

# BTRFS

Présentation du système de fichiers btrfs :

Le système de fichiers btrfs (B-tree File System) est un système de fichiers moderne et avancé développé conjointement par Oracle, Red Hat, Fujitsu, Intel, SUSE, STRATO AG. Il a été conçu pour répondre aux besoins des systèmes de stockage modernes, en offrant de nombreuses fonctionnalités intéressantes :

1. Copie sur écriture (Copy-on-Write) : Btrfs utilise la copie sur écriture, ce qui permet de créer rapidement des instantanés (snapshots) du système de fichiers sans consommer beaucoup d'espace disque.
2. Gestion du stockage : Btrfs permet de gérer facilement le stockage en utilisant des volumes logiques, des sous-volumes et des quotas. Cela facilite la gestion de l'espace disque.
3. Réplication et tolérance aux pannes : Btrfs prend en charge la réplication des données sur plusieurs disques, offrant ainsi une meilleure tolérance aux pannes.
4. Compression des données : Btrfs peut compresser les données en ligne, ce qui permet de réduire l'espace disque utilisé.
5. Déduplication : Btrfs peut détecter et supprimer les doublons de données, ce qui permet également de gagner de l'espace disque.
6. Équilibrage de charge : Btrfs peut automatiquement équilibrer la charge sur plusieurs disques pour optimiser les performances.
7. Vérification de l'intégrité des données : Btrfs peut vérifier l'intégrité des données et les corriger en cas d'erreurs.

## Gestion BTRFS

### Compression

Btrfs prend en charge la compression à la volée (transparent pour l'utilisateur), on renseigne la méthode et le niveau dans `/etc/fstab`.

Pour ma part, j'utilise l'algorithme qui a le meilleur compromis entre compression et performance.

Il suffit de rajouter cette option pour le montage :

```
compress=zstd:3
```

Dans cet exemple, j'utilise l'algorithme zstd et choisis le niveau 3, sachant que l'on peut aller de 1 à 15, niveau 1 très faible taux de compression, mais très grande rapidité, 15 énormes taux de compression, mais rapidité réduite.

On a donc la ligne dans le fichier fstab qui ressemble à ça :

```
UUID=aa78bd8c-d856-4059-b214-2be73e6d99aae / btrfs compress=zstd:3,subvol=/@,defaults 0 0
```

### Documentation officielle.

Pour compresser les données déjà existantes, il faut lancer une défragmentation avec l'option de décompression :

```
sudo btrfs filesystem defragment -r -czstd /
```

L'option `-r` signifie que l'on va faire les sous-répertoires avec, l'option `-c` utilise la compression `zstd` est la méthode.

Pour vérifier la place gagnée, on peut utiliser la commande suivante :

```
sudo btrfs filesystem df /
```

Dans mon cas, voici le retour :

```
Data, single: total=339.01GiB, used=331.45GiB
System, DUP: total=8.00MiB, used=64.00KiB
Metadata, DUP: total=2.00GiB, used=1.22GiB
GlobalReserve, single: total=355.16MiB, used=0.00B
```

Sur la première ligne, on observe que j'ai 339 Go de données qui prend 331 Go sur le disque.

## Lister les volumes

```
sudo btrfs subvolume list ./
```

Dans l'exemple, on liste des sous-volumes qui se trouvent dans `/`.

# Créer/restaurer un instantané.

Avant de commencer, il faut que le système de fichier BTRFS soit monter (de préférence à la racine) pour pouvoir créer et restaurer correctement les instantanés.

Dans mon exemple, j'ai créé un dossier `/btrfs`.

```
sudo mkdir /btrfs
```

Ensuite, j'ai renseigné le point de montage dans mon fichier `/etc/fstab`, la ligne 10 dans l'exemple :

```
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a device; this may
# be used with UUID= as a more robust way to name devices that works even if
# disks are added and removed. See fstab(5).
#
# <file system>      <mount point> <type> <options> <dump> <pass>
UUID=aa78bd8c-d856-4059-b214-2be73e6d99aae /          btrfs compress=zstd:3,subvol=/@,defaults 0 0
UUID=aa78bd8c-d856-4059-b214-2be736d9e9aae /home      btrfs
compress=zstd:3,subvol=/@home,defaults 0 0
UUID=aa78bd8c-d856-4059-b214-2be736d9e9aae /btrfs      btrfs
compress=zstd:3,defaults                0 0
tmpfs                                /tmp      tmpfs defaults,noatime,mode=1777    0 0
```

