. Le signal `SIGKILL` est un signal "brutal" qui force le processus à se terminer immédiatement, sans lui laisser le temps de se terminer proprement. Cependant, il convient de noter que le signal `SIGKILL` ne peut pas être ignoré par le processus et que, par conséquent, il peut entraîner des pertes de données ou des erreurs si le processus est en train de traiter des données importantes.

Voici un exemple d'utilisation de la commande `pkill` avec l'option `-9` :

```
pkill -9 firefox
```

Cette commande terminera tous les processus en cours d'exécution appelés "firefox" en envoyant le signal `SIGKILL` à chacun d'entre eux. Si aucun processus appelé "firefox" n'est en cours d'exécution, la commande ne fera rien. Vous pouvez utiliser des expressions régulières pour cibler des processus avec plus de précision. Par exemple, pour terminer tous les processus qui commencent par "firefox" et qui se terminent par un chiffre, vous pouvez utiliser la commande suivante :

```
pkill -9 "firefox[0-9]"
```

 Copy code

Cette commande terminera tous les processus qui correspondent à l'expression régulière "firefox[0-9]", c'est-à-dire tous les processus qui commencent par "firefox" et qui se terminent par un chiffre.

Service :

SERVICES

Les services (ou daemon) sont des processus lancés et orchestrés par systemd

Système:

- permet de lancer des scripts au démarrage du système, et en parallèles
- Organiser un journal de logs

Ce sont des processus lancés en arrière-plan, souvent ils gèrent des gros programmes nécessaires au fonctionnement du système

Commande systemctl :

Voir la liste des services:

```
$> systemctl
```

Voir des infos sur UN service

```
$> systemctl status [service]
```

démarrer un service

```
$> systemctl start [service]
```

stopper un service

```
$> systemctl stop [service]
```

(restart -> stop+start)

lancer un service au démarrage du système

```
$> systemctl enable [service]
```



que fait la commande commande systemctl



La commande `systemctl` sous Linux est utilisée pour gérer les services système et les unités système. Elle permet de démarrer, d'arrêter, de redémarrer, de recharger la configuration et de vérifier l'état des services et des unités système.

Les services système sont des programmes qui s'exécutent en arrière-plan et qui fournissent des fonctionnalités ou des services au système. Les unités système sont des fichiers de configuration qui décrivent les services et les autres ressources du système, tels que les sockets, les montages de fichiers, etc.

Voici quelques exemples d'utilisation de la commande `systemctl` :

- Pour démarrer un service, vous pouvez utiliser la commande suivante :

 Copy code

```
systemctl start nginx
```

Cette commande démarrera le service appelé "nginx".

- Pour arrêter un service, vous pouvez utiliser la commande suivante :

 Copy code

```
systemctl stop nginx
```

Cette commande arrêtera le service appelé "nginx".

- Pour redémarrer un service, vous pouvez utiliser la commande suivante :

 Copy code

```
systemctl restart nginx
```

Cette commande redémarrera le service appelé "nginx".

- Pour recharger la configuration d'un service, vous pouvez utiliser la commande suivante :

 Copy code

```
systemctl reload nginx
```

Cette commande rechargera la configuration du service appelé "nginx" sans arrêter le service.

- Pour vérifier l'état d'un service, vous pouvez utiliser la commande suivante :

 Copy code

```
systemctl status nginx
```

Cette commande affichera l'état du service appelé "nginx", y compris s'il est en cours d'exécution, arrêté, etc.

```
[darksasuke@T801 ~]$ systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
  Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; preset: disabled)
  Active: inactive (dead)
[darksasuke@T801 ~]$ systemctl start nginx
===== AUTHENTICATING FOR org.freedesktop.systemd.manager =====
Authentification requise pour démarrer « nginx.service ».
Authenticating as: darksasuke
Password:
===== AUTHENTICATION COMPLETE =====
[darksasuke@T801 ~]$ sudo systemctl start nginx
[darksasuke@T801 ~]$
```

lancer un service au demarrage du systeme

\$> systemctl enable [service]

enlever un service de la séquence de démarrage

\$> systemctl disable [service]

Voir les journaux du system (systemd)

\$> journalctl

```
janv. 02 15:32:40 vel-VirtualBox anacron[12671]: Anacron 2.7 started on 2023-01-02
janv. 02 15:32:40 vel-VirtualBox anacron[12671]: Normal exit (0 jobs run)
janv. 02 15:32:40 vel-VirtualBox systemd[1]: anacron.service: Deactivated successfully.
janv. 02 15:57:53 vel-VirtualBox sudo[12767]:      vel : TTY pts/0 ; PWD=/home/vel ; USER=root ; COMMAND=/usr/bin/apt install nginx -y
janv. 02 15:57:53 vel-VirtualBox sudo[12767]: pam_unix(sudo:session): session opened for user root(uid=0) by (uid=1000)
janv. 02 15:57:56 vel-VirtualBox systemd[1]: Reloading.
janv. 02 15:57:56 vel-VirtualBox systemd[1]: Reloading.
janv. 02 15:57:56 vel-VirtualBox systemd[1]: Reloading.
janv. 02 15:57:57 vel-VirtualBox systemd[1]: Starting A high performance web server and a reverse proxy server...
janv. 02 15:57:57 vel-VirtualBox systemd[1]: Started A high performance web server and a reverse proxy server.
janv. 02 15:58:00 vel-VirtualBox sudo[12767]: pam_unix(sudo:session): session closed for user root
lines 968-1009/1009 (END)
```

journalctl

La commande `journalctl` sous Linux permet de lire et de gérer les journaux du système. Les journaux du système sont des fichiers qui enregistrent les événements du système, tels que les erreurs, les avertissements, les connexions réseau, etc. `journalctl` est utilisé pour afficher les journaux du système et pour rechercher des entrées de journal en fonction de différents critères.

Voici quelques exemples d'utilisation de la commande `journalctl` :

- Pour afficher les journaux du système, vous pouvez utiliser la commande suivante :

```
journalctl
```
- Cette commande affichera les journaux du système en ordre chronologique, les entrées les plus récentes en premier.
- Pour afficher les journaux d'un service spécifique, vous pouvez utiliser la commande suivante :

```
journalctl -u nginx
```
- Cette commande affichera les journaux du service appelé "nginx".
- Pour rechercher des entrées de journal contenant un mot clé spécifique, vous pouvez utiliser la commande suivante :

```
journalctl -k error
```
- Cette commande affichera toutes les entrées de journal qui contiennent le mot clé "error".

Vous pouvez utiliser d'autres options avec `journalctl` pour afficher des informations supplémentaires, telles que l'heure de chaque entrée de journal, le nom du service associé à chaque entrée, etc. Vous pouvez également utiliser `journalctl` avec d'autres commandes, telles que `grep` et `awk`, pour filtrer et formater les journaux de manière plus avancée.

Voir les journaux du system (systemd)

\$> journalctl -xe

Voir les journaux d'une seule unité

\$> journalctl -u [UNIT]

```
[darksasuke@T801 ~]$ journalctl --disk-usage
Archived and active journals take up 548.4M in the file system
```

Pour nettoyer le journal : argument --vacuum

```
--vacuum-size=, --vacuum-time= and --vacuum-files= may be combined in a single invocation to enforce any combination of a size, a time and a number of files limit the archived journal files. Specifying any of these three parameters as zero is equivalent to not enforcing the specific limit, and is thus redundant.
```

```
These three switches may also be combined with --rotate into one command. If so, active files are rotated first, and the requested vacuuming operation is executed after. The rotation has the effect that all currently active files are archived (a potentially new, empty journal files opened as replacement), and hence the vacuuming operation has the greatest effect as it can take all log data written so far into account.
```

Pour voir tous les services :

```
vel@vel-VirtualBox:~$ ls /lib/systemd/system/
accounts-daemon.service          fwupd-refresh.service      packagekit-offline-update.service    snapd.seeded.service
acpid.path                         fwupd-refresh.timer     packagekit.service                  snapd.service
acpid.service                      fwupd.service           paths.target                     snapd.snap-repair.service
acpid.socket                       gdm3.service          Plymouth-halt.service            snapd.snap-repair.timer
alsa-restore.service               gdm.service           Plymouth-kexec.service          snapd.socket
alsa-state.service                geoclue.service       Plymouth-log.service             snapd.system-shutdown.service
alsa-utils.service                 getty-pre.target     Plymouth-poweroff.service        sockets.target
anacron.service                   getty@.service         Plymouth-quit.service            sockets.target.wants
anacron.timer                      getty-static.service  Plymouth-quit-wait.service      sound.target
apparmor.service                  getty.target          Plymouth-read-write.service     sound.target.wants
apport-autoreport-path            getty.target.wants   Plymouth-reboot.service        speech-dispatcherd.service
apport-autoreport.service          gpu-manager.service  Plymouth-service                spice-vdagentd.service
apport-autoreport.timer           graphical.target     Plymouth-start.service          spice-vdagentd.socket
apport-forward@.service            graphical.target.wants  Plymouth-switch-root-initramfs.service  spice-vdagent.service
apport-forward.socket             grub-common.service  Plymouth-switch-root.service    ssh.service
apt-daily.service                 grub-initrd-fallback.service polkit.service                  ssh@.service
systemd-reboot.service
systemd-remount-fs.service
systemd-resolved.service
systemd-rfkill.service
systemd-rfkill.socket
systemd-suspend.service
systemd-suspend-then.service
systemd-sysctl.service
systemd-sysusers.service
systemd-timedated.service
systemd-timesyncd.service
systemd-time-wait-sync
systemd-tmpfiles-clean
systemd-tmpfiles-clone
systemd-tmpfiles-set
```

Pour créer un service personnalisé :



Voici les étapes générales à suivre pour modifier un service sous Ubuntu :



1. Identifiez le service que vous souhaitez modifier. Vous pouvez utiliser la commande `systemctl list-units` pour afficher la liste de tous les services disponibles sur votre système.
2. Localisez le fichier de configuration du service. La plupart des services ont un fichier de configuration situé dans le répertoire `/etc/systemd/system/`. Vous pouvez utiliser la commande `systemctl show <service>` pour afficher les informations de configuration du service, y compris le nom et l'emplacement du fichier de configuration.
3. Ouvrez le fichier de configuration du service avec un éditeur de texte. Vous pouvez utiliser un éditeur de texte de votre choix, tel que `nano`, `vi`, etc.
4. Modifiez les paramètres du service comme souhaité. Par exemple, vous pouvez modifier les options de démarrage du service, les dépendances

```
[darksasuke@T801 ~]$ sudo cp /usr/lib/systemd/system/nginx.service /usr/lib/systemd/system/service|
```

```
[Unit]
Description=My test service

[Service]
ExecStart=/home/darksasuke/my_script.sh
```

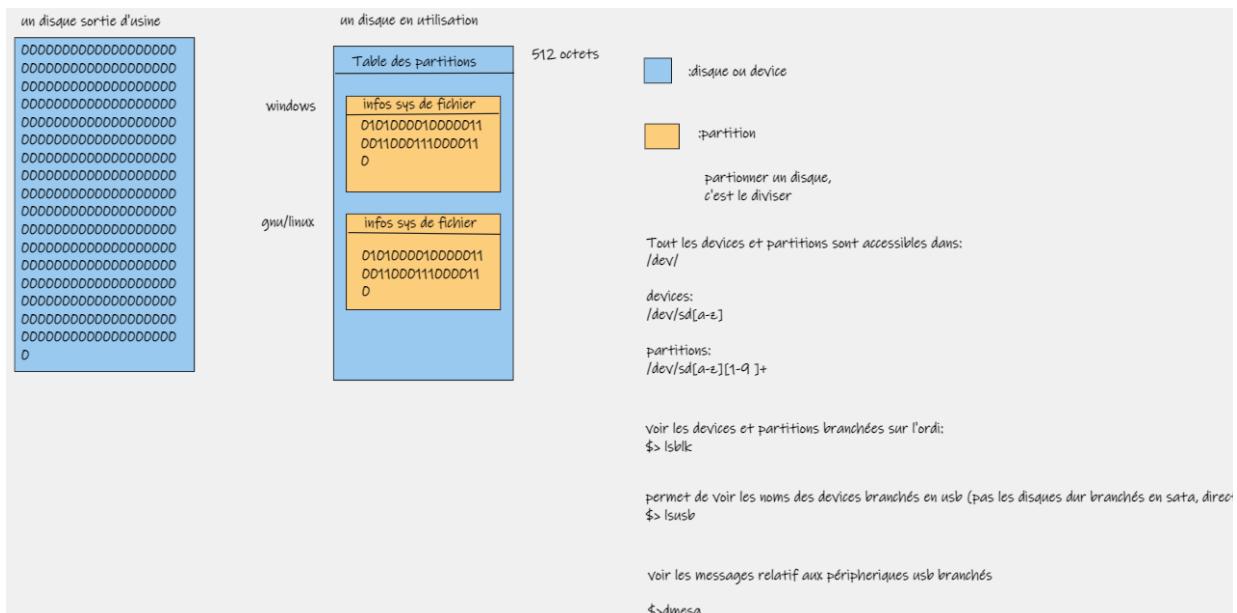
```
#!/bin/bash

while [ a = a ]
do
    echo "bonjour"
    sleep 1
done
```

S'assurer que le script du service soit correctement configuré en exécutable (si besoin -> chmod 755)

Les disques :

Espace de stockage :



Commande **lsblk** :

CH commande lsblk

La commande `lsblk` (list block devices) sous Linux permet de lister les périphériques de stockage de blocs (disques durs, clés USB, etc.) connectés au système. Elle affiche une liste des périphériques de stockage, avec des informations telles que le nom du périphérique, sa taille, son type (partition, système de fichiers, etc.) et son emplacement dans l'arborescence du système de fichiers.

