Configuration de gocryptfs avec pCloud et Montage Automatique

CAZANOVE Romain / CHAN-THIEN-LONG Tao

January 30, 2025

1 Introduction

Ce document décrit les étapes pour configurer `gocryptfs` avec `pCloud` et activer le montage automatique au démarrage, même en cas d'échec.

2 Partie 1 : Configuration de `gocryptfs` avec redémarrage en cas d'échec

2.1 Étape 1 : Installation de `gocryptfs`

1. Mettez à jour les paquets et installez gocryptfs :

```
sudo apt-get update
sudo apt-get install gocryptfs
```

2. Question 1 : Quelle est la commande pour vérifier si gocryptfs est correctement installé ?
Réponse : gocryptfs -h

2.2 Étape 2 : Configuration initiale de `gocryptfs`

1. Créez un répertoire pour contenir les fichiers chiffrés dans votre répertoire pCloud :

```
mkdir ~/pCloudDrive/.PersoPersoEncrypted
```

2. Initialisez le répertoire chiffré :

```
gocryptfs -init ~/pCloudDrive/.PersoPersoEncrypted
```

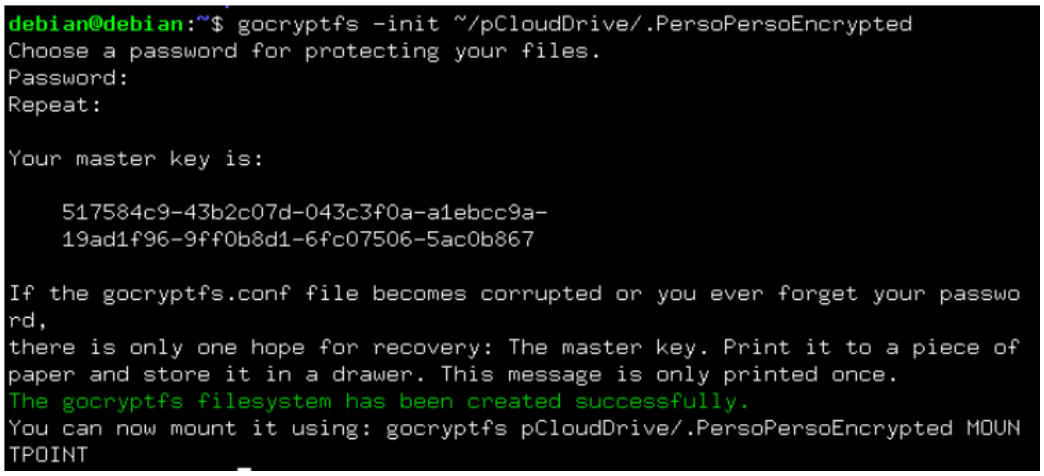


Figure 1: Capture d'écran de l'initialisez le répertoire chiffré

3. Montez le répertoire chiffré sur le répertoire cible :

```
gocryptfs ~/pCloudDrive/.PersoPersoEncrypted ~/home/user/
    PersoPerso
```

4. Question 2 : Quelle est la commande pour démonter un répertoire chiffré monté avec gocryptfs ?
Réponse: fusermount -u /chemin/vers/repertoire/dechiffré

2.3 Étape 3 : Créer un fichier de mot de passe

1. Créez un fichier contenant le mot de passe pour gocryptfs et assurez-vous qu'il est sécurisé :

```
echo "azerty" > /home/debian/.gocryptfs_password
chmod 600 /home/debian/.gocryptfs_password
```

2. Question 3 : Pourquoi est-il important de sécuriser le fichier de mot de passe avec chmod 600 ?
Réponse : Pour faire en sorte que le propriétaire à les seuls droits sur le fichier

2.4 Étape 4 : Créer une unité `systemd`

1.Créez ou modifiez le fichier d'unité systemd pour gérer le montage de gocryptfst :

```
sudo nano /etc/systemd/system/gocryptfs-perso.service
```

Figure 2: Capture d'ecran de l'untié system

2.Rechargez les unités systemd et activez le service :

```
sudo systemctl daemon-reload
sudo systemctl enable gocryptfs-perso.service
sudo systemctl start gocryptfs-perso.service
```

2.5 Étape 5 : Vérifier l'état du service

```
systemctl status gocryptfs-perso.service
```

2.6 Étape 6 : Redémarrer le système

```
sudo reboot
ls /home/debian/PersoPerso
```

