Fichiers III : périphériques en mode bloc

Fichiers : plan

On va observer en 3 niveaux

- arborescence

  `ls -ilah, cd, cp -lar, rm -rf, mv, dd, tar -cxzvf, pwd -P, mkdir -p, rmdir, ln -s, readlink, file, du, find, cat, touch`

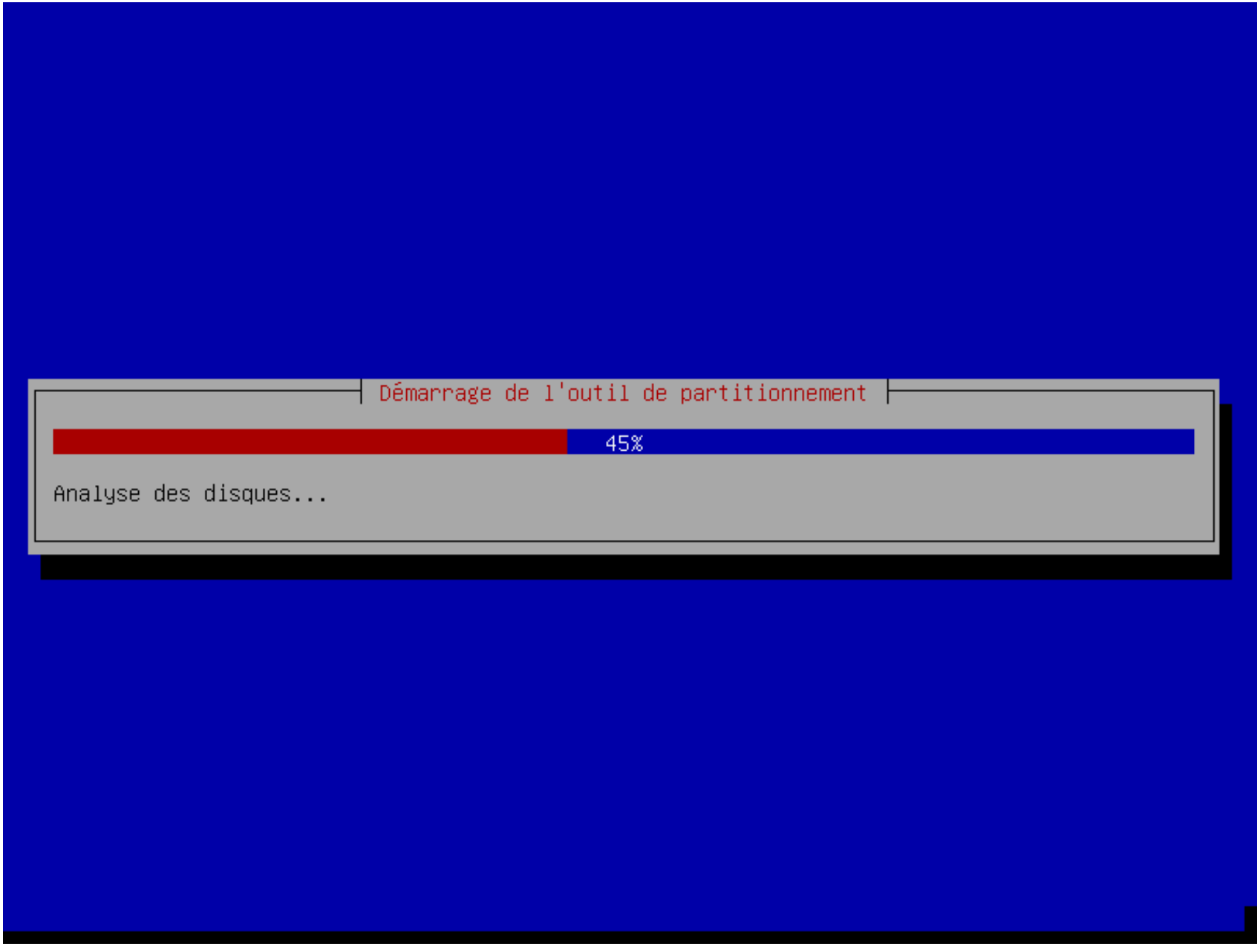
- systèmes de fichiers

  `mkfs -t, fsck, mount, umount, findmnt, stat, df, sync`

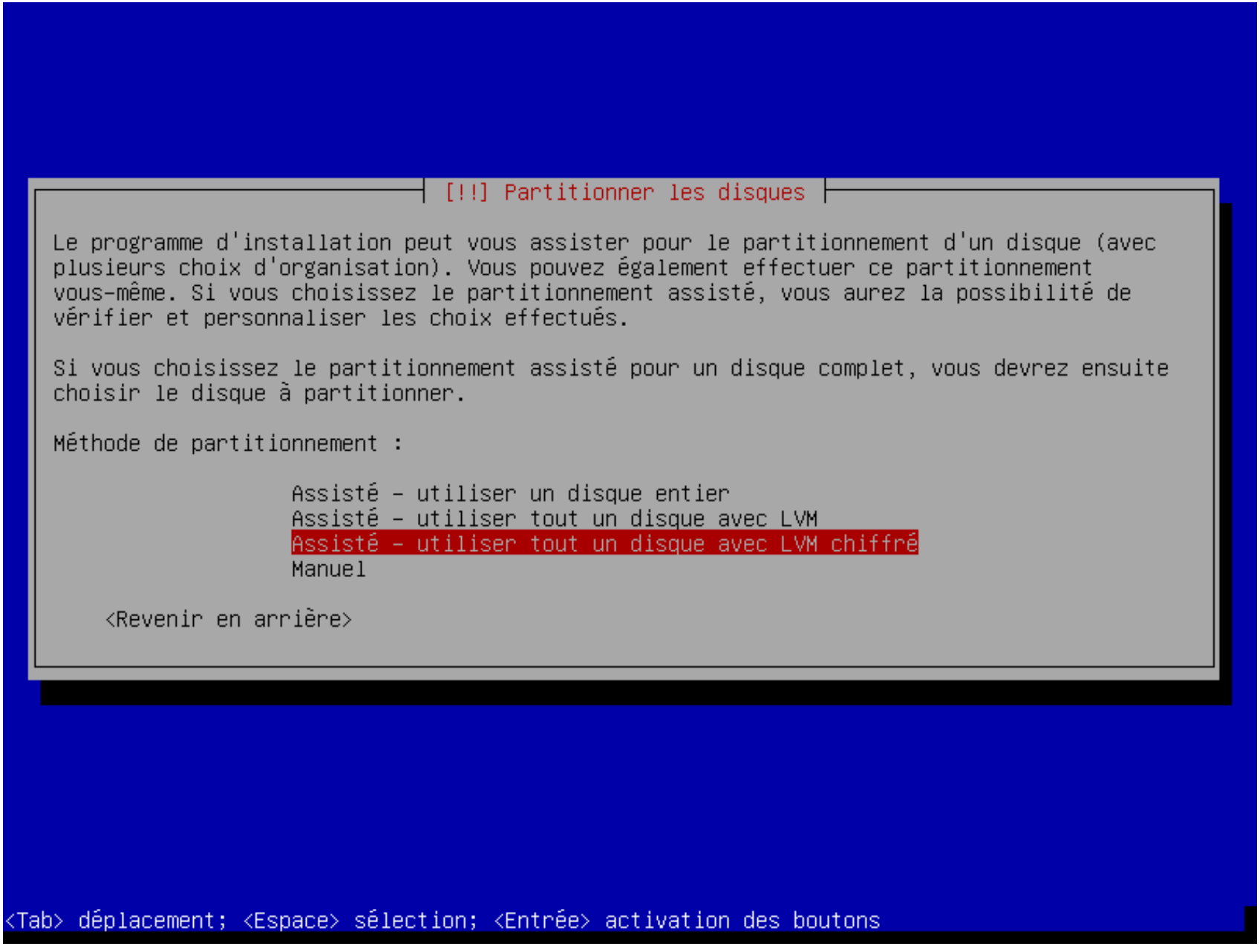
- périphériques bloc (disque, partition, boucle, chiffrement, lvm, raid)

