Disquettes d'amorce : HOWTO

Graham Chapman, grahamc@zeta.org.au
Version française, Cédric Adjih.
v1.01, 6 Février 1995 (v1.01fr1, 3 Mars 1995)

Ce document indique comment créer des disquettes d'amorce, racine, et de maintenance. Ces disquettes peuvent être utilisées comme disquettes de secours ou pour tester de nouveaux noyaux.

Table des matières

1 Remarques du traducteur **3**

2 Introduction **3**

2.1 Pourquoi créer des disquettes d'amorce? . . . 3
2.2 Suggestions et remerciements . . . 4
2.3 Liste des modifications . . . 4

3 Disquettes **5**

3.1 Aperçu des types de disquettes . . . 5
3.2 Amorçage . . . 6
3.2.1 Aperçu . . . 6
3.2.2 Indiquer la racine . . . 6
3.2.3 Copier le noyau sur une disquette d'amorce . . . 7
3.3 Racine . . . 8
3.3.1 Aperçu . . . 8
3.4 Amorce/Racine . . . 8
3.4.1 Disque virtuel en mémoire et disquettes racines . . . 9
3.5 Utilitaires . . . 10

4 Composants **10**

4.1 Systèmes de fichiers . . . 10
4.2 Noyau . . . 11
4.2.1 Compiler un noyau . . . 11
4.3 Périphériques . . . 12

4.4 Répertoires 12
4.4.1 /etc 13
4.4.2 /bin 15
4.4.3 /lib 15
4.5 LILO 15
4.5.1 Aperçu 15
4.5.2 Exemple de fichier configuration LILO 16
4.5.3 Enlever LILO 17
4.5.4 Options de LILO utiles 17

5 Exemples 18
5.1 Liste du contenu de disquettes 18
5.1.1 Liste (ls -lR) du contenu des disquettes amorce/racine. 18
5.1.2 Liste (ls-lR) des répertoires des disquettes utilitaires 24
5.2 Scripts pour créer des disquettes 24
5.2.1 mkroot - Make Root or Boot/Root Diskette 24
5.2.2 mkutil - Crée une disquette utilitaires 26

6 FAQ 27
6.1 Comment faire une disquette d'amorce avec le pilote XXX? 27
6.2 Comment faire pour mettre à jour ma disquette d'amorce avec un nouveau noyau? 28
6.3 Comment enlever LILO pour amorcer DOS? 28
6.4 Comment puis-je amorcer si j'ai perdu et mon noyau et ma disquette d'amorce? 28
6.5 Comment puis-je faire des copies de mes disquettes d'amorce ou racine? 29
6.6 Comment puis-je amorcer sans taper "ahaxxxx=nn,nn,nn" à chaque fois? 30
6.7 J'ai monté ma partition disque. Comment en utiliser les programmes? 31

7 Références 32
7.1 LILO - Lanceur Linux (LInux LOader) 32
7.2 Les FAQ et les HOWTOs Linux 32
7.3 Scripts de secours 33
7.4 SAR - Search And Rescue : cherche et restaure. 33
7.5 La distribution Slackware 33

1 Remarques du traducteur

((NdT: dans ce document les remarques de traduction/du traducteur sont mises entre « `((NdT:` » et « `))` ».

Il faut noter que l'utilisateur auquel ce HOWTO s'adresse plutôt est l'utilisateur ayant une connexion Internet par laquelle il récupère la plupart des fichiers (TCP/IP, voire UUCP). C'est moins fréquent en France, et si ce n'est pas votre cas, certaines choses peuvent se faire d'une autre façon. Par exemple si vous avez un lecteur de CD + un CD Linux, vous pouvez toujours essayer de copier le contenu d'une disquette de secours d'un CD sur votre partition de swap (au lieu de faire une disquette racine) ; ou bien réinstaller un système Linux minimal sur cette partition de swap.

D'autre part la configuration du noyau et des pilotes peut être sensiblement différente si vous lisez ce document à une époque où différents « modules » vous sont directement accessibles (les noyaux >= 1.1.85 ont déjà un support pour des modules « longue durée »).

Par ailleurs une alternative pour lancer un noyau Linux depuis DOS n'est pas indiquée ici : il s'agit de LOADLIN. Si cela présente un intérêt pour vous, récupérez son archive (`lodlin15.zip` ou `lodlin15.tar.gz`), et lisez la documentation l'accompagnant.

Correspondance anglais/français utilisée dans la traduction :

- boot : amorce, amorçage, amorcer, démarrage, démarrer.
- package : paquetage.
- root : racine.
- root device : périphérique racine.
- boot/root disk : disquettes (d') amorce/racine.
- backup : archive.
- boot loader : chargeur, lanceur.
- ramdisk : disque virtuel en mémoire, disque mémoire.
- library : bibliothèque.
- dynamic library : bibliothèque partagée.

))

2 Introduction

2.1 Pourquoi créer des disquettes d'amorce ?

Les diquettes d'amorce sont utiles dans certaines situations comme :

- Test d'un nouveau noyau.

– Récupération du système après un plantage disque ou système. Un tel plantage peut aller du secteur de boot perdu à un crash des têtes du disque.

Il y a plusieurs façon de créer des disquettes d'amorce :

– Utiliser une disquette d'une distribution comme la Slackware. Ceci permet au moins d'amorcer.

– Utiliser un utilitaire prévu justement pour créer les disquettes de secours.

– Apprendre ce qui nécessaire pour chaque type de disquette, et créer les siennes soi-même.

J'ai choisi la dernière possibilité : les faire soi-même. Ce qui permet de plus d'apprendre beaucoup sur la façon dont Linux fonctionne.

Les utilisateurs expérientés de Linux n'auront pas vraiment besoin de ce document. En revanche, les utilisateurs non initiés à l'administration d'un système Linux et qui veulent se prémunir de la perte de leur disque racine (ou d'autres problèmes encore) pourront y trouver quelque intérêt.

Une note à propos des versions utilisées : ce document a été mis à jour pour les paquetages suivants :

– Linux 1.1.73

– LILO 0.15

2.2 Suggestions et remerciements

Toute suggestion, critique ou commentaire, sur le contenu de ce document est la bienvenue. Prevenez-moi si vous découvrez une erreur ou un oubli. J'aimerais remercier les personnes suivantes pour avoir corrigé des erreurs et fourni des idées utiles :

```
Randolph Bentson
Bjxrn-Helge Mevik
Johannes Stille
```

2.3 Liste des modifications

v1.01fr1, 3 Mars 1995 = traduction de v1.01

v1.01, 6 février 1995

– Correction : PAS DE `cp <kernel file> /dev/fd0` - ceci effacerait tout système de fichiers sur la disquette.

- Correction : Mettre boot.b et les fichiers map sur la disquette.
- Ajout : option -dp avec cp pour éviter certains problèmes.
- Changement : restructuration pour améliorer la lisibilité.
- Ajout : il est à présent possible d'utiliser un système de fichiers `ext2` sur les disquettes racines.
- Changement : il est possible à présent de séparer les disquettes d'amorce et les disquettes racines.
- Ajout : partie « remerciements » dans l'introduction.
- Ajout: FAQ.

v1.0, 2 janvier 1995

- Conversion pour répondre aux standards des HOWTOs.
- Ajout d'une nouvelle section « Liste des modifications ».
- Diverses corrections mineures.

v0.10, 1er novembre 1994 Version originale, « brouillon » (draft).

3 Disquettes

3.1 Aperçu des types de disquettes

Les disquettes de type amorce se classent en 4 catégories. La plus grande partie de ce texte peut aussi bien s'appliquer à des disques durs.

Aperçu des types de disquettes et de leur utilisation :

amorce (boot)

Une disquette contenant un noyau qui peut être lancé. La disquette peut contenir un système de fichiers et utiliser un lanceur, ou elle peut contenir uniquement un noyau au début de la disquette.

racine (root)

Une disquette avec un système de fichiers contenant tout ce qui est nécessaire pour démarrer un système Linux. Elle ne contient pas obligatoirement un noyau ou un lanceur.

Cette disquette peut être utilisée pour utiliser le système indépendamment de tout autre disque, une fois que le noyau s'est lancé. Une possibilité spéciale du noyau permet de monter une disquette racine séparée après avoir démarré, cette disquette étant copiée dans un disque virtuel en mémoire.

Vous pouvez utiliser ce type de disquette pour vérifier l'intégrité d'un autre disque sans le monter, ou encore pour rétablir un autre disque après un plantage disque ou une perte de fichiers.

amorce/racine (boot/root)

Une disquette identique à une disquette racine, sauf qu'elle contient un noyau et un lanceur. On peut l'utiliser pour amorcer et utiliser le système. L'avantage de ce type de disquette est la compacité ; tout ce qui est nécessaire tient sur une seule disquette. Néammoins, l'augmentation progressive de la taille des différents fichiers fait qu'il n'est pas toujours possible de tout faire tenir dans une seule disquette.

utilitaires (utility)

Une disquette qui contient un système de fichiers mais qui n'est pas destinée à être montée comme une disquette racine. Vous utiliserez ce type de disquette si vous avez trop d'utilitaires pour pouvoir tout mettre sur votre disquette racine.

Le terme « disquette *utilitaires* » ne s'applique réellement qu'à des disquettes contenant véritablement des paquetages de récupération/restauration.

3.2 Amorçage

3.2.1 Aperçu

Tous les PC démarrent en exécutant une routine du BIOS chargeant le secteur 0 du cylindre 0 du disque/disquette d'amorce.

Sur la plupart des disques amorçables, le secteur 0 du cylindre 0 contient soit :

- une routine d'un lanceur comme LILO qui repère l'emplacement du noyau, le charge et l'exécute pour effectuer le démarrage proprement dit.
- le lancement d'un noyau d'un système d'exploitation, Linux par exemple.

Si un noyau Linux est écrit sur une disquette via le périphérique /dev/fd0 alors le premier secteur de celle-ci sera le premier secteur du noyau Linux, et ce secteur se chargera de continuer l'initialisation en chargeant le reste du noyau et en démarrant Linux. Pour une description plus détaillée du contenu du secteur d'amorce, voir la documentation de lilo-0.5 ou des versions suivantes.