3 Partie 2 : Configuration avec continuation en cas d'échec

3.1 Étape 1 : Création d'un script pour demander le mot de passe

Créez un script :

```
sudo nano /usr/local/sbin/mount_gocryptfs.sh
```

Ajoutez-y le contenu suivant :

```
#!/bin/bash
# Demander le mot de passe
echo "Entrez le mot de passe pour gocryptfs :"
read -s PASSWORD

# Verifier si le mot de passe est vide
if [ -z "$PASSWORD" ]; then
    echo "Le mot de passe ne peut pas etre vide."
    exit 1
fi

# Monter le repertoire chiffre
echo "$PASSWORD" | /usr/bin/gocryptfs --extpass "echo $PASSWORD" /home/debian/pCloudDrive/.PersoPersoEncrypted /home/debian/PersoPerso

#   Verifier si le montage a reussi
if mountpoint -q /home/debian/PersoPerso; then
    echo "Montage reussi."
else
    echo "Echec du montage."
    exit 1
fi
```

Rendez le script exécutable :

```
sudo chmod +x /usr/local/sbin/mount_gocryptfs.sh
```

3.2 Étape 2 : Créer une unité systemd pour la demande de mot de passe

Créez et éditez le fichier suivant :

```
sudo nano /etc/systemd/system/gocryptfs-perso.service
```

Ajoutez-y le contenu suivant :

```
[Unit]
Description=Mount gocryptfs filesystem for PersoPerso
After=network.target home.mount
Wants=home.mount

[Service]
Type=idle
```

```
ExecStart=/usr/local/sbin/mount_gocryptfs.sh
User=debian
Group=debian
StandardInput=tty-force
StandardOutput=journal
RemainAfterExit=true
Restart=on-failure
RestartSec=5s
StartLimitInterval=0
StartLimitBurst=3

[Install]
WantedBy=default.target
```

Rechargez et démarrez le service :

```
sudo systemctl daemon-reload
sudo systemctl enable gocryptfs-perso.service
sudo systemctl start gocryptfs-perso.service
```

3.3 Étape 3 : Redémarrer le système

```
sudo reboot
systemctl status gocryptfs-perso.service
```

Après le redémarrage, vérifiez les journaux pour confirmer que le système continue de démarrer même en cas de mot de passe incorrect :

```
sudo journalctl -u gocryptfs-perso.service
```

TP2: Créer et chiffrer une partition sous Debian

4 Partie 1 : Préparation de la partition

4.1 Identification des disques et partitions

Pour identifier le disque sur lequel vous souhaitez créer une partition, utilisez la commande suivante :

```
lsblk
```

Cette commande fournit une liste des disques et partitions disponibles sur le système, comme illustré ci-dessous :

```
NAME    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda       8:0    0   20G  0 disk
|-sda1    8:1    0   19G  0 part /
|-sda2    8:2    0    1K  0 part
|-sda5    8:5    0  975M  0 part
sr0      11:0    1   51M  0 rom
```

Interprétation :

- `sda` : Représente le disque principal (20 Go).
- `sda1` : Partition principale montée sur `/` (19 Go).
- `sda2` : Partition de taille minimale (1 Ko).
- `sda5` : Une partition supplémentaire (975 Mo).
- `sr0` : Lecteur optique (51 Mo).

4.2 Création d'une nouvelle partition

Après avoir identifié le disque, créez une nouvelle partition en utilisant la commande `fdisk` :

```
fdisk /dev/sdX
```

Remplacez `/dev/sdX` par le nom de votre disque (par exemple `/dev/sda`). Lors de l'exécution, vous verrez un message semblable à :

```
Welcome to fdisk (util-linux 2.38.1).
Changes will remain in memory only, until you decide to write
   ↪  them.
Be careful before using the write command.

This disk is currently in use - repartitioning is probably a
   ↪ bad idea.
It's recommended to umount all file systems, and swapoff all
   ↪ swap
partitions on this disk.
```

Interprétation : Ce message rappelle que toute modification sera temporaire tant que vous n'utilisez pas la commande `w` pour "ecrire" les modifications. Il conseille egalement de démonter les systèmes de fichiers et de désactiver les partitions swap avant de poursuivre.

5 Initialisation du chiffrement avec LUKS

5.1 Installation de `cryptsetup`

Pour installer le paquet `cryptsetup`, exécutez les commandes suivantes :

```
[root@machina]# apt update
[root@machina]# apt install cryptsetup
```

Listing 1: Installation de cryptsetup

5.2 Initialisation du chiffrement

Pour initialiser le chiffrement sur une partition, utilisez la commande suivante en remplaçant `/dev/sdX1` par le chemin de votre partition :

```
[root@machina]# cryptsetup luksFormat /dev/sdX1

WARNING!
========
This will overwrite data on /dev/sda3 irrevocably.

Are you sure? (Type 'YES' in capital letters): YES
Enter passphrase for /dev/sda3:
Verify passphrase:
```

Listing 2: Initialisation du chiffrement LUKS

Note importante : Si la partition dépasse une taille de `16777216 bytes`, le chiffrement ne pourra pas être effectué. Il sera alors nécessaire de redimensionner la partition avec des outils comme `gparted`.

