Le HOWTO du noyau de Linux (Kernel HOWTO)

par Brian Ward, ward@blah.tu-graz.ac.at Version 0.80, 26 Mai 1997

(Adaptation française par Eric Dumas dumas@freenix.fr, 4 Juin 1997). Il s'agit d'un guide détaillé sur la configuration du noyau, sa compilation et ses mises à jour pour les plates-formes x86.

Table des matières

1 Introduction **4**

1.1 Lisez ceci en premier (enfin bon, c'est un conseil)! 4

1.2 Un mot sur le style 4

2 Quelques questions... avec leurs réponses **5**

2.1 Quel est le rôle du noyau? 5

2.2 Pourquoi voudrais-je mettre à jour mon noyau? 5

2.3 Quel genre de matériel supportent les nouveaux noyaux? 5

2.4 Quelle version de gcc ou de libc utiliser? 5

2.5 Que sont les modules chargeables (loadable modules)? 5

2.6 De combien d'espace disque ai-je besoin? 6

2.7 Combien de temps ça prend? 6

3 Comment réellement configurer le noyau? **6**

3.1 Récupérer les sources 6

3.2 Installer les sources 7

3.3 Configurer le noyau 7

3.3.1 Kernel math emulation (Emulation du coprocesseur Arithmétique) 8

3.3.2 Normal (MFM/RLL) disk and IDE disk/cdrom support (Gestion Standard des disques durs) 8

3.3.3 Networking support (Gestion du Réseau) 8

3.3.4 Limit memory to low 16MB (Vous avez moins de 16 Mo) 9

3.3.5 System V IPC 9

3.3.6 Processor type (386, 486, Pentium, PPro) 9

3.3.7 SCSI support (Gestion SCSI) 9

3.3.8 Network device support (Gestion de périphériques réseau) . . . 9
3.3.9 Filesystems (Systèmes de gestion de fichiers) . . . 9
3.3.10 Character devices (Périphériques caractères) . . . 11
3.3.11 Sound card (Carte sons) . . . 11
3.3.12 Autres options de configuration . . . 11
3.3.13 Kernel hacking . . . 11
3.4 Garçon, la suite! (Le Makefile) . . . 12

4 Compiler le noyau **12**
4.1 Nettoyer et chercher les dépendances . . . 12
4.2 La compilation . . . 12
4.3 Les autres options du « make » . . . 12
4.4 Installer le noyau . . . 13

5 Mettre à jour le noyau **13**
5.1 Appliquer un patch . . . 13
5.2 Si quelque chose ne fonctionne pas . . . 14
5.3 Comment se débarasser des fichiers .orig? . . . 15
5.4 Autres patches . . . 15

6 Paquetages supplémentaires **15**
6.1 kbd . . . 16
6.2 util-linux . . . 16
6.3 hdparm . . . 16
6.4 gpm . . . 16

7 Quelques pièges **16**
7.1 make clean . . . 16
7.2 Noyaux énormes ou lents . . . 16
7.3 Le noyau ne compile pas . . . 17
7.4 La nouvelle version du noyau ne boot plus! . . . 17
7.5 Vous avez oublié de lancer LILO, ou bien votre système ne boote plus du tout . . . 18
7.6 Il me dit « warning: bdflush not running » . . . 19
7.7 Il m'indique qu'il ne trouve pas certains symboles et il n'arrive pas à compiler . . . 19

7.8 Je n'arrive pas à faire marche mon CD-ROMM IDE/ATAPI CD-ROM 19
7.9 Le noyau me dit des insanités à propos de requètes obsolètes ! 20
7.10 Le Firewall ne fonctionne pas dans la version 1.2.0 20
7.11 Ce n'est pas une image noyau compressée ! 20
7.12 Problèmes avec la console après mise à jour à la version 1.3.x 20
7.13 Le noyau ne semble pas pouvoir compiler après une mise à jour 20
7.14 Increasing limits 21

8 Note concernant la mise à jour pour les noyaux 2.0 **21**

9 Modules **21**
9.1 Installer les fonctionnalités modules 21
9.2 Modules distribués avec le noyau 22

10 Autres options de configuration **22**
10.1 Configuration générale 22
10.2 Options de réseau 23

11 Trucs et astuces **23**
11.1 Rediriger la sortie standard des commandes make ou patch 23
11.2 Installations diverses du noyau 23
11.3 Mises à jours des noyaux 24

12 D'autres HOWTO qui peuvent être très utiles **24**

13 Divers **25**
13.1 Auteur 25
13.2 Reste à faire 25
13.3 Contributions 25
13.4 Copyright, traduction 26

1 Introduction

Faut-il que vous lisiez ce document ? Oui, si vous avez les symptômes suivants :

- « Arg ! cette archive TrucBidule-46.5.6 nécessite la version du noyau 1.1.193 et je n'ai que la version 1.0.9 ! »
- Il y a un pilote de périphérique dont vous avez besoin dans l'un des nouveaux noyaux.
- Vous n'avez aucune idée de la manière dont il faut s'y prendre pour recompiler le noyau.
- « Est-ce que ce truc dans le fichier *README* décrit toute la marche à suivre ? »
- Vous avez essayé, ca ne marche pas !
- Vous devez donner un conseil à une personne qui a certains problèmes.

1.1 Lisez ceci en premier (enfin bon, c'est un conseil) !

Certains exemples présentés dans ce document présupposent que vous possédez les programmes GNU `tar`, `find` et `xargs`. Ils sont assez standard, et ne devraient pas vous poser trop de problème. On suppose également que vous connaissez la structure de votre système de fichiers. Si vous ne le savez pas, il est impératif que vous gardiez une copie sur papier du résultat de l'exécution de la commande `mount` (ou bien une copie du fichier `/etc/fstab`, si vous savez le lire). Cette information est importante, et ne change pas à moins que vous ne repartionniez votre disque, en ajoutiez un nouveau, réinstalliez votre système, ou quelque chose de similaire.

La dernière version du noyau à l'heure à laquelle j'écris ces mots est la 2.0.30, ce qui signifie que les références et les exemples donnés correspondent à cette version. Même si j'ai essayé de faire en sorte que ce document soit aussi indépendant que possible des versions, le noyau est en perpétuel développement, donc si vous récupérez une nouvelle version, il risque d'y avoir quelques différences. Cela ne devrait pas poser trop de problèmes mais risque de semer un peu la confusion.

Il existe deux versions des sources du noyau : une version dite « stable » et une version de développement. Les version stables ont des numéros de version paires : 1.0.x, 1.2.x sont des versions stables... tout comme la 2.0. Ces noyaux sont considérés comme étant les plus stables et sans erreurs. Les noyaux de développement (1.1.x, 1.3.x, etc) sont des noyaux de tets, pour les développeurs, les testeurs... mais c'est noyaux peuvent être très bugués. Vous avez été avertit.

