The Loopback Root Filesystem HOWTO

par Andrew M. Bishop, *amb@gedanken.demon.co.uk*
traduction par Eric Cano *Eric.Cano@cern.ch*
v1.0.0, 12 juin 1998, traduction octobre 1998

Ce HOWTO explique comment utiliser le périphérique loopback pour faire une installation sur un système de fichier Linux natif, qui peut résider dans une partition DOS sans repartitionnement. D'autres utilisations de cette technique sont aussi présentées.

Contents

1 Principes des périphériques "loopback" et des disques virtuels

Je vais d'abord décrire quelques-uns des principes généraux utilisés pour la mise en place d'un système de fichier en loopback comme racine.

1.1 Périphérique loopback

Sous Linux, un **périphérique loopback** est un périphérique virtuel, qui peut être utilisé comme tout autre périphérique.

Des exemples de périphériques normaux sont les partitions de disques durs comme `/dev/hda1`, `/dev/hda2`, `/dev/sda1`, ou des disques entiers comme les disquettes `/dev/fd0`, etc... Ce sont tous des périphériques qui peuvent contenir une structure de fichiers et de répertoires. Ils peuvent être formatés avec le système de fichier voulu (ext2fs, msdos, ntfs, etc...) puis montés.

Le périphérique loopback associe un fichier à un périphérique complet. Ce fichier peut appartenir à un autre système de fichiers.

Il peut alors être monté comme tout autre périphérique cité plus haut. Pour cela le périphérique appelé `/dev/loop0` ou `/dev/loop1` ou etc... est associé au fichier, puis ce nouveau périphérique virtuel est monté.

1.2 Les disques virtuels

Sous Linux, il est aussi possible d'avoir un autre type de périphérique virtuel monté en tant que système de fichiers, c'est le **disque virtuel** (*ramdisk*).

Dans ce cas, le périphérique ne se réfère pas à un élément du matériel, mais à une portion de la mémoire qui est mise de côté dans ce but. La mémoire allouée ainsi n'est jamais swapée sur le disque, mais reste dans le cache disque.

Un disque virtuel peut être créé à tout moment en écrivant dans le périphérique correspondant `/dev/ram0` ou `/dev/ram1`, etc... Il peut alors être formaté et monté de la même façon que le périphérique loopback.

Quand un disque virtuel est utilisé pour l'amorçage (comme c'est souvent le cas avec les disquettes d'installation de Linux et les disquettes d'urgence), l'image du disque (le contenu complet du disque sous forme d'un seul fichier) peut être stocké sur la disquette de démarrage sous une forme compressée. L'image est automatiquement détectée par le noyau quand celui-ci démarre et décompressée dans le disque virtuel avant d'être montée.

1.3 Le disque virtuel initial

Le **disque virtuel initial** est, sous Linux, un autre mécanisme important dont nous aurons besoin pour utiliser le périphérique loopback comme système de fichier racine.

Quand le disque virtuel initial est utilisé, l'image du système de fichiers est copiée dans la mémoire et montée pour que les fichiers soient accessibles. Un programme sur ce disque virtuel (appelé `linuxrc`) est lancé. Une fois terminé un nouveau périphérique est monté comme système de fichiers racine. Le disque virtuel précédent existe toujours, il est monté sur le répertoire `/initrd` si celui-ci est présent, ou accessible à travers le périphérique `/dev/initrd`.

C'est un comportement peu habituel, puisque la séquence de démarrage normale se lance depuis la partition racine choisie et continue à tourner ainsi. Avec l'option de disque virtuel initial, la partition racine a la possibilité de changer avant que ne commence la séquence de démarrage principale.

1.4 Le système de fichiers racine

Le système de fichiers racine est le périphérique qui est monté en premier, et qui apparaît donc dans le répertoire appelé / après le démarrage.

Il y a un certain nombre de complications a propos du système de fichiers racine, qui sont dues au fait qu'il contient tous les fichiers. Au boot, les scripts `rc` sont lancés ; ce sont soit les fichiers dans `/etc/rc.d` ou `/etc/rc?.d`, suivant la version du programme `/etc/init`.

Quand le système a démarré, il n'est plus possible de démonter la partition racine ou d'en changer car tout les programmes l'utiliseront plus ou moins. C'est pourquoi le disque virtuel initial est si utile, puisqu'il peut être utilisé de façon telle que la partition racine finale n'est pas la même que celle qui est chargée au moment de l'amorçage.