5.3 Déverrouillage de la partition

Après avoir configuré le chiffrement, vous devez déverrouiller la partition avec un nom d'identifiant. Voici la commande correspondante :

```
[root@machina]# cryptsetup open /dev/sdX1
    ↪ crypted_partition
```

Listing 3: Déverrouillage de la partition

L'utilisateur doit entrer la phrase de passe définie lors de l'étape précédente pour déverrouiller la partition.

5.4 Formatage de la partition

Une fois la partition chiffrée et déverrouillée, il est nécessaire de la formater pour la rendre utilisable. Voici un exemple de formatage en ext4 :

```
sudo mkfs.ext4 /dev/mapper/crypted_partition
mke2fs 1.47.0 (5-Feb-2023)
Creating filesystem with 2048 k blocks and 256 inodes
Allocating group tables: done
Writing inode tables: done
Creating journal (1024 blocks): done
Writing superblocks and filesystem accounting information
    ↪ : done
```

Listing 4: Formatage de la partition en ext4

6 Questions et interprétations

6.1 Pourquoi est-il important de ne pas oublier la phrase de passe LUKS ?

La phrase de passe LUKS est essentielle pour accéder aux données chiffrées. Si elle est oubliée :

- Toutes les données deviennent inaccessibles.

- Il n'existe pas de méthode standard pour récupérer les données sans cette phrase de passe.
- Une réinitialisation ou un formatage de la partition entraînera une perte complète des données.

6.2 Pourquoi est-il nécessaire de formater la partition après chiffrement ?

Après le chiffrement, le système de fichiers précédent devient illisible. Le formatage est donc requis pour :

- Créer un système de fichiers compatible pour stocker les données.
- Supprimer les éventuelles données résiduelles.
- Garantir la fonctionnalité et la sécurité de la partition.

Partie 4 : Configuration pour le montage automatique au démarrage

Pour configurer le montage automatique d'une partition chiffrée au démarrage, suivez les étapes ci-dessous.

Étape 1 : Obtenir l'UUID de la partition chiffrée

Utilisez la commande suivante pour identifier l'UUID de la partition chiffrée. Remplacez `/dev/sdX1` par le chemin de votre partition.

```
[root@machine]# blkid /dev/sdX1
```

Listing 5: Obtenir l'UUID avec blkid

Exemple de résultat :

```
root@vraisae:/home/caza# sudo blkid /dev/sda3
/dev/sda3: UUID="fab5c707-d31d-4ade-a215-c0b174cabff8"
    ↪ TYPE="crypto_LUKS"
```

Explication : La commande `blkid` affiche des informations sur les partitions, y compris leur `UUID` (identifiant unique), leur type (ici `crypto_LUKS`), et d'autres métadonnées.

Étape 2 : Modifier le fichier `/etc/crypttab`

Ouvrez le fichier `/etc/crypttab` avec un éditeur de texte, par exemple `nano` :

```
[root@machine]# nano /etc/crypttab
```

Listing 6: Modifier /etc/crypttab

Ajoutez la ligne suivante en remplaçant `UUID=xxxx-yyyy-zzzz-aaaa` par l'UUID obtenu à l'étape précédente :

```
crypted_partition  fab5c707-d31d-4ade-a215-c0b174cabff8=
  ↪ xxxx-yyyy-zzzz-aaaa none luks
```

Listing 7: Ligne à ajouter dans /etc/crypttab

Explication :

- `crypted_partition` : Nom logique de la partition chiffrée (vous pouvez le personnaliser).
- `UUID=xxxx-yyyy-zzzz-aaaa` : L'UUID de la partition chiffrée.
- `none` : Indique qu'aucune clé externe n'est utilisée (un mot de passe sera demandé si nécessaire).
- `luks` : Spécifie que la partition utilise le chiffrement LUKS.

Étape 3 : Configurer le montage dans `/etc/fstab`

Enfin, pour monter automatiquement la partition une fois déchiffrée, assurez-vous que le fichier `/etc/fstab` contient une entrée pour le périphérique déchiffré (souvent monté sous `/dev/mapper/crypted_partition`).

```
nano /etc/fstab
```

Listing 8: Modifier /etc/fstab

```
/dev/mapper/crypted_partition /mnt ext4 defaults 0 0
```

Listing 9: Modifier /etc/fstab

Explication :

- `/dev/mapper/crypted_partition` : Chemin vers le périphérique déchiffré.
- `/mnt` : Point de montage où la partition sera accessible.
- `ext4` : Type du système de fichiers (à ajuster selon votre configuration).
- `defaults 0 0` : Options par défaut pour le montage.