1.2 Un mot sur le style

`Un texte qui ressemble à celui-ci` est soit un message qui apparaît sur votre écran, soit un fichier, ou soit quelque chose qui peut être directement saisi, comme une commande, ou des options pour une commande (si vous possédez une version texte de ce document, il n'y a pas de différence), Les commandes et autre textes devant être saisis sont fréquemment quotés (‘ ’), ce qui provoque le problème classique de la ponctuation : si un tel signe apparaît la fin d'un phrase entre quotes, les gens mettent souvent un ‘.’ à la commande car le style américain indique de mettre le point à l'intérieur des quotes. Même si le bon sens est de retirer le

point, bon nombre de personnes ne s'en souviennent pas, et donc je le mettrai à l'extérieur des quotes dans ces cas-là. Par exemple, j'écrirai '`make config`', et non '`make config.`'

2 Quelques questions... avec leurs réponses

2.1 Quel est le rôle du noyau ?

Le noyau Unix joue le rôle d'un chef d'orchestre entre vos programmes et votre matériel. Premièrement, il gère la mémoire pour tous les programmes en cours d'exécution (processus), et s'assure qu'ils occupent tous une part équitable (ou non) du temps processeur. En plus, il fournit une interface (simple à utiliser) aux programmes pour communiquer avec votre matériel (appels système).

Le noyau a bien sûr un rôle plus important que celui décrit, mais ces fonctions de bases sont les plus importantes à connaître.

2.2 Pourquoi voudrais-je mettre à jour mon noyau ?

Les nouveaux noyaux offrent plus de facilité pour communiquer avec de plus en plus de matériels (c'est-à-dire qu'ils gèrent de plus en plus de périphériques...), ils peuvent avoir une meilleure gestion des processus, tourner plus rapidement que les anciennes versions, être plus stables et ils corrigent les erreurs des versions précédentes. Bon nombre de personnes mettent à jour leurs noyaux car ils veulent avoir les nouveaux pilotes de périphériques et les corrections d'erreurs.

2.3 Quel genre de matériel supportent les nouveaux noyaux ?

Jetez un coup d'oeil sur le Hardware-HOWTO. Vous pouvez aussi regarder le fichier « `config.in` » dans les sources de **Linux**, ou juste les trouver lorsque vous essayez « `make config` », ce qui vous permet de voir tous les matériels supportés par le code source du noyau standard, mais pas tout ce que **Linux** supporte. En effet, beaucoup de périphériques assez utiles (tels les périphériques PCMCIA, et quelques lecteurs de cartouches) sont des modules chargeables maintenus et distribués séparément.

2.4 Quelle version de gcc ou de libc utiliser ?

Linus recommande une version de gcc dans le fichier `README` fournis avecle code source. Si vous ne possédez pas cette version, la documentation dans la version recommandée de gcc devrait vous indiquer si vous avez besoin de mettre à jour votre libc. Ce n'est pas une opération compliquée à réaliser, mais il est important de suivre les instructions.

2.5 Que sont les modules chargeables (loadable modules) ?

Ce sont des parties du noyau (pilotes de périphériques généralement) qui ne sont pas compilées dans celui-ci. On peut les compiler séparément, les insérer, les retirer du noyau à n'importe quel moment. En raison de

cette souplesse, c'est devenu la méthode préférée pour coder certaines caractéristiques du noyau. Bon nombre de pilotes de périphériques tels que PCMCIA et les gestionnaires de cartouches QIC-80/40 sont des modules chargeables.

2.6 De combien d'espace disque ai-je besoin?

Cela dépend de la configuration de votre système. Tout d'abord, le code source compressé de **Linux** était de 6 pour la version 2.0. Vous voulez généralement le garder dans un coin après l'avoir décompressé. Décompressé, cela fait environ 24 Mo. Mais ce n'est pas tout - vous aurez besoin de plus de place pour le compiler. Cela dépend du nombre de périphériques que vous avez configuré dans le noyau. Par exemple, sur une de mes machines, j'ai les fonctionnalités réseau, le gestionnaire pour une 3Com 3C509, et trois gestionnaires de systèmes de fichiers configurés et cela prend environ 30 Mo. Sur un autre système, sans périphérique de réseau (mais toujours avec les fonctionnalités réseau) et une carte son, la place occupée est même supérieure. Comme en général un nouveau noyau est certain d'avoir un code source plus important que les précédents, prévoyez un grand espace disque pour le compiler surtout si vous avez beaucoup de matériel (et au prix du disque dur aujourd'hui, la meilleure aide que je peux vous donner en cas de manque de place disque et d'en acheter un autre).

2.7 Combien de temps ça prend?

Pour beaucoup de gens, la reponse est : « C'est bien long ! ». La vitesse de votre machine et la mémoire dont vous disposez déterminent le temps de compilation, mais cela est un peu proportionnel à la configuration matérielle que vous avez choisie pour le noyau. Sur un 486DX4/100 avec 16Mo de RAM, avec un noyau de la série 1.2, avec cinq gestionnaires de systèmes de fichiers différents, support réseau et le support pour les cartes son, cela prend environ 20 minutes. Sur un 386DX/40 (8 Mo RAM) avec une configuration similaire, cela prend presque 1 heure et demie ! Il est généralement recommandé de se faire un petit café, regarder la télévision, faire la cuisine, ou tout ce que vous voudrez pour vous occuper pendant que votre machine compile le noyau. Vous pouvez connaître quelqu'un avec une machine plus rapide qui puisse vous le compiler si votre machine est vraiment lente.

3 Comment réellement configurer le noyau?

3.1 Récupérer les sources

Vous pouvez récupérer les sources via ftp depuis `ftp.funet.fi` dans le répertoire `/pub/Linux/PEOPLE/Linus`, sur un miroir de ce site, ou d'autres sites. (NdT: en France, `ftp.ibp.fr` dans `/pub/linux/kernel/sources`). Le noyau est typiquement appelé `linux-x.y.z.tar.gz`, où `x.y.z` est le numéro de la version. Les nouvelles versions (plus performantes?) ainsi que les patches se trouvent dans des repertoires tels que « `v1.1` » et « `v1.2` ». Le plus grand chiffre est la dernière version, et peut être généralement considérée comme étant une version de tests, ce qui signifie que si jamais vous vous sentez mal à l'aise avec des version alpha ou béta, vous devriez vous cantonner au mise à jours officielles.

Il est *fortement* suggéré d'utiliser un site miroir au lieu de ftp.funet.fi ! En voici une courte liste :

```
USA :           tsx-11.mit.edu:/pub/linux/sources/system
USA :           sunsite.unc.edu:/pub/Linux/kernel
UK :            sunsite.doc.ic.ac.uk:/pub/unix/Linux/sunsite.unc-mirror/kernel
Autriche :      ftp.univie.ac.at:/systems/linux/sunsite/kernel
Allemagne :     ftp.Germany.EU.net:/pub/os/Linux/Local.EUnet/Kernel/Linus
Allemagne :     sunsite.informatik.rwth-aachen.de:/pub/Linux/PEOPLE/Linus
France :        ftp.ibp.fr:/pub/linux/kernel/sources
Australie:      sunsite.anu.edu.au:/pub/linux/kernel
```

En général, les miroirs du site `sunsite.unc.edu` sont des endroits très recommandés. Le fichier `/pub/Linux/MIRRORS` contient la liste de tous les miroirs connus. Si vous n'avez pas d'accès ftp, une liste de systèmes BBS possédant **Linux** est postée périodiquement dans comp.os.linux.announce. Essayez de mettre la main dessus !