Pour monter la partition.

```
sudo systemctl daemon-reload
sudo mount -a
```

## Créer un instantané.

```
sudo btrfs subvolume snapshot /btrfs/@ /btrfs/@-$(date +%d-%m-%Y_%H-%M-%S)
```

Dans l'exemple, on a fait un instantané de la racine du système, la seconde partie est le chemin de l'instantané, à la fin, j'ai placé une variable pour l'horodater.

## Restaurer un instantané.

Pour restaurer un instantané, il suffit d'utiliser la commande `mv`, le but de la manœuvre est d'intervertir le sous-volume de base avec l'instantané, puis on redémarre.

```
# Ont isole le sous-volume
sudo mv /btrfs/@ /btrfs/@_old

# ont déplace l'instantané en lieu et place du sous-volume
sudo mv /btrfs/@-05-02-2023_13-02-56 /btrfs/@

# ont redémarre l'ordinateur sur l'instantané
reboot
```

## Démarrer sur un instantané sans restauration.

Il est possible de démarrer sur un instantané sans passer par une restauration.

Il faut avant tout connaître le nom de l'instantané sur lequel ont désire démarré.

Dans ces manipulations, il se peut que la disposition clavier soit en QWERTY, [voici un comparatif](#).

Il faut également modifier le fichier `fstab` et renseigner le subvolume souhaiter dans le subvolume lui-même, exemple pour `@-05-02-2023_13-02-56` :

```
sudo nano /btrfs/@-05-02-2023_13-02-56/etc/fstab
```

Modifier la partie `subvol=/@` en `subvol=/@-05-02-2023_13-02-56` :

```
UUID=aa78bd8c-d856-4059-b214-2be73e6d99aae /          btrfs  compress=zstd:3,subvol=/@-05-02-2023_13-02-56,defaults 0 0
```

## Systemd-boot.

Sur systemd-boot, au démarrage, restez appuyé sur la touche `ECHAP` du clavier.

Le menu systemd-boot apparait, laisser la ligne en surbrillance par défaut et appuyer sur la touche `e` pour accéder au mode édition.



Remplacer le `@` par l'instantané souhaité.

Solus 4.7 Endurance  
EFI Default Loader  
Reboot Into Firmware Interface

```
root=PARTUUID=c52d6fc0-5369-400f-9a9d-21a7c699d39b rootflags=subvol=@-05-02-2023_13-02-56 rd.vcons
```

Touche **ENTRER** pour démarrer sur l'instantané.

## GRUB.

Sur GRUB, au démarrage, restez appuyé sur la touche **SHIFT** du clavier.

Le menu GRUB apparait, laisser la ligne en surbrillance par défaut et appuyer sur la touche **e** pour accéder au mode édition.

La ligne qui nous intéresse est celle commençant par **linux**.