Question 4 : Utilité de l'UUID dans la configuration de `/etc/crypttab` et `/etc/fstab`

L'**UUID** (Universal Unique Identifier) est utilisé dans les fichiers de configuration comme `/etc/crypttab` et `/etc/fstab` pour identifier de manière unique une partition ou un volume, indépendamment de son emplacement physique (par exemple, le nom du périphérique tel que `/dev/sda1`).

Cela garantit une configuration stable et fiable, même si l'ordre des périphériques change lors du démarrage ou si de nouveaux disques sont ajoutés. En utilisant l'UUID, on évite les erreurs potentielles liées à la modification des noms de périphériques, assurant ainsi que les partitions ou volumes chiffrés et montés sont toujours correctement associés et configurés.

Partie 5 : Redémarrage et tests de la configuration

Après avoir configuré les fichiers `/etc/crypttab` et `/etc/fstab`, il est nécessaire de redémarrer le système pour tester la configuration. Vous devriez être invité à entrer la phrase de passe pour déverrouiller la partition chiffrée.

Pour vérifier que la partition est correctement montée, utilisez l'une des commandes suivantes :

```
[root@machine]# lsblk
```

Listing 10: Vérifier le montage avec lsblk

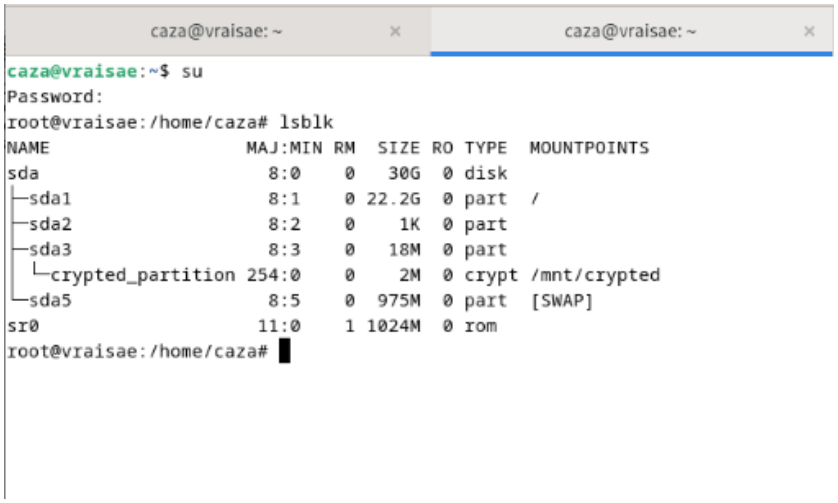
Explication : La commande `lsblk` affiche une vue hiérarchique des périphériques de stockage et de leurs partitions, avec les points de montage associés. Si la partition déchiffrée est montée correctement, elle apparaîtra comme un sous-périphérique (par exemple, `/dev/mapper/crypted_partition`) avec un point de montage (colonne `MOUNTPOINT`).

Vous pouvez également utiliser la commande suivante pour des informations supplémentaires :

```
[root@machine]# df -h
```

Listing 11: Vérifier avec df

Explication : La commande `df -h` liste les systèmes de fichiers montés avec leurs tailles et leurs points de montage. Assurez-vous que la partition déchiffrée est bien listée avec le point de montage attendu.



Question 5

Quelle commande utilisez-vous pour vérifier que la partition chiffrée est bien montée ? Justifiez votre choix.

Pour vérifier qu'une partition chiffrée est bien montée, la commande `lsblk` est particulièrement adaptée. Elle permet d'afficher une vue hiérarchique des périphériques de stockage, indiquant clairement les partitions, leurs volumes déchiffrés (comme ceux gérés par LUKS) et leurs points de montage.

Cette commande est idéale pour les systèmes chiffrés, car elle montre le lien entre le périphérique brut (par exemple, `/dev/sda1`) et le volume logique déchiffré (comme `/dev/mapper/volume_chiffre`). Cela permet de confirmer que le processus de déchiffrement et de montage a été correctement effectué.