Si vous cherchez des informations générales sur Linux et ses distributions, consultez le site `http://www.linux.org`.

(NdT : pour la France, voir la liste postée dans le forum de discussion `fr.comp.os.linux.annonces`)

3.2 Installer les sources

Passez en `su` (super utilisateur) ou bien logez-vous en tant que « `root` », et allez dans le répertoire `/usr/src`. Si vous avez déjà installé les sources du noyau, il y aura toujours un répertoire appelé « `linux` », contenant l'ensemble des sources du noyau. Si vous avez de la place disque et que vous voulez jouer la carte sécurité, il est préférable de conserver ce répertoire intact. Un système assez pratique consiste à renommer votre répertoire en fonction du numéro de la version. La commande « `uname -r` » vous indiquera le numéro de la version du noyau actuellement en fonction.

Par exemple, si « `uname -r` » donne « `1.0.9`, » vous pourrez déplacer votre répertoire (avec `mv`) « `linux` » en « `linux-1.0.9` ». Si vous êtes du genre téméraire, détruisez le répertoire. Dans n'importe quel cas, soyez sûr qu'il n'y ait aucun répertoire « `linux` » dans `/usr/src` avant de décompresser et d'installer les sources.

Dans `/usr/src`, installez les sources en faisant « `tar zxpvf linux-x.y.z.tar.gz` » (si vous avez uniquement un fichier `.tar`, sans de `.gz` à la fin,« `tar xvf linux.x.y.z.tar` » marchera sans problème.). Vous verrez la liste des fichiers se dérouler sous vos yeux. Lorsque c'est terminé, il y aura un nouveau répertoire « `linux` ». Allez dans **Linux** et jetez un coup d'oeil au fichier `README`. Il y a une section « `INSTALLING the kernel` ». Suivez les intructions lorsque nécessaire - prenez garde à ce que les liens soient en place et que les fichiers `.o` soient détruits, etc.

3.3 Configurer le noyau

Note : il s'agit d'une répétition, clarification du paragraphe correspondant dans le fichier `README` de Linus.

La commande « `make config` » lance un script qui va vous poser un certain nombre de questions. Il a besoin de bash, donc vérifiez que vous possédez bien `/bin/bash`, `/bin/sh`, ou `$BASH`.

Il existe un certain nombre d'autres possibilités autres que `make config` et il est possible que vous les trouviez plus simple à utiliser. Pour ceux qui utilisent X, vous pouvez essayer `make xconfig` si vous avez au préalable installé Tk (*click-o-rama* - Nat). `make menuconfig` est destiné à ceux qui ont installé (n)curses et qui préfèrent utiliser un menu en mode texte. Ces interfaces ont un avantage: si vous vous trompez lors de la configuration, vous pouvez facilement corriger l'erreur sans avoir à tout relancer.

Vous êtes alors prêt à répondre aux questions, la plupart du temps par « `y` » ou par « `n` ». Certaines options non critiques ou évidentes ne seront pas décrites ici. Consultez le paragraphe « Autres options de configuration » pour une courte descriptions des options non détaillées ici.

Dans la version 2.0, une option «? » est présente: elle permet d'obtenir une aide pour l'option désirée. Les commentaires sont sûrement les informations les plus à jour sur le gestionnaire.

(NdT: les titres suivants correspondent aux questions qui vous sont posées. Une traduction française se trouve entre parenthèses.)

3.3.1 Kernel math emulation (Emulation du coprocesseur Arithmétique)

Si vous ne possédez pas de coprocesseur arithmétique (autrement dit, vous ne disposez que d'un 386 ou d'un 486SX), répondez « `y` ». Si au contraire, vous en avez un et que vous répondez « `y` », ce n'est pas grave: le coprocesseur sera utilisé et l'émulation ignorée. La seule conséquence est une taille supérieure du noyau. On m'a dit que l'émulations du coprocesseur est lente, bien que cela n'ait rien à voir avec ce paragraphe, c'est un point qu'il faut tenir en compte surtout si vous trouvez que le système X Window est lent.

3.3.2 Normal (MFM/RLL) disk and IDE disk/cdrom support (Gestion Standard des disques durs)

Vous devez à peu près toujours répondre oui à cette question. Cela signifie que le noyau pourra gérer les disques standards des PC comme IDE, que bon nombre de gens possèdent. Ceci n'inclut pas les disques SCSI: ils sont sélectionnables plus tard dans la configuration.

Une question porte sur les gestionnaires de périphériques « old disk-only » et « new IDE ». Vous devez choisir l'un des deux. La principale différence est que le vieux gestionnaire de périphérique ne supporte qu'au plus deux disques sur une seule interface. Le nouveau gestionnaire gère une deuxième interface IDE et les disques CD-ROM IDE/ATAPI , et il est plus gros de 4 Ko que l'ancien. Cela signifie qu'il est possible qu'il puisse avoir quelques bogues, mais il peut améliorer les performances de votre disque, notamment si vous possédez du matériel récent (comme de l'EIDE).

3.3.3 Networking support (Gestion du Réseau)

En principe, vous ne devriez répondre « `y` » que si votre machine se trouve sur un réseau tel qu'Internet ou si vous voulez utiliser SLIP, PPP, term, etc. pour communiquer avec un accès Internet externe. Toutefois, bon nombre de paquetages (tels que le système X Window) a besoin de ce support réseau même si votre machine n'y est pas relié; dans ce cas, répondez « `y` ». Plus tard, on vous demandera si vous voulez avoir le support TCP/IP. De même, répondez « `y` », à moins que vous ne soyez sûr de ce que vous faites.

3.3.4 Limit memory to low 16MB (Vous avez moins de 16 Mo)

Il y a quelques contrôleurs DMA bogués pour 386 qui, au delà de 16Mo de mémoire, ne peuvent rien adresser correctement. Dans ce cas assez rare, répondez « `y` ».

3.3.5 System V IPC

L'une des meilleures définition des IPC (Communication inter-processus) se trouve dans l'index du livre de Perl. Cela n'est pas surprenant : certains développeurs Perl l'utilisent pour permettre aux processus de communiquer entre eux, comme d'autres paquetages(comme par exemple DOOM), donc ce n'est pas vraiment une bonne idée de répondre « `n` » à cette question à moins que vous ne sachiez exactement ce que vous faites.

3.3.6 Processor type (386, 486, Pentium, PPro)

Dans les anciens noyaux, il s'agissait de l'option Use -m486 flag for 486-specific optimizations.

Cette option permettait d'optimiser certaines optimisations pour un type de processeur, le noyau fonctionnant normalement avec les autres processeurs. Le noyau généré avait toutefois une taille légèrement supérieure. Dans les nouveaux noyaux, cette différence de taille n'est plus d'actualité donc vous pouvez indiquer le processeur que vous allez utiliser avec le nouveau noyau. Un noyau « 386 » fonctionnera avec toutes les machines.