Voici un exemple d'utilisation de la commande `lsblk` :

```
$ lsblk
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda        8:0    0 931.5G  0 disk
└─sda1     8:1    0   512M  0 part /boot/efi
└─sda2     8:2    0    16M  0 part
└─sda3     8:3    0 931.0G  0 part
  └─vg0-root 254:0  0 930.5G  0 lvm  /
sdb        8:16   0   3.7T  0 disk
```

```
● vel@vel-VirtualBox:~$ lsblk
NAME  MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
loop0  7:0    0    4K  1 loop /snap/bare/5
loop1  7:1    0 243.1M  1 loop /snap/code/116
loop2  7:2    0 116.7M  1 loop /snap/core/14399
loop3  7:3    0  63.2M  1 loop /snap/core20/1738
loop4  7:4    0 163.3M  1 loop /snap/firefox/1635
loop5  7:5    0  63.3M  1 loop /snap/core20/1778
loop6  7:6    0 346.3M  1 loop /snap/gnome-3-38-2004/119
loop7  7:7    0  91.7M  1 loop /snap/gtk-common-themes/1535
loop8  7:8    0 238.4M  1 loop /snap/firefox/2211
loop9  7:9    0  45.9M  1 loop /snap/snap-store/582
loop10 7:10   0  49.6M  1 loop /snap/snapd/17883
loop11 7:11   0  284K  1 loop /snap/snapd-desktop-integration/14
loop12 7:12   0  304K  1 loop /snap/snapd-desktop-integration/49
```

```

loop13 7:13 0 400,8M 1 loop /snap/gnome-3-38-2004/112
sda     8:0   0    20G 0 disk
└─sda1  8:1   0    1M 0 part
└─sda2  8:2   0    513M 0 part /boot/efi
└─sda3  8:3   0   19,5G 0 part /var/snap/firefox/common/host-hunspell
/
sdb     8:16  0    15G 0 disk
sr0    11:0   1   3,6G 0 rom
sr1    11:1   1  1024M 0 rom

```

CH La différence entre sda et sdb sur linux



Sur un système Linux, les lettres `sda`, `sdb`, `sdc`, etc. sont utilisées pour identifier les périphériques de stockage de blocs (disques durs, clés USB, etc.). Ces lettres sont utilisées pour distinguer les différents périphériques de stockage qui sont connectés au système.

Par exemple, si vous avez un disque dur principal et une clé USB connectés au système, le disque dur principal sera généralement identifié comme `sda`, tandis que la clé USB sera identifiée comme `sdb`. Si vous connectez un autre périphérique de stockage, comme un disque dur externe, il sera identifié comme `sdc`, et ainsi de suite.

Les lettres `sda`, `sdb`, etc. ne sont pas associées à un emplacement physique précis sur le système, mais plutôt à l'ordre de détection des périphériques de stockage par le système.

Par exemple, si le disque dur principal est détecté en premier et la clé USB en second, le disque dur sera identifié comme `sda` et la clé USB comme `sdb`, même si la clé USB est physiquement connectée à un port USB différent.

Il est important de noter que les lettres `sda`, `sdb`, etc. ne sont pas nécessairement utilisées de manière cohérente d'un système à l'autre. Par exemple, sur un autre système, le disque dur principal

Commande **lsusb** :

CH commande lsusb



La commande `lsusb` (list USB devices) sous Linux permet de lister les périphériques USB connectés au système. Elle affiche une liste des périphériques USB, avec des informations telles que le fabricant, le numéro de produit, le numéro de bus et le numéro d'interface.



Voici un exemple d'utilisation de la commande `lsusb` :

Copy code

```
$ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 004: ID 046d:c077 Logitech, Inc. M105 Optical Mouse
Bus 001 Device 003: ID 0bda:0129 Realtek Semiconductor Corp. RTS5129 Card Reader Controller
Bus 001 Device 002: ID 046d:0825 Logitech, Inc. Webcam C270
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Dans cet exemple, la commande `lsusb` a listé cinq périphériques USB connectés au système, y compris une souris, un lecteur de carte, une webcam et deux hubs USB. Les informations affichées incluent le nom du fabricant, le numéro de produit et le numéro de bus et d'interface du périphérique.

Vous pouvez utiliser d'autres options avec `lsusb` pour afficher des informations supplémentaires sur les périphériques USB, telles que le nom du produit, la version du firmware,

```
● vel@vel-VirtualBox:~$ lsusb
Bus 001 Device 002: ID 80ee:0021 VirtualBox USB Tablet
Bus 001 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
```

Commande **dmesg** :

CH commande dmesg

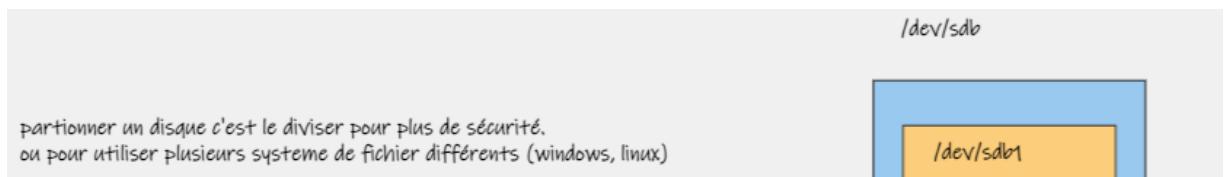
 La commande `dmesg` (display message or driver message) sous Linux permet de lire les messages du noyau du système. Ces messages incluent des informations sur l'état du système, les erreurs, les avertissements et d'autres événements importants qui se produisent pendant le démarrage du système et au cours de son fonctionnement.

Voici un exemple d'utilisation de la commande `dmesg` :

```
$ dmesg
[    0.00000] Initializing cgroup subsys cpuset
[    0.00000] Initializing cgroup subsys cpu
[    0.00000] Initializing cgroup subsys cpacct
[    0.00000] Linux version 4.15.0-112-generic (buildd@lcy01-amd64-024) (gcc
version 7.5.0 (Ubuntu 7.5.0-3ubuntu1~18.04)) #113-Ubuntu SMP Tue Jul 14 16:06:44
UTC 2020
[    0.00000] Command line: BOOT_IMAGE=/boot/vmlinuz-4.15.0-112-generic
root=UUID=f93d1d1c-67d1-4fd9-a2f3-ca05b34c1a17 ro quiet splash vt.handoff=7
[    0.00000] KERNEL supported cpus:
[    0.00000]   Intel GenuineIntel
[    0.00000]   AMD AuthenticAMD
[    0.00000]   Hygon HygonGenuine
[    0.00000]   Centaur CentaurHauls
[    0.00000] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point
registers'
[    0.00000] x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
[    0.00000] x86/fpu:
```

```
● vel@vel-VirtualBox:~$ sudo dmesg
[sudo] Mot de passe de vel :
[    0.00000] Linux version 5.15.0-56-generic (buildd@lcy02-amd64-004) (gcc (Ubuntu 11.3.0-1ubuntu1~22.04) 11.3.0, GNU ld (GNU Binutils
5.15.64)
[    0.00000] Command line: BOOT_IMAGE=/boot/vmlinuz-5.15.0-56-generic root=UUID=57498ffc-b00c-478d-90b5-e9d7ab988b43 ro quiet splash
[    0.00000] KERNEL supported cpus:
[    0.00000]   Intel GenuineIntel
[    0.00000]   AMD AuthenticAMD
[    0.00000]   Hygon HygonGenuine
[    0.00000]   Centaur CentaurHauls
[    0.00000]   zhaoxin Shanghai
[    0.00000] x86/fpu: x87 FPU will use FXSAVE
[    0.00000] signal: max sigframe size: 1440
[    0.00000] BIOS-provided physical RAM map:
[    0.00000] BIOS-e820: [mem 0x0000000000000000-0x000000000009fbff] usable
[    0.00000] BIOS-e820: [mem 0x00000000000000f0-0x000000000000ffff] reserved
[    0.00000] BIOS-e820: [mem 0x0000000000000000-0x000000000000ffff] reserved
[    0.00000] BIOS-e820: [mem 0x0000000000100000-0x000000000000ffff] usable
[    0.00000] BIOS-e820: [mem 0x00000000bfff0000-0x00000000bfffffff] ACPI data
[    0.00000] BIOS-e820: [mem 0x00000000fec00000-0x00000000fec0ffff] reserved
[    0.00000] BIOS-e820: [mem 0x00000000fec00000-0x00000000fec0ffff] reserved
[    0.00000] BIOS-e820: [mem 0x00000000ffff0000-0x00000000ffffffff] reserved
[    0.00000] NX (Execute Disable) protection: active
[    0.00000] SMBIOS 2.5 present.
[    0.00000] DMI: Innotek GmbH VirtualBox/VirtualBox, BIOS VirtualBox 12/01/2006
[    0.00000] Hypervisor detected: KVM
[    0.00000] kvm-clock: Using msrs 4b564d01 and 4b564d00
[    0.00000] kvm-clock: cpu 0, msr 7ee01001, primary cpu clock
[    0.00004] kvm-clock: using sched offset of 6668888675 cycles
[    0.00000] clocksource: kvm-clock: mask: 0xffffffffffffffff max_cycles: 0x1cd42e4dff, max_idle_ns: 881590591483 ns
[    0.00000] tsc: Detected 2304.000 MHz processor
```

Partition :



UNE TABLE DE PARTITIONS
https://fr.wikipedia.org/wiki/GUID_Partition_Table

On va utiliser tout le temps GPT

on a maintenant /dev/sdb1

créer une table de partition et des partitions sur sdb

```
$> fdisk /dev/sdb
$> cfdisk /dev/sdb
$> parted /dev/sdb
```

Table de partitionnement

文 A

[Article](#) [Discussion](#)

[Lire](#) [Modifier](#) [Modifier le code](#)

Une **table de partitionnement** (*partition table*, *partition map*) est stockée sur un disque dur (ou équivalent) et contient les informations nécessaires à la division d'un [disque dur](#) en tranches (*slice*) ou partitions (*partition*).

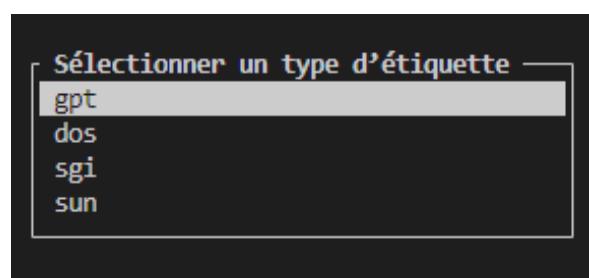
Le terme **table de partitionnement** est souvent associé à celle contenue dans le [Master boot record](#) mais peut aussi désigner plusieurs autres tables de partition comme [GUID Partition Table](#), [Apple partition map](#) ([en](#)).

Créer partition :

Repérer le disque et envoyer la commande :