6.3 2.2.4.6. Partie 6 : Tests d'accès et démontage

Créez un fichier de test pour vérifier les permissions, puis démontez et verrouillez la partition:

```
[root@machina]# touch /mnt/crypted/testfile
[root@machina]# umount /mnt/crypted
[root@machina]# cryptsetup close crypted_partition
:root@vraisae:/home/caza# sudo touch /mnt/crypted/
   ↪ testfile
```

```
:root@vraisae:/home/caza# sudo umount /mnt/crypted
:root@vraisae:/home/caza# cryptsetup close
   ↪ crypted_partition
bash: cryptsetup: command not found
:root@vraisae:/home/caza# sudo cryptsetup close
   ↪ crypted_partition
:root@vraisae:/home/caza#
```

6.4 Question 6 : Que signifie verrouiller une partition avec `cryptsetup close` et pourquoi est-ce utile ?

Verrouiller une partition avec la commande `cryptsetup close` signifie désactiver l'accès au volume déchiffré associé à une partition chiffrée. Cette opération ferme le mapper logique créé lors du déchiffrement (par exemple, `/dev/mapper/volume_chiffre`), rendant les données inaccessibles tant que la partition n'est pas de nouveau déverrouillée avec la clé appropriée. Cette action est utile pour plusieurs raisons. Elle renforce la sécurité en empêchant tout accès non autorisé aux données sensibles lorsqu'elles ne sont pas nécessaires. Elle libère également les ressources système associées au volume déchiffré, comme les descripteurs de fichiers et les points de montage. Enfin, cela garantit que la partition redevient complètement chiffrée, offrant ainsi une protection immédiate en cas de perte ou de vol du support de stockage.

Conclusion et questions de synthèse

Vous avez appris à chiffrer une partition sous Debian, à la configurer pour un montage automatique, et à vérifier que tout fonctionne correctement.

Question de synthèse 1 : Quelle est l'importance du chiffrement d'une partition pour la sécurité des données ?

Question de synthèse 2 : En cas de perte de la phrase de passe, que se passerait-il pour les données stockées sur la partition chiffrée ?

Réponse aux questions de synthèse

Le chiffrement d'une partition est essentiel pour garantir la sécurité des données, en particulier dans un contexte où des informations sensibles doivent être protégées contre les accès non autorisés. En chiffrant une partition, les données qu'elle contient deviennent illisibles sans la phrase de passe ou la clé

appropriée, ce qui protège efficacement contre le vol, la perte de matériel ou l'intrusion.

Cependant, cette sécurité a une contrepartie importante : en cas de perte de la phrase de passe, il devient impossible d'accéder aux données stockées sur la partition chiffrée. Sans cette clé, les informations restent définitivement chiffrées et donc inutilisables, soulignant l'importance de sauvegarder la phrase de passe ou une clé de récupération dans un endroit sûr.

2.2.6. Chapitre : Vérifications et dépannage de LUKS

Question 1 :Pourquoi est-il important d'examiner les logs lorsque le montage automatique d'une partition chiffrée échoue ?

Il est important d'examiner les logs lorsque le montage automatique d'une partition chiffrée échoue, car ils fournissent des informations détaillées sur les erreurs survenues pendant le processus. Les journaux système, accessibles via des outils comme `journalctl` ou `dmesg`, permettent de repérer des problèmes spécifiques, tels qu'un mot de passe incorrect, un fichier de clé manquant, une mauvaise configuration des fichiers `crypttab` ou `fstab`, ou encore des dépendances non satisfaites au démarrage. En identifiant précisément la cause de l'échec, il devient possible de corriger le problème rapidement et efficacement, ce qui est essentiel pour garantir l'accès aux données chiffrées en toute sécurité.

Question 2 : Expliquez comment vous interprétez les résultats de `lsblk` pour vérifier que votre partition chiffrée est bien montée.

Pour vérifier qu'une partition chiffrée est bien montée à l'aide de la commande `lsblk`, il faut examiner la hiérarchie des périphériques affichée. Une partition chiffrée apparaîtra comme un périphérique parent avec un sous-périphérique correspondant à son déchiffrement. Par exemple, si la partition chiffrée est nommée `sda2`, vous devriez voir un sous-périphérique avec un nom logique (souvent défini lors de la configuration, comme `cryptroot`) indiqué comme étant monté dans un point de montage spécifique (par exemple `/mnt`, ou un autre chemin). La colonne `MOUNTPOINT` permettra de confirmer si la partition déchiffrée est montée, et la colonne `TYPE` indiquera des informations comme `crypt` pour la partition chiffrée et `part` ou `lvm` pour le type

de sous-périphérique. Si aucun point de montage n'est visible, cela indique que la partition n'est pas montée correctement.

Question 3 : Pourquoi est-il important de tester l'écriture et l'accès aux données avant de considérer une partition chiffrée comme opérationnelle ?