3.3.7 SCSI support (Gestion SCSI)

Si vous avez un périphérique SCSI, répondez « `y` ». On vous demandera alors de plus amples renseignements sur la gestion des CD-ROM, des disques, et de quel type de contrôleur SCSI vous possédez. Consultez SCSI-HOWTO pour plus de détails.

3.3.8 Network device support (Gestion de périphériques réseau)

Si vous avez une carte réseau, ou si vous voulez utiliser SLIP, PPP, ou un port parallèle, répondez « `y` » . Le script de configuration vous demandera alors quel type de carte vous possédez et quels protocoles vous voulez inclure.

3.3.9 Filesystems (Systèmes de gestion de fichiers)

On vous demandera si vous désirez gérer un certain nombre de gestionnaires de fichiers. Il y a :

Standard (minix) - Les distributions ne créent pas de gestionnaires de fichiers de type minix, et bon nombre de personnes ne s'en servent pas, mais cela peut être une bonne idée de le configurer. Quelques programmes de « sauvetage de disques » l'utilisent, et encore aujourd'hui, un bon nombre de disquettes risquent avoir ce systeme de fichiers, puisque que ce système est optimal pour les disquettes.

Extended fs - C'était la première version du gestionnaire de fichiers étendu, qui n'est plus utilisée.

Second extended - Très largement utilisé dans les nouvelles distributions. Vous avez sûrement ce système sur votre machine. (NdT : et puis c'est un Français qui l'a fait alors... !)

xiafs filesystem - A une époque, il était assez utilisé mais maintenant, je ne connais personne qui l'utilise.

msdos - Si vous voulez utiliser les partitions de votre disque MS-DOS, ou monter des disquettes au format MS-DOS, répondez « `y` ».

umsdos - C'est une extension du gestionnaire de fichiers MS-DOS, avec des caractéristiques semblables à celles d'Unix, telles la longueur des noms. Ce n'est pas très utile pour des gens (comme moi) qui n'utilisent pas MS-DOS.

/proc - Un des systèmes de fichiers les plus puissants (idée effrontément chipée aux Laboratoires Bell, je pense). Ca n'a rien à voir avec vos partitions disques, mais c'est une interface de gestionnaire de fichiers avec le noyau et les processus. Un grand nombre d'outils de visualisation de processus (comme « `ps` » l'utilisent. Si vous l'avez installé, essayez « `cat /proc/meminfo` » ou « `cat /proc/devices` ». Quelques shells (comme rc) utilisent `/proc/self/fd` (connu comme `/dev/fd` sur d'autres systèmes) pour les entrées/sorties. Vous devriez répondre « `y` » à ceci : de nombreux programmes standard **Linux** en ont besoin.

NFS - Si vous êtes en réseau et que vous voulez partager des fichiers, répondez « `y` ».

ISO9660 - Format rencontré sur de nombreux CD-ROM. Si vous possédez un CD-ROM et que vous comptez vous en servir sous Linux, répondez « `y` ».

OS/2 HPFS - A l'heure ou j'écris ce document, c'est un gestionnaire de fichiers en lecture seule pour OS/2 HPFS.

System V et Coherent - pour les partitions de type System V et les systèmes Coherent (il y a d'autres variantes pour Unix PC).

Mais je ne sais pas quels sont les gestionnaires de fichiers dont j'ai besoin ! Bon, tapez « `mount` », cela va vous afficher un truc comme ça :

```
blah# mount
/dev/hda1 on / type ext2 (defaults)
/dev/hda3 on /usr type ext2 (defaults)
none on /proc type proc (defaults)
/dev/fd0 on /mnt type msdos (defaults)
```

Observez chacune des lignes. Le mot à coté de « `type` » est le nom du gestionnaire de fichier. Dans cet exemple, mes partitions `/` et `/usr` sont du type ext2fs, j'utilise `/proc`, et il y a un lecteur de disquettes monté en utilisant le gestionnaire de fichier MS-DOS (beuurrrk !).

Essayez « `cat /proc/filesystems` » si vous avez /proc. Cela vous donnera la liste des gestionnaire de fichiers que votre noyau gère.

La configuration de certains systèmes de fichiers rarement utilisés et qui ne sont pas absolument nécessaire peuvent augmenter la taille du noyau. Voir la section sur les modules chargeables pour résoudre ce problème, ainsi que la section sur les gros noyau.

3.3.10 Character devices (Périphériques caractères)

Vous trouverez ici les périphériques pour gérer vos imprimantes, souris bus, souris PS/2 (beaucoup de portables utilisent une souris constructeur de type PS/2), quelques gestionnaires de cassettes, et d'autres périphériques de type « caractère ». Répondez « `y` » quand vous le jugez bon.

Note: « Selection » est un programme qui vous permet d'utiliser la souris à l'extérieur de X-Windows pour effectuer des opérations de copier-coller entre les consoles virtuelles. C'est assez utile si vous avez une souris série, car cela s'interface bien avec X-Windows, mais vous devrez faire quelques trucs pour que cela fonctionne avec les autres. Le support Selection a été une option de configuration pendant longtemps mais c'est désormais un standard.

Note 2: « Selection » est désormais considéré comme obsolète. « gpm » est le nom du nouveau programme. Il peut faire plus de chose, comme gérer plus de souris, etc.

3.3.11 Sound card (Carte sons)

Si aimez entendre `biff` aboyer, répondez « `y` », et plus tard, un autre programme de configuration va se compiler puis vous demander ce que vous désirez installer. (Un mot sur la configuration des cartes sons: lorsqu'il vous demande si vous voulez installer la version entière du gestionnaire, vous pouvez répondre '`n`' et donc économiser de la mémoire en insérant uniquement les caractéristiques dont vous avez besoin.). Il est grandement recommandé de consulter le Sound-Howto pour plus de détail consernant la gestion des cartes sons.

3.3.12 Autres options de configuration

Toutes les options de configuration ne sont pas détaillées dans ce document car elles changent trop souvent ou bien sont réellement évidentes (par exemple, le support de la carte 3Com 3C509 permet la compilation du gestionnaire.). Il existe une liste assez complète de ces options (plus une manière de les insérer dans le script `Configure`) mise à jour par Axel Boldt (`axel@uni-paderborn.de`) sur l'URL suivante:

```
http://math-www.uni-paderborn.de/~axel/config_help.html
```

ou bien via ftp anonyme:

```
ftp://sunsite.unc.edu/pub/Linux/kernel/config/krnl_cnfg_hlp.x.yz.tgz
```

où `x.yz` est le numéro de la version.

Pour les noyaux plus récentts (2.0.x et supérieurs), ce fichier a été intégré dans les sources.

3.3.13 Kernel hacking

Extrait du README de Linus:

La configuration « kernel hacking » provoque généralement un noyau plus gros ou plus lent (ou les deux), et peut même rendre le noyau moins stable en configurant quelques routines pour essayer de chercher les défauts du code, pour trouver les problèmes du noyau (kmalloc()). Donc, vous devrez répondre « n » à cette question si vous créez un noyau pour un système sûr.

3.4 Garçon, la suite ! (Le Makefile)

Après avoir fait `make config`, un message vous indique que le noyau est configuré, et qu'il faut « regarder le Makefile général pour des configurations supplémentaires » , etc.