Une autre méthode pour copier un noyau sur une disquette d'amorce est de créer un système de fichiers, non pas comme périphérique racine, mais simplement pour installer LILO, ceci permettant de préciser des options du démarrage.

3.2.2 Indiquer la racine

Le noyau doit d'une façon ou d'une autre savoir quelle partition de quel disque utiliser comme racine. On peut fournir cette information de plusieurs façons:

- En définissant ROOT_DEV = <périphérique> dans le makefile du noyau Linux, et en le recompilant (pour des conseils sur la recompilation du noyau, lire la FAQ Linux et regarder dans /usr/src/linux). Des commentaires dans le makefile décrivent les choix possibles pour <périphérique>. En lançant l'utilitaire rdev :

```
rdev <nom de fichier> <peripherique>
```

Ceci fixera le périphérique racine du noyau contenu dans `<nom de fichier>` comme étant `<périphérique>`. Exemple :

```
rdev Image /dev/sda1
```

et le périphérique racine du noyau dans Image deviendra la première partition sur le premier disque SCSI.

Il y a plusieurs façons de lancer `rdev`. Essayez :

```
rdev -?
```

et vous verrez comment utiliser cette commande.

Il n'y a en général pas besoin de fixer le périphérique racine pour une disquette car le noyau utilisé pour l'amorce pointe probablement déjà vers le bon périphérique. Mais il peut arriver que vous ayez un noyau d'une autre machine (provenant par exemple d'une distribution), ou que vous vouliez utiliser le noyau pour lancer une disquette racine. Ca ne coûte rien d'essayer : utilisez `rdev` pour vérifier le périphérique racine d'un fichier contenant un noyau, avec :

```
rdev -r <nom de fichier>
```

Le périphérique racine peut être changé dans le noyau par d'autres moyens que `rdev`. Pour plus d'informations, voir la FAQ à la fin de ce document.

3.2.3 Copier le noyau sur une disquette d'amorce

Une fois que le noyau est configuré il doit être copié sur la disquette d'amorce.

Si la disquette ne va pas contenir de système de fichiers, alors le noyau doit être copié en utilisant ainsi la commande `dd` :

```
dd if=<nom de fichier> of=<peripherique>

ou      <nom de fichier> est le nom du noyau
et      <peripherique> est le peripherique du lecteur
        de disquette, habituellement /dev/fd0
```

Le paramètre origine (« seek parameter ») de la commande `dd` n'a pas à être utilisé. Le fichier doit être copié au début, sur le secteur de d'amorce (secteur 0, cylindre 0).

Le nom du périphérique varie. De nombreux systèmes ont `/dev/fd0` comme alias (sous une forme ou sous une autre) du « vrai » périphérique correspondant au lecteur de disquette par défaut. Par exemple, si c'est un lecteur 3 1/2 par défaut (« lecteur A: » sous DOS), le nom du périphérique sera `/dev/fd0H1440`, mais habituellement, `/dev/fd0` correspondra au même périphérique.

Si le noyau doit être copié sur une disquette d'amorce contenant un système de fichiers, alors : il faut monter sur un répertoire adéquat d'un système de fichiers lui-même déjà monté. Puis utiliser la commande `cp`. Par exemple :

```
mount -t ext2 /dev/fd0 /mnt
```

```
cp Image /mnt
umount /mnt
```

3.3 Racine

3.3.1 Aperçu

Un disque/une disquette racine contient un système Linux complet et fonctionnel, mais sans nécessairement inclure un noyau. En d'autres termes, la disquette peut ne pas être amorçable, mais une fois que le noyau fonctionne, la disquette racine contient tout ce qui est nécessaire pour un système Linux complet. Pour pouvoir être utilisée ainsi, la disquette doit inclure un minimum :

- Un système de fichiers.
- un ensemble minimal de répertoires : `dev`, `proc`, `bin`, `etc`, `lib`, `usr`, `tmp`.
- un ensemble d'utilitaires de base : `bash` (pour avoir un shell), `ls`, `cp`, etc...
- un ensemble minimum de fichiers de configuration : `rc`, `inittab`, `fstab`, etc...
- Les bibliothèques partagées des utilitaires.

Bien sûr, un système de fichiers ne devient utile que lorsque vous pouvez lancer un programme à partir de celui-ci, et une disquette racine n'est en principe utile que si vous :

- Vérifiez un système de fichiers sur un autre disque, par exemple votre partition racine de votre disque dur. Vous devez amorcer Linux à partir d'un autre disque/d'une autre disquette, par exemple avec une disquette racine. Puis vous pouvez lancer fsck sur votre partition disque racine originale alors qu'elle n'est pas montée.
- Restaurez tout ou partie de votre disque original à partir d'une archive en utilisant les utilitaires d'archivage et de compression comme `cpio`, `tar`, `gzip` et `ftape`.

3.4 Amorce/Racine

C'est la même chose qu'une disquette racine, avec en plus un noyau et un chargeur comme LILO.

Avec ce système, le noyau est copié sur la disquette racine, et LILO est alors lancé pour installer une configuration qui pointe vers le noyau de la disquette. Lors de l'amorçage, LILO chargera et lancera le noyau à partir de la disquette.

Plusieurs fichiers doivent être copiés sur la disquette pour que cette méthode fonctionne. Les détails de ces fichiers et de la configuration de LILO, ainsi qu'un exemple fonctionnel sont donnés ci-dessous dans la section intitulée « LILO ».

3.4.1 Disque virtuel en mémoire et disquettes racines

Pour qu'une disquette racine soit efficace, vous devrez pouvoir la lancer depuis un disque virtuel en mémoire. Ceci évite d'avoir le système fonctionnant dans une boîte de sardine, comme ce serait le cas avec une disquette.

Le noyau Linux inclus une option avec laquelle sous certaines circonstance il copiera automatiquement le contenu de la disquette racine en disque-mémoire, puis selectionnera la racine comme étant le disque mémoire au lieu de la disquette. Les deux avantages majeurs :

- Le système est beaucoup plus rapide.
- Le lecteur de disquette est libéré, et permet d'utiliser d'autres disquettes.

Pour pouvoir utiliser cette possibilité il faut que :

- Le système de fichiers de la disquette soit un système de fichiers `minix` ou `ext2`. Le système de fichiers `ext2` est le plus utilisé. Notez que si vous avez un noyau Linux antérieur à 1.1.73, vous devriez jeter un coup d'oeil aux commentaires dans la section ci-dessous, intitulée « Systèmes de fichiers », pour voir si votre noyau supportera `ext2`. Si votre noyau est ancien, il se peut que vous ayez à utiliser `minix`.
- Un disque mémoire doit être choisi lors de la configuration du noyau et de taille supérieure à la taille de la disquette.

Un disque mémoire peut être configuré dans le noyau de plusieurs façons :

- En décommentant la macro `RAMDISK` dans le makefile du noyau Linux :

  ```
  RAMDISK = -DRAMDISK=1440
  ```

  pour définir un disque mémoire de 1440 x 1K blocs, la taille d'une disquette 3 1/2 haute densité.
- En lançant l'utilitaire `rdev`, disponible sur la plupart des systèmes Linux. Cet utilitaire affiche ou fixe les valeurs de certaines options du noyau comme la taille d'un disque mémoire. Pour définir un disque mémoire de 1440 blocs dans le noyau contenu dans le fichier Image, entrer :

  ```
  rdev -r Image 1440
  ```

  Ceci peut changer dans l'avenir, bien sûr. Pour voir ce dont votre version de `rdev` est capable, entrez la commande :

  ```
  rdev -?
  ```

  et `rdev` devrait afficher ses options.
- En utilisant le lanceur LILO pour définir la taille à l'amorçage. Par exemple avec le paramètre :

  ```
  ramdisk = 1440
  ```

  demande un disque mémoire de 1440 x 1K blocks à l'amorçage.

– En interrompant l'amorçage automatique de LILO et en ajoutant `ramdisk=1440` à la ligne de commande. Ce pourrait être :

```
vmlinux ramdisk=1440
```

Voir la section sur LILO pour plus d'informations.

– En éditant le fichier noyau et en changeant les valeurs du début du fichier qui contiennent la taille du disque mémoire. Ce n'est réellement à faire qu'en dernier recours, mais c'est possible. Voir la FAQ à la fin de ce document pour plus d'informations.

La méthode la plus simple est de définir les paramètres (dont la taille du disque mémoire) dans le fichier de configuration de LILO, parce que vous aurez de toute façon à écrire un tel fichier.

La configuration de LILO est décrite brièvement dans la section intitulée « LILO », mais il est recommandé d'utiliser la dernière version stable de LILO de votre site ftp le plus proche, et de lire la documentation qui va avec.

3.5 Utilitaires

Souvent une disquette n'est pas suffisante pour stocker tous les logiciels dont vous aurez besoin pour analyser, réparer et restaurer les disques corrompus. Quand vous inclurez `tar`, `gzip`, `e2fsck`, `fdisk`, `Ftape` et autres, ils rempliront à eux seuls

une disquette entière, peut-être plus si vous voulez beaucoup d'utilitaires.

Ceci signifie qu'un ensemble de secours contient souvent une disquette « utilitaires », avec un système de fichiers contenant tous les fichiers supplémentaires requis. Cette disquette peut être montée sur un répertoire pratique, comme `/usr`, sur un système amorce/racine.

Créer un système de fichiers est réellement simple, et est décrit ci-dessous dans la section « Systèmes de fichiers ».

4 Composants

4.1 Systèmes de fichiers

Le noyau Linux accepte maintenant deux systèmes de fichiers pour des disques racines copiés automatiquement dans des disques mémoire. `ext2` a été ajouté quelque part entre les versions 1.1.17 et 1.1.57 (j'ignore quand exactement). Si vous avez un noyau correspondant, cherchez dans `/usr/src/linux/drivers/block/ramdisk.c` la chaîne « `ext2` ». Si elle n'y est pas, alors il vous faudra utiliser un système de fichiers `minix`, et en conséquence, la commande `mkfs`.