Il est important de tester l'écriture et l'accès aux données sur une partition chiffrée avant de la considérer comme opérationnelle pour garantir qu'elle fonctionne correctement et qu'aucune erreur n'interfère avec son utilisation. Cela permet de vérifier que le chiffrement et le déchiffrement des données se font sans problème, que les permissions sont bien configurées et que la configuration du système de fichiers est compatible avec les besoins. Ces tests permettent également de détecter des problèmes potentiels, comme une corruption du système de fichiers, des lenteurs excessives ou des erreurs dans les mécanismes de déchiffrement, afin de les corriger avant la mise en production. Ainsi, ils assurent l'intégrité, la sécurité et la fiabilité des données stockées.

Question 4 : Décrivez une méthode pour identifier la source d'un problème si la partition chiffrée ne se monte pas automatiquement au démarrage.

Pour identifier la source du problème lorsque la partition chiffrée ne se monte pas automatiquement au démarrage, commencez par examiner les journaux système à l'aide de la commande `journalctl -b` pour afficher les événements du dernier démarrage et recherchez des messages spécifiques en filtrant avec des mots-clés comme `cryptsetup` ou `mount`. Cela permettra de détecter des erreurs telles qu'un mot de passe incorrect, une clé manquante ou un problème lié à la configuration de `crypttab` ou `fstab`.

TP3 : Intégration des deux méthodes de chiffrement

2.3.1. Partie 1 : Préparation de la partition chiffrée avec LUKS

Commencez par créer une partition chiffrée en suivant les étapes du premier TP, utilisant `cryptsetup` et LUKS. Voici les commandes de base pour préparer et monter la partition chiffrée :

```
[root@machina]# lsblk
```

Explications des commandes et étapes

Étape 1 : Utilisation de `fdisk` pour gérer les partitions

La commande suivante permet de lister ou manipuler les partitions sur un disque spécifique.

```
[root@machina]# fdisk /dev/sdX
root@vraisae:/home/caza# sudo fdisk /dev/sda

Welcome to fdisk (util-linux 2.38.1).
Changes will remain in memory only, until you decide to
   ↪ write them.
Be careful before using the write command.

This disk is currently in use - repartitioning is
   ↪ probably a bad idea.
It's recommended to umount all file systems, and swapoff
   ↪ all swap
partitions on this disk.
```

Listing 12: Exécution de fdisk

Explication : Cette commande ouvre le disque spécifié (`/dev/sdX` ou `/dev/sda`) dans `fdisk`. Elle affiche un avertissement si le disque est actuellement utilisé, suggérant de démonter les systèmes de fichiers et de désactiver les partitions swap pour éviter tout problème.

Étape 2 : Création d'une partition chiffrée avec `cryptsetup` et LUKS

La commande suivante initialise une partition chiffrée avec le format LUKS.

```
[root@machina]# cryptsetup luksFormat /dev/sdX1
root@vraisae:/home/caza# sudo cryptsetup luksFormat /dev/
    ↪ sda3

WARNING!
========
This will overwrite data on /dev/sda3 irrevocably.

Are you sure? (Type 'yes' in capital letters): YES
Enter passphrase for /dev/sda3:
Verify passphrase:
root@vraisae:/home/caza#
```

Listing 13: Exécution de cryptsetup

Explication : La commande `cryptsetup luksFormat` initialise le chiffrement sur la partition spécifiée (`/dev/sdX1` ou `/dev/sda3`). - Un avertissement informe que toutes les données sur cette partition seront effacées de manière irréversible. - L'utilisateur doit confirmer en tapant `YES` en majuscules. - Ensuite, il est invité à entrer et à vérifier une passphrase qui sera utilisée pour le déchiffrement de la partition.

Étape 3 : Ouverture et formatage de la partition chiffrée

Dans cette étape, nous allons :

- Ouvrir la partition chiffrée en utilisant `cryptsetup`.
- Formater la partition avec le système de fichiers `ext4`.

Ouverture de la partition chiffrée

La commande suivante permet d'ouvrir une partition chiffrée en créant un point de montage logique. La passphrase utilisée dans cet exemple est `caza`.

Explication :

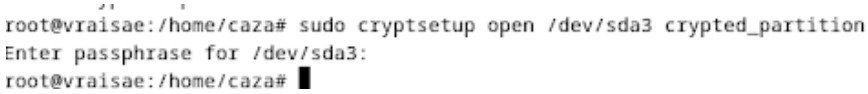


Figure 3: Exemple de commande `cryptsetup open`.

- `cryptsetup open /dev/sdX1 crypted_partition` : Ouvre la partition chiffrée située sur `/dev/sdX1` et la lie au point logique nommé `crypted_partition`.
- L'utilisateur est invité à entrer la passphrase qui ici est (`caza`) pour déchiffrer la partition.