  `(g)parted, lsblk, cryptsetup, lvm, mdadm, losetup, dmsetup`

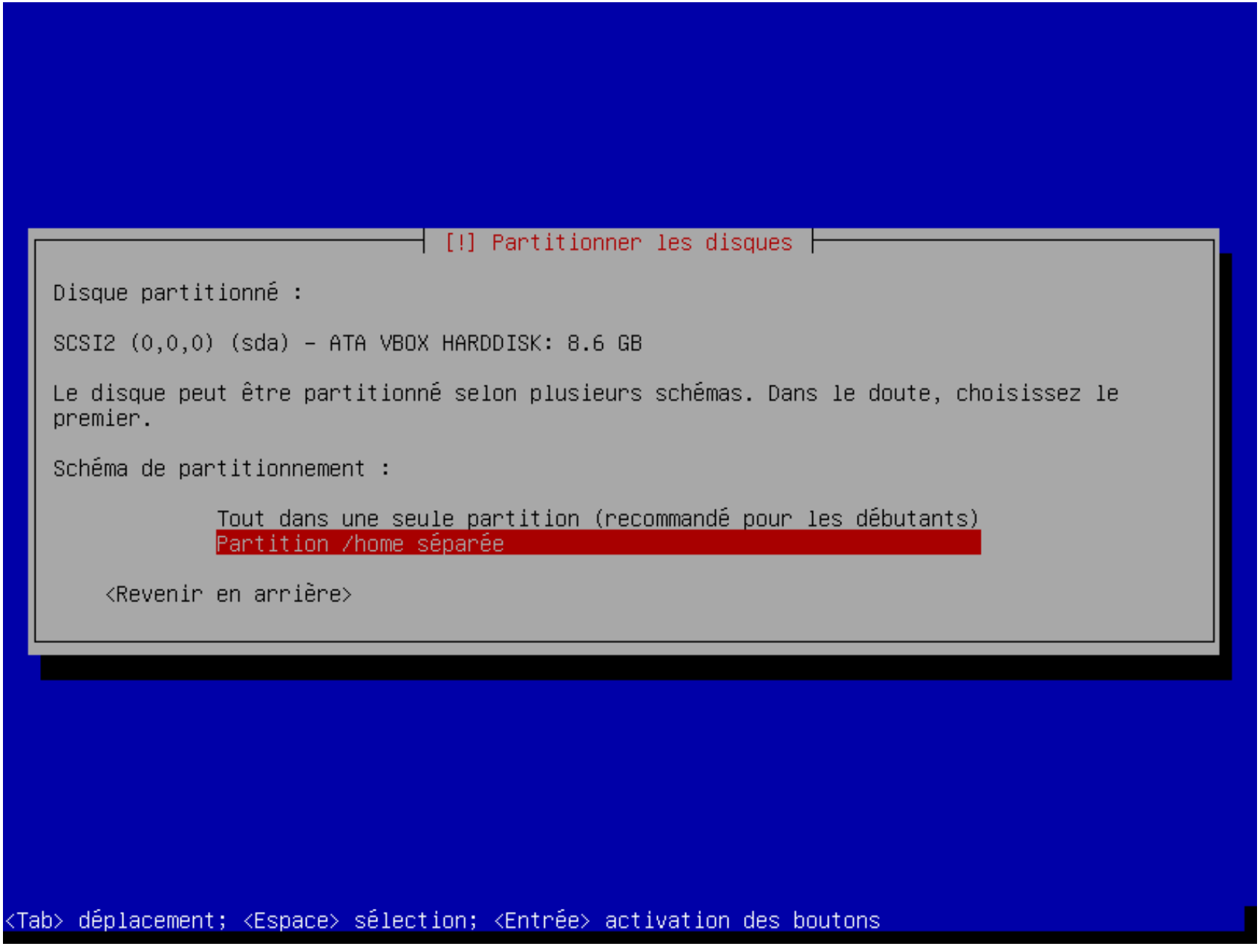
Périphériques en mode bloc : motivation



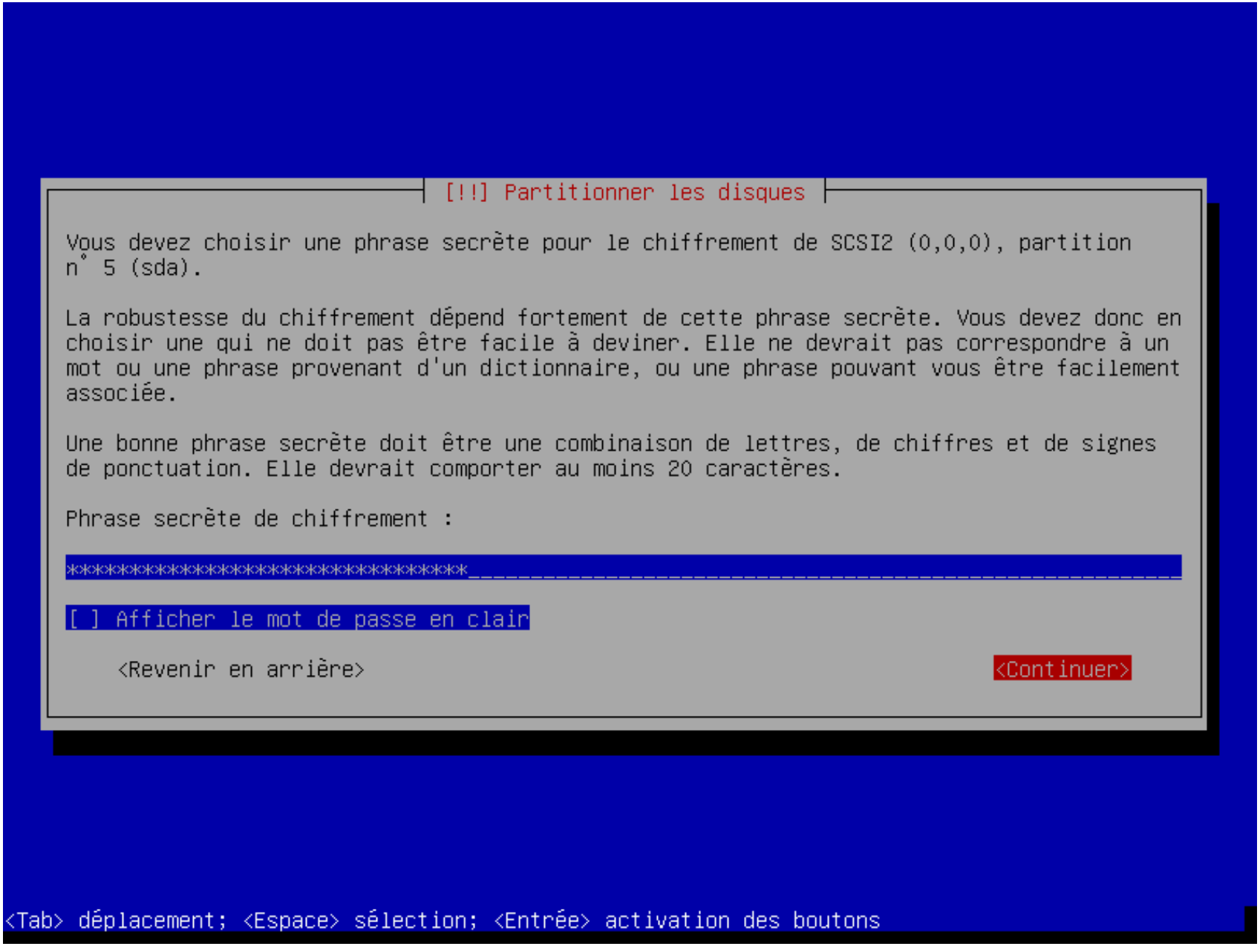
Périphériques en mode bloc : motivation



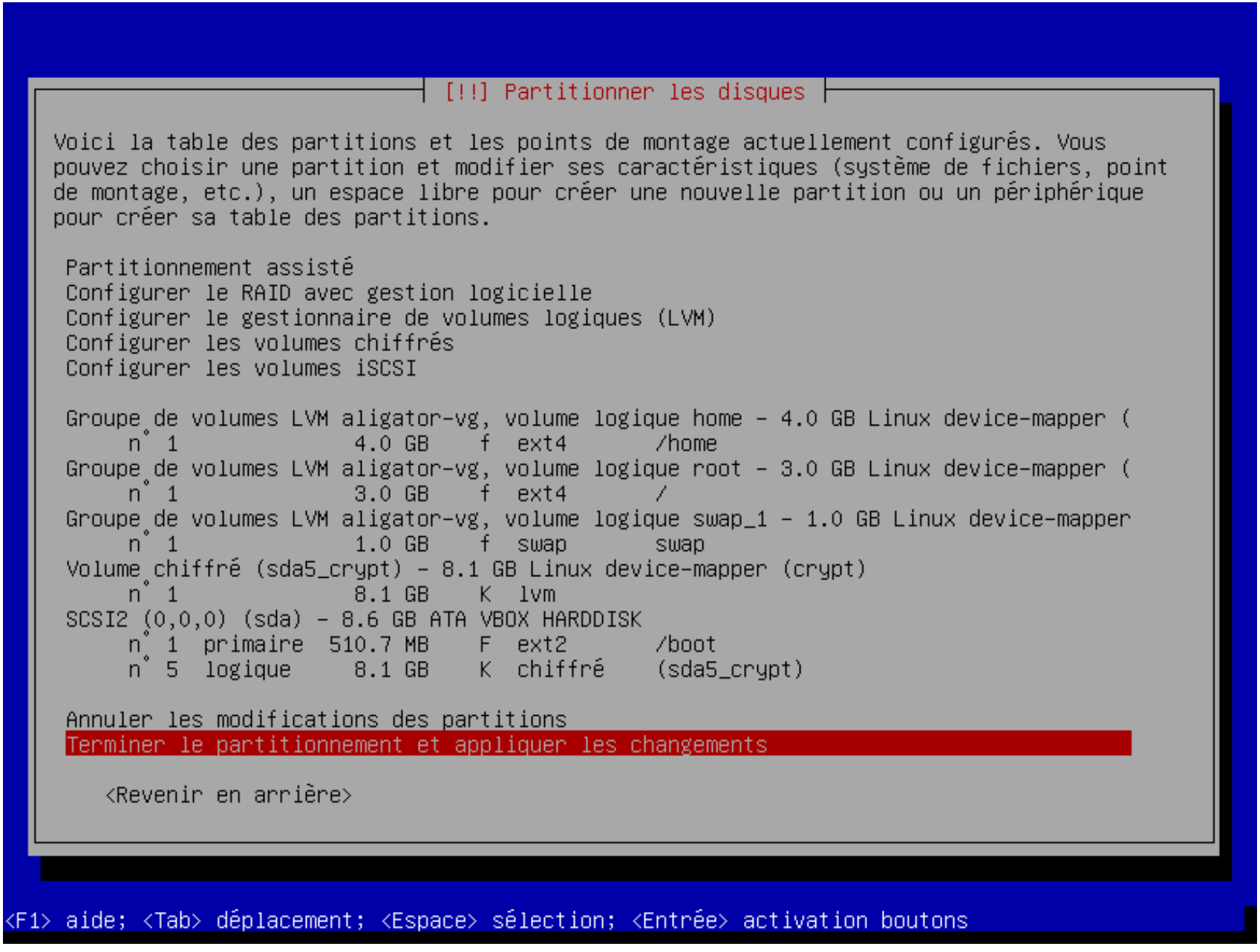
Périphériques en mode bloc : motivation



Périphériques en mode bloc : motivation



Périphériques en mode bloc : motivation



Périphériques en mode bloc : motivation

Le parti pris de ce chapitre est de rendre banal ce genre de constructions en l'interpretant de la façon homogène suivante : les disques, les partitions, les volumes logiques LVM, les volumes chiffrés, le raid, les images disque iso, etc, ne sont que des block devices obtenus les uns à partir des autres en formant un DAG.

Ainsi, vous serez en mesure de comprendre ce qu'il se passe et d'isoler l'origine du problème quand tout se casse.

Demo.

Périphériques en mode bloc

Un *périphérique en mode bloc* (en:*block device*) est un périphérique auquel on accède (lecture/écriture) par blocs d'octets (en général un bloc mesure entre 512 et 4096 octets).

C'est une abstraction de tout ce sur quoi on peut installer un système de fichiers.

Ils représentent les disques, clefs USB, cartes SD, partitions, cd-roms, et divers assemblages de ceux-ci.