GNU GRUB version 2.12

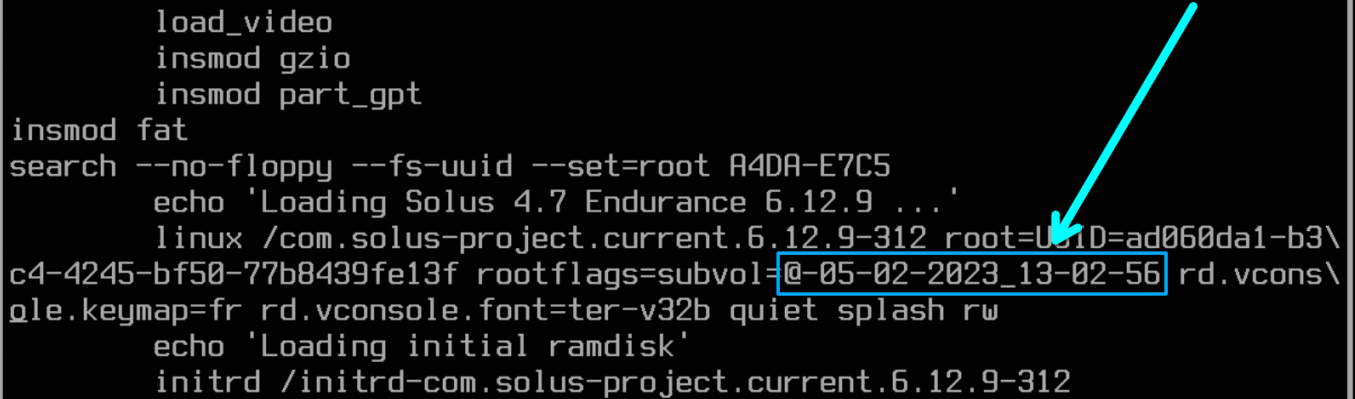
```
load_video
insmod gzio
insmod part_gpt
insmod fat
search --no-floppy --fs-uuid --set=root A4DA-E7C5
echo 'Loading Solus 4.7 Endurance 6.12.9 ...'
linux /com.solus-project.current.6.12.9-312 root=UUID=ad060da1-b3\
c4-4245-bf50-77b8439fe13f rootflags=subvol=@ rd.vconsole.keymap=fr rd.vco\
nsole.font=ter-v32b quiet splash rw
echo 'Loading initial ramdisk'
initrd /initrd-com.solus-project.current.6.12.9-312
```

Minimum Emacs-like screen editing is supported. TAB lists completions. Press Ctrl-x or F10 to boot, Ctrl-c or F2 for a command-line or ESC to discard edits and return to the GRUB menu.

Remplacer le @ par l'instantané souhaité.

GNU GRUB version 2.12

```
load_video
insmod gzio
insmod part_gpt
insmod fat
search --no-floppy --fs-uuid --set=root A4DA-E7C5
echo 'Loading Solus 4.7 Endurance 6.12.9 ...'
linux /com.solus-project.current.6.12.9-312 root=UUID=ad060da1-b3\
c4-4245-bf50-77b8439fe13f rootflags=subvol=@-05-02-2023_13-02-56 rd.vconso\
le.keymap=fr rd.vconsole.font=ter-v32b quiet splash rw
echo 'Loading initial ramdisk'
initrd /initrd-com.solus-project.current.6.12.9-312
```



Minimum Emacs-like screen editing is supported. TAB lists completions. Press Ctrl-x or F10 to boot, Ctrl-c or F2 for a command-line or ESC to discard edits and return to the GRUB menu.

Touche **F10** pour démarrer sur l'instantané.

## Maintenance.

Selon la documentation de Manjaro, le BTRFS n'a pas spécialement besoin de maintenance, il est conseillé de supprimer de temps à autre, de supprimer des instantanés non-utiliser pour libérer de l'espace.

## Statistique

Voici une commande qui permet d'avoir des statistiques sur l'espace :

```
sudo btrfs filesystem usage -h /
```

## Supprimer un instantané

La commande suivante permet de supprimer un instantané :

```
sudo btrfs subvolume delete /btrfs/@-05-02-2023_13-02-56
```

## Informations sur un instantané

Pour obtenir des infos sur un instantané, il suffit de lancer la commande suivant en précisant bien le chemin vers celui-ci, dans l'exemple pour `/btrfs/@`.

```
sudo btrfs sub show /btrfs/@
```

## Déduplication

La restauration de sauvegarde peut créer des fichiers en double dans le système de fichier, la commande suivante permet de supprimer les doublons :

```
sudo duperemove -drh --hashfile=/.duperemove.dat /
```

- **duperemove** : C'est l'outil principal qui effectue la déduplication. Il identifie les fichiers en double et les remplace par des références à un seul fichier, économisant ainsi de l'espace disque.
- **-d** : Cette option indique à **duperemove** de supprimer les fichiers en double qu'il trouve. Sans cette option, l'outil se contenterait d'identifier les fichiers en double sans les supprimer.
- **-r** : Cette option signifie "récursif". Elle indique à **duperemove** de parcourir tous les sous-répertoires du répertoire spécifié (dans ce cas, **/**).
- **-h** pour l'affichage humain des tailles
- **--hashfile=/.duperemove.dat** : Cette option spécifie le chemin vers un fichier de hachage où **duperemove** stocke les informations sur les fichiers déjà traités. Cela permet à l'outil de ne pas avoir à recalculer les hachages des fichiers déjà analysés lors des exécutions ultérieures.
- **/** : C'est le répertoire cible pour l'analyse. Dans ce cas, il s'agit de la racine du système de fichiers, ce qui signifie que **duperemove** va analyser tous les fichiers et répertoires sur le système.