```
vel@vel-VirtualBox:~$ sudo cfdisk /dev/sdb
```

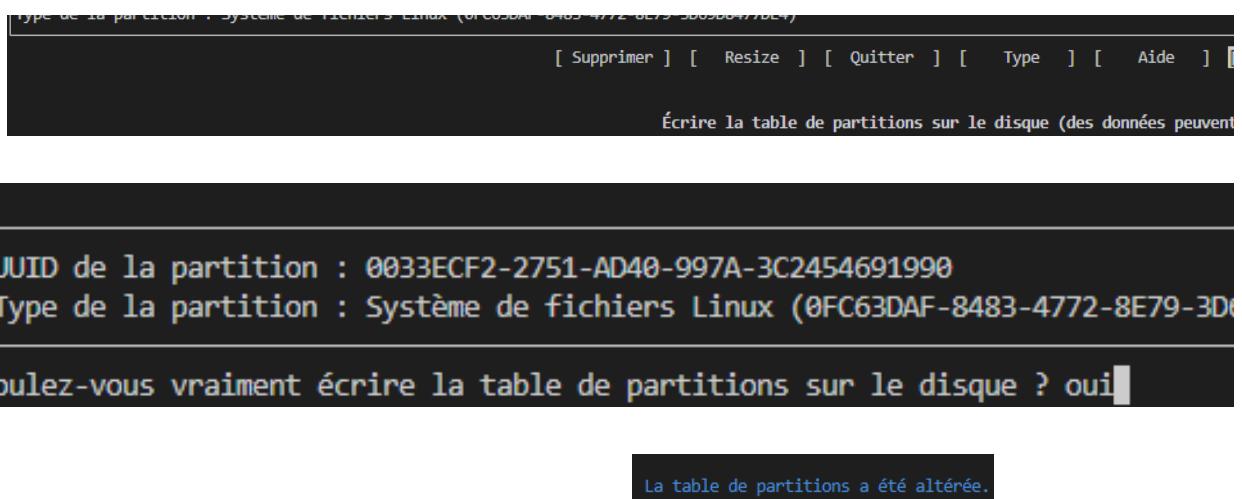
Sélectionner GPT :



| Périphérique | Début | Fin | Secteurs | Taille |
|--------------|----------|----------|----------|--------|
| /dev/sdb1 | 2048 | 10487807 | 10485760 | 5G |
| Espace libre | 10487808 | 31457246 | 29969439 | 18G |

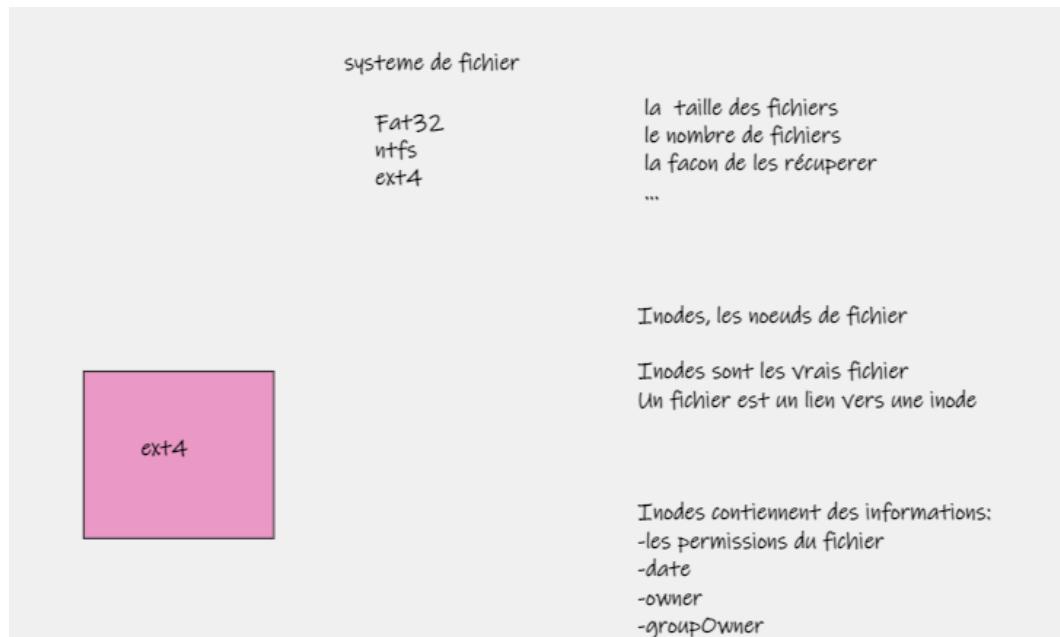
Il faut écrire avant de quitter :

```
UUID de la partition : 0033ECF2-2751-AD40-997A-3C2454691990
Type de la partition : Système de fichiers Linux (0FC63DAE-8483-4772-9E20-2069D8477DE1)
```



On vérifie nos manipulations en tapant la commande **LSBLK**

```
● vel@vel-VirtualBox:~$ lsblk
  NAME   MAJ:MIN RM    SIZE RO TYPE MOUNTPOINTS
  loop0    7:0    0     4K  1 loop /snap/bare/5
  loop1    7:1    0  243,1M  1 loop /snap/code/116
  loop2    7:2    0  116,7M  1 loop /snap/core/14399
  loop3    7:3    0   63,2M  1 loop /snap/core20/1738
  loop4    7:4    0  163,3M  1 loop /snap/firefox/1635
  loop5    7:5    0   63,3M  1 loop /snap/core20/1778
  loop6    7:6    0  346,3M  1 loop /snap/gnome-3-38-2004/119
  loop7    7:7    0  91,7M  1 loop /snap/gtk-common-themes/1535
  loop8    7:8    0  238,4M  1 loop /snap/firefox/2211
  loop9    7:9    0   45,9M  1 loop /snap/snap-store/582
  loop10   7:10   0   49,6M  1 loop /snap/snapd/17883
  loop11   7:11   0   284K  1 loop /snap/snapd-desktop-integration/14
  loop12   7:12   0   304K  1 loop /snap/snapd-desktop-integration/49
  loop13   7:13   0  400,8M  1 loop /snap/gnome-3-38-2004/112
  sda      8:0    0    20G  0 disk
    └─sda1   8:1    0     1M  0 part
    └─sda2   8:2    0   513M  0 part /boot/efi
    └─sda3   8:3    0  19,5G  0 part /var/snap/firefox/common/host-hunspell
        /
  sdb      8:16   0    15G  0 disk
    └─sdb1   8:17   0     5G  0 part
    └─sdb2   8:18   0   200M  0 part
    └─sdb3   8:19   0     8G  0 part
    └─sdb4   8:20   0   1,8G  0 part
  sr0     11:0   1    3,6G  0 rom
  sr1     11:1   1  1024M  0 rom
```



 ntfs

 NTFS (New Technology File System) est un système de fichiers utilisé principalement sous Windows. Il a été développé par Microsoft et est devenu le système de fichiers par défaut pour les versions récentes de Windows.

NTFS offre de nombreuses fonctionnalités avancées par rapport à d'autres systèmes de fichiers, telles que la possibilité de stocker des fichiers de taille supérieure à 4 Go, la gestion des permissions d'accès aux fichiers et la prise en charge de la compression de fichiers. Il est également connu pour être un système de fichiers stable et fiable.

Bien que NTFS soit principalement utilisé sous Windows, il peut également être utilisé sur d'autres systèmes d'exploitation, y compris Linux. Pour utiliser NTFS sous Linux, vous pouvez utiliser un pilote de lecteur NTFS, tel que ntfs-3g, qui permet de lire et d'écrire sur des partitions NTFS depuis Linux. Cependant, il peut y avoir certains problèmes de compatibilité avec les fichiers NTFS sous Linux, il est donc recommandé de sauvegarder vos données importantes avant de les utiliser.

 EXT4 (Fourth Extended File System) est un système de fichiers utilisé principalement sous Linux. Il a été développé en remplacement du système de fichiers EXT3 et offre de nombreuses améliorations par rapport à celui-ci, notamment une meilleure gestion de l'espace disque et des performances améliorées.

EXT4 permet de stocker des fichiers de taille allant jusqu'à 16 To et prend en charge des noms de fichiers de 255 caractères. Il offre également une meilleure gestion des erreurs et de la tolérance aux pannes, ce qui le rend plus fiable que d'autres systèmes de fichiers.

EXT4 est le système de fichiers par défaut pour de nombreuses distributions Linux, y compris Ubuntu et CentOS. Vous pouvez utiliser la commande `mkfs.ext4` pour formater une partition en EXT4 ou la commande `mount` pour monter une partition EXT4 dans l'arborescence du système de fichiers.

 Un inode (index node) est un objet utilisé par le système de fichiers EXT4 (et d'autres systèmes de fichiers) pour stocker des informations sur les fichiers et les répertoires. Chaque fichier et répertoire sur le système a son propre inode, qui contient des informations telles que la taille du fichier, la date de modification, les permissions d'accès, etc.

Les inodes sont stockés dans une table sur le disque, et chaque inode a un numéro unique qui lui est associé. Lorsque vous accédez à un fichier ou à un répertoire, le système de fichiers utilise le numéro d'inode pour trouver les informations sur l'objet dans la table d'inodes.

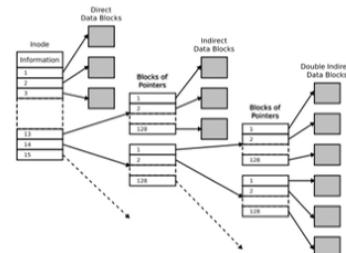
Le nombre d'inodes disponibles sur un système de fichiers EXT4 est limité par la taille de la table d'inodes. Lorsque vous créez un système de fichiers EXT4, vous pouvez spécifier le nombre d'inodes à créer. Si vous prévoyez de stocker de nombreux petits fichiers sur ce système, vous devriez augmenter le nombre d'inodes afin de ne pas manquer d'espace dans la table d'inodes.

Vous pouvez utiliser la commande `tune2fs` pour afficher et modifier les paramètres système de fichiers EXT4, y compris le nombre d'inodes. Vous pouvez également utiliser la commande `df -i` pour afficher le nombre d'inodes utilisés et disponibles sur un système de fichiers.

Inodes [edit]

Every file or directory is represented by an inode. The term "inode" comes from "index node" (over time, it became i-node and then inode).^[1] It contains information about the size, permission, ownership, and location on disk of the file or directory.

Example of ext2 inode structure:



Quote from the Linux kernel documentation for ext2:

/dev/sdb

créer des systèmes de fichiers

\$> mkfs

Mettre un système de fichier ext4 sur une partition sdb1

/dev/sdb1

file system
ext4

/

\$> mkfs.ext4 /dev/sdb1

informations sur le système de fichier

\$> dumpe2fs /dev/sdb1

informations sur la taille et l'espace utilisé sur le système de fichier monté

\$> df -h

informations sur le nombre d'inodes utilisées/disponibles sur un système de fichier monté

\$> df -i

ajuster/modifier la taille d'un système de fichier:

\$> resize2fs [FS]

réparer un système de fichier:

\$> e2fsck [FS]

Commande `mkfs`:

 La commande `mkfs` (make file system) sous Linux permet de créer un nouveau système de fichiers sur un périphérique de stockage, tel qu'un disque dur ou une clé USB. Elle permet de formater le périphérique et d'y créer une structure de fichiers de base, telle que l'arborescence du système de fichiers.

Voici un exemple d'utilisation de la commande `mkfs` pour créer un système de fichiers EXT4 sur une partition `/dev/sda1`:

```
$ mkfs.ext4 /dev/sda1
```

Copy code

La commande `mkfs` accepte de nombreuses options pour personnaliser la création du système de fichiers. Par exemple, vous pouvez utiliser l'option `-L` pour donner un nom à la partition, l'option `-m` pour réservé une certaine quantité d'espace sur la partition pour les fichiers temporaires, etc.

Il est important de noter que l'utilisation de la commande `mkfs` effacera toutes les données précédemment présentes sur le périphérique de stockage. Assurez-vous de sauvegarder toutes les données importantes avant de formater le périphérique.

Pour formater la partition :

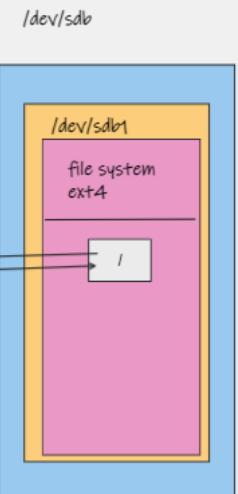
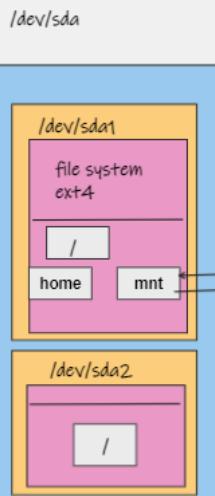
```
STO      11.0   1  5.0D  0  MDT /cdrom  
ubuntu@ubuntu:~$ mkfs.ext4 /dev/sdb1
```

Résultat de la commande `sudo mkfs.ext4 /dev/sdb1` :

```
● vel@vel-VirtualBox:~$ sudo mkfs.ext4 /dev/sdb1  
[sudo] Mot de passe de vel :  
mke2fs 1.46.5 (30-Dec-2021)  
En train de créer un système de fichiers avec 1310720 4k blocs et 327680 i-noeuds.  
UUID de système de fichiers=460d3def-7bff-4ec0-8c16-960c6d442ed0  
Superblocs de secours stockés sur les blocs :  
32768, 98304, 163840, 229376, 294912, 819200, 884736  
  
Allocation des tables de groupe : complété  
Écriture des tables d'i-noeuds : complété  
Création du journal (16384 blocs) : complété  
Écriture des superblocs et de l'information de comptabilité du système de  
fichiers : complété
```

accéder à un système de fichier

Il faut le monter
"connecter les deux arbres de fichiers"



Pour monter la partition : commande `mount`



La commande `mount` sous Linux permet de monter un périphérique de stockage (disque dur, clé USB, etc.) dans l'arborescence du système de fichiers. Cela signifie que le périphérique de stockage devient accessible depuis le système et que vous pouvez accéder aux fichiers qu'il contient.

Voici un exemple d'utilisation de la commande `mount` pour monter une partition EXT4 située sur `/dev/sda1` dans le répertoire `/mnt` :

Copy code

```
$ mount -t ext4 /dev/sda1 /mnt
```

La commande `mount` accepte de nombreuses options pour personnaliser le montage du périphérique. Par exemple, vous pouvez utiliser l'option `-o` pour spécifier des options de montage, comme `ro` pour monter le périphérique en lecture seule ou `noatime` pour ne pas mettre à jour la date de dernier accès aux fichiers.

Pour démonter un périphérique, vous pouvez utiliser la commande `umount`. Par exemple :

