Exemple concret d'installation de debian en Full Disk Encryption avec une machine en efi.

F.S. G.

Création le 24 Octobre 2022
Compilation du 26 octobre 2022.

Résumé

Ce document contient les commandes saisies lors de la dernière installation d'un disque dur externe comme disque système en vue de son utilisation sur les ordinateurs soit du travail, soit syndicaux.

1 La préparation du support

1.1 Formatage du disque cible

`fdisk /dev/sdc`

objectif. Créer deux partitions sur le disque :
- un disque dont la table de partition sera de type gpt,
- une partition type EFI de 64 Mo – le test à 32M montre des problèmes possibles d'après fdisk – qui va être la partition démarrable,
- une partition type Linux qui sera tout le reste et va accueillir le système.

Évidemment si le disque a déjà accueilli une table de partition antérieure, il va être demandé à plusieurs occasions de confirmer l'action destructrice ou créatrice de table de partitions ou de partitions.

La table de partitions. La table de partitions de type GPT.
g

Partition EFI. Suite de touches pour la création de la partition EFI d'une taille de 64 Mo.
n puis Enter Enter +64M Enter t 1

La partition Linux. Suite de touches pour la création de la partition Linux occupant le reste du disque.
n, Enter, Enter, Enter, Enter,

Sortie. Sortie de `fdisk`, enregistrement et synchronisation.
w

1.2 Création de la partition chiffrée.

Création de la partition chiffrée type Luks version 1 [1] en utilisant les paramètres optimum, d'où le benchmark.

```
cryptsetup benchmark
cryptsetup luksFormat -y -v --type=luks1 --hash=sha256 --key-size=256 \
--cipher=aes-xts-plain64 /dev/sdc2
```

La compilation de cryptsetup par les équipes de debian rend optimal le chiffrement avec une clé de 256 bits et un hashage sha256.

1.3 Ouverture de la partition chiffrée

Le nom qui sera donné à cette partition sera *StarFleet* !

```
cryptsetup luksOpen /dev/sdc2 StarFleet
```

Désormais la partition est atteignable par le *node* `/dev/mapper/StarFleet` pour toute action.

1.4 Création du lvm dans la partition chiffrée

Dans la partition `/dev/mapper/StarFleet` vont être créés le volume logique – système de fichiers – par la commande `pvcreate` puis le nom d'un groupe de volumes au nom de *NCC1701* par la commande `vgcreate`.

```
pvcreate /dev/mapper/StarFleet
vgcreate NCC1701 /dev/mapper/StarFleet
```

1.5 Création des partitions dans le lvm

Création de trois partitions dans le groupe de volumes *NCC1701*, l'une désignée par *racine* d'une taille de 50 Go, l'autre, *memoire* d'une taille de 8 Go et enfin *utilisateurs* occupant tous le reste du disque.

```
lvcreate NCC1701 -L 50G -n racine
lvcreate NCC1701 -L 8G -n memoire
lvcreate NCC1701 -l +100%FREE -n utilisateurs
```

Les partitions seront dédiées aux usages suivants :

- la partition *racine* accueillera l'ensemble des logiciels, paramètres, réglages et autres caches,
- la partition *memoire* sera dédiée à la mémoire additionnelle, le *swap*,
- la partition *utilisateurs* recevra les données et paramètres des utilisateurs.

1.6 Formatage de toutes les partitions

Formatage de la partition EFI en FAT32, des partitions racine et utilisateurs en EXT4, et de la partition memoire en SWAP.

```
mkfs.fat -F32 /dev/sdc1
mkfs.ext4 /dev/NCC1701/racine
mkfs.ext4 /dev/NCC1701/utilisateurs
mkswap /dev/NCC1701/memoire
```

1. Grub2 ne sait pas à ce jour et à ma connaissance gérer les disques entièrement chiffrés en Luks version 2.

1.7 démontage du swap et de l'efi de l'hôte, activation du swap de la cible

Afin d'éviter toute interférence avec le futur système, démontage de la partition EFI actuelle, puis du swap actuel et activation du swap du futur système.

```
umount /boot/efi
swapoff -a
swapon /dev/NCC1701/memoire
```

1.8 création des points de montage pour l'installation

Je vais accrocher le futur système comme une arborescence prenant racine au dossier `/mnt`.

```
mkdir -p /mnt/{boot/efi,home}
```

J'ai besoin de créer la sous-arborestcence suivante :

```
/ (actuellement /mnt, future racine)
+-- home (utilisateurs futurs)
+-- boot/
....+-- efi (partition EFI placée au bon endroit)
```

1.9 montage des différentes partitions

Puisque les points d'accrochage des partitions de la cible sont créés, il est temps de les y accrocher, en suivant l'ordre d'imbrication.

```
mount /dev/NCC1701/racine /mnt
mount /dev/NCC1701/utilisateurs /mnt/home
mount /dev/sdc1 /mnt/boot/efi
```