1.5 La séquence d'amorçage de Linux

Pour montrer comment le disque virtuel initial opère pendant la séquence de démarrage, l'ordre des événements est présenté ci dessous.

1. Le noyau est chargé en mémoire, ceci est effectué par `LILO` ou `LOADLIN`. Vous pouvez voir le message `Loading...` pendant que ceci arrive.
2. L'image du disque virtuel est chargée en mémoire, à nouveau ceci est réalisé par `LILO` ou `LOADLIN`. Vous pouvez voir le message `Loading...` à nouveau quand ceci arrive.
3. Le noyau est initialisé, y compris la lecture des options de ligne de commande et le montage du disque virtuel en tant que racine.
4. Le programme `/linuxrc` est lancé sur le disque virtuel initial.
5. Le périphérique racine est changé pour celui spécifié dans les paramètres du noyau.
6. Le programme `/etc/init` est lancé, et va exécuter la séquence de démarrage paramétrable par l'utilisateur.

Ceci est juste une version simplifiée de ce qui arrive, mais c'est suffisant pour expliquer comment le noyau démarre et où le disque virtuel est utilisé.

2 Comment créer un périphérique loopback.

Maintenant que les principes généraux ont été présentés, la méthode pour créer le périphérique loopback peut être expliquée.

2.1 Pré-requis

La création du périphérique loopback va nécessiter un certain nombre de choses.

- Un système Linux installé.
- Un moyen pour copier des gros fichiers sur la partition DOS de destination.

Le point le plus important est l'accès à un système Linux déjà installé. Ce point est nécessaire car le périphérique loop ne peut être créé que sous Linux. Cela signifie qu'il ne sera pas possible d'installer un système à partir de rien. Le système Linux que vous utilisez devra être capable de compiler un noyau.

Une fois le périphérique loopback créé, il représentera un gros fichier. J'ai utilisé un fichier de 80 Mo, mais si c'était suffisant pour un terminal X, ça ne sera sans doute pas suffisant pour une utilisation plus importante. Ce fichier doit être copié sur la partition DOS, donc un réseau ou beaucoup de disquettes seront mis a contribution.

Vous aurez besoin des logiciels suivants :

- `LOADLIN` version 1.6 ou supérieure
- Une version de `mount` qui supporte les périphériques loopback
- Une version du noyau qui inclut les options requises.

Tout ceci devrait être disponible en standard sur des installations récentes de Linux.

2.2 Création du noyau Linux

J'ai créé le périphérique loopback avec le noyau Linux version 2.0.31, d'autres versions devraient faire l'affaire, mais elles devront avoir au moins les options listées ci-dessous configurées.

Les options du noyau que vous devrez sélectionner sont les suivantes :

- RAM disk support (`CONFIG_BLK_DEV_RAM`).
- Initial RAM disk (initrd) support (`CONFIG_BLK_DEV_INITRD`).
- Loop device support (`CONFIG_BLK_DEV_LOOP`).
- fat fs support (`CONFIG_FAT_FS`).
- msdos fs support (`CONFIG_MSDOS_FS`).

Les deux premières sont le disque virtuel lui-même et le disque virtuel initial. La suivante est le support pour les périphériques loopback. Les deux dernières sont le support pour les systèmes de fichiers msdos, qui est requis pour monter des partitions DOS.

La compilation d'un noyau sans modules est la plus simple, mais si vous voulez utiliser les modules ça devrait être possible, bien que je ne l'aie pas essayé. Si vous utilisez des modules, vous devez configurer les options précédentes dans le noyau, et non comme des modules.

Le code source du noyau lui-même devra être modifié d'une façon très simple. La version 2.0.34 du noyau telle que fournie ne permet pas au périphérique loopback d'être utilisé comme racine. Une très petite modification du noyau peut rendre ceci possible.

Le fichier `/init/main.c` a juste besoin qu'on lui ajoute une seule ligne comme montré dans la version modifiée ci-dessous. La ligne qui dit `"loop", 0x0700` est celle qui a été ajoutée.

```
static void parse_root_dev(char * line)
{
        int base = 0;
        static struct dev_name_struct {
                const char *name;
                const int num;
```

```
        } devices[] = {
                { "nfs",     0x00ff },
                { "loop",    0x0700 },
                { "hda",     0x0300 },

...

                { "sonycd",  0x1800 },
                { NULL, 0 }
        };

...

}
```

Une fois le noyau configuré, il devra être compilé pour produire une fichier `zImage` (`make zImage`). Ce fichier devrait être `arch/i386/boot/zImage` une fois compilé.