On peut lister les périphériques en mode bloc et leurs relations avec la commande:

```
$ lsblk
```

On obtient une vue arborescente du DAG des relations entre block devices.

Partitionnement de périphériques bloc

On peut partitionner un périphérique bloc (en général un disque) en plusieurs sous périphériques bloc. Ça permet de :

- faire cohabiter plusieurs systèmes de fichiers sur un seul disque
- d'isoler certaines parties du système
- faciliter certaines tâches (sauvegardes, mise en lecture seule)

Partitionner un périphérique bloc revient à écrire une table de partitions au début du périphérique indiquant diverses infos (signature, version, checksum, padding, etc) les adresses des partitions qui se trouveront sur le reste du périphérique.

Pour créer une table de partitions:

```
# parted
```

(voir aussi `sfdisk`, `fdisk`, `cfdisk`, `gparted`)

Partitionnement de périphériques bloc

Il y a différents types de tables de partitions avec leurs spécificités, par exemple :

- dans le cas de `msdos`, la table de partition tient sur le premier bloc de 512Mo (MBR), de sorte qu'au plus 4 partitions primaires sont possibles, et qu'elle ne peut adresser que des petits disques (2,199 To max).
- dans le cas de `GPT`, la table de partitions est répétée en fin de périphérique et le premier bloc est laissé vide.

Péripheriques boucles (en: loop devices)

On peut promouvoir un fichier régulier en un block device:

```
# losetup -f <fichier>

# losetup -d /dev/loop<num>
```

Utilisations :

- accéder à une image ISO d'un CD ou DVD (cf TP virtualbox).
- ajouter une partition d'échange (en: swap) rapidement.
- récupérer les données d'un disque endommagé en travaillant sur une image pour ne pas abmier le disque encore plus.
- pour se faire la main avec des disques virtuels en RAM et tout jeter (cf TP).

Cryptsetup (Linux Unified Key Setup, LUKS)

Étant donné un périphérique bloc `<block-device>`, on peut y installer un périphérique bloc chiffré. Ce périphérique bloc chiffré ne sera utilisable que lorsque la phrase de passe sera donnée.

- installer un périphérique bloc chiffré dans `<block-device>` :

```
# cryptsetup luksFormat <block-device>
```

- déchiffrer le périphérique bloc chiffré installé dans `<block-device>` et le rendre accessible comme un nouveau périphérique bloc accessible par le chemin `/dev/mapper/<name>` :

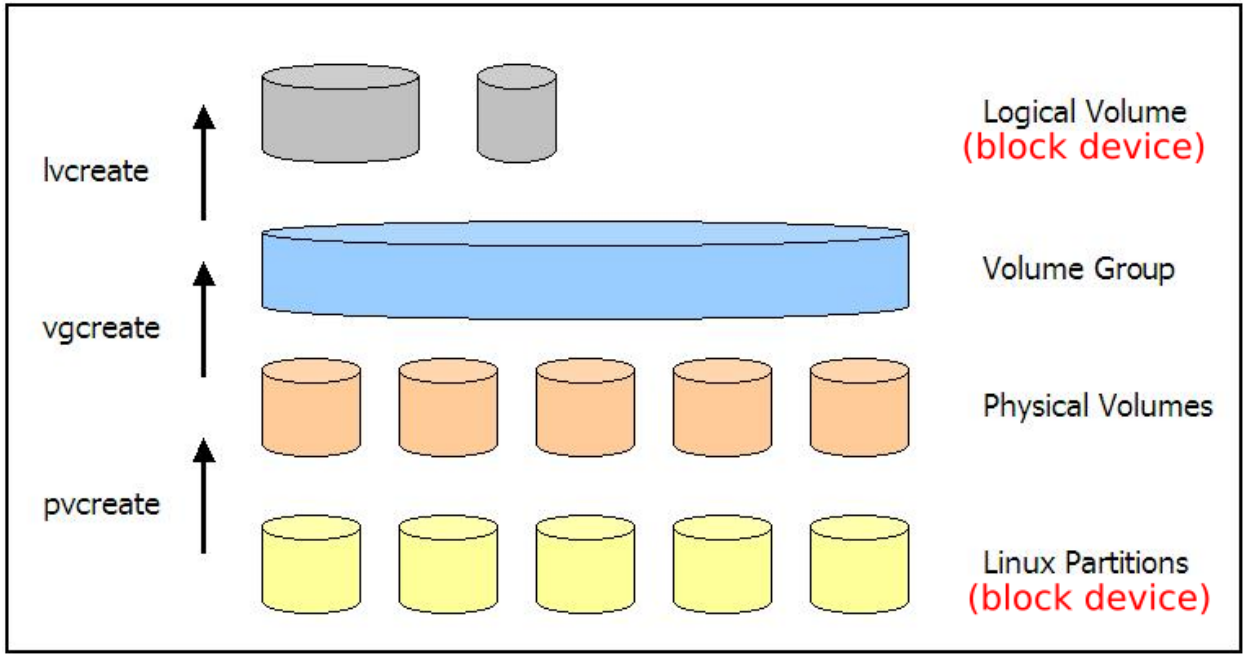
```
# cryptsetup open <block-device> <name>
```