## Scripts et configuration

### Instantanés programmé

Sur Solus les mises à jour sont déployées le vendredi tard dans la nuit, je souhaite donc in instantané automatique chaque vendredi.

Voici le script que systemd va lancer tous les jours, le script contient une condition, c'est qu'on soit le 5<sup>e</sup> jour de la semaine.

Remplacer **CHEMIN** par le chemin voulu et enregistré à l'endroit voulu.

```
#!/bin/bash

# Définir le chemin où les instantanés sont stockés
SNAPSHOT_DIR="/btrfs"
INDEX_FILE="/CHEMIN/index.txt"
```

```

ENTRIES_DIR="/CHEMIN"

# Vérifie si aujourd'hui est un vendredi (5)
if [ "$(date +%u)" -eq 5 ]; then
    # Vérifie si un instantané a déjà été créé aujourd'hui
    if ls ${SNAPSHOT_DIR}/@-$(date +%d-%m-%Y)* 1> /dev/null 2>&1; then
        echo "Un instantané a déjà été créé aujourd'hui."
        exit 0
    fi

    # Crée un nouvel instantané
    btrfs subvolume snapshot ${SNAPSHOT_DIR}/@ ${SNAPSHOT_DIR}/@-$(date +%d-%m-%Y_%H-%M-%S)

    # Garde un historique du noyau utilisé lors de la création de cet instantané
    echo "@-$(date +%d-%m-%Y_%H-%M-%S) kernel $(uname -a | awk '{print $3}')" >> ${INDEX_FILE}
    chown 1000:1000 ${INDEX_FILE}

    # Partie spécifique à Solus, récupère les entrées de systemd-boot
    clr-boot-manager mount-boot
    cp -r /boot/loader/entries ${ENTRIES_DIR}/entries-$(uname -a | awk '{print $3}')
    umount /boot

    # Ajoutez d'autres commandes ici
else
    # Vous pouvez laisser cette ligne vide si vous ne voulez rien faire
    :
fi

```

Ensuite, il faut créer le service et le timer pour systemd.

Voici l'emplacement où stocker les fichiers : `/etc/systemd/system/` pour l'exemple, je vais les nommer `btrfs_friday_snapshot`

Pour le service, je crée dans le dossier `/etc/systemd/system/` le fichier `btrfs_friday_snapshot.service`

```

[Unit]
Description=instantané BTRFS exécuté tous les vendredis

[Service]
Type=oneshot

```



```
ExecStart=/CHEMIN VERS DE SCRIPT/btrfs_friday_snapshot.sh
```

```
[Install]
```

```
WantedBy=multi-user.target
```

Pour le timer, je crée dans le dossier `/etc/systemd/system/` le fichier `btrfs_friday_snapshot.timer`

```
[Unit]
```

```
Description=Timer pour exécuter un instantané BTRFS tous les vendredis
```

```
[Timer]
```

```
OnCalendar=daily
```

```
Persistent=true
```

```
[Install]
```

```
WantedBy=timers.target
```

Ensuite, on relance, active et démarre le service systemd.

```
sudo systemctl daemon-reload
sudo systemctl start btrfs_friday_snapshot.timer
sudo systemctl enable btrfs_friday_snapshot.timer
```

On peut contrôler qu'il soit bien lancé avec la commande :

```
systemctl list-timers
```

## Suppression programmé des instantanés

Après avoir lancer une automatisation pour créer des instantanés automatiquement, il est intéressant de pouvoir supprimer automatiquement les plus ancien.