Pour créer un système de fichiers sur disquette sur mon système, je lance la commande suivante :

```
mke2fs /dev/fd0
```

La commande `mke2fs` détectera automatiquement l'espace disponible et ne requiert donc pas de paramètres supplémentaires. ((NdT: vérifiez à deux fois avant de lancer un `mke2fs`. mke2fs est pratiquement irréversible))

Une manière simple de tester le résultat est alors d'essayer de monter la disquette. Si c'est du `ext2`, alors :

```
mount -t ext2 /dev/fd0 /<repertoire vide>
```

devrait marcher.

4.2 Noyau

4.2.1 Compiler un noyau

Dans la plupart des cas, il devrait être possible de copier votre noyau courant et d'amorcer la disquette à partir de celui-ci. Par contre il y a des cas où vous voudrez construire une disquette à part.

Une des raisons est la taille. Le noyau est l'un des fichiers les plus gros dans un système minimal, donc si vous voulez créer une disquette d'amorce/racine, vous devrez réduire la taille du noyau autant que possible. Le noyau accepte maintenant le changement de disquette après l'amorçage et avant de monter la racine, donc il n'est plus nécessaire de mettre le noyau et le reste sur la même disquette ; dans ce cas ces commentaires ne s'appliquent pas.

Il y a deux moyens de réduire la taille du noyau :

- Le compiler avec le minimum d'options nécessaire, et laisser tomber tout le reste. Le réseau est un bon candidat à la suppression, ainsi que le support pour les périphériques dont vous n'avez pas besoin pour votre système amorce/racine.
- Le compresser en utilisant l'option standard « noyau compressé » du makefile :

  ```
  make zImage
  ```

  Référez-vous à la documentation du source du noyau pour une description à jour sur la compilation de noyaux compressé. ((NdT: et aussi le Kernel-HOWTO)). Note : le source du noyau est habituellement dans `/usr/src/linux`.

Ayant déterminé l'ensemble minimum des options à inclure dans un noyau, il se peut que vous en ayez plusieurs à ajouter. L'utilisation la plus fréquente d'une disquette amorce/racine est probablement l'examen et la récupération d'un système de fichiers corrompu. Pour cela il se peut que vous ayez besoin de support du noyau.

Par exemple, si vos archives sont toutes mises sur bande en utilisant Ftape pour accéder à votre lecteur de bande, et si d'aventure vous perdiez votre disque racine, vous seriez dans l'impossibilité de récupérer vos archives. Vous auriez alors à réinstaller Linux et Ftape, puis tenter de lire et extraire vos précieuses archives.

Il est probablement souhaitable de garder une copie de la même version des utilitaires d'archivages utilisés pour écrire les archives pour ne pas perdre du temps et de l'énergie à installer des versions ne pouvant pas les lire.

Bonne idée : ajouter à votre noyau de disquette amorce/racine quelque support I/O que vous ayez ajouté à votre noyau pour l'archivage. Notez toutefois que le module `Ftape` (tout du moins la version que j'ai) est assez gros et ne tiendra pas sur une disquette amorce/racine. Vous pouvez le mettre sur une disquette utilitaires, comme décrit ci-dessous dans la section « Ajouter des disquettes utilitaires ».

La procédure pour compiler le noyau est décrite dans la documentation qui vient avec le noyau. Elle est simple à suivre, donc commencez par regarder dans `/usr/src/linux`. Note : si vous avez des problèmes de compilation du noyau, vous ne devriez peut-être pas essayer de créer des disquettes de amorce/racine de toute façon.

4.3 Périphériques

Un répertoire `/dev` contenant tous les fichiers spéciaux pour tous les périphériques utilisés par le système est obligatoire pour tout système Linux. Le répertoire lui-même est un répertoire normal, et peut être créé par la commande `mkdir`. Les fichiers périphériques spéciaux doivent par contre être créés par la commande `mknod`.

Il y a un raccourci heureusement : plutôt que de faire un zillion de `mknod`, copiez le contenu du répertoire `/dev` et supprimez ce dont vous n'avez pas besoin. Le seul impératif est de copier les fichiers en utilisant l'option `-R`. Ceci copiera le répertoire sans tenter de copier le contenu des fichiers. Notez que si vous utilisez la minuscule `-r`, le résultat sera très différent puisque vous copiez le contenu de tous vos périphériques (disques, lecteurs de cd). Ou tout du moins la partie pouvant tenir sur une disquette. Donc attention et utilisez :

```
cp -dpR /dev /mnt
```

en supposant que la disquette soit montée sur `/mnt`. Les options `dp` font que les liens symboliques sont copiés comme des liens, plutôt que, assez peu judicieusement, leur contenu ; et que les attributs du fichier sont préservés, en particulier les droits d'accès, et les groupes et propriétaires.

Si vous voulez créer les fichiers de périphérique sur la disquette à la force du poignet, faites un `ls -l` pour afficher les numéros de périphérique mineurs et majeurs et créez-les sur disquette avec `mknod`. Beaucoup de distributions incluent un script `MAKEDEV` dans le répertoire `/dev`. Il peut être utilisé pour créer les fichiers périphériques mais il est probablement plus facile de copier le `/dev`, particulièrement dans le cas de disques de secours.

4.4 Répertoires

Il se peut qu'avec simplement `/dev`, `/proc` et `/etc` vous puissiez amorcer un système Linux - je ne sais pas, je n'ai pas essayé. Par contre un ensemble minimum raisonnable de répertoires consiste en :

/dev

Requis pour les entrées/sorties avec les périphériques

/proc

Utilisé par la commande `ps`

/etc

Répertoire des fichiers de configuration système

/bin

Utilitaires exécutables faisant partie du système

/lib

Bibliothèques partagées

/mnt

Répertoire sur lequel monter les partitions à maintenir

/usr

Applications et utilitaires supplémentaires

Notez que l'arborescence présentée ici est valable uniquement pour la disquette racine. Voyez le *Linux File System Standard* pour des informations plus précises sur la façon de structurer un système Linux « standard ».

Quatre de ces répertoires se créent très simplement :

- `/dev` comme décrit ci-dessus dans la section « Périphériques ».
- `/proc` doit juste exister, donc il suffit de le créer avec `mkdir`.
- Deux autres, `/mnt` et `/usr` sont (aussi) des répertoires où des partitions sont montées, utilisés plus tard quand le système amorce/racine tournera. Donc, il faut aussi juste les créer.

Les trois répertoires restants sont décrits dans les sections qui suivent.

4.4.1 /etc

Ce répertoire doit contenir un certain nombre de fichiers de configuration. Sur la plupart des systèmes, ceux-ci peuvent être divisés en trois groupes :

- Ceux qui sont absolument requis, comme `rc`, `fstab`, `passwd`.
- Ceux qui sont peut-être nécessaires, mais personne n'en est sûr.
- Ceux qui n'ont aucune utilité.

Les fichiers secondaires peuvent être identifiés avec la commande :

```
ls -trul
```

qui donne les fichiers dans l'ordre inverse de date de dernier accès. Les fichiers non utilisés ne sont évidemment pas nécessaires pour une disquette racine.

Sur mes disquettes racines, j'ai réduit le nombre de fichiers à 15. Ceci réduit le travail à trois ensembles de fichiers :

- Ceux que je dois configurer pour un système amorce/racine :

```
rc      le script de demarrage systeme
fstab   liste des partitions a monter
inittab parametres pour le processus init, le
        premier processus lance lors du demarrage.
```

- ceux que je réduis pour un système amorce/racine :

```
passwd  liste des utilisateurs
shadow  contient les mots de passes
((NdT: dans un systeme avec les shadow-passwords))
```

  Ceux-ci devraient être raccourcis sur les systèmes sécurisés pour éviter de de copier les mots de passe des utilisateurs hors du système, et de telle manière que lorsque vous amorcez depuis la disquette, les « login » non désirés soient rejetés avec toute la vigueur nécessaire.

- Les autres. Ils marchent pour l'instant, donc je les laisse.

A part ceux-ci, je n'ai eu réellement qu'à adapter deux fichiers, et ce qu'ils doivent contenir est étonnement court.

- `rc` devrait contenir :

```
#!/bin/sh
/etc/mount -av
/bin/hostname amorce_racine
```

  et vous n'avez pas besoin en réalité d'utiliser `hostname`. Même `mount` n'est nécessaire que pour monter `/proc` pour `ps` ; Linux tournerait sans. ((NdT: vous voudrez probablement ajouter quelque chose comme `/bin/loadkeys /etc/fr-latin1.fr` pour avoir un clavier français. La présence de `mount` dans `/etc` comme celle de `loadkeys` dans `/bin` et celle de `fr-latin1.map` dans `/etc` n'est pas très conforme au « Linux FileSystem Standard ». Sur votre partition disque, ce serait mal)).

- `fstab` devrait contenir :

```
/dev/fd0        /               ext2    defaults
/proc           /proc           proc    defaults
```

  Je ne crois pas que la première ligne soit nécessaire non plus, mais il se trouve que si je l'enlève, `mount` ne montera pas `/proc`.

Inittab devrait être correct tel quel, à moins que vous ne vouliez vous assurer que des utilisateurs ne puissent se loger via le port série. Pour empêcher cela, mettez en commentaire les lignes de `/etc/getty` qui ont des `ttys` ou des `ttyS` à la fin de la ligne. Quoiqu'il en soit, il n'est pas totalement inutile de laisser les/des `tty` ((tout court)) de façon à ce que vous puissez quand même vous loger vous.

Pour le reste, copier simplement tous les fichiers texte dans le répertoire `/etc` ainsi que tous les exécutables dont vous n'êtes pas sûr qu'ils ne soient pas utiles. Comme modèle, voir l'exemple dans la section « Exemple de liste (ls-lR) d'une disquette amorce/racine ».

4.4.2 /bin

C'est un endroit pratique pour mettre les utilitaires supplémentaires dont vous avez besoin pour effectuer les opérations de base ; des utilitaires comme `ls`, `mv`, `cat`, `dd`, etc...

