Créer un conteneur/coffre-fort chiffré

Introduction

L'idée ici est de créer un petit fichier coffre-fort de 20 Mo, mais rien n'empêche évidemment d'utiliser un fichier de plusieurs Go suivant vos besoins.

Notre coffre-fort sera remplit au départ de nombre aléatoires, et servira à stocker des dossiers et fichiers confidentiels.

On utilisera ici un cryptage fort, avec un mot de passe très long, et surtout : une clé décalée dans le conteneur, pour accroître la difficulté à casser le coffre-fort via une attaque par dictionnaire.

Bien entendu, rien n'empêche ensuite de mettre un coffre dans le coffre, puis un coffre dans le coffre dans le coffre, etc.

À noter l'on peut aussi utiliser cette technique pour chiffrer des partitions entières non formatées via ***/etc/crypttab***...

Création du conteneur

1. Créez un dossier de travail :

```
mkdir chambreforte
```

2. Allez dans la chambre forte :

```
cd chambreforte
```

3. Créez le conteneur (taille à adapter suivant vos besoins…) :

```
dd if=/dev/urandom of=coffre1.dat bs=1M count=20
```

4. Recherchez le prochain périphérique de bouclage libre :

```
FREE_LOOP=$(losetup -f)
```

5. Liez le périphérique de bouclage avec le conteneur :

```
losetup ${FREE_LOOP} coffre1.dat
```

6. Vérifiez le résultat :

```
losetup -a
```

7. Donnez en ligne le décalage de la clé (multiple de 512 octets). On prendra ici un décalage de **20**, vu le petit fichier... :

```
read -p "Entrez le décalage : " OFFSET
```

8. Vérifiez le résultat :

```
echo "Décalage : $OFFSET"
```

9. Ouvrez le conteneur en donnant un mot de passe complexe et surtout long ! Notez que par défaut, `cryptsetup` utilise les options `-c aes-xts-plain64 -s 512 -h sha256` :

```
cryptsetup open --type=plain -o ${OFFSET} ${FREE_LOOP} moncoffre1
```

10. On vire la variable OFFSET de la mémoire par sécurité :

```
unset OFFSET
```

11. Vérifiez le résultat :

```
cryptsetup status moncoffre1
ls -l /dev/mapper/moncoffre1
df
lsblk --fs
```

12. Si tout est ok, formatez le conteneur (remarquez l'absence de phase de partitionnement) :

```
mkfs.ext4 /dev/mapper/moncoffre1
```

13. Montez le conteneur chiffré :

```
mount /dev/mapper/moncoffre1 /mnt
```

14. Créez un petit fichier de test pour remplissage :

```
echo "coucou" > /mnt/foret.txt
cat /mnt/foret.txt
```

15. Démontez le conteneur chiffré :

```
umount /mnt
```

16. Vérifiez le résultat :

```
df
lsblk --fs
```

17. Fermez le conteneur :

```
cryptsetup close moncoffre1
cryptsetup status moncoffre1
```

18. Retirez le périphérique de bouclage :

```
losetup -d ${FREE_LOOP}
```

19. Revenez au dossier parent :

```
cd ..
```

Réouverture du conteneur

Dans cette seconde phase, la logique est la même, sauf qu'on ne formate pas le conteneur, sinon on perd les données...

```
cd chambreforte
FREE_LOOP=$(losetup -f)
losetup ${FREE_LOOP} coffre1.dat
losetup -a
read -p "Entrez le décalage : " OFFSET
cryptsetup open --type=plain -o ${OFFSET} $
{FREE_LOOP} moncoffre1
unset OFFSET
cryptsetup status moncoffre1
mount /dev/mapper/moncoffre1 /mnt
cat /mnt/foret.txt
```

Démontage final

```
umount /mnt
cryptsetup close moncoffre1
losetup -d ${FREE_LOOP}
cd ..
```

Conclusion

GNU/Linux possède un outil particulièrement performant et souple pour créer des conteneurs chiffrés, qu'ils soient sous forme de fichier coffre-fort, de partition entière, de disque dur autonome, voire de grappes RAID !

Une utilisation classique de `cryptsetup` est le chiffrement de sa partition ***/home*** pour protéger ses données en cas de vol. À noter que le chiffrement est « transparent » côté performances de la

machine, et ne pose donc pas de problèmes particuliers, même sur des matériels anciens.

Pour chiffrer la partition racine en revanche, la méthode ici proposée n'est pas complètement adaptée pour deux raisons :

- ➔ GRUB a besoin d'accéder à un ***/boot*** visible pour charger le noyau et le disque virtuel de démarrage (*RAMFS*), ce qui nécessite donc de prévoir une partition non chiffrée dédiée au ***/boot***.
- ➔ `cryptsetup` utilise les fichiers ***/etc/crypttab*** et ***/etc/fstab*** pour ouvrir les conteneurs chiffrés au démarrage de la machine. Là encore, si ***/etc*** n'est pas accessible au démarrage, le montage ne fonctionnera pas...

Maintenant si vous cherchez un outil permettant de gérer des disques durs de sauvegarde chiffrés accessibles depuis plusieurs systèmes d'exploitation, `veracrypt` est un outil intermédiaire en mode graphique plutôt intéressant, fonctionnant sous les 3 OS majeurs.