1.10 mise à jour des paquets, installation des outils

J'installe dans l'hôte les outils nécessaires pour pouvoir installer le système de base dans la cible.

```
apt update
apt install wget debootstrap
```

1.11 installation de la base dans la cible

La commande `debootstrap` est celle qui fonctionne pour installer le système de base. L'ordre des paramètres est important ! Notez que grâce à cela il est possible aussi d'installer le système pour une autre architecture via le paramètre `-arch=` de choisir la distribution, ici *bullseye* de préciser le point où est accroché la cible, j'ai choisi `/mnt` mais par exemple dans l'installateur classique debian celui-ci choisit `/target` et enfin le serveur web contenant les paquets de la distribution debian à installer.

```
debootstrap --arch=amd64 bullseye /mnt http://ftp.fr.debian.org/debian/
```

1.12 Montages nécessaires pour le chroot

Le `chroot` (**ch**ange **root**) nécessite avant son exécution que certains *nodes* du système soient liés intrinsèquement.

```
mount --rbind /dev /mnt/dev
mount --rbind /sys /mnt/sys
mount --rbind /proc /mnt/proc
mount --make-rslave /mnt/dev
mount --make-rslave /mnt/sys
mount --make-rslave /mnt/proc
```

1.13 copie du fichier de résolution et chroot

Avant de passer dans le nouvel environnement via `chroot` copie du fichier contenant le serveur dns pour pouvoir ensuite utiliser la connexion internet de l'hôte afin de télécharger la suite des paquets.

```
cp -v /etc/resolv.conf /mnt/etc/
chroot /mnt /bin/bash
```

2 la configuration dans le chroot

Toute cette partie se situe dans le nouvel environnement.

2.1 mot de passe de root

Comme j'ai souvent tendance à l'oublier, autant commencer par donner à l'administrateur du futur système un mot de passe.

```
passwd root
```

2.2 modification des hosts et hostname

Ensuite il est temps de configurer le nom réseau à cette nouvelle machine, je l'appelle *Enterprise* pour rester dans la même thématique que le nom du volume chiffré et celui du groupe de volumes.

```
echo Enterprise > /etc/hostname
echo "127.0.1.1    Enterprise" >> /etc/hosts
```

Ce nom est attribué dans le fichier `hostname` mais aussi ajouté dans le fichier `hosts` avec l'adresse spécifique 127.0.1.1 demandée par certains environnements.

2.3 édition du /etc/apt/sources.list et mise à jour des paquets

Il est indispensable, surtout lorsque le système utilise du matériel non-libre ou que des logiciels nécessitant des pilotes non-libres voire non-libres eux-mêmes, de modifier la liste des sources logicielles située dans le fichier `sources.list`

```
nano -cil /etc/apt/sources.list
```

Exemple. Fichier `/etc/apt/sources.list`.

```
deb http://ftp.fr.debian.org/debian bullseye main contrib non-free
deb http://ftp.fr.debian.org/debian bullseye-updates main contrib non-free
deb http://ftp.fr.debian.org/debian bullseye-proposed-updates main contrib non-free
deb http://ftp.fr.debian.org/debian bullseye-backports main contrib non-free
deb http://ftp.fr.debian.org/debian-security bullseye-security main contrib non-free
```

2.4 mise à jour des paquets de la cible

Puis vient la mise à jour de la liste des versions de logiciels et des logiciels déjà installés, ce qui arrive parfois.

```
apt update
apt upgrade -y
```

2.5 installation d'outils et de la configuration des locales et de la timezone

Il est ensuite temps d'installer de quoi configurer la régionalisation du système, et un outil `pciutils` qui va identifier le matériel présent. On peut aussi ajouter `usbtils` si tout le matériel n'est pas retrouvé grâce à pciutils.

```
apt install locales console-setup pciutils
dpkg-reconfigure tzdata
dpkg-reconfigure locales
dpkg-reconfigure console-setup
```

Note. `tzdata` a été installé déjà dans le *base-system* avec `debootstrap`.

2.6 Ajout des informations vers les fichier /etc/crypttab et /etc/fstab

```
echo "StarFleet UUID=$(blkid -o value -s UUID /dev/sdc2) none luks" >> /etc/cryptab
blkid | grep NCC1701 >> /etc/fstab
blkid | grep sdc1 >> /etc/fstab
nano -cil /etc/fstab
```

Ces informations permettent de générer un fichier fstab correct. La ligne concernant crypttab permet de configurer le fichier directement sans avoir à l'ouvrir. Un mot de passe sera nécessaire au démarrage pour ouvrir la partition chiffrée qui accueille le lvm. Les quatre champs du fichier `crypttab` sont séparés par des tabulations ce qui ne se voit pas forcément dans la ligne montrée ci-avant.