## Script

```
#!/bin/bash

# Répertoire où se trouvent les instantanés BTRFS
SNAPSHOT_DIR="/btrfs" # Modifiez ce chemin selon votre configuration

# Nombre d'instantanés à conserver
KEEP_SNAPSHOTS=10 # Modifiez cette valeur pour ajuster le nombre d'instantanés à conserver
```

```
# Compte le nombre d'instantanés
SNAPSHOT_COUNT=$(btrfs subvolume list "$SNAPSHOT_DIR" | awk '/@-/ {print $9}' | wc -l)

# Si le nombre d'instantanés dépasse la limite, supprime les plus anciens
if [ "$SNAPSHOT_COUNT" -gt "$KEEP_SNAPSHOTS" ]; then
    # Liste les instantanés, trie par date et supprime les plus anciens
    btrfs subvolume list "$SNAPSHOT_DIR" | awk '/@-/ {print $9}' | sort | head -n -"$KEEP_SNAPSHOTS" | xargs -r -
    I {} btrfs subvolume delete "$SNAPSHOT_DIR/{"
fi
```

A noter :

**SNAPSHOT\_DIR** correspond au répertoire des instantanés.

**KEEP\_SNAPSHOTS** correspond au nombre d'instantanés à conserver.

## Programation

Voici les fichiers à créer dans `/etc/systemd/system/`

`manage_delete_old_btrfs_snapshot.service`

```
[Unit]
Description=Gestionnaire d'instantanés BTRFS

[Service]
Type=oneshot
ExecStart=/root/manage_delete_old_btrfs_snapshot.sh

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

`manage_delete_old_btrfs_snapshot.timer`

```
[Unit]
Description=Timer pour le gestionnaire d'instantanés BTRFS

[Timer]
OnBootSec=10min
#OnUnitActiveSec=30min
Persistent=true
```

```
[Install]
```

```
WantedBy=timers.target
```

Activez et démarrez le timer avec les commandes suivantes :

```
sudo systemctl enable manage_delete_old_btrfs_snapshot.timer
```

```
sudo systemctl start manage_delete_old_btrfs_snapshot.timer
```

Vous pouvez vérifier le statut du timer et du service avec les commandes suivantes :

```
sudo systemctl status manage_delete_old_btrfs_snapshot.timer
```

```
sudo systemctl status manage_delete_old_btrfs_snapshot.service
```

# RAID

Dans cette section, prenez en compte le fait qu'elle est écrite dans le sens où vous avez déjà une bonne base du terminal et que vous connaissez les commandes `lsblk`, `blkid`, `fdisk` ... en rapport avec la gestion des disques.

Voici les différents RAID proposés par BTRFS ([source](#)) :

- single: Une seule copie des informations. 100% de l'espace disque est utilisé. Le second disque ne sera utilisé que lorsque le premier est rempli. Le troisième que lorsque les deux premiers seront remplis.
- dup: Chaque disque contient les données en double.
- raid0: Une seule copie des informations. 100% de l'espace disque est utilisé. Tous les disques seront utilisés simultanément.
- raid1: Copie en double. 50% de l'espace disque est utilisé. Un disque peut tomber en panne.
- raid1c3: Copie en triple. 33% de l'espace disque est utilisé. Deux disques peuvent tomber en panne.
- raid1c4: Copie en quadruple. 25% de l'espace disque est utilisé. Trois disques peuvent tomber en panne.
- raid10: Mélange de RAID1 et RAID0.
- raid5: Une copie et une copie de parité distribuée. Un disque peut tomber en panne. L'espace disque utilisable est  $100 \times (\mathbf{N}-1) / \mathbf{N} \%$ . **N** étant le nombre de disques. **Ce mode n'est pas encore stable, donc pas conseillé.**
- raid6: Une copie et deux copies de parité distribuées. Deux disques peuvent tomber en panne. L'espace disque utilisable est  $100 \times (\mathbf{N}-2) / \mathbf{N} \%$ . **N** étant le nombre de disques.

## Création

Nous allons d'abord créer le système de fichier BTRFS sur chaque disque qui fera partie du RAID.

Dans l'exemple, je souhaite faire un RAID1 avec deux disques dur `SDB` et `SDC`.

```
sudo mkfs.btrfs /dev/sdb /dev/sdb
```

Je vais maintenant monter mon premier disque dans le point de montage créé préalablement, `/raid01` pour moi.

```
sudo mount /dev/sdb /raid01
```

Je rajoute désormais le deuxième disque à ce point de montage.

```
sudo btrfs device add /dev/sdc /raid01
```

À présent, on lance l'équilibrage et on indique le type de RAID.