Voir la section intitulée « Exemple de liste (ls-lR) d'une disquette amorce/racine » pour la liste des fichiers que j'ai mis dans mon répertoire `/bin`. Vous remarquerez peut-être que je n'ai mis aucun des utilitaires requis pour récupérer depuis une archive, comme `cpio`, `tar`, `gzip`, etc... En effet, je les mets sur une disquette utilitaires pour économiser l'espace de la disquette amorce/racine. Une fois que j'ai démarré avec celle-ci, elle est copiée dans le disque mémoire laissant le lecteur de disquette libre pour monter une autre disquette (la disquette utilitaires donc). Je la monte habituellement dans `/usr`.

La création d'une disquette utilitaire est décrite ci-dessous dans la section intitulée « Ajouter des disquettes utilitaires ».

4.4.3 /lib

Deux bibliothèques doivent être présentes pour lancer de nombreux programmes sous Linux :

- `ld.so`
- `libc.so.4`

Si elles ne sont pas trouvées dans votre répertoire `/lib`, alors le système est dans l'impossibilité de démarrer. Si vous avez de la chance peut-être verrez-vous un message vous expliquant pourquoi.

Elles doivent être présentes dans un répertoire existant, `/lib`. Remarquez que `libc.so.4` peut être un lien symbolique vers une bibliothèque avec un numéro de version inclus dans le nom du fichier. Si vous faites :

```
ls -l /lib
```

vous verrez quelque chose comme :

```
libc.so.4 -> libc.so.4.5.21
```

Dans ce cas, la bibliothèque libc dont vous avez besoin est `libc.so.4.5.21`. ((NDT : naturellement il se peut même que ce soit un fichier différent de `libc.so.4` comme `libc.so.5`))

4.5 LILO

4.5.1 Aperçu

Pour que l'amorce/racine soit utile, il faut qu'elle soit amorçable. Pour cela, la manière la plus simple (la seule ?) est d'installer un lanceur, qui est un morceau de programme contenu dans la disquette secteur 0, cylindre 0. Voir la section « Disquette d'amorce » pour un aperçu du déroulement du démarrage.

LILO est un lanceur disponible dans n'importe quel site ftp Linux. Il vous permet de déterminer entre autres :

- Le périphérique à utiliser pour la racine.

– S'il faut utiliser un disque virtuel en mémoire.

4.5.2 Exemple de fichier configuration LILO

C'est un endroit pratique pour fixer certaines options de démarrage du noyau. Mon fichier de configuration LILO racine/amorce, utilisé avec LILO 0.15 est :

```
boot = /dev/fd0
install = ./mnt/boot.b
map = ./mnt/lilo.map
delay = 50
message = ./mnt/lilo.msg
timeout = 150
compact
image = ./mnt/vmlinux
        ramdisk = 1440
        root = /dev/fd0
```

Remarquez que boot.b, lilo.msg et le noyau doivent d'abord avoir été copiés sur la disquette en utilisant une commande comme :

```
cp /boot/boot.b ./mnt
```

Si ça n'est pas fait, et si le disque dur n'est pas disponible, LILO ne fonctionnera pas correctement à l'amorçage. Mauvaise idée : créer une disquette de secours qui a besoin du disque dur pour se lancer.

Je lance lilo avec la commande :

```
/sbin/lilo -C <fichier configuration>
```

Je le lance dans répertoire qui contient le répertoire `mnt` sur lequel j'ai monté la disquette ((NdT: i.e. `/`). Ceci signifie que j'indique à LILO d'installer un lanceur sur le périphérique d'amorçage (`/dev/fd0` dans ce cas), pour lancer un noyau dans le répertoire racine de la disquette.

J'ai aussi spécifié que je voulais le périphérique racine comme étant la disquette et qu'un disque mémoire serait créé avec 1440 1K blocs, la taille d'une disquette. Comme j'avais créé un système de fichiers `ext2` sur la disquette, ceci complète les conditions requises pour que Linux puisse copier au démarrage le périphérique d'amorce dans un disque mémoire.

Les caractéristiques du disque virtuel en mémoire sont décrites dans la section intitulée « Disque virtuel en mémoire et disquettes racines » plus haut.

Ajoutons aussi que le paramètre « single » fait que Linux ne démarrera qu'en mode simple utilisateur. Ceci peut être utile pour empêcher les utilisateurs de se connecter à partir des ports série. ((NdT: voir `telinit` et `/etc/inittab` pour en savoir plus sur le mode simple utilisateur)).

J'utilise aussi des commandes `DELAY`, `MESSAGE` et `TIMEOUT`, de telle sorte que lorsque je lance le disque, LILO me donne la possibilité d'entrer des options en ligne de commande si je le désire. Je n'en n'ai pas besoin pour l'instant, mais qui sait de quoi demain sera fait ? Je peux avoir besoin de monter une autre partition ou de monter un système de fichiers en lecture seule.

Le fichier message que j'utilise contient :

```
Linux - Disquette d'amorce/racine
=================================

Entrer une ligne de commande de la forme :

      vmlinux [ options ]

Si rien n'est entre, Linux sera charge avec les
parametres par defaut apres 15 secondes.
```

Cela permet simplement de me rappeler des options que j'ai choisies.

Les lecteurs sont invités à consulter la documentation de LILO, avec attention, avant d'installer quoi que ce soit. Il est assez facile de détruire des partitions si vous utilisez le mauvais paramètre `boot =`. Si vous manquez d'expérience, NE lancez PAS LILO à moins que vous compreniez tout, et que vous ayez vérifié trois fois vos paramètres.

4.5.3 Enlever LILO

Puisque nous parlons de LILO, voici une chose à ajouter : si vous avez un problème avec LILO sur un disque contenant DOS, vous pouvez toujours remplacer le secteur d'amorçage par le lanceur DOS en utilisant la commande DOS :

```
FDISK /MBR
```

où MBR signifie « Master Boot Record ». Remarquez que certains puristes sont en désaccord avec cette interprétation et peut-être à raison, mais en tout cas, ça marche.

4.5.4 Options de LILO utiles

LILO a plusieurs options utiles à garder en tête lors de la création de disquettes d'amorce.

- Option en ligne de commande - vous pouvez entrer des options en ligne de commande pour définir le périphérique racine, la taille du disque mémoire, des paramètres pour les périphériques, ou d'autres choses encore. Si vous mettez une commande `DELAY = nn` dans votre fichier de configuration LILO, alors ce dernier s'arrêtera pour vous permettre de choisir le noyau à charger, ainsi que sur la même ligne toute option. Par exemple :

  ```
  vmlinux aha152x=0x340,11,3,1 ro
  ```

  passera les paramètres `aha152x` suivants au pilote du disque aha152x SCSI (sous réserve que ce pilote ait été inclus dans le noyau lors de sa compilation), et demandera à ce que la racine soit montée en lecture seule (`ro`).

– L'option « lock » en ligne de commande. Cette option demande à LILO de conserver la ligne de commande entrée comme la ligne de commande par défaut à utiliser pour tous les prochains amorçages. C'est particulièrement utile lorsque vous avez un périphérique qui ne peut pas être sélectionné automatiquement. En utilisant lock, vous pouvez éviter d'avoir à entrer les paramètres du périphérique à chaque démarrage.

  Par exemple :

```
vmlinux aha152x=0x340,11,3,1 root=/dev/sda8 ro lock
```

– Commande de configuration APPEND. Elle vous permet de donner une liste de paramètres de périphériques. C'est une alternative à lock. Remarque : toutes les parties de la forme <nom>=<valeur> DOIVENT être mises entre guillemets. Par exemple :

```
APPEND = "aha152x=0x340,11,3,1"
```

– Commande de configuration DELAY. Elle oblige LILO à faire une pause d'un certain nombre de dixièmes de seconde, et permet à l'utilisateur d'interrompre l'amorçage automatique de la ligne de commande par défaut pour en entrer une autre.

5 Exemples

5.1 Liste du contenu de disquettes

Il s'agit de la liste des contenus des répertoires et des fichiers que je garde sur mon disque dur pour créer des disquettes amorce/racine et utilitaires. Cela donne une idée de quels fichiers mettre dans /etc et /bin sur ses disquettes.

Des exemples de scripts dans la section suivante utilisent ces répertoires et fichiers comme modèles pour créer les disquettes.