- désactiver le périphérique bloc chiffré `<name>` :

```
# cryptsetup close <name>
```

Logical Volume Management (LVM2)

On peut aussi recoller des block devices et les redécouper pour en faire d'autres block devices, qui apparaissent dans `/dev/mapper/`.



LVM permet aussi de faire des snapshots (fr:instantanés), redimensionner à chaud, etc.

```
$ man 8 lvm
```

Autocompletion : `pv[TAB]`, `vg[TAB]`, `lv[TAB]`

Redundant Arrays of Inexpensive Disks (RAID)

Permet de combiner plusieurs block devices en un seul de sorte à augmenter les performances et/ou augmenter la tolérance aux pannes (redondance).

https://fr.wikipedia.org/wiki/RAID_(informatique)

https://en.wikipedia.org/wiki/Standard_RAID_levels

```
# apt install mdadm
# man mdadm
```

Simuler des erreurs avec dmsetup

Intérêt : tester le comportement d'un programme ou d'une infrastructure et leur robustesse aux erreurs.

On peut par exemple alterner une période de 10 secondes sans erreur et une période de 100 secondes avec des erreurs

```
# dmsetup create <nom> << EOF
0 <taille> flakey <device> 0 10 100
```

On peut introduire une erreur permanente au niveau du 9e "secteur" :

```
# dmsetup create <nom> << EOF
0 8 linear <device> 0
8 1 error
9 <taille - 9> linear <device> 9
EOF
```

- https://www.kernel.org/doc/Documentation/device-mapper/dm-flakey.txt
- https://www.kernel.org/doc/Documentation/device-mapper/linear.txt

Empilement de périphériques bloc

L'ordre dans lequel vous allez empiler les couches dépend de ce que vous voulez faire.