Donc, jetez un coup d'oeil au Makefile. Vous ne changerez probablement rien, mais ça ne peut pas vous faire de mal ! Vous pouvez également changer ici les options en rapport avec la commande « `rdev` », une fois que le noyau est prêt.

4 Compiler le noyau

4.1 Nettoyer et chercher les dépendances

Lorsque le script de configuration va se terminer, il vous demandera de faire un « `make dep` » et un « `make clean` ». C'est-à-dire de nettoyer les répertoire et d'effectuer la création des dépendances, pour les fichiers d'include et ce genre de choses. Pour cela, faites « `make dep` ». Cela ne prend pas trop de temps à moins que vous n'ayez une machine très lente. Pour les anciennes versions, lorsque c'est terminé faites un « `make clean` ». Cela détruit tous les fichiers objets et d'autre fichiers que le noyau compile. *N'oubliez pas cette étape.*

4.2 La compilation

Après avoir cherché les dépendances et avoir nettoyé l'arborescence, vous pouvez lancer la compilation par « `make zImage` » ou « `make zdisk` » (C'est cette partie qui prend du temps). « `make zImage` » va compiler le noyau, et laisser le fichier dans le répertoire `arch/i386/boot`. Le noyau s'appelle « `zImage` ». C'est le nouveau noyau compressé. « `make zdisk` » fait la même chose, mais copie le noyau sur une disquette que vous aurez sûrement insérée dans le lecteur « A: ».

« `zdisk` » est assez commode pour tester de nouveaux noyaux ; s'il ne fonctionne pas du tout, enlevez juste la disquette et relancez la machine avec un ancien noyau. C'est également une méthode pratique si vous avez détruit accidentellement votre noyau. Vous pouvez également vous en servir pour installer de nouveaux systèmes, en copiant directement le contenu d'un disque sur un autre.

Tous les noyaux assez récents sont compressés, comme l'indique le « `z` » en début des noms. Un noyau compressé est automatiquement décompacté lors de l'exécution.

4.3 Les autres options du « make »

« `make mrproper` » est un « `make clean` » plus puissant. Cela est parfois nécessaire, vous devriez le faire à

chaque patch. Regardez la section sur les modules à propos du « `make modules` ».

4.4 Installer le noyau

Une fois que vous avez un nouveau noyau qui fonctionne comme vous le voulez, il est grand temps de l'installer. Un grand nombre de personnes utilisent LILO (LInux LOader). « `make zlilo` » installera votre noyau, lancera LILO, et laissera votre machine prête à rebooter, mais seulement si votre machine est configurée de cette manière sur votre machine : le noyau s'appelle `/vmlinuz`, lilo se trouve dans `/sbin`, et votre fichier de configuration pour LILO (`/etc/lilo.conf`) correspond à ces données.

Autrement, vous devrez utiliser LILO directement. C'est un produit assez facile à installer, mais qui peut dérouter en raison de son fichier de configuration. Regardez le fichier de configuration (soit `/etc/lilo/config` pour les versions les plus anciennes, soit `/etc/lilo.conf` pour les nouvelles), et regardez son contenu. Le fichier de configuration doit ressembler à cela :

```
image = /vmlinux
    label = Linux
    root = /dev/hda1
    ...
```

La ligne « `image =` » sert à indiquer le nom du noyau utilisé. Beaucoup de gens utilisent `/vmlinuz`. « `label` » est utilisé par LILO pour indiquer sur quel noyau ou sur quel système d'exploitation vous allez booter (nom logique), et « `root` » est la racine de ce système d'expoitation. Faites une sauvegarde de votre vieux noyau (quel que soit son nom), et copiez le nouveau noyau (zImage) à sa place : « `cp zImage /vmlinuz` » si vous utilisez « `/vmlinuz` ». Puis relancez lilo. Sur les systèmes récents, vous pouvez uniquement lancer « `lilo` », mais sur de plus anciens, vous devez faire un `/etc/lilo/install` ou `/etc/lilo/lilo -C /etc/lilo/config`.

Si vous désirez en savoir plus sur la configuration de LILO, ou si vous voulez avoir LILO, récupérez la dernière version sur votre site FTP favori et suivez les instructions.

Pour booter sur l'un de vos anciens noyaux situés sur votre disque dur (une autre manière de prévenir un problème du à un nouveau noyau), copiez les lignes suivantes « `image = xxx` » à la fin du fichier de configuration LILO, et changez « `image = xxx` » par « `image = yyy` », où « `yyy` » est le chemin complet du noyau de secours. Puis changez « `label = zzz` » en « `label = linux-sauvegarde` » et relancez `lilo`. Vous devrez surement rajouter une ligne dans le fichier de configuration indiquant « `delay=x` », où x est le temps en dixième de secondes qui indique à LILO le temps qu'il doit attendre avant d'amorcer la machine. Vous pouvez l'interrompre (avec la touche shift par exemple), et taper le mot clef de l'image du noyau de sauvegarde (au cas ou un problème se produit).

5 Mettre à jour le noyau

5.1 Appliquer un patch

Les nouvelles versions du noyau sont distribuées sous la forme de patches. Par exemple, si vous possédez la version 1.1.45, et que vous remarquez qu'il existe un « `patch46.gz` », cela signifie que vous pouvez passer à

la version 1.1.46 en appliquant ce patch. Vous devriez faire avant une sauvegarde de votre arborescence des sources de **Linux** (« `make clean` » puis « `cd /usr/src; tar zcf old-tree.tar.gz linux` » va produire une archive compressée).

Poursuivons avec cet exemple et supposons que vous ayez mis le fichier « `patch46.gz` » dans `/usr/src`. Allez dans `/usr/src` et faites un « `zcat patch46.gz | patch -p0` » (ou « `patch -p0 < patch46` » si le patch est compressé). Vous verrez alors une liste de messages vous indiquant les essais de modifications. Cela marche ou pas. (En principe oui !). Généralement, cela va trop vite pour les lire, et on ne sait pas trop si ça a marché. Dans ce cas, vous devriez ajouter l'option `-s` à `patch`, ce qui lui indique qu'il ne doit afficher que les erreurs. (Vous n'avez pas grand chose à faire des « héhé, mon ordinateur est en train de faire des quelque chose...! »). Pour regarder ce qui se passe, allez dans `/usr/src/linux` et cherchez les fichiers ayant pour extension `.rej`. Quelques versions de patch (vieilles versions) laissent les fichiers rejetés avec `#` pour extension. Vous pouvez utiliser « `find` » pour les trouver :

```
find .  -name '*.rej' -print
```

vous en donnera la liste avec le chemin pour y accéder.

Si tout a marché, faites un « `make clean` », « `config`, » et « `dep` » comme décrit dans les sections 3 et 4.

Il y a quelques options pour la commande `patch`. Comme indiqué ci-dessus, `patch -s` supprime tous les messages sauf les erreurs. Si vous stockez les sources de votre noyau dans un autre répertoire que `/usr/src/linux`, un `patch -p1` dans ce répertoire fera les choses proprement. Les autres options sont bien documentées dans les pages de manuel.