5.1.1 Liste (ls -lR) du contenu des disquettes amorce/racine.

Liste (ls -lR) du répertoire boot_disk pour les disquettes amorce/racine :

```
total 226
drwxr-xr-x   2 root     root         1024 Oct  8 13:40 bin/
drwxr-xr-x   2 root     root         3072 Sep  8 16:37 dev/
drwxr-xr-x   2 root     root         1024 Oct  8 12:38 etc/
drwxr-xr-x   2 root     root         1024 Sep 10 14:58 lib/
-rw-r--r--   1 root     root       297956 Jan 25 21:55 vmlinux

boot_disk/bin:
total 366
-rwxr-xr-x   1 root     root         4376 Sep  9 21:34 cat*
-rwxr-xr-x   1 root     root         4112 Sep  9 21:34 chown*
-rwxr-xr-x   1 root     root        12148 Sep  9 21:34 cp*
```

```
-rwxr-xr-x   1 root     root         4376 Sep  9 21:34 cut*
-rwxr-xr-x   1 root     root         7660 Sep  9 21:34 dd*
-rwxr-xr-x   1 root     root         4696 Sep  9 21:34 df*
-rwx--x--x   1 root     root         1392 Sep 10 14:13 hostname*
-rwxr-xr-x   1 root     root         5252 Sep  9 21:34 ln*
-rwsr-xr-x   1 root     root         6636 Sep  9 21:34 login*
-rwxr-xr-x   1 root     root        13252 Sep  9 21:34 ls*
-rwxr-xr-x   1 root     root         4104 Sep  9 21:34 mkdir*
-rwxr-xr-x   1 root     root        21504 Sep 10 15:27 more*
-rwxr-xr-x   1 root     root         6744 Sep  9 21:34 mv*
-rwxr-xr-x   1 root     root         9780 Sep  9 21:34 ps*
-rwxr-xr-x   1 root     root         5076 Sep  9 21:34 rm*
-r-xr-xr-x   1 root     root        12604 Sep  9 21:34 sed*
-rwxr-xr-x   1 root     root       222208 Sep  9 21:34 sh*
-rws--x--x   1 root     root        16464 Sep  9 21:34 su*
-rwxr-xr-x   1 root     root         1216 Sep  9 21:34 sync*

boot_disk/dev:
total 73
-rwxr-xr-x   1 root     root         8331 Sep  8 16:31 MAKEDEV*
crw-r--r--   1 root     root      10,   3 Sep  8 16:31 bmouseatixl
crw-r--r--   1 root     root      10,   0 Sep  8 16:31 bmouselogitec
crw-r--r--   1 root     root      10,   2 Sep  8 16:31 bmousems
crw-r--r--   1 root     root      10,   1 Sep  8 16:31 bmouseps2
crw-------   1 root     root       0,   0 Sep  8 16:31 boot0
crw-r--r--   1 root     root       4,   0 Sep  8 16:31 console
crw-r--r--   1 root     root       5,  64 Sep  8 16:31 cua0
crw-r--r--   1 root     root       5,  65 Sep  8 16:31 cua1
crw-r--r--   1 root     root       5,  66 Sep  8 16:31 cua2
crw-r--r--   1 root     root       5,  67 Sep  8 16:31 cua3
brw-r--r--   1 root     root       2,   0 Sep  8 16:31 fd0
brw-r--r--   1 root     root       2,  12 Sep  8 16:31 fd0D360
brw-r--r--   1 root     root       2,  16 Sep  8 16:31 fd0D720
brw-r--r--   1 root     root       2,  28 Sep  8 16:31 fd0H1440
brw-r--r--   1 root     root       2,  12 Sep  8 16:31 fd0H360
brw-r--r--   1 root     root       2,  16 Sep  8 16:31 fd0H720
brw-r--r--   1 root     root       2,  16 Sep  8 16:31 fd0Q720
brw-r--r--   1 root     root       2,   4 Sep  8 16:31 fd0d360
brw-r--r--   1 root     root       2,   8 Sep  8 16:31 fd0h1200
brw-r--r--   1 root     root       2,  20 Sep  8 16:31 fd0h360
brw-r--r--   1 root     root       2,  24 Sep  8 16:31 fd0h720
brw-r--r--   1 root     root       2,  24 Sep  8 16:31 fd0q720
brw-r--r--   1 root     root       2,   1 Sep  8 16:31 fd1
brw-r--r--   1 root     root       2,  13 Sep  8 16:31 fd1D360
brw-r--r--   1 root     root       2,  17 Sep  8 16:31 fd1D720
brw-r--r--   1 root     root       2,  29 Sep  8 16:31 fd1H1440
brw-------   1 root     root       2,  31 Sep  8 16:31 fd1H1722
brw-r--r--   1 root     root       2,  13 Sep  8 16:31 fd1H360
```

```
brw-r--r--   1 root     root       2,  17 Sep  8 16:31 fd1H720
brw-r--r--   1 root     root       2,  17 Sep  8 16:31 fd1Q720
brw-r--r--   1 root     root       2,   5 Sep  8 16:31 fd1d360
brw-r--r--   1 root     root       2,   9 Sep  8 16:31 fd1h1200
brw-r--r--   1 root     root       2,  21 Sep  8 16:31 fd1h360
brw-r--r--   1 root     root       2,  25 Sep  8 16:31 fd1h720
brw-r--r--   1 root     root       2,  25 Sep  8 16:31 fd1q720
brw-r-----   1 root     root       3,   0 Sep  8 16:31 hda
brw-r-----   1 root     root       3,   1 Sep  8 16:31 hda1
brw-r-----   1 root     root       3,   2 Sep  8 16:31 hda2
brw-r-----   1 root     root       3,   3 Sep  8 16:31 hda3
brw-r-----   1 root     root       3,   4 Sep  8 16:31 hda4
brw-r-----   1 root     root       3,   5 Sep  8 16:31 hda5
brw-r-----   1 root     root       3,   6 Sep  8 16:31 hda6
brw-r-----   1 root     root       3,   7 Sep  8 16:31 hda7
brw-r-----   1 root     root       3,   8 Sep  8 16:31 hda8
brw-r-----   1 root     root       3,  64 Sep  8 16:31 hdb
brw-r-----   1 root     root       3,  65 Sep  8 16:31 hdb1
brw-r-----   1 root     root       3,  66 Sep  8 16:31 hdb2
brw-r-----   1 root     root       3,  67 Sep  8 16:31 hdb3
brw-r-----   1 root     root       3,  68 Sep  8 16:31 hdb4
brw-r-----   1 root     root       3,  69 Sep  8 16:31 hdb5
brw-r-----   1 root     root       3,  70 Sep  8 16:31 hdb6
brw-r-----   1 root     root       3,  71 Sep  8 16:31 hdb7
brw-r-----   1 root     root       3,  72 Sep  8 16:31 hdb8
crw-r-----   1 root     root       1,   2 Sep  8 16:31 kmem
brw-------   1 root     root      12,   0 Sep  8 16:31 loop0
brw-------   1 root     root      12,   1 Sep  8 16:31 loop1
crw-r--r--   1 root     root       6,   0 Sep  8 16:31 lp0
crw-r--r--   1 root     root       6,   1 Sep  8 16:31 lp1
crw-r--r--   1 root     root       6,   2 Sep  8 16:31 lp2
brw-r--r--   1 root     root      12,   0 Sep  8 16:31 mcd0
crw-r-----   1 root     root       1,   1 Sep  8 16:31 mem
crw-r--r--   1 root     root       5,  65 Sep  8 16:31 modem
crw-r--r--   1 root     root       5,  64 Sep  8 16:31 mouse
crw-r--r--   1 root     root      27,   4 Sep  8 16:31 nrft0
crw-r--r--   1 root     root      27,   5 Sep  8 16:31 nrft1
crw-r--r--   1 root     root      27,   6 Sep  8 16:31 nrft2
crw-r--r--   1 root     root      27,   7 Sep  8 16:31 nrft3
crw-------   1 root     root       9, 128 Sep  8 16:31 nrmt0
crw-r--r--   1 root     root       1,   3 Sep  8 16:31 null
crw-r-----   1 root     root       6,   0 Sep  8 16:31 par0
crw-r-----   1 root     root       6,   1 Sep  8 16:31 par1
crw-r-----   1 root     root       6,   2 Sep  8 16:31 par2
crw-r-----   1 root     root       1,   4 Sep  8 16:31 port
crw-r--r--   1 root     root      10,   1 Sep  8 16:31 ps2aux
crw-r--r--   1 root     root       4, 128 Sep  8 16:31 ptyp0
crw-r--r--   1 root     root       4, 129 Sep  8 16:31 ptyp1
```

```
crw-r--r--   1 root     root       4, 130 Sep  8 16:31 ptyp2
crw-r--r--   1 root     root       4, 131 Sep  8 16:31 ptyp3
crw-r--r--   1 root     root       4, 132 Sep  8 16:31 ptyp4
crw-r--r--   1 root     root       4, 133 Sep  8 16:31 ptyp5
crw-r--r--   1 root     root       4, 134 Sep  8 16:31 ptyp6
crw-r--r--   1 root     root       4, 135 Sep  8 16:31 ptyp7
crw-r--r--   1 root     root       4, 136 Sep  8 16:31 ptyp8
crw-r--r--   1 root     root       4, 137 Sep  8 16:31 ptyp9
crw-r--r--   1 root     root       4, 138 Sep  8 16:31 ptypa
crw-r--r--   1 root     root       4, 139 Sep  8 16:31 ptypb
crw-r--r--   1 root     root       4, 140 Sep  8 16:31 ptypc
crw-r--r--   1 root     root       4, 141 Sep  8 16:31 ptypd
crw-r--r--   1 root     root       4, 142 Sep  8 16:31 ptype
crw-r--r--   1 root     root       4, 143 Sep  8 16:31 ptypf
brw-r-----   1 root     root       1,   0 Sep  8 16:31 ram
crw-r--r--   1 root     root      27,   0 Sep  8 16:31 rft0
crw-r--r--   1 root     root      27,   1 Sep  8 16:31 rft1
crw-r--r--   1 root     root      27,   2 Sep  8 16:31 rft2
crw-r--r--   1 root     root      27,   3 Sep  8 16:31 rft3
crw-------   1 root     root       9,   0 Sep  8 16:31 rmt0
brw-r-----   1 root     root       8,   0 Sep  8 16:31 sda
brw-r-----   1 root     root       8,   1 Sep  8 16:31 sda1
brw-r-----   1 root     root       8,   2 Sep  8 16:31 sda2
brw-r-----   1 root     root       8,   3 Sep  8 16:31 sda3
brw-r-----   1 root     root       8,   4 Sep  8 16:31 sda4
brw-r-----   1 root     root       8,   5 Sep  8 16:31 sda5
brw-r-----   1 root     root       8,   6 Sep  8 16:31 sda6
brw-r-----   1 root     root       8,   7 Sep  8 16:31 sda7
brw-r-----   1 root     root       8,   8 Sep  8 16:31 sda8
brw-r-----   1 root     root       8,  16 Sep  8 16:31 sdb
brw-r-----   1 root     root       8,  17 Sep  8 16:31 sdb1
brw-r-----   1 root     root       8,  18 Sep  8 16:31 sdb2
brw-r-----   1 root     root       8,  19 Sep  8 16:31 sdb3
brw-r-----   1 root     root       8,  20 Sep  8 16:31 sdb4
brw-r-----   1 root     root       8,  21 Sep  8 16:31 sdb5
brw-r-----   1 root     root       8,  22 Sep  8 16:31 sdb6
brw-r-----   1 root     root       8,  23 Sep  8 16:31 sdb7
brw-r-----   1 root     root       8,  24 Sep  8 16:31 sdb8
brw-------   1 root     root       8,  32 Sep  8 16:31 sdc
brw-------   1 root     root       8,  33 Sep  8 16:31 sdc1
brw-------   1 root     root       8,  34 Sep  8 16:31 sdc2
brw-------   1 root     root       8,  35 Sep  8 16:31 sdc3
brw-------   1 root     root       8,  36 Sep  8 16:31 sdc4
brw-------   1 root     root       8,  37 Sep  8 16:31 sdc5
brw-------   1 root     root       8,  38 Sep  8 16:31 sdc6
brw-------   1 root     root       8,  39 Sep  8 16:31 sdc7
brw-------   1 root     root       8,  40 Sep  8 16:31 sdc8
brw-------   1 root     root       8,  48 Sep  8 16:31 sdd
```

```
brw-------   1 root     root       8,  49 Sep  8 16:31 sdd1
brw-------   1 root     root       8,  50 Sep  8 16:31 sdd2
brw-------   1 root     root       8,  51 Sep  8 16:31 sdd3
brw-------   1 root     root       8,  52 Sep  8 16:31 sdd4
brw-------   1 root     root       8,  53 Sep  8 16:31 sdd5
brw-------   1 root     root       8,  54 Sep  8 16:31 sdd6
brw-------   1 root     root       8,  55 Sep  8 16:31 sdd7
brw-------   1 root     root       8,  56 Sep  8 16:31 sdd8
brw-------   1 root     root       8,  64 Sep  8 16:31 sde
brw-------   1 root     root       8,  65 Sep  8 16:31 sde1
brw-------   1 root     root       8,  66 Sep  8 16:31 sde2
brw-------   1 root     root       8,  67 Sep  8 16:31 sde3
brw-------   1 root     root       8,  68 Sep  8 16:31 sde4
brw-------   1 root     root       8,  69 Sep  8 16:31 sde5
brw-------   1 root     root       8,  70 Sep  8 16:31 sde6
brw-------   1 root     root       8,  71 Sep  8 16:31 sde7
brw-------   1 root     root       8,  72 Sep  8 16:31 sde8
brw-r--r--   1 root     root      11,   0 Sep  8 16:31 sr0
brw-r-----   1 root     root      11,   1 Sep  8 16:31 sr1
brw-r-----   1 root     root      11,   2 Sep  8 16:31 sr2
brw-r-----   1 root     root       3,   1 Sep  8 16:31 swap
crw-r--r--   1 root     root       5,   0 Sep  8 16:31 tty
crw-r--r--   1 root     root       4,   0 Sep  8 16:31 tty0
crw-------   1 root     root       4,   1 Sep  8 16:31 tty1
crw-r--r--   1 root     root       4,   2 Sep  8 16:31 tty2
-rw-r--r--   1 root     root           20 Sep  8 16:31 tty21
crw-r--r--   1 root     root       4,   3 Sep  8 16:31 tty3
crw-r--r--   1 root     root       4,   4 Sep  8 16:31 tty4
crw-r--r--   1 root     root       4,   5 Sep  8 16:31 tty5
crw-r--r--   1 root     root       4,   6 Sep  8 16:31 tty6
crw-------   1 root     root       4,   7 Sep  8 16:31 tty7
crw-------   1 root     root       4,   8 Sep  8 16:31 tty8
crw-r--r--   1 root     root       4,  64 Sep  8 16:31 ttyS0
crw-r--r--   1 root     root       4,  65 Sep  8 16:31 ttyS1
crw-r--r--   1 root     root       4,  66 Sep  8 16:31 ttyS2
crw-r--r--   1 root     root       4, 192 Sep  8 16:31 ttyp0
crw-r--r--   1 root     root       4, 193 Sep  8 16:31 ttyp1
crw-r--r--   1 root     root       4, 194 Sep  8 16:31 ttyp2
crw-r--r--   1 root     root       4, 195 Sep  8 16:31 ttyp3
crw-r--r--   1 root     root       4, 196 Sep  8 16:31 ttyp4
crw-r--r--   1 root     root       4, 197 Sep  8 16:31 ttyp5
crw-r--r--   1 root     root       4, 198 Sep  8 16:31 ttyp6
crw-r--r--   1 root     root       4, 199 Sep  8 16:31 ttyp7
crw-r--r--   1 root     root       4, 200 Sep  8 16:31 ttyp8
crw-r--r--   1 root     root       4, 201 Sep  8 16:31 ttyp9
crw-r--r--   1 root     root       4, 202 Sep  8 16:31 ttypa
crw-r--r--   1 root     root       4, 203 Sep  8 16:31 ttypb
crw-r--r--   1 root     root       4, 204 Sep  8 16:31 ttypc
```

```
crw-r--r--   1 root     root       4, 205 Sep  8 16:31 ttypd
crw-r--r--   1 root     root       4, 206 Sep  8 16:31 ttype
crw-r--r--   1 root     root       4, 207 Sep  8 16:31 ttypf
-rw-------   1 root     root        63488 Sep  8 16:31 ttys0
crw-r--r--   1 root     root       4,  67 Sep  8 16:31 ttys3
crw-r--r--   1 root     root       1,   5 Sep  8 16:31 zero

boot_disk/etc:
total 173
-rw-r--r--   1 root     root           53 Sep  8 18:48 boot.env
-rwxr-xr-x   1 root     root        27408 Sep  8 18:48 e2fsck*
-rwxr-xr-x   1 root     root        18540 Sep  8 18:48 fdisk*
-rw-r--r--   1 root     root           69 Oct  8 12:27 fstab
-r-x------   1 root     root        13312 Sep  8 18:48 getty*
-rw-r--r--   1 root     root          334 Sep  8 18:48 group
-rw-r--r--   1 root     root           12 Sep  8 18:48 host.conf
-rw-r--r--   1 root     root           62 Sep  8 18:48 hosts
-r-x------   1 root     root         6684 Sep  8 18:48 ifconfig*
-rwxr-xr-x   1 root     root        11492 Sep  8 18:48 init*
-rw-r--r--   1 root     root         1017 Sep  9 22:12 inittab
-rw-r--r--   1 root     root            0 Oct  8 12:19 issue
-rw-r-----   1 root     root         5137 Sep  8 18:48 login.defs
-rwxr-xr-x   1 root     root        14028 Sep  8 18:48 mke2fs*
-rwxr-x---   1 root     root         2436 Sep  8 18:48 mkswap*
-rwxr-xr-x   1 root     root        11288 Sep  8 18:48 mount*
-rw-r--r--   1 root     root          327 Sep  8 18:48 passwd
-rwxr-xr-x   1 root     root          383 Sep 10 16:02 profile*
-rw-r--r--   1 root     root           94 Sep  8 18:48 protocols
-rwxr-xr-x   1 root     root          334 Oct  8 12:27 rc*
-rwxr-xr-x   1 root     root         9220 Sep  8 18:48 reboot*
-r-x------   1 root     root         4092 Sep  8 18:48 route*
-rw-r--r--   1 root     root           20 Sep  8 18:48 securetty
-rw-r--r--   1 root     root         9749 Sep  8 18:48 services
-rw-r--r--   1 root     root           36 Sep  8 18:48 shells
-rwxr-xr-x   1 root     root        13316 Sep  8 18:48 shutdown*
-rwxr-xr-x   1 root     root         2496 Sep  8 18:48 swapoff*
-rwxr-xr-x   1 root     root         2496 Sep  8 18:48 swapon*
-rw-r--r--   1 root     root         5314 Sep  8 18:48 termcap
-rwxr-xr-x   1 root     root         5412 Sep  8 18:48 umount*
-rw-r--r--   1 root     root          224 Sep  8 18:48 utmp
-rw-r--r--   1 root     root          280 Sep  8 18:48 wtmp

boot_disk/lib:
total 629
-rwxr-xr-x   1 root     root        17412 Sep 10 14:58 ld.so*
-rwxr-xr-x   1 root     root       623620 Sep  8 18:33 libc.so.4*
```