Exemple. Le fichier `/etc/crypttab` dans ma config.

```
StarFleet    UUID=4037ccc6-4410-4fff-80ee-b2099a46c33e    none    luks
```

Ensuite on s'occupe du fichier `fstab` comme le montre la dernière commande des quatre lignes précédentes :

Exemple. Fichier `/etc/fstab`

```
/dev/NCC1701/racine  /   ext4   rw,errrors=remount-ro  0  1
/dev/NCC1701/memoire swap  swap  sw  0  0
/dev/NCC1701/utilisateurs /home  ext4  defaults  0  2
UUID=(uuid de /dev/sdc1)  /boot/efi  vfat  umask=0077  0  2
```

2.7 Édition de la configuration du clavier.

Vient le moment de configurer le clavier.

```
nano -w /dec/default/keyboard
```

Dans le fichier peut-être que la ligne `XKBLAOUT` est déjà en `"fr"`, mais dans les lignes qui viennent voici la configuration telle que je l'ai laissée au moment d'enregistrer le fichier.

Exemple. Voici les lignes telles qu'elles doivent être configurées pour le support du clavier français.

```
XKBMODEL="pc105"
XKBLAYOUT="fr"
XKBVARIANT="latin9"
```

2.8 Installation du noyau et des outils divers

C'est l'heure d'installer le noyau. La mention `firmware-***` correspond à la recherche de firmwares particuliers pour certaines cartes via la commande `lspci` et en recherchant par exemple les lignes spécifiques contenant des mots tels que "VGA" pour la carte graphique, "Network" ou "Wireless Network" pour le wifi ou encore "Ethernet" pour la carte réseau filaire.

```
apt install linux-image-amd64 lvm2 cryptsetup dhcpcd5 sudo grub-efi-amd64 \
firmware-***
```

C'est la raison pour laquelle cette mention `firmware-***` a été déportée en seconde ligne mais doit être écrite à la suite de `\` puisque sinon la ligne allait sortir de la page.

2.9 Modification du fichier de configuration de grub

Avant de poursuivre il faut modifier le fichier de configuration de grub, situé à `/etc/default/grub`, je vais lui ajouter, donc en fin de fichier, la valeur de l'UUID de la partition chiffrée qui doit être déchiffrée au démarrage par un mot de passe saisi manuellement. Ce travail est l'objet de la première ligne. La seconde ligne permet l'édition de ce fichier pour y déplacer cet UUID puis d'ajouter des fichiers

```
echo $(blkid -o value -s UUID /dev/sdc2) >> /etc/default/grub
nano -w /etc/default/grub
```

Il est possible de procéder autrement selon les capacités et aptitudes de chaque personne.

Voici les modifications apportées :

`GRUB\_CMDLINE_LINUX=""` devient

```
GRUB_CMDLINE_LINUX="cryptdevice=UUID=(uuid de /dev/sdc2):StarFleet"
```

ajouter au fichier :

```
GRUB_ENABLE_CRYPTODISK=y
GRUB_DISABLE_OS_PROBER=true
```

2.10 Régénération de l'initramfs

Une fois le fichier `/etc/default/grub` modifié, il est temps de générer l'initramfs, image linux minimaliste en charge d'être copiée en mémoire vive au démarrage du système afin qu'elle vérifie le système qui sera ensuite lancé pour l'utilisateur.

```
update-initramfs -u
```

Si aucune erreur n'est signalée, il est temps ensuite de passer à la suite !

2.11 Installation de grub avec support de l'efi

Désormais il est temps d'installer grub, puisque c'est le *bootloader* que je choisis en lui passant les bons paramètres qui déborderaient de la ligne, aussi ai-je coupé cette ligne en deux.

```
grub-install --target=x86_64-efi --efi-directory=/boot/efi \
 --bootloader-id=debian --uefi-secure-boot /dev/sdc
```

Ne vous trompez pas de disque ! Le disque cible est bien *sdc*.

Puis vient la phase de génération de la configuration de grub.

```
grub-mkconfig -o /boot/grub/grub.cfg
```

Les commandes chez debian sont traditionnellement : `update-grub` ou `update-grub2` mais ce sont en fait des alias pour la commande précédente.

2.12 ajout de l'utilisateur courant

Pour le démarrage viendra ensuite un utilisateur courant, il n'est pas nécessaire de l'ajouter au groupe des administrateurs (*sudo*) mais je le fais ici.

```
adduser utilisateur
adduser utilisateur sudo
```

2.13 ajout de quelques paquets pour le futur système

Comme je souhaite au démarrage avoir une interface graphique et quelques logiciels, je choisis ici de les installer,

```
apt install xorg lightdm lightdm-gtk-greeter xfce4 xfce4-goodies \
network-manager network-manager-gnome wpasupplicant dialog etc...
```

Remarque. `apt-cache search task-` pour afficher tous les métapaquets permettant d'installer des configurations prêtes à l'emploi ou presque.

3 Sortie du chroot et reboot de la machine.

D'abord sortie de l'environnement `chroot`é

```
exit
```

Puis redémarrage de la machine, c'est la méthode la plus propre actuellement testée pour détacher le disque cible de la machine hôte.

```
reboot
```

Normalement vous avez désormais un disque externe contenant une debian chiffrée minimaliste prête à être utilisée sur une machine fonctionnant en EFI.