5.2 Si quelque chose ne fonctionne pas

(Note : cette section traite plutôt des noyaux assez anciens)

Le problème le plus fréquent qui se présente est lorsqu'un patch modifie le fichier « `config.in` ». Dans ce cas, ça ne marche pas très bien, car vous avez changé les options pour mieux coller à votre machine. En principe, ça ne devrait plus trop se produire, mais avec les anciennes versions... Pour résoudre ce problème, jetez un coup d'oeil au fichier `config.in.rej` et regardez son contenu. Le changement sera indiqué par « `+` » et « `-` » au début d'une ligne. Regardez ces lignes et retenez si elles sont marquées « `y` » ou « `n` ». Maintenant, éditez `config.in`, et changez les « `y` » en « `n` » et les « `n` » en « `y` » lorsque cela est nécessaire. Faites un

```
patch -p0 < config.in.rej
```

et si cela fonctionne, alors vous pouvez continuer avec la configuration et la compilation. Le fichier `config.in.rej` restera, mais vous pouvez le détruire.

Si vous avez d'autres problèmes, vous avez dû installer un patch défectueux. Si la commande patch indique « `previously applied patch detected: Assume -R?` », vous êtes probablement en train d'appliquer un patch déjà appliqué. Si vous répondez « `y` », cela risque détruire votre source (chose qui peut ne pas être une mauvaise idée dans un premier temps...).

Pour revenir en arrière (dépatcher), faites un « `patch -R` » sur le patch original.

La meilleure chose à faire lorsqu'un patch détruit tout, est de repartir d'un noyau initial tout neuf ! (par exemple, à partir des fichiers `linux-x.y.z.tar.gz`).

5.3 Comment se débarasser des fichiers .orig?

Après avoir appliqué quelques patches, les fichiers `.orig` vont commencer à s'empiler. Par exemple, j'en étais à la version 1.1.51 et la dernière fois que j'avais fait le ménage, c'était avec la version 1.1.48 (je crois...). Détruire les fichiers .orig a permis de récupérer près d'un Méga octets.

```
find .  -name '*.orig' -exec rm -f {} ';'
```

fera cela pour vous. Quelques versions de `patch` qui utilisent `#` pour les rejets utilisent un tilde à la place de `.orig`.

Il y a d'autres manières (meilleures?) pour se débarrasser des fichiers .orig qui utilisent le programme GNU `xargs` :

```
find .  -name '*.orig' | xargs rm
```

ou la méthode sûre mais un peu plus verbeuse :

```
find . -name '*.orig' -print0 | xargs --null rm --
```

5.4 Autres patches

Il y a toujours d'autres patches (je les appellerais « non-standards ») que ceux que distribue Linus. Si vous les appliquez, les patches Linus risquent de ne plus marcher correctement et vous serez obligé soit de les enlever, soit d'adapter les patches. C'est généralement un travail assez pénible pour les novices, aussi revenir aux anciens sources avant d'appliquer les patches de **Linux** semble être une bonne solution. Après, vous pouvez regarder si les patches non standards fonctionnent. S'ils ne fonctionnent pas, soit revenez à l'ancienne version, soit essayez de modifier le patch pour le faire fonctionner, ou alors attendez qu'un nouveau patch arrive.

Quelle est l'utilité des patches qui ne trouvent pas dans les distributions standards ? Vous le saurez en écoutant les autres. J'ai l'habitude d'utiliser le patch « noblink » car j'ai horreur des curseurs qui clignotent (Ce patch est (ou bien était) mis à jour fréquemment pour les nouveaux noyaux). Avec les gestionnaires de périphériques étant de plus développés sous la forme de modules chargeables, le terme patch « non standards » a de moins en moins de signification.

6 Paquetages supplémentaires

Votre noyau de **Linux** a bon nombre de caractéristiques qui ne sont pas expliquées dans le noyau lui-même ; elles sont typiquement utilisées par des paquetages externes. Une liste des plus utiles est donnée ici.

6.1 kbd

La console de **Linux** possède un bon nombre de caractéristiques. Elle peut changer les polices de caractères, remapper, permuter les modes vidéos (dans les noyaux les plus récents), etc. Le package kbd possède des programmes qui permettent à l'utilisateur de faire tout cela, plus une tonne de fontes et de types de claviers. (NdT : fr.map pour le clavier français).

6.2 util-linux

Rik Faith (`faith@cs.unc.edu`) met à jour une grande quantité d'outils **Linux** qui sont appelés util-linux. Disponibles via ftp anonyme sur sunsite.unc.edu dans `/pub/Linux/system/misc`, il contient des programmes tels que `setterm`, `rdev`, et `ctrlaltdel`, qui sont dépendants du noyau. Comme le dit Rik , *n'installez pas sans réfléchir*, vous n'avez pas besoin de tout installer dans la distribution, et cela pourrait vous causer de graves ennuis.

6.3 hdparm

Plusieurs packages sont livrés avec le noyau ainsi que des programmes. Les patches et les programmes pour optimiser et jouer avec vos disques sont distribués séparément.

6.4 gpm

gpm est un gestionnaire de souris. Ce programme vous permet d'effectuer du couper-coller en mode texte entre les fenêtres virtuelles ainsi que d'autres opérations avec un grand nombre de types de souris.

7 Quelques pièges

7.1 make clean

Si votre noyau a un comportement surnaturel (ça m'est arrivé!), il y a des chances pour que vous ayez oublié de faire un « make clean ». Les symptômes peuvent être, un effondrement de votre système, des problèmes d'entrées sorties étranges, une chute des performances, des reboot aléatoires,... Vérifiez que avez également fait un make dep.

7.2 Noyaux énormes ou lents

Si votre noyau consomme beaucoup de mémoire, ou s'il est réellement gros, ou bien s'il faut une éternité pour le compiler même lorsque vous utilisez votre nouveau 786DX6/440, c'est que vous avez configuré un bon nombre de périphériques dont vous n'avez pas besoin. Si vous ne les utilisez pas, ne les configurez pas car cela prend beaucoup de place en mémoire. Si vous avez moins de 16 Mo de mémoire, répondez bien y à la question « `limit memory to low 16MB` ». Cela améliore assez bien les performances (surtout sur un système possédant 4Mo). Le symptôme le plus visible est l'augmentation sensible du fonctionnement du swap. Si votre

disque fait beaucoup de bruit, et qu'il ne s'agit pas d'un de ce vieux disques Fujitsu Eagles qui fait le bruit d'un avion lors de son attérissage lorsque vous l'éteignez, jetez un coup d'oeil à votre configuration.

Vous pouvez calculer la taille mémoire que le noyau utilise en prenant la mémoire totale de votre machine, et en soustrayant la valeur de la mémoire totale (« total mem ») dans `/proc/meminfo` ou bien avec la commande « `free` ». Vous pouvez trouver ces renseignement avec la commande « `dmesg` » (ou bien en regardant le fichier log du noyau, lorsqu'il existe sur votre système).C'est une ligne de ce genre :

```
Memory: 15124k/16384k available (552k kernel code, 384k reserved, 324k data)
```

Mon 386 (qui a été volontairement mal configuré) indique :

```
Memory: 7000k/8192k available (496k kernel code, 384k reserved, 312k data)
```

Si malgrè tout vous avez un noyau trop gros, faîtes un `make bzimage`. Vous devrez toutefois avoir installé une version récente de LILO.