((NdT: vous voudrez probablement ajouter `loadkeys` dans un répertoire d'exécutables , ex: `/bin`, et `fr-latin1.map` dans un répertoire données, ex: `/etc`, pour pouvoir passer en mode « clavier français ».

Chez moi `fr-latin1.map` est dans `/usr/lib/kbd/keytables`))

5.1.2 Liste (ls-lR) des répertoires des disquettes utilitaires

La liste ls-lR du répertoire util_disk :

```
total 1
drwxr-xr-x   2 root     root         1024 Sep 10 16:05 bin/

util_disk/bin:
total 897
-rwxr-xr-x   1 root     root        41984 Sep 10 14:11 cpio*
-rwxr-xr-x   1 root     root       504451 Sep  9 21:39 ftape.o*
-rwxr-xr-x   1 root     root        63874 Sep  9 21:40 gzip*
-rwxr-xr-x   1 root     root        13316 Sep  9 21:34 insmod*
-rwxr-xr-x   1 root     root           58 Sep  9 21:34 lsmod*
-rwxr-xr-x   1 root     root         3288 Sep  9 21:34 mknod*
-rwxr-xr-x   1 root     root         9220 Sep  9 21:34 rmmod*
-rwxr-xr-x   1 root     root       226308 Sep  9 22:13 tar*
```

5.2 Scripts pour créer des disquettes

Il y a deux scripts :

- mkroot : crée une disquette racine ou amorce/racine.
- mkutil : crée une disquette utilitaires.

Tout deux sont prévus pour être lancés dans le répertoire père de `boot_disk` et `util_disk`, chacun contenant tout ce qui doit être copié sur sa disquette. Remarquez que ces scripts NE détermineront/copieront PAS automatiquement tous les fichiers pour vous. _Vous_ choisissez quels sont les fichiers nécessaires, créez les répertoires et copiez les fichiers dans les répertoires. Les scripts sont des exemples qui copieront le contenu de ces répertoires. Attention, ce sont des scripts simples, et ne sont pas prévus pour le novice.

Les deux scripts contiennent des variables de configuration au début ; ce qui vous permet de les adapter aisement pour fonctionner n'importe où. D'abord, créez les répertoires modèles et copiez les fichiers. Puis vérifiez les variables de configuration au début de chaque script, en les changeant partout où c'est nécessaire.