```
$ umount /mnt
```

Il est important de noter que vous ne pouvez généralement pas démonter un périphérique si des fichiers ouverts sur celui-ci sont en cours d'utilisation. Assurez-vous de fermer tous les fichiers ouverts sur le périphérique.

Il faut le monter dans le répertoire /mnt/ -> répertoire existe spécialement pour ça

```
vel@vel-VirtualBox:~$ sudo mount /dev/sdb1 /mnt/
```

En vérifiant avec lsblk -> on constate que sdb1 est sur /mnt/

```
sdb      8:16    0    15G  0 disk '
└─sdb1   8:17    0     5G  0 part /mnt
  └─sdb2   8:18    0   200M  0 part
  └─sdb3   8:19    0     8G  0 part
  └─sdb4   8:20    0   1,8G  0 part
sr0     11:0    1   3,6G  0 rom
sr1     11:1    1  1024M 0 rom
```

```
ubuntu@ubuntu:~$ touch /mnt/file
touch: cannot touch '/mnt/file': Permission denied
ubuntu@ubuntu:~$ sudo !!
sudo touch /mnt/file
ubuntu@ubuntu:~$ cp Movies/piratesofcaraibes /mnt/
```

Pour démonter une partition :

```
ubuntu@ubuntu:~$ sudo umount /dev/sdb1
ubuntu@ubuntu:~$ sudo umount /mnt/
```

```
ubuntu@ubuntu:~$ lsblk
NAME  MAJ:MIN RM    SIZE RO TYPE MOUNTPOINT
loop0   7:0    0  2.2G  1 loop /rofs
loop1   7:1    0    4K  1 loop /snap/bare/5
loop2   7:2    0   62M  1 loop /snap/core20/1611
loop3   7:3    0 346.3M  1 loop /snap/gnome-3-38-2004/115
loop4   7:4    0  91.7M  1 loop /snap/gtk-common-themes/1535
loop5   7:5    0  54.2M  1 loop /snap/snap-store/558
loop6   7:6    0   47M  1 loop /snap/snapd/16292
sda     8:0    0   25G  0 disk
  └─sdb    8:16   0   1.4G  0 disk
    ├─sdb1   8:17   0     1G  0 part
    ├─sdb2   8:18   0   200M  0 part
    ├─sdb3   8:19   0   100M  0 part
    └─sdb4   8:20   0 147.2M  0 part
sr0     11:0   1   3.6G  0 rom  /cdrom
```

```

ubuntu@ubuntu:~$ ls /mnt/
ubuntu@ubuntu:~$ 

● vel@vel-VirtualBox:~$ sudo umount /dev/sdb1
● vel@vel-VirtualBox:~$ lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
loop0    7:0    0   4K  1 loop /snap/bare/5
loop1    7:1    0 243,1M 1 loop /snap/code/116
loop2    7:2    0 116,7M 1 loop /snap/core/14399
loop3    7:3    0 63,2M 1 loop /snap/core20/1738
loop4    7:4    0 163,3M 1 loop /snap/firefox/1635
loop5    7:5    0 63,3M 1 loop /snap/core20/1778
loop6    7:6    0 346,3M 1 loop /snap/gnome-3-38-2004/119
loop7    7:7    0 91,7M 1 loop /snap/gtk-common-themes/1535
loop8    7:8    0 238,4M 1 loop /snap/firefox/2211
loop9    7:9    0 45,9M 1 loop /snap/snap-store/582
loop10   7:10   0 49,6M 1 loop /snap/snapd/17883
loop11   7:11   0 284K 1 loop /snap/snapd-desktop-integration/14
loop12   7:12   0 304K 1 loop /snap/snapd-desktop-integration/49
loop13   7:13   0 400,8M 1 loop /snap/gnome-3-38-2004/112
sda      8:0    0   20G  0 disk
└─sda1   8:1    0     1M 0 part
└─sda2   8:2    0   513M 0 part /boot/efi
└─sda3   8:3    0 19,5G 0 part /var/snap/firefox/common/host-hunspell
/
sdb      8:16   0   15G  0 disk
└─sdb1   8:17   0     5G 0 part
└─sdb2   8:18   0   200M 0 part
└─sdb3   8:19   0     8G 0 part
└─sdb4   8:20   0   1,8G 0 part
sr0     11:0   1   3,6G 0 rom
sr1     11:1   1 1024M 0 rom

```

Si on démonte une partition, elle emporte les fichiers placé à l'intérieur avec elle, on peut donc s'amuser à démonter/remonter les partition comme bon nous semble pour les placer à différents endroit sur le système

Pour avoir des informations sur les partitions montée : **df -h**

```

● vel@vel-VirtualBox:~$ df -h
          Sys. de fichiers Taille Utilisé Dispo Uti% Monté sur
tmpfs            299M    1,5M  297M  1% /run
/dev/sda3        20G     12G  6,2G  66% /
tmpfs            1,5G      0  1,5G  0% /dev/shm
tmpfs            5,0M    4,0K  5,0M  1% /run/lock
/dev/sda2        512M    5,3M 507M  2% /boot/efi
tmpfs            299M     80K 298M  1% /run/user/127
tmpfs            299M    72K 298M  1% /run/user/1000

```

Pour avoir des information sur une partition : commande **dumpe2fs**

La commande **dumpe2fs** sous Linux permet d'afficher les informations sur un système de fichiers EXT2, EXT3 ou EXT4. Elle peut être utilisée pour afficher des informations détaillées sur la structure du système de fichiers, telles que le nombre d'inodes, la taille des blocs, les options de montage, etc.

Voici un exemple d'utilisation de la commande **dumpe2fs** pour afficher les informations sur un système de fichiers EXT4 situé sur **/dev/sda1** :

```
$ dumpe2fs /dev/sda1
```

Copy code

La commande **dumpe2fs** accepte également de nombreuses options pour afficher des informations spécifiques sur le système de fichiers. Par exemple, vous pouvez utiliser l'option **-h** pour afficher un résumé des informations principales, l'option **-b** pour afficher la liste des blocs libres ou l'option **-i** pour afficher la liste des inodes libres.

Vous pouvez utiliser la commande **dumpe2fs** pour obtenir des informations sur l'état et la structure d'un système de fichiers EXT2, EXT3 ou EXT4, ce qui peut être utile pour dépanner.

Il est recommandé de faire des sauvegardes régulières pour éviter de perdre des données ou optimiser les performances du système. Cependant, veillez à ne pas modifier les informations affichées par la commande `dump2fs`, car cela pourrait endommager le système de fichiers.

```
● vel@vel-VirtualBox:~$ sudo dump2fs /dev/sdb1
dump2fs 1.46.5 (30-Dec-2021)
Filesystem volume name: <none>
Last mounted on: <not available>
Filesystem UUID: 460d3def-7bff-4ec0-8c16-960c6d442ed0
Filesystem magic number: 0xEF53
Filesystem revision #: 1 (dynamic)
Filesystem features: has_journal ext_attr resize_inode dir_index
Filesystem flags: signed_directory_hash
Default mount options: user_xattr acl
Filesystem state: clean
Errors behavior: Continue
Filesystem OS type: Linux
Inode count: 327680
Block count: 1310720
Reserved block count: 65536
Overhead clusters: 42072
Free blocks: 1268642
Free inodes: 327669
First block: 0
Block size: 4096
Fragment size: 4096
Group descriptor size: 64
Reserved GDT blocks: 639
Blocks per group: 32768
Fragments per group: 32768
Inodes per group: 8192
Inode blocks per group: 512

```

```
ubuntu@ubuntu:~/noisette$ df -ih
Filesystem      Inodes IUsed IFree IUse% Mounted on
udev            231K   489  230K    1% /dev
tmpfs           248K   859  247K    1% /run
/dev/sr0          0     0     0   100% /cdrom
/dev/loop0       198K  198K    0  100% /rofs
/cow             248K  3.0K  245K    2% /
tmpfs           248K    1  248K    1% /dev/shm
tmpfs           248K     7  247K    1% /run/lock
tmpfs           248K    19  247K    1% /sys/fs/cgroup
tmpfs           248K    42  247K    1% /tmp
/dev/loop1        29     29    0  100% /snap/bare/5
/dev/loop2        12K    12K    0  100% /snap/core20/1611
/dev/loop4        75K    75K    0  100% /snap/gtk-common-themes/1535
/dev/loop3        18K    18K    0  100% /snap/gnome-3-38-2004/115
```

la commande dd

dd
permet de lire un flux d'entrée et de l'écrire sur une sortie
il va copier Bit par Bit

\$> dd if=[file] of=[file]

nettoyer un disque (usine):

\$> dd if=/dev/zero of=/dev/sdb

effacer la table des partitions:

\$> dd if=/dev/zero of=/dev/sdb bs=512 count=1

```
copier des disques ou des partitions  
$> dd if=/dev/sdb of=save.part
```

dd et gzip pour compresser

```
$> dd if=/dev/sdb1 | gzip > save.gzip
```

```
ouvrir le contenu d'un .iso/ creer une clef bootable  
$> dd if=archlinux.iso of=/dev/sdb
```

creer des mots de passe aleatoire

```
$> dd if=/dev/urandom of=motdepasse bs=1000 count=1
```



La commande `dd` sous Linux est un outil en ligne de commande qui permet de copier des fichiers et de convertir les données. Elle est souvent utilisée pour créer des images de disques, cloner des disques durs, etc.

Voici un exemple d'utilisation de la commande `dd` pour créer une image de disque d'un disque dur situé sur `/dev/sda` et l'enregistrer dans un fichier `/tmp/sda.img` :

Copy code

```
$ dd if=/dev/sda of=/tmp/sda.img
```

La commande `dd` accepte de nombreuses options pour personnaliser la copie des données. Par exemple, vous pouvez utiliser l'option `bs=` pour spécifier la taille des blocs de données à lire et à écrire, l'option `count=` pour spécifier le nombre de blocs à copier, etc.

Il est important de noter que la commande `dd` peut être dangereuse si elle est utilisée de manière incorrecte, car elle peut écraser des données importantes. Assurez-vous de vérifier les chemins d'accès de entrée et de sortie avant de l'utiliser et de faire une sauvegarde de vos données importantes.

dev/zero -> Nettoyage usine : Il n'y a que des 0 qui seront écrit dessus

Procédure :

Démonter toute les partitions du disque qu'on veut formater :

```
ubuntu@ubuntu:~$ umount /dev/sdb1  
umount: /home/ubuntu/chocolat: umount failed: Operation not permitted  
ubuntu@ubuntu:~$ sudo !!  
sudo umount /dev/sdb1  
ubuntu@ubuntu:~$ sudo umount /dev/sdb2  
ubuntu@ubuntu:~$ sudo umount /dev/sdb3  
ubuntu@ubuntu:~$ lsblk  
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT  
loop0 7:0 0 2.2G 1 loop /rofs  
loop1 7:1 0 4K 1 loop /snap/bare/5  
loop2 7:2 0 62M 1 loop /snap/core20/1611  
loop3 7:3 0 346.3M 1 loop /snap/gnome-3-38-2004/115  
loop4 7:4 0 91.7M 1 loop /snap/gtk-common-themes/1535  
loop5 7:5 0 54.2M 1 loop /snap/snap-store/558  
loop6 7:6 0 47M 1 loop /snap/snapd/16292  
sda 8:0 0 25G 0 disk  
sdb 8:16 0 1.4G 0 disk  
|---sdb1 8:17 0 1G 0 part  
|---sdb2 8:18 0 200M 0 part  
|---sdb3 8:19 0 100M 0 part  
|---sdb4 8:20 0 147.2M 0 part  
sr0 11:0 1 3.6G 0 rom /cdrom  
ubuntu@ubuntu:~$
```

```
ubuntu@ubuntu:~$ sudo dd if=/dev/zero of=/dev/sdb
```

```
ubuntu@ubuntu:~$ sudo dd if=/dev/zero of=/dev/sdb
dd: writing to '/dev/sdb': No space left on device
3015075+0 records in
3015074+0 records out
1543717888 bytes (1.5 GB, 1.4 GiB) copied, 74.1426 s, 20.8 MB/s
ubuntu@ubuntu:~$ lsblk
NAME  MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
loop0  7:0    0   2.2G 1 loop /rofs
loop1  7:1    0     4K 1 loop /snap/bare/5
loop2  7:2    0   62M 1 loop /snap/core20/1611
loop3  7:3    0 346.3M 1 loop /snap/gnome-3-38-2004/115
loop4  7:4    0  91.7M 1 loop /snap/gtk-common-themes/1535
loop5  7:5    0  54.2M 1 loop /snap/snap-store/558
loop6  7:6    0   47M 1 loop /snap/snapd/16292
sda    8:0    0   25G 0 disk
sdb    8:16   0   1.4G 0 disk
sr0   11:0   1   3.6G 0 rom  /cdrom
ubuntu@ubuntu:~$
```

On voit que notre partition **sdb** est revenu au paramètre d'usine

```
ubuntu@ubuntu:~$ sudo dd if=/dev/zero of=/dev/sdb bs=2048 count=1
1+0 records in
1+0 records out
2048 bytes (2.0 kB, 2.0 KiB) copied, 0.0107891 s, 190 kB/s
ubuntu@ubuntu:~$ lsblk
NAME  MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
loop0  7:0    0   2.2G 1 loop /rofs
loop1  7:1    0     4K 1 loop /snap/bare/5
loop2  7:2    0   62M 1 loop /snap/core20/1611
loop3  7:3    0 346.3M 1 loop /snap/gnome-3-38-2004/115
loop4  7:4    0  91.7M 1 loop /snap/gtk-common-themes/1535
loop5  7:5    0  54.2M 1 loop /snap/snap-store/558
loop6  7:6    0   47M 1 loop /snap/snapd/16292
sda    8:0    0   25G 0 disk
sdb    8:16   0   1.4G 0 disk
sr0   11:0   1   3.6G 0 rom  /cdrom
ubuntu@ubuntu:~$
```

! Ce type de commande ci-dessous va écrire des 0 sans s'arrêter -> peut faire crash la machine sur laquelle on est !