Formatage de la partition avec ext4

Après avoir ouvert la partition chiffrée, nous devons la formater avec un système de fichiers compatible, comme `ext4`.

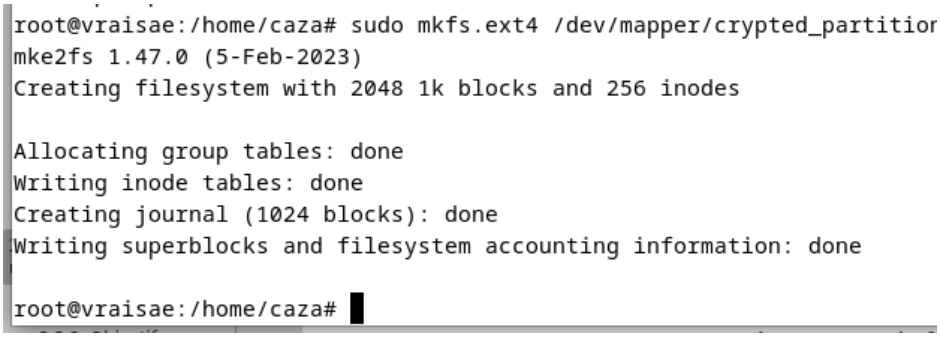


Figure 4: Exemple de commande `mkfs.ext4`.

Explication :

- `mkfs.ext4 /dev/mapper/crypted_partition` : Crée un système de fichiers `ext4` sur la partition logique déchiffrée située dans `/dev/mapper/crypted_partitio`
- La commande initialise la partition, alloue des tables de groupe, écrit les tables des inodes et crée un journal pour la gestion des métadonnées.

```
1 sudo mount /dev/mapper/crypted_partition /mnt/crypted
```

Listing 14: Commande pour monter une partition chiffrée

Cette commande monte une partition chiffrée localisée à `/dev/mapper/crypted_partition` sur le point de montage `/mnt/crypted`. Elle suppose que la partition chiffrée

a été préalablement déverrouillée avec une solution comme `cryptsetup` pour être accessible via `/dev/mapper`. Le répertoire `/mnt/crypted` doit exister et être prêt à accueillir la partition montée. Comme il n'y a pas eu d'erreur la partition est bien montée sur /mnt/crypted

Question : Pourquoi utiliser un chiffrement sur la partition avant d'y ajouter un autre niveau de chiffrement par répertoire ?

L'utilisation d'un chiffrement au niveau de la partition combiné à un chiffrement supplémentaire par répertoire répond à des besoins spécifiques en matière de sécurité et de gestion des données sensibles. Le chiffrement de la partition assure une protection globale, en sécurisant toutes les données qu'elle contient, y compris les métadonnées du système de fichiers comme les noms, tailles, et hiérarchies des fichiers.

Cependant, dans certains cas, un second niveau de chiffrement par répertoire est utile pour isoler des données particulièrement sensibles. Cela permet de gérer des accès différenciés : par exemple, un utilisateur peut avoir accès à certaines parties de la partition mais être restreint d'accéder à des répertoires spécifiques qui nécessitent un mot de passe ou une clé additionnelle.

Cette double protection renforce également la résilience contre des failles potentielles du chiffrement principal. Si, par exemple, la partition chiffrée est compromise, les données critiques dans les répertoires protégés restent inaccessibles sans la clé spécifique. Cette approche est donc particulièrement adaptée à des environnements exigeant une sécurité granulaire ou un compartimentage strict des données.

Partie 2 : Installation et préparation de gocryptfs

Installation de gocryptfs

Installez le paquet `gocryptfs` si ce n'est pas déjà fait :

```
[root@machina]# apt update
[root@machina]# apt install gocryptfs
```

Créez ensuite un dossier dans la partition chiffrée pour y stocker le répertoire chiffré gocryptfs :

```
[root@machina]# mkdir /mnt/crypted/gocryptfs_data
[root@machina]# mkdir /mnt/crypted/gocryptfs_mount
```

```
root@vraisae:/home/caza# sudo apt install gocryptfs
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  gocryptfs
0 upgraded, 1 newly installed, 0 to remove and 23 not upgraded.
Need to get 1,794 kB of archives.
After this operation, 8,405 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian bookworm/main amd64 gocryptfs amd64 2.3-1+b4
[1,794 kB]
Fetched 1,794 kB in 2s (997 kB/s)
Selecting previously unselected package gocryptfs.
(Reading database ... 174455 files and directories currently installed.)
Preparing to unpack .../gocryptfs_2.3-1+b4_amd64.deb ...
Unpacking gocryptfs (2.3-1+b4) ...
Setting up gocryptfs (2.3-1+b4) ...
```