5.2.1 mkroot - Make Root or Boot/Root Diskette

```
# mkroot: cree une disquette amorce/racine en creant un systeme
#       de fichiers sur celle-ci, puis la monte et copie les
#       fichiers d'un modele.
#       Note : le modele a copier doit deja exister. Changez-les
```

```
#       variables de configuration pour adapter a votre systeme
#
# utilisation : mkroot [nokernel]
#               Si le parametre est omis, le noyau et LILO sont
#               copies.

# Copyright original :
# Copyright (c) Graham Chapman 1994. All rights reserved.
# Permission is granted for this material to be freely
# used and distributed, provided the source is acknowledged.
# No warranty of any kind is provided. You use this material
# at your own risk.
#
# Fichier modifie (traduction) par Cedric Adjih.

# Variables de configuration...
BOOTDISKDIR=./boot_disk       # nom du repertoire "disquette d'amorce"
MOUNTPOINT=./mnt              # repertoire sur lequel monter la disquette
LILODIR=/sbin                 # repertoire contenant lilo
LILOBOOT=/boot/boot.b         # secteur d'amorce de lilo
LILOMSG=./lilo.msg            # message qu'affichera lilo lors de l'amorce
LILOCONFIG=./lilo.conf        # parametres lilo pour une amorce/racine
DISKETTEDEV=/dev/fd0          # peripherique disquette

echo $0 : cree une disquette d'amorce/racine
echo Attention : les donnees de la disquette seront detruites.
echo Inserez une disquette dans $DISKETTEDEV et tapez une touche...
read anything

mke2fs $DISKETTEDEV
if [ $? -ne 0 ]
then
        echo Erreur en executant mke2fs.
        exit
fi

mount -t ext2 $DISKETTEDEV $MOUNTPOINT
if [ $? -ne 0 ]
then
        echo Impossible de monter la disquette.
        exit
fi

# Copie des repertoires contenant des fichiers.
for i in bin etc lib
do
        cp -dpr $BOOTDISKDIR/$i $MOUNTPOINT
done
```

```
# Copie dev *sans* essayer de copier les fichiers dedans.
cp -dpR $BOOTDISKDIR/dev $MOUNTPOINT

# Cree les fichiers necessaires.
mkdir $MOUNTPOINT/proc
mkdir $MOUNTPOINT/tmp
mkdir $MOUNTPOINT/mnt
mkdir $MOUNTPOINT/usr

# Copie le noyau.
if [ "$1" != "nokernel" ]
then
        echo "Copie du noyau."
        cp $BOOTDISKDIR/vmlinux $MOUNTPOINT
        echo Noyau copie.

        # Installe lilo
        cp $LILOBOOT $MOUNTPOINT
        cp $LILOMSG $MOUNTPOINT
        $LILODIR/lilo -C $LILOCONFIG
        echo LILO installe.
fi

umount $MOUNTPOINT

echo Disquette prete.
```

5.2.2 mkutil - Crée une disquette utilitaires

```
# mkutil: cree une disquette utilitaires en creant un systeme
#       de fichiers sur celle-ci, puis en la monte et copie les
#       fichiers d'un modele.
#       Note : le modele a copier doit deja exister. Changez les
#       variables de configuration pour adapter a votre systeme
#

# Copyright original :
# Copyright (c) Graham Chapman 1994. All rights reserved.
# Permission is granted for this material to be freely
# used and distributed, provided the source is acknowledged.
# No warranty of any kind is provided. You use this material
# at your own risk.
#
# Fichier modifie (traduction) par Cedric Adjih.

# Configuration variables...
UTILDISKDIR=./util_disk       # nom du repertoire contenant le modele
```

```
MOUNTPOINT=./mnt              # repertoire sur lequel monter la disquette
DISKETTEDEV=/dev/fd0          # peripherique disquette

echo $0 : cree une disquette utilitaire
echo Attention : les donnees de la disquette seront detruites.
echo Inserez une disquette dans $DISKETTEDEV et tapez une touche...
read anything

mke2fs $DISKETTEDEV
if [ $? -ne 0 ]
then
        echo Erreur en executant mke2fs.
        exit
fi

# Any file system type would do here
mount -t ext2 $DISKETTEDEV $MOUNTPOINT
if [ $? -ne 0 ]
then
        echo Impossible de monter la disquette.
        exit
fi

# copy the directories containing files
cp -dpr $UTILDISKDIR/bin $MOUNTPOINT

umount $MOUNTPOINT

echo Disquette utilitaire prete.
```

6 FAQ

6.1 Comment faire une disquette d'amorce avec le pilote XXX ?

Le plus simple est peut-être de récupérer un noyau de la distribution Slackware de votre crêmerie la plus proche. Les noyaux de la Slackware sont des noyaux standards qui tentent d'inclure le plus de pilotes possibles. Si vous avez un contrôleur SCSI ou IDE, vous avez des chances de les trouver là.

Allez dans le répertoire `a1` et choisissez sun noyau IDE ou SCSI, selon le type de contrôleur que vous avez. Vérifiez dans le fichier `xxxxkern.cfg` si les pilotes nécessaires sont inclus ; si c'est le cas, récupérez donc `xxxxkern.tgz`, installez-le comme décrit plus haut sur la disquette d'amorce.

((NDT : vous pouvez trouver d'autres noyaux dans `XXXX/distributions/slakinst/kernels/*` dont la liste des périphériques contenus est cette fois dans le `config.in`. D'autre part les noyaux de la Slackware ne sont pas forcement investis de super-pouvoirs, donc le noyau que vous compilez vous-même fera sûrement l'affaire)).

Vous devez alors vérifier le périphérique racine du noyau, en utilisant la commande `rdev` :

```
rdev vmlinuz
```

`rdev` affichera alors la racine du noyau. Si ce n'est pas celle que vous voulez, utilisez encore une fois rdev pour la changer. Par exemple, le noyau que j'ai essayé avait pour racine `/dev/sda3`, mais ma partition scsi racine était `/dev/sda8`. Pour utiliser une disquette comme racine, vous aurez à utiliser la commande :

```
rdev vmlinuz /dev/fd0
```

Si vous voulez aussi savoir comment créer une disquette racine Slackware, c'est hors du sujet de ce document ; je suggère donc que vous vous référiez au *Linux Install Guide* ((NdT: dont le titre de la version française par René Cougnenc est *Bien débuter sous Linux*)), ou aux fichiers de la distribution Slackware. Voyez la section intitulée « Références ».

6.2 Comment faire pour mettre à jour ma disquette d'amorce avec un nouveau noyau ?

Pour une disquette sans système de fichiers, copiez simplement le noyau sur votre disquette d'amorce avec `dd` ; sinon, pour une disquette amorce/racine, utilisez `cp`. Référez-vous à la section de ce document intitulée « Amorce » pour des détails sur la création d'une disquette d'amorce. La description s'applique aussi bien à la mise à jour d'un noyau sur une disquette d'amorce.

6.3 Comment enlever LILO pour amorcer DOS ?

Ce n'est pas vraiment un sujet de ce document, mais la question est posée si souvent ! Donc voilà, sous DOS utilisez :

```
FDISK /MBR
```

MBR signifie Master Boot Record, et il remplace le secteur d'amorce par celui de DOS, sans toucher à la table des partitions. Certains puristes ne sont pas d'accord, mais même l'auteur de LILO, Werner Almesberger le suggère. C'est simple et efficace.

Vous pouvez aussi utiliser la commande `dd` pour copier l'archive qu'a faite LILO du secteur d'amorce ; voir la documentation de LILO.

6.4 Comment puis-je amorcer si j'ai perdu et mon noyau et ma disquette d'amorce ?

Si vous n'avez pas de disquette d'amorce, la méthode la plus simple est de récupérer un noyau de la Slackware pour votre type de contrôleur (IDE ou SCSI) comme décrit précédemment dans « Comment faire une disquette d'amorce avec un pilote XXX ? ». Vous pouvez même démarrer votre système avec ce noyau, et réparer les dommages.

Le noyau que vous obtiendrez peut ne pas avoir le périphérique racine pointant sur la bonne partition chez vous. Par exemple, le noyau standard SCSI de la Slackware pointe sur `/dev/sda2`. Chez moi ma partition racine est `/dev/sda8`. Dans ce cas cela doit être changé.

Vous pouvez toujours changer le périphérique racine et les options disque mémoire du noyau, même si tout ce que vous avez est un noyau et un autre système d'exploitation comme DOS.

`rdev` change les paramètres du noyau en modifiant les valeurs à des emplacements fixes du fichier contenant le noyau. Vous pouvez faire de même avec un éditeur hexadécimal, quel que soit le système d'exploitation que vous utilisez. Par exemple « Norton Utilities Disk Editor » sous DOS. Vous vérifierez et, le cas échéant, modifierez les octets du noyau aux endroits suivants :

```
0x01F8  Octet de poids faible de la taille du disque memoire.
0x01F9  Octet de poids fort de la taille du disque memoire.
0x01FC  Numero de peripherique, mineur - cf plus bas
0X01FD  Numero de peripherique, majeur - cf plus bas
```

La taille du disque mémoire est le nombre de blocs du disque virtuel en mémoire à créer. Si vous voulez amorcer avec une disquette racine, fixer celle-ci à 1440 en base 10, c'est-à-dire 0x05A0 : donc mettez l'octet 0x01F8 du fichier à 0xA0 et celui en 0x01F9 à 0x05. Ceci réservera assez d'espace pour une disquette.

Le numéros de périphérique majeurs et mineurs doivent être ceux du périphérique contenant votre disquette racine. Des valeurs utiles :

```
peripherique    majeur mineur
/dev/fd0            2     0    premier lecteur de disquette
/dev/hda1           3     1    partition 1 du 1er disque IDE
/dev/sda1           8     1    partition 1 du 1er disque SCSI
/dev/sda8           8     8    partition 8 on 1er disque SCSI
```

Une fois que vous avez fixé ces paramètres, vous pouvez écrire le fichier sur une disquette en utilisant soit « Norton Utilities Disk Editor », soit un programme appelé `rawrite.exe`. Ce programme est inclus dans plusieurs distributions dont la SLS et la Slackware. C'est un programme DOS qui copie un fichier sur une disquette « brute », en commençant par le secteur d'amorce, au lieu de l'écrire sur le système de fichier. Si vous utilisez « Norton Utilities », vous devez écrire le fichier sur une disquette physique commençant au début de la disquette.

6.5 Comment puis-je faire des copies de mes disquettes d'amorce ou racine?

Il n'est pas souhaitable de n'avoir qu'un seul jeu de disquettes de secours. 2 ou 3 doivent être conservées au cas où l'une d'entre elles serait illisible.