```
ubuntu@ubuntu:~$ dd if=/dev/zero of=file
^C1399683+0 records in
1399683+0 records out
716637696 bytes (717 MB, 683 MiB) copied, 6.7481 s, 106 MB/s
```

Exercice :

```
home > darksasuke >  EDITOR=makefile
1  ##### EXERCICE #####
2  # maKey.sh
3  # un script qui permet la creation d'une cle USB partitionnée.
4
5
6
7  # l'utilisateur tape: $>sudo bash maKey.sh /dev/sde I
8
9  # -on lui demande si il veut nettoyer complètement la clé. (dd)
10
11 # -on lui montre la taille de la clé et on demande combien de partition il veut des
12
13 # -on divise l'espace disponible en part à peu près égales.
14
```

```

15  # -on met un systeme de fichier sur les partitions (mkfs.ext4,mkfs.vfat)
16
17  # -on lui affiche les partitions crées ( lsblk | grep )
18

```

Correction exercice :

```

#!/bin/bash
function help0{
    echo "MAKEY"
    echo "Creation automatique de clés"
    echo " utilisation:"
    echo "--clean   nettoyer la table de partition"
    echo "-nb      nombre de partitions à créer"
    echo "--device=[device]  device à utiliser"
}

for arg in $@ #
do
    if [[ $arg =~ ^--device=(.+)]]
    then
        device=${BASH_REMATCH[1]}
    elif [[ $arg =~ ^--clean ]]
    then
        clean=true
    elif [[ $arg =~ ^-(0-9) ]]
    then
        nb_parts=${BASH_REMATCH[1]}
    else
        help
        exit 1
    fi
done

echo $device

if [ -z $device ]
then
    echo "il faut indiquer un device"
    echo "--device=[device]"
    echo ""
    help
    exit 2
fi

# vérification du device
if [ -z $device ]
then
    echo "il faut indiquer un device"
    echo "--device=[device]"
    echo ""
    help
    exit 2
fi

lsblk $device &> /dev/null

if [ $? -ne 0 ]
then
    echo "impossible de travailler avec: '$device'"
    echo "vérifiez qu'il sagit bien d'un device"
    echo "et qu'il soit bien branché sur la machine"
    exit
fi

# vérification du nombre de partitions

if [ -z $nb_parts ]
then
    echo "Il faut indiquer un nombre de partitions"
    help
    exit
fi

# ON COMMENCE
echo "Travaille sur: $device"
read -p "Voulez-vous continuer? Y " choix
if [ -n "$choix" ]
then

```

```

echo "On annule"
exit
fi

# nettoyer si on a mit --clean

if [ -n $clean ]
then
    dd if=/dev/zero of=$device bs=2048 count=1 > /dev/null
fi

# creer la table de partition
parted --script $device mklabel gpt

# creer les partitions

percent=$((100/$nb_parts))

for i in $(seq 0 $((nb_parts-2)) )
do
    start=$((0+$i*$percent))
    end=$((start+$percent))
    parted --script $device mkpart primary ext4 $start% $end%
    mkfs.ext4 $device$((i+1))
done

[ $nb_parts -eq 1 ] && end=0 && i=0

parted --script $device mkpart primary ext4 $end% 100%
mkfs.ext4 $device$((i+1))

echo ""
echo "[TERMINÉ]"
lsblk $device
#####

```

La commande **dd** est intéressant pour effectuer des sauvegardes :

```

513  sudo dd if=/dev/sdb of=save_table bs=2048 count=1
514  ls save_table
515  lsblk
516  sudo dd if=/dev/zero of=/dev/sdb bs=2048 count=1
517  lsblk
518  sudo dd if=save_table of=/dev/sdb

```

- La ligne 513 permet d'effectuer une sauvegarde de la partition sélectionnée
- On vérifie que le fichier est bien présent avec
- Reset de la partition avec la ligne 516
- On récupère la sauvegarde avec la commande **dd** (ligne 518)

Correction exercice :

Exemple gestion argument en bash :

```

$ makey.sh
1  #!/bin/bash
2
3  for arg in $@ #
4  do
5      if [[ $arg =~ ^--device=(.+) ]]
6      then
7          device=${BASH_REMATCH[1]}
8          if [[ $arg =~ ^--clean ]]
9          then
10         device=true

```

```

11      if [[ $arg =~ ^--([0-9]) ]]
12      then
13          nb_parts=${BASH_REMATCH[1]}
14      fi
15  done

```

Le chiffrement : cryptsetup

cryptsetup est un utilitaire en ligne de commande qui permet de configurer et gérer les disques cryptés sur un système GNU/Linux. Il peut être utilisé pour créer et ouvrir des volumes chiffrés, ainsi que pour changer leurs mots de passe et les paramètres de chiffrement. Il prend en charge plusieurs algorithmes de chiffrement tels que AES, Blowfish, et Serpent, et peut être utilisé avec différents types de supports de stockage tels que des disques durs, des clés USB et des partitions de disque.

Le chiffrement avec cryptsetup

créer un conteneur chiffré

```
$> cryptsetup luksFormat /dev/sdb
```

ouvrir un conteneur chiffré

```
$> cryptsetup open /dev/sdb [nom]
```

les conteneurs chiffrés sont accessibles dans:

```
$> /dev/mapper/[nom]
```

Fermer des conteneurs chiffrés

```
$> cryptsetup close /dev/mapper/[nom]
```

créer un conteneur chiffré avec un mdp aléatoire:

```
$> dd if=/dev/urandom of=motdepasse bs=1000 count=1
$> cryptsetup luksFormat /dev/sdb --key-file motdepasse
$> cryptsetup open /dev/sdb [nom] --key-file motdepasse
```

Faire un backup du header LUKS (pour une restauration plus tard)

```
$> cryptsetup luksHeaderBackup /dev/sdb --header-backup-file ~/backup/file.img
```

Restoration du header (si il y a un problème)

```
cryptsetup -v --header ~/backup/file.img open /dev/sdb [nom]
```

Le chiffrement d'un conteneur va supprimer toutes les données dessus

Ouvrir un conteneur chiffré :

```
[darksasuke@archlinux makey]$ sudo cryptsetup open
/dev/sdb crypted_part
Saisissez la phrase secrète pour /dev/sdb :
[darksasuke@archlinux makey]$ lsblk
NAME      MAJ:MIN RM    SIZE RO TYPE  MOUNTPOINTS
sda        8:0     0 476,9G  0 disk
└─sda1     8:1     0   513M  0 part   /boot
└─sda2     8:2     0   93,1G  0 part
└─sda3     8:3     0   150G  0 part
```

```

└─sda4          8:4      0 149,5G  0 part   /
  └─sdb          8:16     1   3,8G  0 disk
    └─crypted_part

```

Les conteneur chiffré sont stocké dans le dossier **/dev/mapper/**

```
[darksasuke@archlinux makey]$ ls /dev/mapper/
control crypted_part
```

On créer deux partition (une chiffré, l'autre non)

```

sdb          8:16     1   3,8G  0 disk
└─sdb1       8:17     1     1G  0 part
  └─sdb2       8:18     1   2,7G  0 part
nvme0n1     259:0    0 953,9G  0 disk
└─nvme0n1p1 259:1    0   499M  0 part
└─nvme0n1p2 259:2    0   128M  0 part
└─nvme0n1p3 259:3    0 797,2G  0 part
└─nvme0n1p4 259:4    0   9,5G  0 part
[darksasuke@archlinux ~]$ sudo mkfs.ext4 /dev/sdb2
mke2fs 1.46.5 (30-Dec-2021)
En train de créer un système de fichiers avec 720384 4k blocs e
t 180224 i-noeuds.
UUID de système de fichiers=c0440c9d-eb0e-409d-91aa-1dedaa5c16c
3
Superblocs de secours stockés sur les blocs :
      32768, 98304, 163840, 229376, 294912

Allocation des tables de groupe : complété

Écriture des tables d'i-noeuds : complété

Création du journal (16384 blocs) :
```

```
[darksasuke@archlinux ~]$ sudo cryptsetup luksFormat /dev/sdb1

ATTENTION !
=====
Cette action écrasera définitivement les données sur /dev/sdb1.

Êtes-vous sûr ? (Typez « yes » en majuscules) : YES
Saisissez la phrase secrète pour /dev/sdb1 :
Vérifiez la phrase secrète :
```

```
[darksasuke@archlinux ~]$ sudo cryptsetup open /dev/sdb1 crypte
d
Saisissez la phrase secrète pour /dev/sdb1 :
[darksasuke@archlinux ~]$ lsblk
NAME      MAJ:MIN RM  SIZE RO TYPE  MOUNTPOINTS
sda        8:0      0 476,9G  0 disk
└─sda1     8:1      0  513M  0 part   /boot
└─sda2     8:2      0  93,1G  0 part
└─sda3     8:3      0 150G  0 part
└─sda4     8:4      0 149,5G  0 part   /
sdb        8:16     1   3,8G  0 disk
└─sdb1     8:17     1     1G  0 part
  └─crypted 254:0  0 1008M  0 crypt
  └─sdb2     8:18     1   2,7G  0 part
```

```

└─sda2      8:1    0   513M  0 part
nvme0n1    259:0   0 953,9G  0 disk
└─nvme0n1p1 259:1   0   499M  0 part
└─nvme0n1p2 259:2   0   128M  0 part
└─nvme0n1p3 259:3   0 797,2G  0 part
└─nvme0n1p4 259:4   0   9,5G  0 part
[darksasuke@archlinux ~]$ |

```

Après avoir mis un système de fichier sur le conteneur : on monte le conteneur

```

[darksasuke@archlinux ~]$ sudo mount /dev/mapper/crypted dos1
[darksasuke@archlinux ~]$ lsblk
NAME      MAJ:MIN RM    SIZE RO TYPE  MOUNTPOINTS
sda        8:0    0 476,9G  0 disk
└─sda1     8:1    0   513M  0 part  /boot
└─sda2     8:2    0   93,1G  0 part
└─sda3     8:3    0   150G  0 part
└─sda4     8:4    0 149,5G  0 part  /
sdb        8:16   1   3,8G  0 disk
└─sdb1     8:17   1   1G  0 part
└─crypted 254:0   0 1008M  0 crypt /home/darksasuke/dos1
└─sdb2     8:18   1   2,7G  0 part  /home/darksasuke/dos2
nvme0n1   259:0   0 953,9G  0 disk
└─nvme0n1p1 259:1   0   499M  0 part
└─nvme0n1p2 259:2   0   128M  0 part
└─nvme0n1p3 259:3   0 797,2G  0 part
└─nvme0n1p4 259:4   0   9,5G  0 part
[darksasuke@archlinux ~]$ |

```

Possible de faire le chiffrement avec un fichier avec l'argument **-key-file**

```

[darksasuke@archlinux ~]$ vim mdp
[darksasuke@archlinux ~]$ sudo cryptsetup luksFormat /dev/sdb -
--key-file mdp
ATTENTION: Le périphérique /dev/sdb contient déjà une signature
pour une partition « gpt ».

ATTENTION !
=====
Cette action écrasera définitivement les données sur /dev/sdb.

Êtes-vous sûr ? (Typez « yes » en majuscules) :

```

Pour ouvrir le fichier :

```

[darksasuke@archlinux ~]$ sudo cryptsetup open /dev/s
e --key-file mdp

```

Résumé des diverses commandes passées :

```

479  sudo mount /dev/mapper/crypted dos1
480  lsblk
481  sudo umount dos1
482  sudo umount dos2
483  lsblk

```

```

484 sudo cryptsetup close /dev/mapper/crypted
485 lsblk
486 vim mdp
487 sudo cryptsetup luksFormat /dev/sdb --key-file mdp
488 lsblk
489 sudo cryptsetup open /dev/sdb noisette --key-file mdp
490 lsblk
491 sudo cryptsetup close /dev/mapper/crypted
492 sudo cryptsetup close /dev/mapper/noisette
493 dd if=/dev/urandom of=mdp bs=2000 count=4
494 sudo cryptsetup luksFormat /dev/sdb --key-file mdp
495 lsblk
496 sudo cryptsetup open /dev/sdb noisette
497 sudo cryptsetup luksFormat /dev/sdb --key-file mdp
498 sudo cryptsetup open /dev/sdb noisette --key-file mdp
499 lsblk
500 cat mdp
501 history

```

- Si un dd se passe mal, il est possible de récupérer le fichier à partir du moment où l'on a fait un backup du header ! Si le header est altéré on ne pourra plus récupérer le fichier

```

HEADER BACKUP
luksHeaderBackup <device> --header-backup-file <file>

Stores a binary backup of the LUKS header and keyslot area.
See cryptsetup-luksHeaderBackup(8) .