Figure 5: Capture de l'installation de gocryptfs

```
root@vraisae:/home/caza# apt update
Hit:1 http://deb.debian.org/debian bookworm InRelease
Hit:2 http://deb.debian.org/debian bookworm-updates InRelease
Hit:3 http://security.debian.org/debian-security bookworm-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
23 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@vraisae:/home/caza# apt install gocrytfs
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

Figure 6: Capture de la mise à jour du terminal

```
root@vraisae:/home/caza# mkdir /mnt/crypted/gocryptfs_data
```

Figure 7: Creation du dossier `gocryptfs_data`

```
root@vraisae:/home/caza# mkdir /mnt/crypted/gocryptfs_mount
```

Figure 8: Creation du dossier `gocryptfs_mount`

```
root@vraisae:/home/caza# gocryptfs -init /mnt/crypted/gocryptfs_data
Choose a password for protecting your files.
Password:
Repeat:

Your master key is:

    271b5eb1-3eaa5b9d-de3435a6-a1e05b3d-
    20a03570-90c33132-42ae64fd-6d9223f7

If the gocryptfs.conf file becomes corrupted or you ever forget your password,
there is only one hope for recovery: The master key. Print it to a piece of
paper and store it in a drawer. This message is only printed once.
The gocryptfs filesystem has been created successfully.
You can now mount it using: gocryptfs /mnt/crypted/gocryptfs_data MOUNTPOINT
```

Figure 9: Initialisation

```
root@vraisae:/home/caza# gocryptfs /mnt/crypted/gocryptfs_data /mnt/crypted/gocryptfs_mount
Password:
Decrypting master key
Filesystem mounted and ready.
```

Figure 10: Affectation de du dossier chiffre à la partition chiffre

Partie 3 : Configuration du répertoire chiffré avec gocryptfs

Initialisez le répertoire chiffré gocryptfs à l'intérieur de la partition chiffrée :

```
[root@machina]# gocryptfs -init /mnt/crypted/
    ↪ gocryptfs_data
```

```
[root@machina]# gocryptfs /mnt/crypted/gocryptfs_data /
    ↪ mnt/crypted/gocryptfs_mount
```

Question 2 : Quel est l'intérêt d'ajouter un répertoire chiffré gocryptfs dans une partition déjà chiffrée par LUKS ?

Réponse :

L'ajout d'un répertoire chiffré gocryptfs dans une partition déjà chiffrée par LUKS permet de combiner les avantages des deux méthodes de chiffrement :

- **Chiffrement en deux couches :** Cela offre une sécurité renforcée, car les données sont chiffrées une première fois par LUKS au niveau du disque, puis une deuxième fois par `gocryptfs` au niveau des fichiers.

```
root@vraisae:/home/caza# touch /mnt/crypted/gocryptfs_mount/testfile
```

Figure 11: creation du fichier testfile

- **Gestion des fichiers individuels :** Avec `gocryptfs`, il est possible de chiffrer et de manipuler des fichiers spécifiques de manière flexible, ce qui n'est pas possible avec LUKS qui agit au niveau de la partition.
- **Portabilité :** Les fichiers chiffrés par `gocryptfs` peuvent être copiés et montés sur un autre système, contrairement à LUKS qui nécessite de déverrouiller l'intégralité de la partition.

Partie 4 : Tests et vérifications

Testez le bon fonctionnement du répertoire chiffré gocryptfs en créant un fichier de test dans `/mnt/crypted/ gocryptfs_mount` :

```
[root@machina]# touch /mnt/crypted/gocryptfs_mount/
    ↪ testfile
```

Ensuite, démontez le répertoire `gocryptfs` et la partition chiffrée pour vous assurer que les données sont bien protégées :

```
[root@machina]# fusermount -u /mnt/crypted/
    ↪ gocryptfs_mount
```

```
[root@machina]# umount /mnt/crypted
```

Question 3 : Décrivez l'intérêt des tests
Réponse :
Ces tests permettent de vérifier :

- que le montage du répertoire `gocryptfs` est correctement configuré et opérationnel ;
- que les données sont bien protégées lorsque le répertoire `gocryptfs` et la partition chiffrée sont démontés ;
- que la combinaison de `LUKS` et `gocryptfs` offre une solution de chiffrement robuste et fonctionnelle.

```
root@vraisae:/home/caza# fusermount -u /mnt/crypted/gocryptfs_mount
root@vraisae:/home/caza# umount /mnt/crypted
```

Figure 12: Démontage du dossier gogryptfs