```
sudo btrfs balance start -dconvert=raid1 -mconvert=raid1 /raid01
```

Vérification.

```
sudo btrfs filesystem show /raid01
```

Voici un résultat normal :

```
Label: none  uuid: 6f27cdf0-f9e8-4577-a1a8-541598274825
  Total devices 2 FS bytes used 8.00GiB
  devid    1 size 20.00GiB used 9.28GiB path /dev/vdb
  devid    2 size 20.00GiB used 9.28GiB path /dev/vdc
```

Un résultat avec problème sur `/dev/vdc` :

```
Label: none  uuid: 6f27cdf0-f9e8-4577-a1a8-541598274825
  Total devices 2 FS bytes used 8.00GiB
  devid    1 size 20.00GiB used 9.28GiB path /dev/vdb
  devid    3 size 0 used 0 path /dev/vdc MISSING
```

## Remplacement d'un disque défectueux

La première étape est de changer physiquement le disque défectueux.

Une fois fait, il faut démonter le raid avec la commande `umount`, puis remonter en mode dégradé en lecture/écriture.

```
sudo moumount -o degraded,rw /raid01
```

Ajouter le disque dans le RAID, dans l'exemple le nouveau disque est `sdd`.

```
sudo btrfs device add /dev/sdd /raid01 -f
```

Puis, on retire le disque absent du RAID.

```
sudo btrfs device delete missing /raid01
```

Puis, on relance l'équilibrage des données.

```
sudo btrfs balance start -usage=50 /raid01
```

## Sauvegarde/restauration d'instantané

Le but est de pouvoir sauvegarder et restaurer un sous-volume à chaud, le principe est de créer un instantané en lecture seul et de l'envoyer dans un fichier.

### Sauvegarde

Pour créer une sauvegarde, il faut d'abord créer un instantané en lecture seul :

```
sudo btrfs subvolume snapshot -r @ @-save
```

L'option `-r` permet de créer l'instantané en `RO` (lecture-seul), une autre solution pour un instantané déjà existant est :

```
sudo btrfs property set -f /btrfs/@-save ro true
```

Pour vérifier le status

```
btrfs property get -ts /btrfs/@-save
```

Avec la commande la fonction `send`, on peut exporter l'instantané dans un fichier :

```
sudo btrfs send @-save > /chemin/vers/la/destination/@-save.dump
```

### Restauration

Voici la commande pour restaurer, `/btrfs` étant le point de montage global de la partition BTRFS :

```
cat /chemin/vers/la/destination/@-save.dump | sudo btrfs receive /btrfs
```

Variante pour le même résultat.

```
sudo btrfs receive -v /btrfs < /chemin/vers/la/destination/@-save.dump
```

L'option `-v` est là pour voir le détail (verbeux).

Mettre l'instantané en `RW` (Lecture-Écriture) :

```
sudo btrfs property set -f /btrfs/@-save ro false
```

Pour vérifier le status

```
btrfs property get -ts /btrfs/@-save
```

## Conversion d'un système de fichier existant en BTRFS.

Voici une retranscription des étapes que l'on a faites sur un PC lors de l'atelier Linux.

### Image disque

La première étape est de créer une image disque en cas de corruption du système de fichier lors de la conversion.

Nous avons utilisé Rescuzilla pour effectuer cette opération.

### Contrôle et réparation du système de fichier

Dans la [documentation de BTRFS](#), il est conseillé de faire une vérification et réparation du système de fichier existant.

Les opérations suivantes vont se faire via un LiveCD dans notre cas, nous avons utilisé un live de Linux Mint.

Nous avons fait la commande `lsblk` pour identifier la partition que nous voulons convertir, dans notre cas c'est sda2.

Nous avons lancé la commande : `sudo e2fsck -fvy /dev/sda2`

### Conversion

Nous avons lancé la conversion avec la commande suivante :

```
sudo btrfs-convert /dev/sda2
```

Avec 200 Go de données, l'opération a mis moins de 30 mins sur un disque SSD.

## Création des sous-volumes

Maintenant que la conversion est terminée, nous allons créer un sous-volume `@` et `@home`.

Nous avons monté la partition et ouvert un terminal à la racine de cette partition fraîchement convertis.

Nous avons créé les sous-volumes.