Le moyen le plus simple de faire des copies de disquettes, y compris pour les disquettes amorce et les disquettes utilitaires, est d'utiliser `dd` pour copier le contenu de la disquette dans un fichier sur votre disque dur, puis d'utiliser ce même fichier pour faire la copie dans l'autre sens avec une disquette différente.

Pour copier l'original, utilisez la commande :

```
dd if={$<$}peripherique{$>$} of={$<$}nom de fichier{$>$}
```

```
ou      {$<$}peripherique{$>$} = nom du peripherique du lecteur
        de disquette
et      {$<$}nom de fichier{$>$} = nom du fichier que vous
        voulez copier
```

Par exemple, pour copier `/dev/fd0` dans un fichier appelé `/tmp/disquette.copie`, on utilisera

```
dd if=/dev/fd0 of=/tmp/disquette.copie
```

Sans le paramètre `count`, comme ici, ceci signifie qu'une disquette entière de 2880 blocs (3 1/2 haute densité) sera copiée.

Pour copier le fichier obtenu sur une nouvelle disquette, mettez la nouvelle disquette dans le lecteur et utilisez la commande inverse :

```
dd if=<nom de fichier> of=<peripherique>
```

Remarquez que j'ai supposé qu'il n'y avait qu'un seul lecteur de disquette. Si vous en avez deux de même type, vous pouvez copier des disquettes en utilisant :

```
dd if=/dev/fd0 of=/dev/fd1
```

6.6 Comment puis-je amorcer sans taper "ahaxxxx=nn,nn,nn" à chaque fois?

Quand un périphérique disque n'est pas détecté automatiquement, il est nécessaire de fournir au noyau une liste de paramètres comme :

```
aha152x=0x340,11,3,1
```

Cette liste de paramètres peut être fournie de plusieurs manières avec LILO :

- En l'entrant en ligne de commande à chaque démarrage via LILO. Mais, cela lasse.
- En utilisant le mot clé `lock` de LILO pour que celui-ci enregistre la ligne de commande comme étant celle par défaut. Elle sera alors utilisée à chaque démarrage (sauf si l'utilisateur en entre une autre).
- En utilisant l'instruction `APPEND` dans le fichier de configuration de LILO. Attention, la liste des paramètres doit être mise entre guillemets (").

Un exemple de ligne de commande utilisant une liste de paramètres :

```
vmlinux aha152x=0x340,11,3,1 root=/dev/sda1 lock
```

Ceci passera la liste des paramètres périphériques ainsi que le périphérique racine à utiliser (`/dev/sda1`) au noyau. La ligne de commande sera aussi conservée et utilisée dans tous les démarrages ultérieurs.

Un exemple de commande `APPEND` est :

```
APPEND = "aha152x=0x340,11,3,1"
```

Notez que que la liste des paramètres NE doit PAS être entre guillemets sur la ligne de commande, mais DOIT l'être dans la commande `APPEND`.

Notez aussi que pour que la liste des paramètres soit effective, le noyau doit contenir le pilote approprié. Sinon les paramètres ne serviront à rien. Vous devrez recompiler le noyau pour inclure le pilote requis. Pour des détails sur la recompilation du noyau, voyez dans `/usr/src/linux` le fichier `README`, lisez la FAQ Linux, et l'Installation-HOWTO ((et le Kernel-HOWTO)). Ou bien vous vous procurerez un noyau adéquat pour vos périphériques.

Les lecteurs sont fortement invités à lire la documentation de LILO avant de s'essayer à l'installation de LILO. Une utilisation sans précaution de la commande `BOOT` peut endommager les partitions.

6.7 J'ai monté ma partition disque. Comment en utiliser les programmes?

((NdT: section ajoutée.

En supposant que vous ayez amorcé sur une disquette, que votre partition racine soit votre disquette, et que vous puissiez monter votre partition disque ex-partition racine par exemple sur `/mnt`.

C'est ce qui se passe si vous essayez de récupérer une partition racine sur le disque dont un des fichiers vitaux manque pour amorcer.

Vous pouvez bien sûr exécuter directement un programme de votre partition racine, par exemple `/mnt/usr/local/bin/mlist`. Ou vous pouvez ajouter `/mnt/bin`, `/mnt/usr/bin`, ... à votre variable d'environnement `PATH`. Mais il y a des chances pour qu'un programme qui recherche des fichiers dans `/usr/lib` par exemple ne fonctionne pas directement (ex: l'éditeur `emacs`).

Une solution est d'utiliser `chroot` pour lancer un programme en changeant sa partition racine. Par exemple :

```
( cd /mnt ; /mnt/usr/sbin/chroot /mnt /usr/bin/emacs /etc/fstab )
```

lancera `chroot` supposé être ici dans `/usr/sbin` de la partition disque, qui d'abord fixera la racine du programme à `/mnt` : alors tous les accès au système de fichiers ajouteront `/mnt` aux chemins spécifiés. Ainsi lorsque `chroot` lancera l'exécution de son deuxième argument `/usr/bin/emacs`, le noyau Linux traduira celui-ci en `/mnt/usr/bin/emacs`. De même lorsque l'éditeur `emacs` lancé avec l'argument `/etc/fstab`, essayera d'ouvrir ce fichier, le système traduira le chemin du fichier en `/mnt/etc/fstab`.

Il peut-être utile de lancer quelque chose comme :

```
( cd /mnt ; /mnt/usr/sbin/chroot /mnt /bin/bash -login )
```

qui permet d'avoir `bash` comme sur son ancien système, ou encore :

```
( cd /mnt ; /mnt/usr/sbin/chroot /mnt /sbin/agetty 38400 tty7 )
```

qui permet d'avoir un « login: » comme sur son ancienne partition racine, sur la console 7 (taper `Alt-F7`). (7 étant une console sur laquelle il n'y a pas déjà de « login: », à remplacer par un nombre adéquat chez vous).))

7 Références

Dans cette section vvv est utilisé dans le nom des paquetages à la place du numéro de version pour éviter d'indiquer des versions spécifiques. Quand vous récupérez un paquetage, prenez toujours la dernière version, à moins que vous n'ayez de bonnes raisons.

7.1 LILO - Lanceur Linux (LInux LOader)

Ecrit par Werner Almesberger. Excellent lanceur ; sa documentation contient de l'information sur le secteur d'amorce ainsi qu'une description des premières étapes du démarrage.

Ftp: tsx-11.mit.edu:/pub/linux/packages/lilo/lilo.vvv.tar.gz ainsi qu'à sunsite.unc.edu, et d'autres miroirs. ((NdT: Depuis la France il est conseillé d'utiliser un miroir en France comme ftp.ibp.fr)).

7.2 Les FAQ et les HOWTOs Linux

Ils sont disponibles dans de nombreux endroits. Regarder dans les conférences Usenet news.answers et comp.os.linux.announce ((NdT: ainsi que dans comp.os.linux.answers)).

((NdT: les sites des Etats-Unis sont reproduits ici à titre indicatif et parce qu'ils sont dans l'original. Vous trouverez l'intégralité des fichiers dans les miroirs en France :

```
------------------------------------------------------------
ftp.ibp.fr ( 132.227.60.2 ) qui "mirrorise" tsx-11.mit.edu.

Les messages du newsgroup/conference "fr.comp.os.linux" sont
archives sur ftp.ibp.fr et sur renux.

Principaux sites FTP Linux en FRANCE:

    ftp.ibp.fr          132.227.60.2     /pub/linux
    ftp.fdn.org         193.55.4.75      /pub/Linux
    ftp.loria.fr        152.81.10.10     /pub/linux
    ftp.univ-angers.fr  193.49.144.1     /pub/Linux
------------------------------------------------------------
```

Il est fortement conseillé d'utiliser les miroirs locaux. Par exemple vous pourrez remplacer dans ce qui suit tsx-11.mit.edu par ftp.ibp.fr, sunsite.unc.edu par ftp.loria.fr.

De plus ftp.ibp.fr au moins contient un répertoire /pub/linux/french dans lequel sont archivés les documents en français dont ce texte (normalement : difficile de le savoir avec certitude au moment précis de la traduction du texte :-)), les autres HOWTOs, le guide d'installation *Bien débuter sous Linux*, et d'autres encore)).

Ftp: sunsite.unc.edu:/pub/Linux/docs

– La FAQ est dans /pub/linux/docs/faq/linux-faq

– Les HOWTOs sont dans `/pub/Linux/docs/HOWTO`

Pour WWW, lancez la page de documentation Linux :

`http://sunsite.unc.edu/mdw/linux.html`

Si vous êtes désespéré, envoyez un courrier électronique à :

`mail-server@rtfm.mit.edu`

avec le mot « help » dans le contenu du message, puis suivez les instructions que vous recevrez.

Note : si vous n'avez pas lu la FAQ Linux et les documents associés comme le Linux Installation HOWTO et le *Linux Install Guide* ((NdT: *Bien débuter sous Linux*)), alors vous ne devriez sans doute pas essayer de créer des disquettes d'amorce.

7.3 Scripts de secours

Ecrits par Thomas Heiling. Ce sont des scripts pour produire des disquettes d'amorce et des disquettes d'amorce/racine. Ils dépendent de versions spécifiques d'autres programmes comme LILO, donc il se peut que vous ayez à les adapter à votre système. Mais ils restent un point de départ utile pour des scripts plus élaborés que ceux de ce document.

Ftp : `sunsite.unc.edu:/pub/Linux/system/Recovery/rescue.tgz` ((NdT: ou comme toujours dans un site miroir comme `ftp.loria.fr`))

7.4 SAR - Search And Rescue : cherche et restaure.

Ecrit par Karel Kubat. `SAR` produit une disquette de secours en utilisant plusieurs techniques pour minimiser l'espace requis sur une disquette. Le manuel donne aussi une description du démarrage/login de Linux.

Ftp : `ftp.icce.rug.nl:/pub/unix/SAR-vvv.tar.gz`

Le manuel est disponible en format WWW :

`http://www.icce.rug.nl/karel/programs/SAR.html`

7.5 La distribution Slackware

Hormis le fait d'être une des plus populaires distributions Linux, c'est aussi un bon endroit pour obtenir un noyau standard. Elle est disponible presque partout ; il est inutile donc que je donne des adresses ici.