2.3 Création du périphérique disque virtuel initial

Le disque virtuel initial est simplement crée comme un périphérique loopback au départ. Vous devrez faire ceci en tant que root. Les commandes que vous devez exécuter sont listées ci dessous, elles supposent être lancées depuis le répertoire principal de root (`/root`).

```
mkdir /root/initrd
dd if=/dev/zero of=initrd.img bs=1k count=1024
mke2fs -i 1024 -b 1024 -m 5 -F -v initrd.img
mount initrd.img /root/initrd -t ext2 -o loop
cd initrd
[créez les fichiers]
cd ..
umount /root/initrd
gzip -c -9 initrd.img > initrdgz.img
```

Il y a un certain nombre d'étapes, mais on peut les décrire comme ceci.

1. Créez un point de montage pour le disque virtuel initial (un répertoire vide).
2. Créez un fichier vide de la taille requise. Ici j'ai utilisé 1024ko, vous pourriez avoir besoin de plus ou de moins suivant le contenu. (la taille est le dernier paramètre).
3. Créez un système de fichiers ext2 dans le fichier vide.
4. Montez le ficher au point de montage, ceci utilise le périphérique loopback.
5. Changez le répertoire courant pour le périphérique loopback.
6. Créez les fichiers requis (voir plus bas pour les détails).
7. Sortez du périphérique loopback monté.
8. Démontez le périphérique.
9. Créez une version compressée pour l'utiliser plus tard.

Contenu du disque virtuel initial

Les fichiers dont vous avez besoin sur le disque virtuel représentent le minimum nécessaire pour pouvoir d'exécuter une commande.

- `/linuxrc` Le fichier qui est lancé pour monter le système de fichiers msdos (voir plus bas).
- `/lib/*` L'éditeur de liens dynamiques et les librairies dont les programmes ont besoin.
- `/etc/*` Le cache utilisé par l'éditeur de liens dynamiques (pas strictement requis, mais ça l'empêche de se plaindre).
- `/bin/*` Un interpréteur de commandes (`ash` car il est plus petit que `bash`). Les programmes `mount` et `losetup` pour manipuler le disque DOS et configurer les périphériques loopback.
- `/dev/*` Les périphériques qui seront utilisés. Vous avez besoin de `/dev/zero` pour `ld-linux.so`, `/dev/hda*` pour monter le disque msdos et `/dev/loop*` pour les périphériques loopback.
- `/mnt` Un répertoire vide pour y monter le disque msdos.

Le contenu du disque virtuel initial que j'ai utilisé est énuméré ci-dessous. Ces fichiers représentent environ 800ko, une fois pris en compte l'espace perdu par les structures du système de fichiers.

```
total 18
drwxr-xr-x   2 root     root         1024 Jun  2 13:57 bin
drwxr-xr-x   2 root     root         1024 Jun  2 13:47 dev
drwxr-xr-x   2 root     root         1024 May 20 07:43 etc
drwxr-xr-x   2 root     root         1024 May 27 07:57 lib
-rwxr-xr-x   1 root     root          964 Jun  3 08:47 linuxrc
drwxr-xr-x   2 root     root        12288 May 27 08:08 lost+found
drwxr-xr-x   2 root     root         1024 Jun  2 14:16 mnt

./bin:
total 168
-rwxr-xr-x   1 root     root        60880 May 27 07:56 ash
-rwxr-xr-x   1 root     root         5484 May 27 07:56 losetup
-rwsr-xr-x   1 root     root        28216 May 27 07:56 mount
lrwxrwxrwx   1 root     root            3 May 27 08:08 sh -> ash

./dev:
total 0
brw-r--r--   1 root     root       3,   0 May 20 07:43 hda
brw-r--r--   1 root     root       3,   1 May 20 07:43 hda1
brw-r--r--   1 root     root       3,   2 Jun  2 13:46 hda2
brw-r--r--   1 root     root       3,   3 Jun  2 13:46 hda3
brw-r--r--   1 root     root       7,   0 May 20 07:43 loop0
brw-r--r--   1 root     root       7,   1 Jun  2 13:47 loop1
crw-r--r--   1 root     root       1,   3 May 20 07:42 null
crw-r--r--   1 root     root       5,   0 May 20 07:43 tty
crw-r--r--   1 root     root       4,   1 May 20 07:43 tty1
crw-r--r--   1 root     root       1,   5 May 20 07:42 zero

./etc:
total 3
-rw-r--r--   1 root     root         2539 May 20 07:43 ld.so.cache
```

```
./lib:
total 649
lrwxrwxrwx   1 root     root           18 May 27 08:08 ld-linux.so.1 -> ld-linux.so.1.7.14
-rwxr-xr-x   1 root     root        21367 May 20 07:44 ld-linux.so.1.7.14
lrwxrwxrwx   1 root     root           14 May 27 08:08 libc.so.5 -> libc.so.5.3.12
-rwxr-xr-x   1 root     root       583795 May 20 07:44 libc.so.5.3.12

./lost+found:
total 0

./mnt:
total 0
```