HEADER RESTORE
luksHeaderRestore <device> --header-backup-file <file>

Restores a binary backup of the LUKS header and keyslot area from the specified file.
See cryptsetup-luksHeaderRestore(8) .

```

Dès qu'on chiffre -> toujours sauvegarder

 Pour faire un backup du header d'une partition cryptée, vous pouvez utiliser l'utilitaire cryptsetup avec l'option "luksHeaderBackup". Voici comment procéder :

1. Connectez la partition à votre système (par exemple en la branchant à l'aide d'une clé USB).
2. Ouvrez un terminal et utilisez la commande `sudo cryptsetup luksHeaderBackup <device> --header-backup-file <file>` pour créer un fichier de backup du header de la partition. Remplacez "<device>" par le nom de la partition (par exemple "/dev/sda1") et "<file>" par le nom et l'emplacement du fichier de backup que vous souhaitez créer (par exemple "/home/user/header-backup.img").
3. Saisissez votre mot de passe administrateur lorsque vous y êtes invité.
4. Le fichier de backup du header sera créé à l'emplacement spécifié. Vous pouvez le copier à un emplacement sécurisé pour en faire un backup supplémentaire.

Note : cette procédure nécessite les priviléges administrateur, car elle modifie le contenu du disque. Éitez attention à utiliser la option "luksHeaderBackup" avec soin.

du disque. Faites attention à utiliser l'option "luksHeaderBackup" et non pas "luksHeaderDump", qui ne crée pas de backup mais affiche simplement le contenu du header sur la sortie standard.

LVM :

Partitionnement avec LVM

Logical Volume manager

LVM sert à créer/modifier des partitions logiques.
On l'utilise donc à la place de cfdisk/parted

LVM peut:

- créer des partitions qui utilisent plusieurs device
- modifier facilement les tailles des partitions -->
- rajouter/supprimer de l'espace total à utiliser, rajouter un device

PV:Physical Volume

Il faut "déclarer" des volumes physiques pour que lvm puisse les utiliser

```
$> pvcreate [volume]
```

```
$> pvcreate /dev/sdb  
$> pvcreate /dev/sdc
```



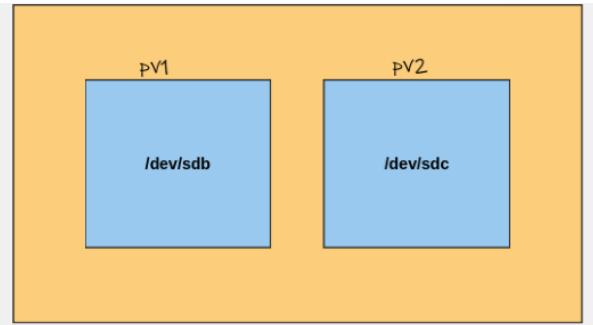
```
[darksasuke@archlinux ~]$ sudo pvcreate /dev/sdb
WARNING: crypto_LUKS signature detected on /dev/sdb at offset 0. Wipe it? [y/n]: y
      Wiping crypto_LUKS signature on /dev/sdb.
WARNING: crypto_LUKS signature detected on /dev/sdb at offset 16384. Wipe it? [y/n]: y
      Wiping crypto_LUKS signature on /dev/sdb.
Physical volume "/dev/sdb" successfully created.
```

```
[darksasuke@archlinux ~]$ pvdisplay
      WARNING: Running as a non-root user. Functionality may be unavailable.
/run/lock/lvm/P_global:aux: open failed: Permission non accordée
[darksasuke@archlinux ~]$ sudo pvdisplay
      "/dev/sdb" is a new physical volume of "3,75 GiB"
      --- NEW Physical volume ---
      PV Name          /dev/sdb
      VG Name
      PV Size         3,75 GiB
      Allocatable     NO
      PE Size          0
      Total PE         0
      Free PE          0
      Allocated PE      0
      PV UUID          C12S5O-i6V5-v0c2-jFwi-THOQ-OByh-20uP7T
```

VG: Virtual Group

on crée un groupe de un ou plusieurs PV

```
$> vgcreate [nom] [PV_de_base]  
$> vgcreate LVTEST /dev/sdb
```



Voir les VG

```
$> vgdisplay
```

Rajouter un PV au VG

```
$> sudo vgextend LVTEST /dev/sdc
```

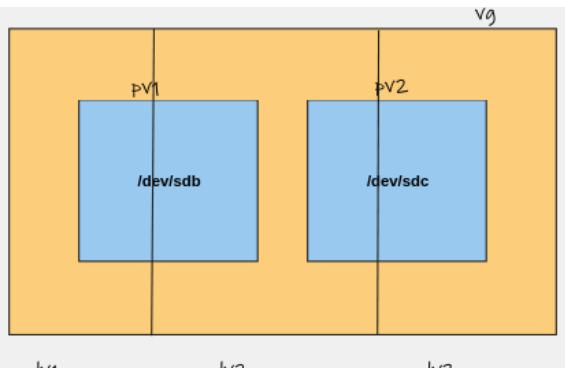
```
[darksasuke@archlinux ~]$ sudo vgcreate my_group /dev/sdb  
  Volume group "my_group" successfully created  
[darksasuke@archlinux ~]$ sudo vgdisplay  
--- Volume group ---  
VG Name           my_group  
System ID  
Format          lvm2  
Metadata Areas      1  
Metadata Sequence No 1  
VG Access        read/write  
VG Status         resizable  
MAX LV            0  
Cur LV            0  
Open LV            0  
Max PV            0  
Cur PV            1  
Act PV            1  
VG Size          <3,75 GiB  
PE Size          4,00 MiB  
Total PE          959  
Alloc PE / Size    0 / 0  
Free  PE / Size    959 / <3,75 GiB  
VG UUID          yfGuix-2NdF-oYz8-LICx-wYXQ-dUfy-X3jpsa
```

LV: Logical Volume

on crée des volumes logiques sur un VG

```
$> lvcreate [VG] -n [lv_name] -l [SIZE]  
$> lvcreate LVTEST -n lv1 -l 33%VG  
$> lvcreate LVTEST -n lv2 -l 33%VG  
$> lvcreate LVTEST -n lv3 -l 100%FREE
```

les LV sont accessible dans :



```
les LV sont accessibles via :
```

```
/dev/mapper/[VG_NAME]-[LV_NAME]
```

```
voir les LV:
```

```
$> lvdisplay
```

```
$> lsblk
```

```
supprimer un lv
```

```
$> lvremove /dev/mapper/LVTEST-lv3
```

```
Agrandir un LV:
```

```
$> lvresize /dev/mapper/LVTEST-lv2 -l +100%FREE
```

```
[darksasuke@archlinux ~]$ sudo lvcreate my_group -n part1 -l 1G
    Invalid argument for --extents: 1G
    Error during parsing of command line.
[darksasuke@archlinux ~]$ sudo lvcreate my_group -n part1 -L 1G
    Logical volume "part1" created.
[darksasuke@archlinux ~]$ lsblk


| NAME             | MAJ:MIN | RM | SIZE   | RO | TYPE | MOUNTPOINTS |
|------------------|---------|----|--------|----|------|-------------|
| sda              | 8:0     | 0  | 476,9G | 0  | disk |             |
| └─sda1           | 8:1     | 0  | 513M   | 0  | part | /boot       |
| └─sda2           | 8:2     | 0  | 93,1G  | 0  | part |             |
| └─sda3           | 8:3     | 0  | 150G   | 0  | part |             |
| └─sda4           | 8:4     | 0  | 149,5G | 0  | part | /           |
| sdb              | 8:16    | 1  | 3,8G   | 0  | disk |             |
| └─my_group-part1 | 254:0   | 0  | 1G     | 0  | lvm  |             |
| nvme0n1          | 259:0   | 0  | 953,9G | 0  | disk |             |
| └─nvme0n1p1      | 259:1   | 0  | 499M   | 0  | part |             |
| └─nvme0n1p2      | 259:2   | 0  | 128M   | 0  | part |             |
| └─nvme0n1p3      | 259:3   | 0  | 797,2G | 0  | part |             |
| └─nvme0n1p4      | 259:4   | 0  | 9,5G   | 0  | part |             |


[darksasuke@archlinux ~]$ |
```

Si je veux ajouter de l'espace à une partition

Je rajoute un PV au VG

```
$> pvcreate /dev/sdd
```

```
$> vgextends LVTEST /dev/sdd
```

Le VG est agrandit de la taille du nouveau PV

Je peux resize un LV

```
$> lvresize /dev/mapper/LVTEST-lv2 -l +100%FREE
```

Attention LVRESIZE ne modifie que la taille de la partition,
pas la taille du système de fichier:

resize le système de fichier:

```
$> resize2fs
```

- Remove et resize :

```
[darksasuke@archlinux ~]$ sudo lvremove /dev/mapper/my_group-part3
Do you really want to remove active logical volume my_group/part3? [y/n]: y
Logical volume "part3" successfully removed.
[darksasuke@archlinux ~]$ lsblk
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda        8:0    0 476,9G 0 disk
└─sda1     8:1    0   513M 0 part /boot
└─sda2     8:2    0   93,1G 0 part
└─sda3     8:3    0   150G 0 part
└─sda4     8:4    0 149,5G 0 part /
sdb        8:16   1   3,8G 0 disk
└─my_group-part1 254:0  0   1,2G 0 lvm
└─my_group-part2 254:1  0   1,2G 0 lvm
nvme0n1    259:0   0 953,9G 0 disk
└─nvme0n1p1 259:1   0   499M 0 part
└─nvme0n1p2 259:2   0   128M 0 part
└─nvme0n1p3 259:3   0 797,2G 0 part
└─nvme0n1p4 259:4   0   9,5G 0 part
[darksasuke@archlinux ~]$ sudo lvresize /dev/mapper/my_group-part2 -l
100%FREE
Size of logical volume my_group/part2 changed from 1,23 GiB (316 ext
ents) to <1,28 GiB (327 extents).
Logical volume my_group/part2 successfully resized.
[darksasuke@archlinux ~]$
```

Vgextend :

```
[darksasuke@archlinux ~]$ sudo vgextend my_group /dev/sdc
Volume group "my_group" successfully extended
[darksasuke@archlinux ~]$ sudo vgdisplay
--- Volume group ---
VG Name           my_group
System ID
Format          lvm2
Metadata Areas   2
Metadata Sequence No 15
VG Access       read/write
VG Status        resizable
MAX LV          0
Cur LV          2
Open LV          0
Max PV          0
Cur PV          2
Act PV          2
VG Size         11,23 GiB
PE Size          4,00 MiB
Total PE        2875
Alloc PE / Size 572 / 2,23 GiB
Free  PE / Size 2303 / <9,00 GiB
VG UUID         yfGuIX-2NdF-oYz8-LICx-wYXQ-dUfy-X3jpsa
```

```
[darksasuke@archlinux ~]$ sudo lvresize /dev/mapper/my_group-part2 -l
+100%FREE
Size of logical volume my_group/part2 changed from 1,00 GiB (256 ext
ents) to <10,00 GiB (2559 extents).
Logical volume my_group/part2 successfully resized.
```

Exercice :

```
CH dd if=/dev/urandom of=mdp_contener bs=50 count=1
```



La commande `dd` que vous avez fournie permet de copier 50 blocs de données aléatoires générées par le périphérique `/dev/urandom` dans un fichier nommé `mdp_contener`.

Le périphérique `/dev/urandom` est un périphérique spécial qui génère des données aléatoires lorsqu'il est lu. Il est souvent utilisé pour générer des mots de passe, des clés de chiffrement, etc.

La commande `dd` accepte également l'option `of` pour spécifier le fichier de sortie et l'option `bs` pour spécifier la taille des blocs de données à lire et à écrire. Dans cet exemple, 50 blocs de données de 50 octets (`bs=50`) sont lus et écrits dans le fichier `mdp_contener`.

Il est important de noter que la commande `dd` peut être dangereuse si elle est utilisée de manière incorrecte, car elle peut écraser des données importantes. Assurez-vous de vérifier les chemins d'accès d'entrée et de sortie avant de l'utiliser et de faire une sauvegarde de vos données importantes.

Important : Quand on a des problèmes de cohérences sur l'affichage des partitions avec lsblk :

La commande **dmsetup** permet de voir les disques afficher :

```
576 sudo dmsetup status
```

On peut remove le device concerné avec la commande :