7.3 Le noyau ne compile pas

Si cela ne compile pas, alors le patch a sûrement échoué, ou bien vous possédez des sources corrompus. Votre version de gcc peut également ne pas être correcte. Soyez sûr que les liens que Linus décrit dans le fichier README sont corrects. En général, si un noyau ne compile pas, c'est qu'un truc ne tourne pas rond, et il est plus que probable que certains outils doivent être reinstallés.

Ou peut-être vous ètes en train de compiler un noyau 1.2.x avec un compilateur ELF (gcc 2.6.3 et supérieur). Si vous obtenez un message du genre `so-and-so undefined` pendant la compilation, il y a des chances que cela soit ce problème. La solution est très simple. Ajoutez ces lignes au début de votre fichier `arch/i386/Makefile`:

```
AS=/usr/i486-linuxaout/bin/as
LD=/usr/i486-linuxaout/bin/ld -m i386linux
CC=gcc -b i486-linuxaout -D__KERNEL__ -I$(TOPDIR)/include
```

Puis, relancez `make dep` et `zImage`.

Dans des cas relativement rares, gcc peut échouer en raison de problèmes de matériel. Le message d'erreur ressemble à un truc assez mysterieux « xxx exited with signal 15 ». Je n'en n'aurais probablement pas parlé si cela ne m'étais arrivé une fois. J'avais un cache mémoire deffectueux et le compilateur avait un résultat plutôt aléatoire. Essayez dans un premier de reinstaller gcc si vous avez des probmlèmes. Vous devriez vous méfier si votre noyaux compile très bien avec les caches externes vidés, une mémoire réduite, etc.

Cela semble géner certaines personnes lorsque je met en doute leur matériel. Je ne le fais pas exprès. Il existe une FAQ dédiée à ce sujet : `http://www.bitwizard.nl/sig11/`.

7.4 La nouvelle version du noyau ne boot plus !

Soit LILO ne fonctionne pas, soit il n'est pas configuré correctement. Une fois, un problème dans le fichier de config m'a posé pas mal de soucis : j'avais mis « `boot = /dev/hda1` » à la place de « `boot = /dev/hda` »

(cela peut sembler ennuyant à première vue, mais une fois que vous avez un fichier de configuration qui fonctionne, vous ne devez pas avoir besoin d'y retoucher).

7.5 Vous avez oublié de lancer LILO, ou bien votre système ne boote plus du tout

Argh ! La meilleure chose à faire est de booter à partir d'une disquette et de préparer une nouvelle disquette de boot (« `make zdisk` » fait cela très bien). Vous avez besoin de savoir où votre partition racine (`/`) se trouve et quel est son type (ext2fs, minix, etc). Dans l'exemple ci-dessous, vous aurez également besoin de connaître la partition des sources du noyau (`/usr/src/linux`), et où il est montée.

Dans cet exemple,la racine `/` est `/dev/hda1`, la partition qui supporte `/usr/src/linux` est `/dev/hda3`, normalement montée sur `/usr`. Ils sont tous deux des systèmes de type ext2fs. L'image du noyau se trouve dans `/usr/src/linux` et elle s'appelle `zImage`.

L'idée est que s'il existe un noyau qui fonctionne dans `/usr/src/linux` et qui s'appelle `zImage`, il est alors possible de l'utiliser pour la nouvelle disquette. Une autre alternative, qui peut marcher ou pas est présentée après cet exemple (cela dépend des systèmes).

En premier, bootez à partir d'une disquette d'installation (boot/root) ou d'une disquette de secours. Ensuite, montez la partition où se trouve le noyau en état de marche.

```
mkdir /mnt
mount -t ext2 /dev/hda3 /mnt
```

Si `mkdir` vous annonce que le répertoire existe, ignorez le message. Maintenant, allez dans le répertoire où se trouve le noyau en état de marche Notez que

```
/mnt + /usr/src/linux/arch/i386/boot - /usr = /mnt/src/linux/arch/i386/boot
```

Insérez une disquette formatée dans le lecteur A: (vérifiez qu'il ne s'agit pas de la disquette boot ou root !), faites une copie de l'image sur le disque et configurez votre partition racine :

```
cd /mnt/src/linux/arch/i386/boot
dd if=zImage of=/dev/fd0
rdev /dev/fd0 /dev/hda1
```

Allez à la racine `/`, et démontez la partition `/usr` :

```
cd /
umount /mnt
```

Maintenant, vous devriez être capables de rebooter votre système normalement à partir de cette disquette. N'oubliez pas de lancer lilo avant de rebooter !

Comme mentionné ci-dessus, il y a une autre manière très pratique. S'il vous arrive d'avoir un noyau fonctionnant dans `/` (`/vmlinuz` par exemple), on peut s'en servir. Supposons que vous remplissiez les conditions

ci-dessus, et que votre noyau s'appelle `/vmlinuz`, faites juste ceci : changez `/dev/hda3` en `/dev/hda1` (la partition `/`), `/mnt/src/linux` en `/mnt`, et `if=zImage` en `if=vmlinuz`. La petite note expliquant comment aller dans `/mnt/src/linux` peut être oubliée.

Utiliser LILO avec de gros disques (avec un nombre de cyclindres supérieur à 1024) peut poser des problème. Consultez le mini-Howto LILO ou d'autre documentation à ce sujet.

7.6 Il me dit « warning: bdflush not running »

Cela peut être un problème assez grave. Avec les noyaux ayant une version supérieure à 1.0 (je n'arrive pas à me souvenir laquelle exactement, mais c'était aux alentours du 20 Avril 1994), un programme appelé « `update` » qui vide périodiquement les buffers du gestionnaire de fichiers a été amélioré. Récupérez les sources de « `bdflush` » (vous pouvez les récupérer là où vous avez trouvé votre noyau), et compilez-le (il vaut mieux fonctionner avec un ancien noyau pendant la compilation et pendant l'installation). Il s'installera tout seul comme « `update` » et le système devrait fonctionner correctement après cela.

7.7 Il m'indique qu'il ne trouve pas certains symboles et il n'arrive pas à compiler

Vous possédez probablement un compilateur ELF (gcc et supérieur) et les sources du noyau 1.2.x. La correction a apporter et d'ajouter les lignes suivantes au début du fichier `arch/i386/Makefile`:

```
AS=/usr/i486-linuxaout/bin/as
LD=/usr/i486-linuxaout/bin/ld -m i386linux
CC=gcc -b i486-linuxaout -D__KERNEL__ -I$(TOPDIR)/include
```

Cela compilera un noyau 1.2.x avec les bibliothèques a.out.

7.8 Je n'arrive pas à faire marche mon CD-ROMM IDE/ATAPI CD-ROM

Aussi étrange que cela puisse paraître, beaucoup de gens n'arrivent pas à faire fonctionner leurs disques ATAPI, tout simplement parce qu'il y a un bon nombre de choses qui ne fonctionnent pas.