La seule étape complexe est la création des périphériques dans `dev`. Utilisez le programme `mknod` pour les créer, et servez vous des périphériques dans `/dev` comme modèles pour les paramètres requis.

Le fichier /linuxrc

Le fichier `/linuxrc` sur le disque virtuel initial est nécessaire pour mettre en place le périphérique loopback, avant de l'utiliser comme racine.

L'exemple suivant essaye de monter `/dev/hda1` comme une partition msdos et en cas de réussite assigne les fichiers `/linux/linuxdsk.img` et `/linux/linuxswp.img` respectivement aux périphériques `/dev/loop0` et `/dev/loop1`.

```
#!/bin/sh

echo INITRD: Essaye de monter /dev/hda1 comme partition msdos

if /bin/mount -n -t msdos /dev/hda1 /mnt; then

   echo INITRD: Montage réussi
   /bin/losetup /dev/loop0 /mnt/linux/linuxdsk.img
   /bin/losetup /dev/loop1 /mnt/linux/linuxswp.img
   exit 0

else

   echo INITRD: Echec du montage
   exit 1

fi
```

Le premier périphérique, `/dev/loop0` deviendra la racine et le second, `/dev/loop1` deviendra la mémoire virtuelle.

Si vous voulez pouvoir écrire sur la partition racine en tant qu'utilisateur normal quand vous aurez fini, alors vous devriez plutôt utiliser `mount -n -t msdos /dev/hda1 /mnt -o uid=0,gid=0,umask=000`. Ceci associera tous les accès à la partition DOS à l'utilisateur root et placera les permissions en conséquences.

2.4 Création du périphérique racine

Le périphérique racine que vous utiliserez est le fichier `linuxdsk.img`. Vous devrez le créer de la même façon que le disque virtuel initial, en plus grand. Vous pouvez y mettre l'installation de Linux de votre choix.

La méthode la plus simple est de copier une installation Linux existante sur ce disque. une alternative est d'y installer une distribution de Linux.

En supposant que ceci est fait, vous avez encore des modifications mineures à apporter.

Le fichier `/etc/fstab` doit référencer les partitions racine et swap en utilisant les deux périphériques loopback qui sont mis en place par le disque virtuel initial.

```
/dev/loop0     /      ext2   defaults 1 1
/dev/loop1     swap   swap   defaults 1 1
```

Ceci permettra de s'assurer que quand le périphérique racine sera utilisé, le noyau ne sera pas induit en erreur sur son emplacement. L'espace de swap pourra ainsi être ajouté de la même façon qu'une partition de swap normale. Vous devez aussi retirer toute autre référence vers un disque racine ou swap.

Si vous voulez être capable d'accéder à la partition DOS après le démarrage de Linux, vous devrez faire quelques petites modifications.

Créez un répertoire appelé `/initrd`, qui sera le point de montage du disque virtuel initial une fois que le système de fichier racine sera monté en loopback.

Créez un lien symbolique appelé `/DOS` qui pointe sur `/initrd/mnt` où la partition DOS sera montée.

Ajoutez un ligne dans le fichier rc qui monte les disques. Il devra lancer la commande `mount -f -t msdos /dev/hda1 /initrd/mnt` ; ceci créera un montage "fictif" de la partition DOS pour que tous les autres programmes (comme `df`) sachent que la partition DOS est montée et où la trouver. Si vous avez utilisé des options différentes dans `/linuxrc`, vous devrez évidemment utiliser les mêmes ici.

Il n'y a plus de raison d'avoir le noyau Linux sur le périphérique racine puisqu'il a été chargé plus tôt. Toutefois, si vous utilisez les modules, vous devrez les inclure normalement sur ce périphérique.