```
578 sudo dmsetup remove my_group-part0
```

Une fois la commande passé : la partition n'apparaît plus quand on tape la commande lsblk

```
loop19      7:19   0  81,7M  1  loop  /snap/teams
sda         8:0    1  7,5G  0  disk
nvme0n1    259:0   0  953,9G  0  disk
└─nvme0n1p1 259:1   0   499M  0  part  /boot/efi
└─nvme0n1p2 259:2   0   128M  0  part
└─nvme0n1p3 259:3   0   793,7G  0  part
└─nvme0n1p4 259:4   0   9,5G  0  part
└─nvme0n1p5 259:5   0   150G  0  part  /var/snap/f
```

Pour rajouter les scripts à la variable d'environnement PATH :

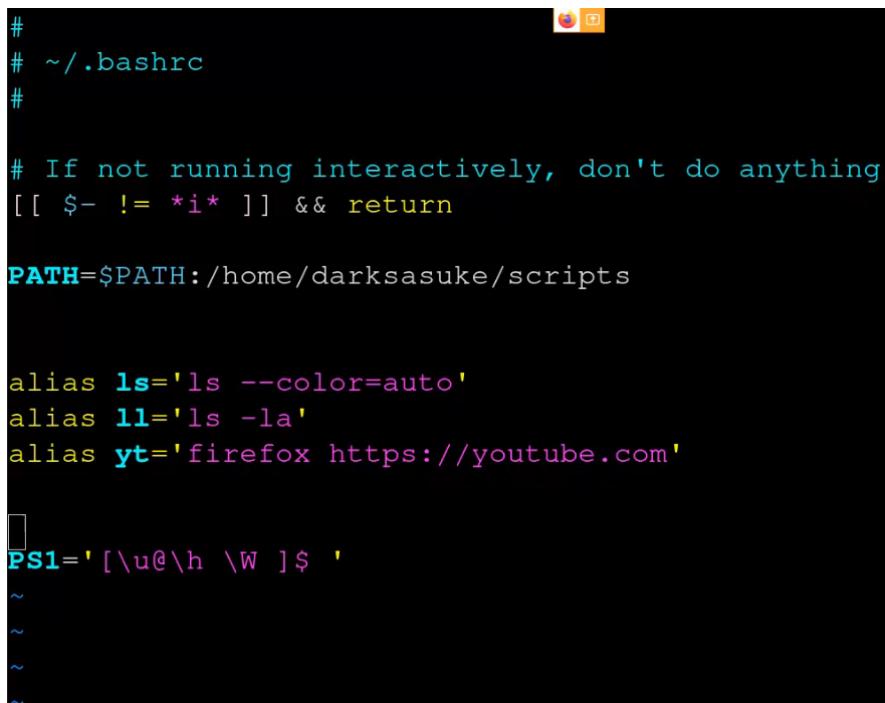
On peut les stocker dans un dossier qu'on va intégrer à la variable d'environnement avec les commandes suivante :

```
[darksasuke@archlinux ~] $ PATH=$PATH:/home/darksasuke/scripts
[darksasuke@archlinux ~] $ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/bin:/usr/bin/site_perl:/usr/bin/vendor_perl
:/usr/bin/core_perl:/home/darksasuke/scripts
[darksasuke@archlinux ~] $ |
```

Les variables d'environnement seront active uniquement tant que le terminal dans lequel a été passé la commande est actif, il faudrait réaliser une action supplémentaire : modifier le fichier **bashrc**

```
[darksasuke@archlinux ~] $ source ~/.bashrc
[darksasuke@archlinux ~]$ vim ~/.bashrc
[darksasuke@archlinux ~]$ source ~/.bashrc
```

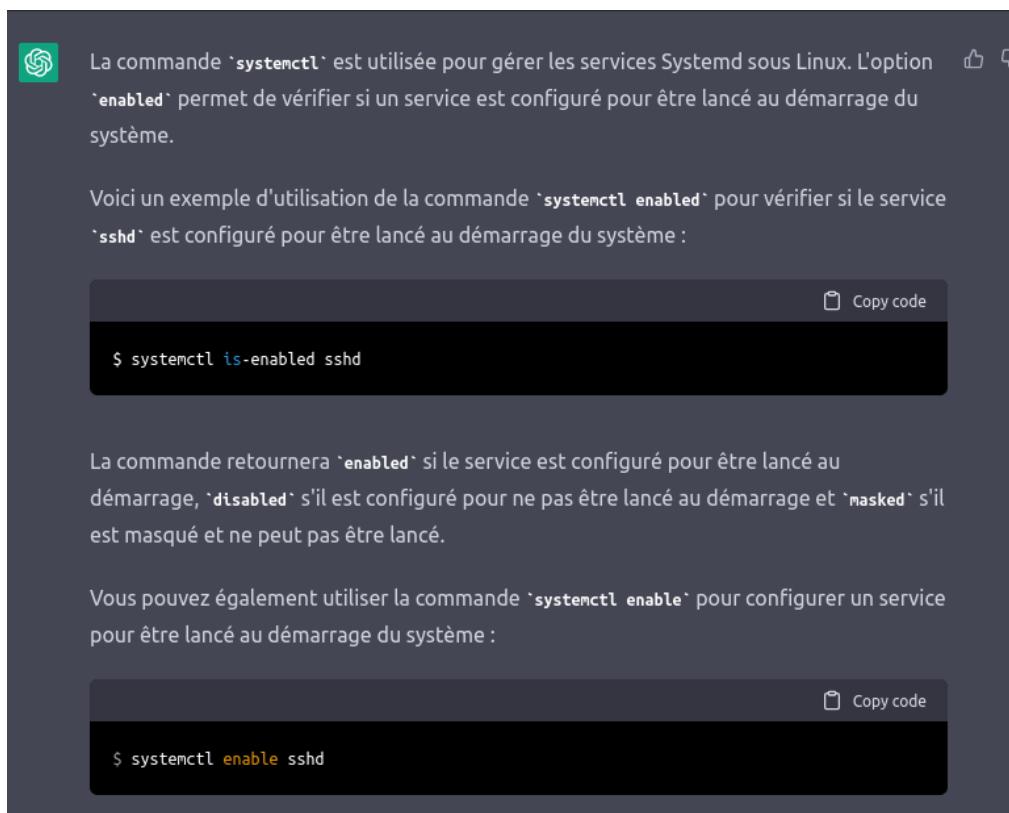
```
[darksasuke@archlinux ~]\$ cd  
[darksasuke@archlinux ~]\$
```



```
#  
# ~/.bashrc  
  
# If not running interactively, don't do anything  
[[ $- != *i* ]] && return  
  
PATH=$PATH:/home/darksasuke/scripts  
  
alias ls='ls --color=auto'  
alias ll='ls -la'  
alias yt='firefox https://youtube.com'  
  
PS1='[\u@\h \W ]\$ '  
~  
~  
~  
~
```

Modifier ligne PATH avec le chemin du dossier

Pour exécuter un logiciel au démarrage :



La commande `systemctl` est utilisée pour gérer les services Systemd sous Linux. L'option `enabled` permet de vérifier si un service est configuré pour être lancé au démarrage du système.

Voici un exemple d'utilisation de la commande `systemctl enabled` pour vérifier si le service `sshd` est configuré pour être lancé au démarrage du système :