```
sudo btrfs subvolume create @  
sudo btrfs subvolume create @home
```

Ensuite, nous avons déplacé le contenu de la racine dans le sous-volume `@` avec la commande `mv`, ont à trouver une astuce pour exclure `@` et `@home` pour les laisser en lieu et place.

```
sudo mv $(ls --ignore=@ --ignore=@home) @
```

Puis, nous avons fait de même pour le home, par contre à ce stade, le home se trouve dans `@`.

```
cd @  
sudo mv home/* /media/live/aa78bd8c-d810-4059-b214-2be46e2e9bde/@home/
```

## Modification du FSTAB

Maintenant que nous avons fait tous les mouvements, nous allons mettre à jour le fstab.

Nous allons repérer la ligne de la racine qui est encore renseignée en EXT4 avec l'ancien UUID :

```
UUID=fd0587f2-d0d0-404c-b4c3-aa358b8dacd6 / ext4 noatime 0 1
```

Nous allons la modifier et ajouter une ligne pour le sous-volume `@home`.

Nous obtenons le nouveau UUID grâce à la commande : `sudo blkid`.

```
UUID=aa78bd8c-d810-4059-b214-2be46e2e9bde / btrfs subvol=/@,defaults 0 0  
UUID=aa78bd8c-d810-4059-b214-2be46e2e9bde /home btrfs subvol=@home,defaults 0 0
```

## Mise à jour de GRUB.

Pour mettre à jour grub, nous avons utilisé l'outil Boot-repair qui est préinstallé dans le live de Linux Mint.

Tout c'est bien déroulé, le PC à démarrer correctement ☑.

Pas encore tester avec `systemd-boot`.

Sources :

<https://www.deltasight.fr/remplacer-disque-raid1-btrfs/>

<https://debian-facile.org/doc:systeme:btrfs#raid-0-1-et-10>

[https://doc.ubuntu-fr.org/btrfs#le\\_raid\\_de\\_disques](https://doc.ubuntu-fr.org/btrfs#le_raid_de_disques)

[https://docs.redhat.com/fr/documentation/red\\_hat\\_enterprise\\_linux/7/html/storage\\_administration\\_guide/btrfs-integrated\\_volume\\_management#btrfs-integrated\\_volume\\_management](https://docs.redhat.com/fr/documentation/red_hat_enterprise_linux/7/html/storage_administration_guide/btrfs-integrated_volume_management#btrfs-integrated_volume_management)

# Scripts BTRFS utiles

Instantanés sans contrôle du jour de la semaine.

```
#!/bin/bash

# Définir le chemin où les instantanés sont stockés
SNAPSHOT_DIR="/btrfs"
INDEX_FILE="/chemin/vers//btrfs_kernel.txt"
ENTRIES_DIR="/chemin/vers/.snapshots_btrfs"
SUDO_USER="utilisateur du système"

# Vérifie si un instantané a déjà été créé aujourd'hui
if ls ${SNAPSHOT_DIR}/@-$(date +%d-%m-%Y)* 1> /dev/null 2>&1; then
    echo "Un instantané a déjà été créé aujourd'hui."
    exit 0
fi

# Crée un nouvel instantané
btrfs subvolume snapshot ${SNAPSHOT_DIR}/@ ${SNAPSHOT_DIR}/@-$(date +%d-%m-%Y_%H-%M-%S)

# Garde un historique du noyau utilisé lors de la création de cet instantané
echo "@-$(date +%d-%m-%Y_%H-%M-%S) kernel $(uname -a | awk '{print $3}')" >> ${INDEX_FILE}
chown 1000:1000 ${INDEX_FILE}
```



```
# Partie spécifique à Solus, récupère les entrées de systemd-boot
clr-boot-manager mount-boot
cp -r /boot/loader/entries ${ENTRIES_DIR}/entries-$(uname -a | awk '{print $3}')
umount /boot

# Notification de création
sudo -u "${SUDO_USER}" DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/${id -u "${SUDO_USER}"} /bus
notify-send -a "Instantané BTRFS" "Instantané @-$(date +%d-%m-%Y_%H-%M-%S) créé"

# Ajoutez d'autres commandes ici
```

---

Revision #42

Created 4 February 2025 19:22:27 by Julien

Updated 4 June 2025 14:39:34 by Julien