2.5 Création du périphérique de mémoire virtuelle.

Le périphérique que vous utiliserez sera le fichier `linuxswap.img`. Le périphérique de mémoire virtuelle *(swap)* est très simple à fabriquer. Créez un fichier vide de la même façon que pour le disque virtuel initial, puis exécutez `mkswap linuxswap.img` pour l'initialiser.

La taille de la mémoire virtuelle dépendra de ce que vous comptez faire avec le système installé, mais je recommanderais entre 8 Mo et la quantité de RAM que vous avez.

2.6 Création du répertoire MSDOS

Les fichiers qui seront utilisés devront être déplacés sur la partition DOS.

Les fichiers qui devront être dans le répertoire DOS appelé `C:\LINUX` sont les suivants :

- `LINUXDSK.IMG` L'image de la partition qui deviendra le périphérique racine.
- `LINUXSWP.IMG` L'espace de mémoire virtuelle.

2.7 Création de la disquette de démarrage.

La disquette de démarrage qui est utilisée est juste une disquette amorçable au format DOS.

On la crée en utilisant `format a: /s` sous DOS.

Sur ce disque vous devrez créer un fichier `AUTOEXEC.BAT` (comme ci-dessous) et copier le noyau, le disque virtuel initial sous forme compressée et l'exécutable `LOADLIN`.

- `AUTOEXEC.BAT` Le fichier de commandes exécuté automatiquement par le DOS.
- `LOADLIN.EXE` L'exécutable du programme `LOADLIN`.
- `ZIMAGE` Le noyau Linux.
- `INITRDGZ.IMG` L'image compressée du disque virtuel initial.

Le fichier `AUTOEXEC.BAT` devrait contenir une seule ligne comme ci-dessous.

```
\loadlin \zImage initrd=\initrdgz.img root=/dev/loop0 ro
```

Ceci spécifie l'image du noyau à utiliser, l'image du disque virtuel initial, et le périphérique racine après que le disque virtuel ait fait son office, avec la partition racine montée en lecture seule.

3 Démarrage du système

Pour démarrer depuis le nouveau périphérique racine, il suffit de faire démarrer le PC sur la disquette préparée plus haut.

Vous verrez les événement suivants se succéder :

1. Chargement du DOS.
2. Démarrage AUTOEXEC.BAT
3. Lancement de LOADLIN
4. Copie du noyau Linux dans la mémoire
5. Le disque virtuel initial est copié en mémoire
6. Le noyau Linux démarre
7. Le fichier `/linuxrc` sur le disque virtuel initial est exécuté
8. La partition DOS est montée, ainsi que les périphériques racine et de swap
9. La séquence de démarrage continue depuis le périphérique loopback

Une fois ceci accompli, vous pouvez retirer la disquette et utiliser le système Linux.

3.1 Problèmes possibles et leurs solutions

Il y a un certain nombre d'étapes de ce processus qui peuvent échouer. Je vais essayer d'expliquer lesquelles, et ce qu'il faut vérifier.

Le démarrage du DOS est facile à reconnaître grâce au message qu'il affiche à l'écran : `Démarrage de MS-DOS...` . Si ceci n'est pas visible, soit la disquette n'est pas amorçable, soit le PC ne démarre pas sur le lecteur de disquettes.

Quand le fichier `AUTOEXEC.BAT` est exécuté, les commandes qu'il contient devraient être affichées sur l'écran par défaut. Dans le cas présent, il n'y a d'une seule ligne dans le fichier, qui lance `LOADLIN`.

Quand `LOADLIN` se lancera, il exécutera deux actions facile à distinguer : premièrement il chargera le noyau en mémoire, ensuite il copiera le disque virtuel en mémoire. Chacune de ces actions est indiquée par un message `Loading...` .

Le noyau commence par se décompresser, ceci peut engendrer des erreur **crc** si l'image du noyau est corrompue. Ensuite, il lancera la séquence d'initialisation qui est très prolixe en messages de diagnostic. Le chargement du périphérique disque virtuel sera aussi visible durant cette phase.

Quand le fichier `/linuxrc` est lancé, il n'y a pas de message de diagnostic, mais vous pouvez les ajouter pour vous aider à debugger. Si cette étape échoue dans le montage du périphérique loopback en tant que périphérique racine, vous verrez un message avertissant qu'il n'y a pas de périphérique racine, et le noyau interrompra son exécution.