```
$ systemctl is-enabled sshd
```

La commande retournera `enabled` si le service est configuré pour être lancé au démarrage, `disabled` s'il est configuré pour ne pas être lancé au démarrage et `masked` s'il est masqué et ne peut pas être lancé.

Vous pouvez également utiliser la commande `systemctl enable` pour configurer un service pour être lancé au démarrage du système :

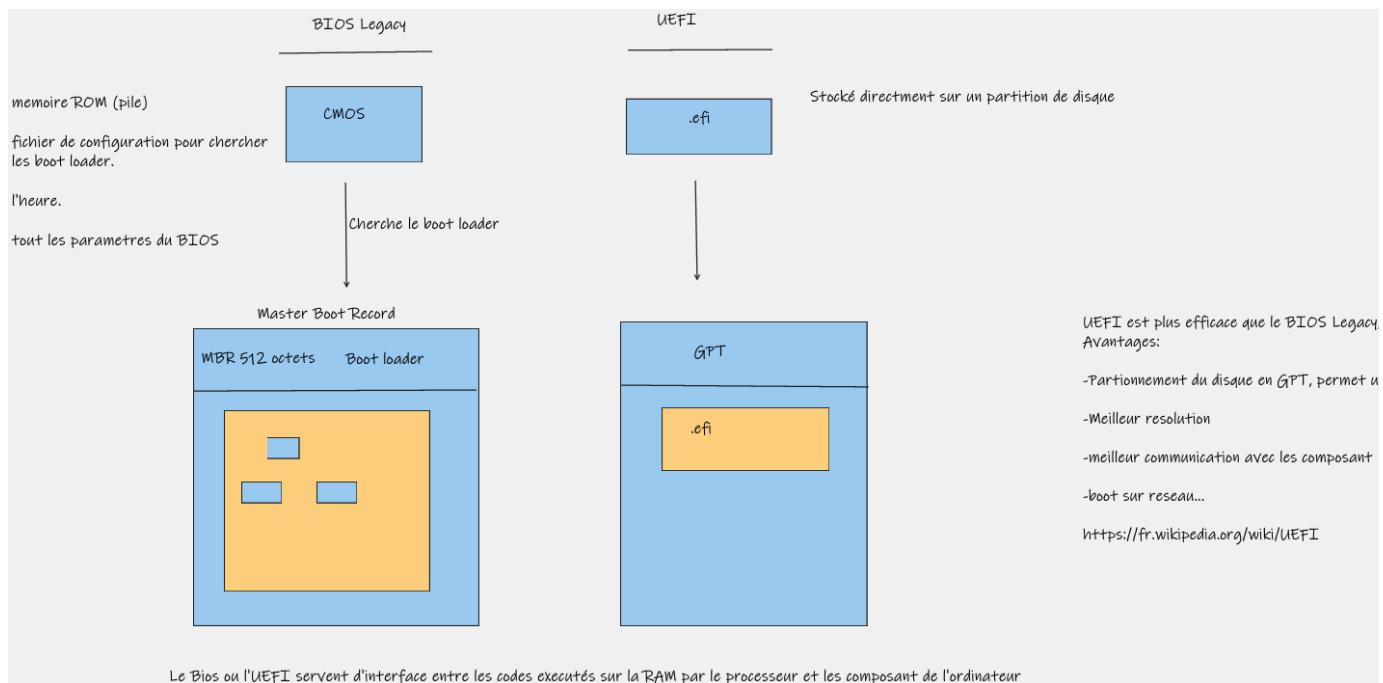
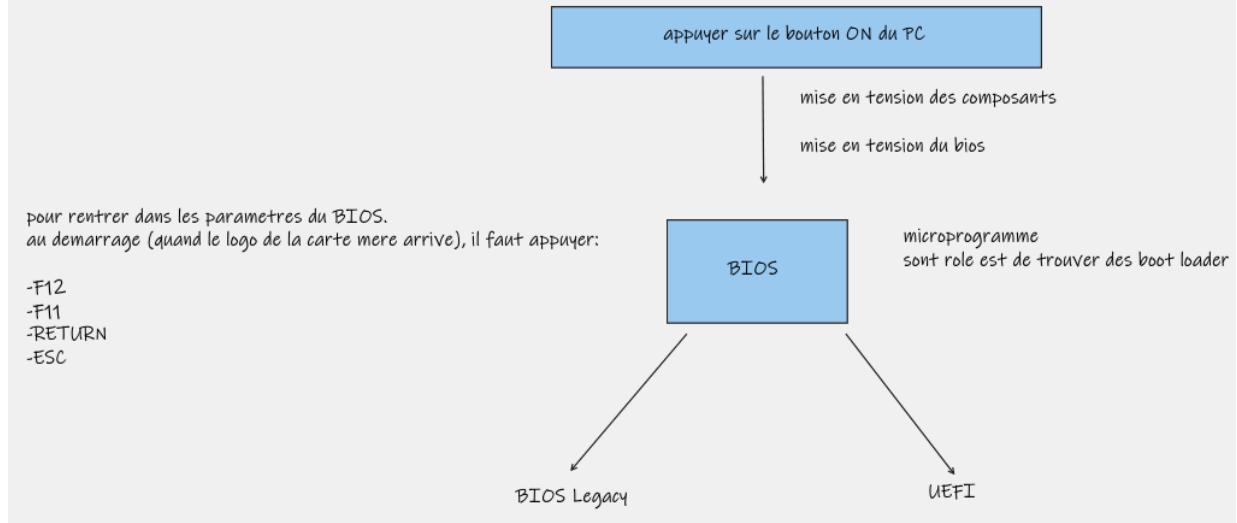
```
$ systemctl enable sshd
```

Avant l'os

SEQUENCE D'ALLUMAGE

boot sequence

La sequence de démarrage



Le Bios ou l'UEFI servent d'interface entre les codes exécutés sur la RAM par le processeur et les composant de l'ordinateur

UEFI

Article Discussion

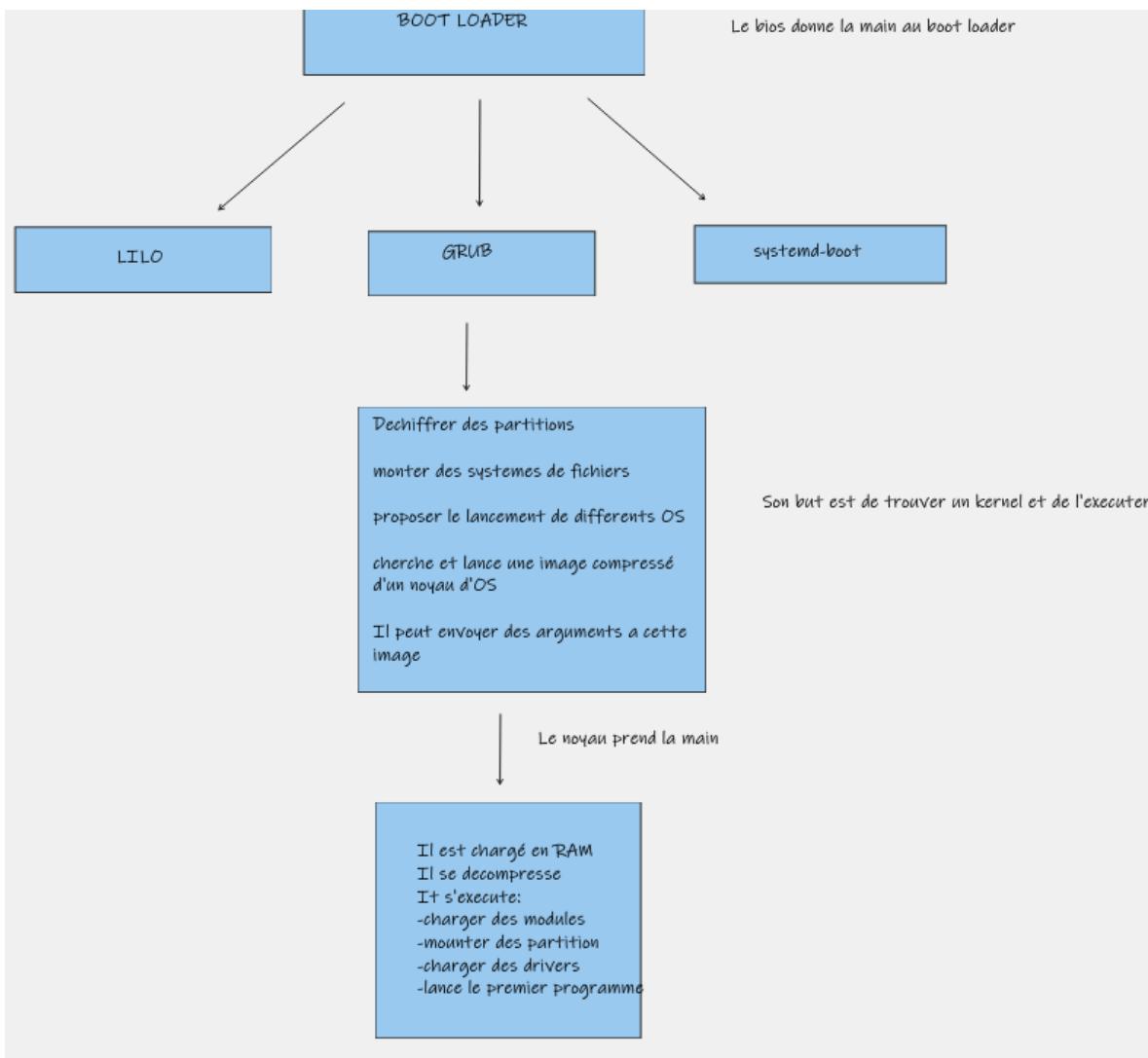
Lire Modifie

Pour les articles homonymes, voir [EFI](#).

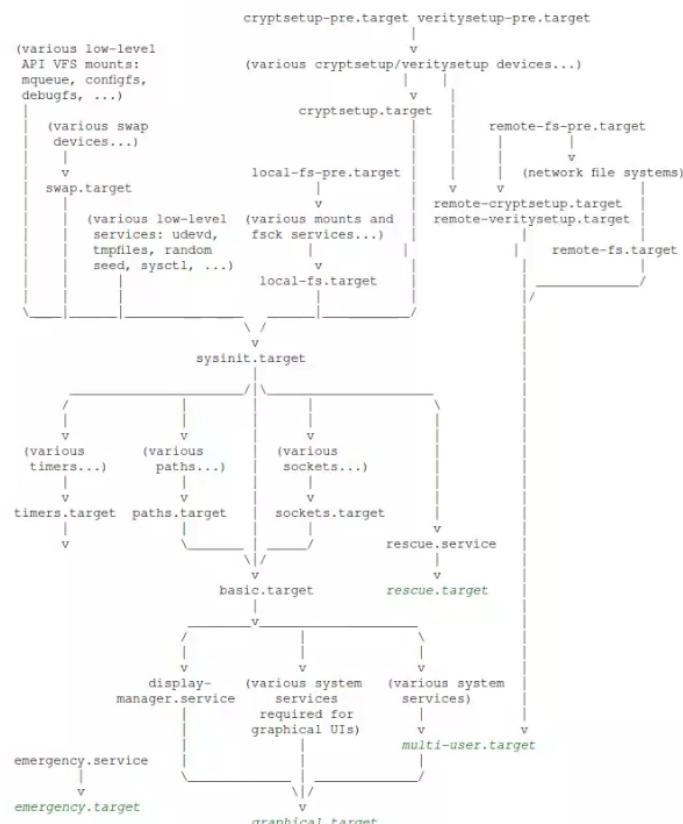
Le standard **UEFI** (de l'anglais **Unified Extensible Firmware Interface**, signifiant en français : « Interface micrologicielle extensible unifiée ») définit une [interface](#) entre le [micrologiciel \(firmware\)](#) et le [système d'exploitation \(OS\)](#) d'un [ordinateur](#). Cette interface succède sur certaines cartes-mères au [BIOS](#).

IL faut désactiver le secure boot pour démarrer une distribution Linux avec UEFI





Service :



timers.target is pulled-in by basic.target asynchronously. This allows timers units to depend on services which become only available later in boot.

SEQUENCE DE DEMARRAGE

startup sequence

Quand l'os prend la main

LINUX

/sbin/init est la parent de tout les scripts, il a l'id 1

systemd lance des programmes

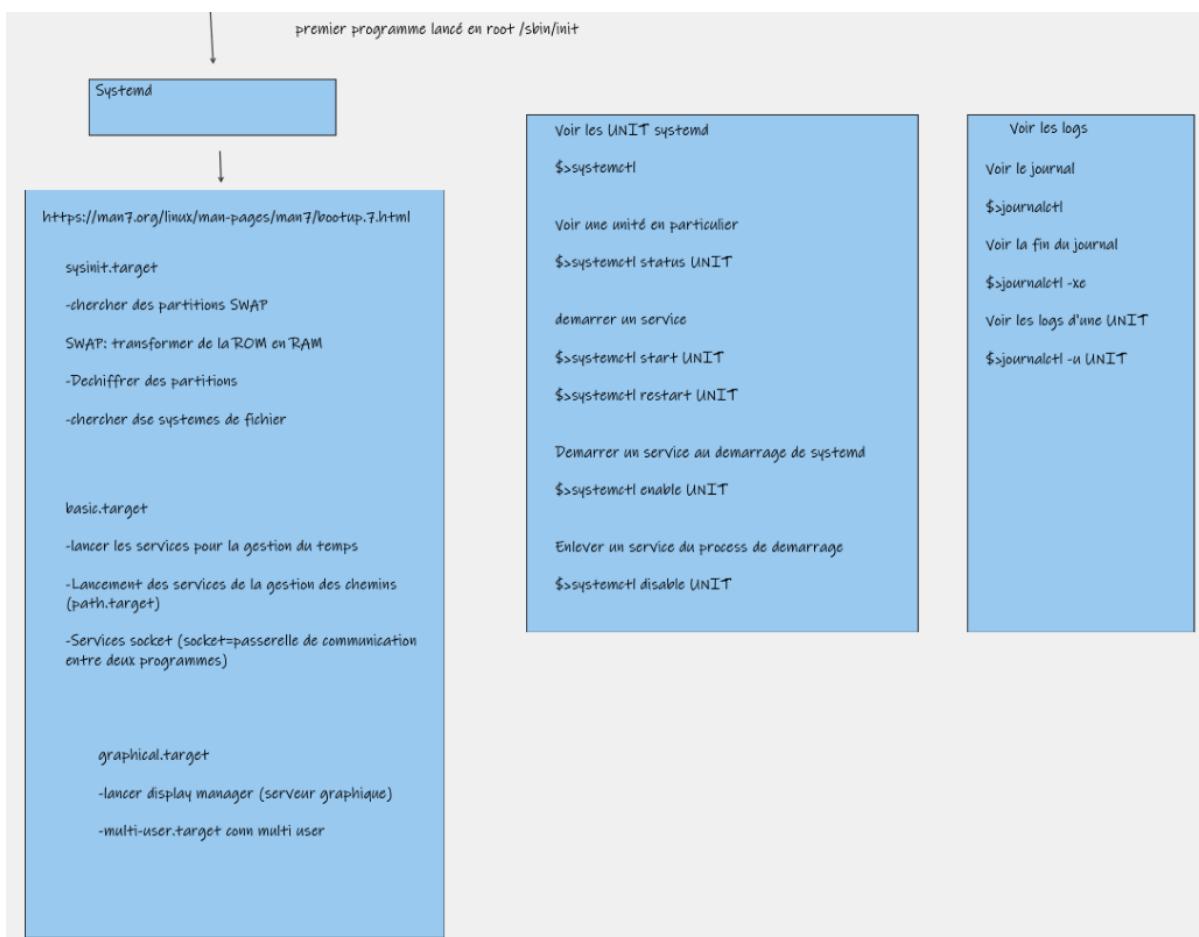
ces programmes sont appelés services

ces services sont paramétrables

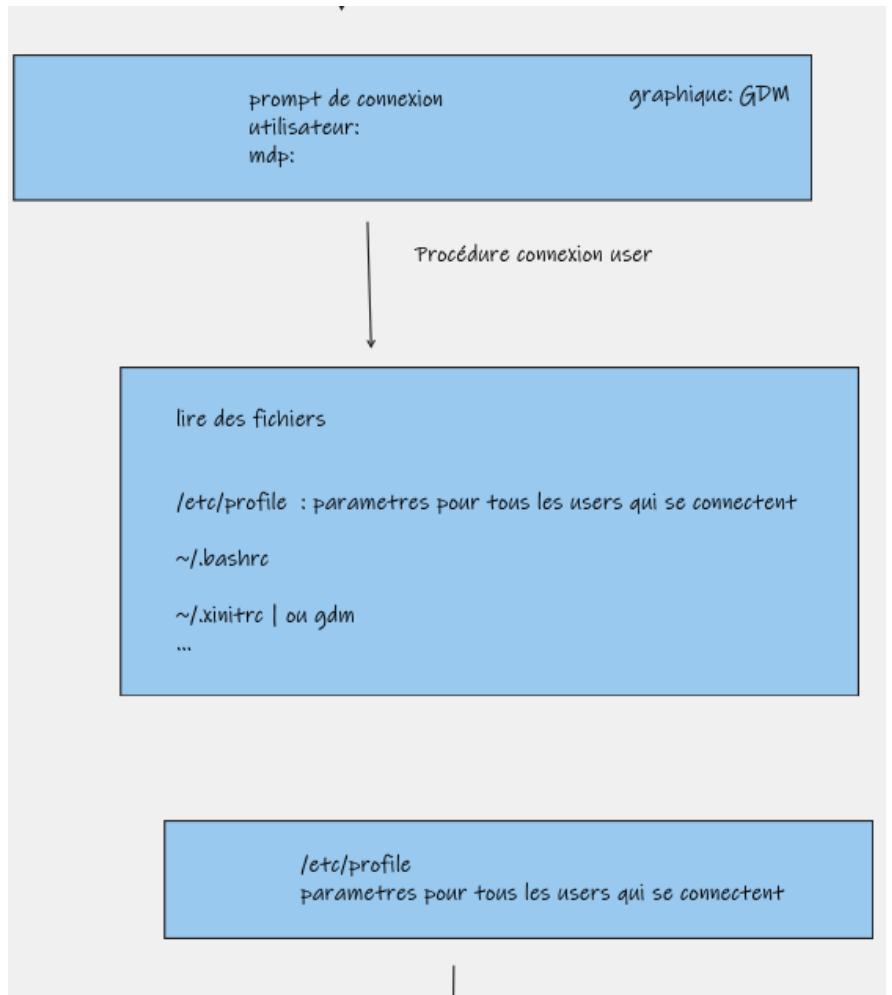
ces services ont des logs

ces services peuvent être lancé en parallèle.

ces services sont nommés 'daemon'



getty -> login



Commande pour changer de tty -> CTRL/ALT/F7