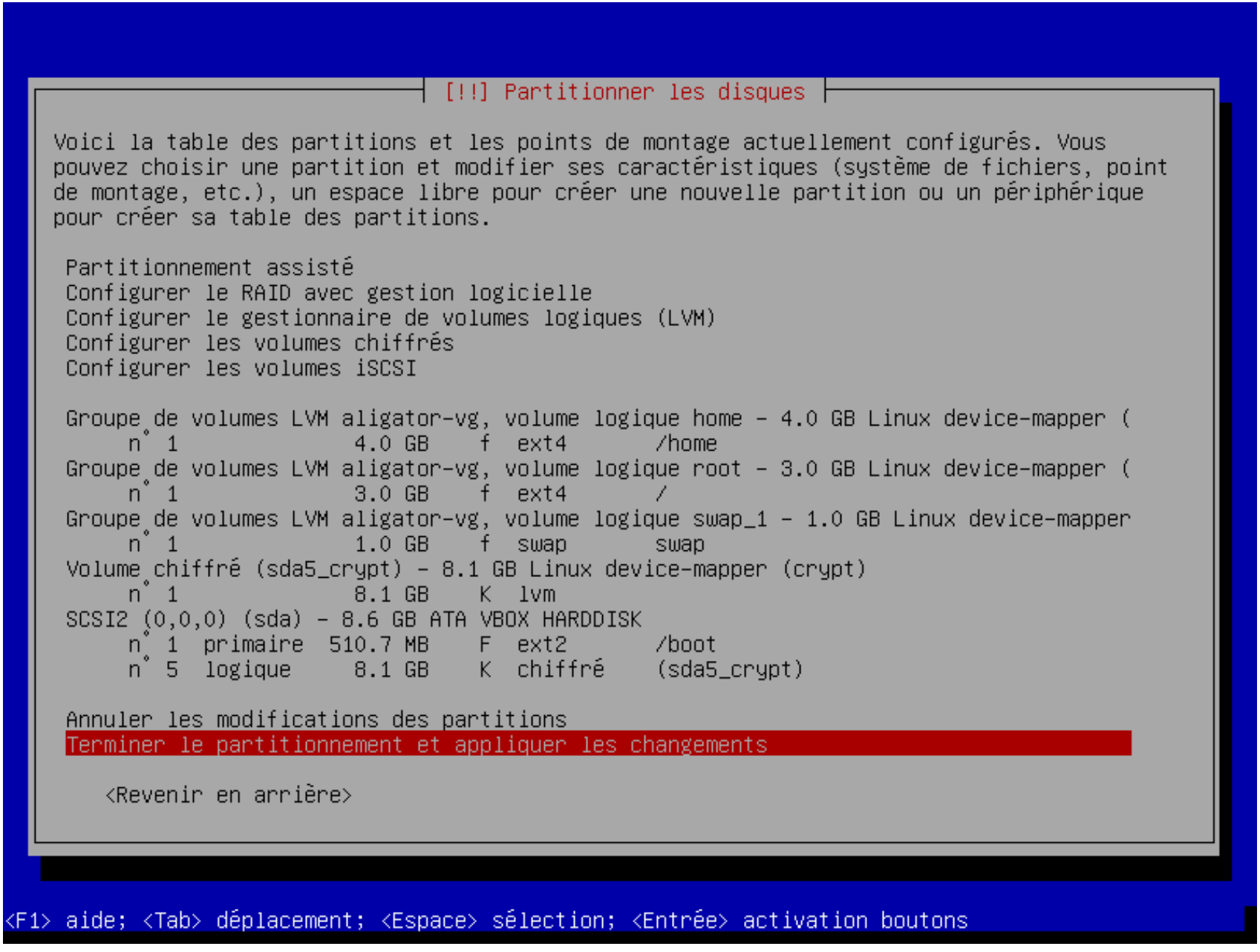
Typiquement, si vous voulez chiffrer votre disque, vous aurez une partition non chiffrée pour `/boot` mais le reste du disque pourra être chiffré avant d'être sépare en différentes partitions pour le reste de / et une partition pour le `swap` (partition d'échange qui peut être utilisée comme une extension de la RAM lorsque celle-ci est pleine), de sorte qu'une seule phrase de passe soit à taper. [Attention à ce que tel quel, le design donné dans cet exemple est sensible à l'*Evil maid attack*]

Certaines combinaisons ont peu d'intérêt: par exemple faire du RAID entre des périphériques bloc situés sur le même disque physique, n'augmentera ni la vitesse, ni ne protégera contre un crash du disque.

Périphériques en mode bloc : debriefing

Identifier dans l'écran de validation suivant le DAG des block devices avec leurs tailles, les filesystems et leurs points de montage dans l'arborescence.

Périphériques en mode bloc : debriefing



Périphériques en mode bloc : debriefing

Ouverture

- On a vu que les périphériques de stockage de masse sont accessibles en tant que fichiers de type `b` dans `/dev/`,
- on a vu que des systèmes de fichiers étaient installés dans certains de ces périphériques,
- on a vu que ces systèmes de fichiers étaient montés les uns sur les autres à partir de / pour constituer ensemble l'arborescence du système.
- Mais alors le système de fichier installé sur / qui donne naissance aux premiers répertoires `/dev`, `/etc`, `/home`, etc provient d'un filesystem installé dans un block device qui se trouve dans `/dev`.
- Y'a pas un serpent qui se mord la queue là ?

Dans un prochain épisode, nous verrons comment cela est possible, et pourquoi la partie "noyau" de la séquence de boot du système se décompose en deux parties.