La séquence de démarrage normale du nouveau système de fichiers racine va maintenant continuer, et cette partie est à nouveau généreuse en messages. Il pourrait y avoir des problèmes dûs au fait que le système de fichiers racine est monté en lecture-écriture, mais l'option de ligne de commande '`ro`' pour `LOADLIN` devrait arranger ça. Un autre problème qui peut apparaître est la confusion de la séquence de démarrage à propos de l'emplacement du système de fichiers racine ; ceci sera probablement dû à un problème avec `/etc/fstab`.

Quand la séquence de démarrage est réalisée, le problème qui reste est que les programmes ne savent pas si la partition DOS est montée ou non. C'est pourquoi c'est une bonne idée d'utiliser une fausse commande mount décrite plus tôt. Ceci rend la vie nettement plus simple si vous voulez accéder au fichiers sur le périphérique DOS.

3.2 Documents de référence

Les document que j'ai utilisés pour créer mon premier périphérique racine en loopback sont :

- Les sources du noyau Linux, en particulier `init/main.c`
- La documentation du noyau Linux, en particulier `Documentation/initrd.txt` et `Documentation/ramdisk.txt`.
- La documentation de `LILO`.
- La documentation de `LOADLIN`.

4 Autres possibilités de périphériques racine en loopback

Une fois que le principe de démarrer sur un système de fichiers dans une partition DOS est acquis, il y a de nombreuses autres choses que l'on peut faire.

4.1 Installation "tout sur un disque DOS"

S'il est possible de charger Linux depuis un fichier sur un disque dur DOS en utilisant une disquette de démarrage, alors il est clair qu'on peut faire la même chose en utilisant le disque dur lui-même.

Un menu de choix de configuration au démarrage peut être utilisé pour donner l'option de lancer `LOADLIN` depuis l'`AUTOEXEC.BAT`. Ceci donnera une séquence de démarrage plus rapide, mais c'est la seule différence.

4.2 Installation démarrée avec LILO

Utiliser `LOADLIN` n'est qu'une des options possibles pour charger un noyau Linux. Il y a aussi `LILO` qui fait pratiquement la même chose, mais sans nécessiter DOS.

Dans ce cas, la disquette au format DOS peut être remplacée par une disquette au format ext2fs. A part cela, les détails restent très similaires, le noyau et le disque virtuel initial étant encore des fichiers sur cette disquette.

La raison pour laquelle j'ai choisi la méthode avec `LOADLIN` est que les arguments qui doivent être données à `LILO` sont légèrement plus complexes. Le contenu de la disquette est aussi plus clair pour un observateur lambda, puisqu'on peut la lire sous DOS.

4.3 Installation VFAT / NTFS

J'ai essayé la méthode NTFS, et je n'ai pas eu de problème avec. Le support du système de fichier NTFS n'est pas une option standard du noyau, mais vous devez appliquer le patch de Martin von Löwis, qui est disponible sur sa page web. *http://www.informatik.hu-berlin.de/~loewis/ntfs/* . Ce logiciel est en version alpha et requiert un patch qui n'est pas totalement trivial à appliquer au noyau, mais pas trop difficile non plus.

Les seuls changements pour les options VFAT ou NTFS sont sur le disque virtuel initial, le fichier `/linuxrc` doit monter un système de fichiers de type vfat ou ntfs plutôt que msdos.

Je ne connais pas de raison pour laquelle ceci ne marcherait pas aussi sur une partition VFAT.

4.4 Installer Linux sans repartitionner

Le processus d'installation de Linux sur un PC avec une distribution standard requiert de démarrer sur une disquette et de repartitionner le disque dur. Cette étape pourrait être remplacée par une disquette de démarrage qui crée un périphérique loopback vide et un fichier de swap. Ceci permettrait à l'installation de procéder normalement, sur le périphérique loopback plutôt que sur une partition.

Ceci pourrait être une alternative à une installation `UMSDOS`, et serait plus efficace pour l'utilisation du disque, puisque l'unité d'allocation minimale sur un système de fichiers ext2 est de 1ko contre 32ko sur une partition DOS. On peut aussi l'utiliser sur des disques VFAT et NTFS qui sinon posent un problème.

4.5 Démarrer depuis un périphérique non amorçable

Cette méthode peut aussi être utilisée pour démarrer un système Linux depuis un périphérique qui n'est pas normalement amorçable.

- CD-Rom
- Disques Zip
- Lecteurs de disques sur port parallèle

Evidemment, de nombreux autres périphériques pourraient être utilisés, les partitions racines en NFS sont déjà incluses dans le noyau comme une option, mais la méthode présentée ici pourrait être utilisée à la place.