Si votre CD-ROM est le seul disque d'une interface IDE particulière il doit être configuré pour étant « maître » ou « esclave ». C'est l'erreur la plus fréquemment rencontrée.

Creative Labs (par exemple) a mis des interfaces IDE sur leurs cartes sons. Toutefois, cela pose un certain problème car certaines personnes ne possèdent qu'une seule interface mais beaucoup ont deux interfaces IDE sur leur carte mère (IRQ15 généralement), donc une pratique commune est de mettre la troisième interface IDE soundblaster à l'IRQ11.

Cela pose un problème avec Linux car les versions 1.2.x ne supportent pas une troisième interface IDE (cela est géré avec les versions 1.3.x mais ce sont des versions de développement, et la troisième interface n'est pas détectée automatiquement). Pour résoudre ce problème, vous avez plusieurs choix.

Si avez déjà un deuxième port IDE, il y a des chances pour que vous ne l'utilisiez pas ou qu'il n'ait pas deux périphériques connectés. Désactivez l'interface ATAPI de la carte son et connectez le disque sur votre seconde interface.

Si vous n'avez pas une seconde interface, l'interface IDE de la carte son est normalement sur l'IRQ 15. Cela devrait fonctionner.

Si pour certaines raisons, vous devez réellement utiliser la « troisième » interface, utilisez un noyau 1.3.x (par exemple 1.3.57), et lisez `drivers/block/README.ide`. Il y a beaucoup plus d'information.

7.9 Le noyau me dit des insanités à propos de requètes obsolètes !

Récupérez une nouvelle version du progamme `route` ainsi que tout autre programme effectuant le même genre de manipulations : `/usr/include/linux/route.h` (qui est actuellement un fichier dans `/usr/src/linux`) a changé.

7.10 Le Firewall ne fonctionne pas dans la version 1.2.0

Mettez à jour à la version 1.2.1.

7.11 Ce n'est pas une image noyau compressée !

N'utilisez pas le fichier `vmlinux` créé dans `/usr/src/linux` comme image de boot :
`[..]/arch/i386/boot/zImage` est la bonne.

7.12 Problèmes avec la console après mise à jour à la version 1.3.x

Changez le mot `dumb` en `linux` dans l'entrée console du fichier `/etc/termcap`. Vous devrez également le faire dans le fichier `terminfi`.

7.13 Le noyau ne semble pas pouvoir compiler après une mise à jour

Le noyau **Linux** contient un certain nombre de fichiers includes (les fichiers se terminant par `.h`) qui se trouvent dans le répertoire `/usr/include`. Ils sont référencés comme cela (ou `xyzzy.h` doit être dans `/usr/include/linux`) :

```
#include <linux/xyzzy.h>
```

Normalement, il y a un lien appelé `linux` dans `/usr/include` sur le répertoire `include/linux` de votre répertoire source **Linux** (`/src/linux/include/linux` dans un système standard). Si ce lien n'existe pas, ou bien pointe au mauvais endroit, bon nombre de programmes ne compileront pas. Si vous décidez que les sources du noyau prennent trop de place sur votre disque et que vous les détruisez, cela sera un problème. Un autre problème qui peut arriver, c'est avec les permissions d'accès aux fichiers. Si votre `root` a un umask qui n'autorise pas les autres utilisateurs de voir ses fichiers par défaut, et que vous désarchiviez les sources

du noyau avec l'option `p` (conserve le mode), les utilisateurs ne pourront pas utiliser le compilateur C. Vous devrez alors utiliser la commande `chmod` pour résoudre le problème. Il est plus facile de résoudre ce problème en réinstallant les fichiers includes. Vous pouvez procéder de la même manière que lors de l'installation des sources au début, en ajoutant un argument :

```
blah# tar zxvpf linux.x.y.z.tar.gz linux/include
```

Notez que « `make config` » va recréer le lien `/usr/src/linux` s'il n'existe pas.

7.14 Increasing limits

Les quelques commandes qui suivent peuvent être assez utiles à ceux qui se demandent comment augmenter certaines limites logicielles imposées par le noyau :

```
echo 4096 > /proc/sys/kernel/file-max
echo 12288 > /proc/sys/kernel/inode-max
echo 300 400 500 > /proc/sys/vm/freepages
```

8 Note concernant la mise à jour pour les noyaux 2.0

Les noyaux 2.0.x ont introduits une certaine quantité de modifications dans l'installation du noyau. Le fichier `Documentation/Changes` situé dans l'arborescence des sources contient un certain nombre d'informations que vous devez regarder lorsque vous effectuez la mise à jour de votre système. Il est fort probable que vous ayez à mettre à jour plusieurs paquetages tels que gcc, libc et SysVInit ainsi que plusieurs fichiers systèmes. Donc, ne paniquez pas !

9 Modules

Les modules chargeables du moyau peuvent économiser de la mémoire et sont faciles à configurer. La portée des modules a augmenté jusqu'à inclure les filesystems, les gestionnaires de cartes ethernet, les lecteurs de cartouches, les imprimantes, etc.

9.1 Installer les fonctionnalités modules

Ces fonctionnalités sont disponibles sur n'importe quel site où vous avez récupéré les sources du noyau : `modules-x.y.z.tar.gz`; choisissez la dernière version `x.y.z` qui est égale ou inférieure à votre version du noyau. Installez-le avec '`tar zxvf modules-x.y.z.tar.gz`', `cd` dans le répertoire crée (`modules-x.y.z`), regardez le fichier `README`, et suivez les instructions d'installation (qui sont en général très simples, du genre, faites `make install`). Vous devriez avoir maintenant les programmes `insmod`, `rmmod`, `ksyms`, `lsmod`, `genksyms`, `modprobe`, et `depmod` dans `/sbin`. Si vous voulez les tester avec le gestionnaire d'exemple « hw » dans `insmod` ; regardez le fichier `INSTALL` dans ce sous-répertoire pour les détails.

`insmod` insère un module dans le noyau en cours d'exécution. Les modules ont généralement une extension `.o`. Le pilote fourni en exemple est appelé `drv_hello.o`, donc, pour l'insérer, `insmod drv_hello.o`'. Pour voir les modules que le noyau utilise, lancer `lsmod`. Il devrait alors afficher :

```
blah# lsmod
Module:        #pages:  Used by:
drv_hello          1
```

'`drv_hello`' est le nom du module, il utilise une page mémoire (4k), et aucun autre module du noyau ne dépend de lui. Pour le retirer, lancer '`rmmod drv_hello`'. Notez que `rmmod` prend comme argument *le nom du module,* pas un nom de fichier ; ce nom peut être obtenu en exécutant `lsmod`.

Les autres programmes de gestion des modules sont documentés (pages de manuel).

9.2 Modules distribués avec le noyau

Avec la version 2.0.30, bon nombre de filesystems, quelques pilotes SCSI, plusieurs gestionnaires de cartes Ethernet, et d'autres périphériques sont des modules chargeables. Pour les utiliser, soyez sûr que vous ne les insérez pas dans la configuration du noyau. Pour cela, ne répondez pas `y` pendant le '`make config`'.

Compilez un nouveau noyau et bootez avec. Puis, `cd` dans `/usr/src/linux` à nouveau, et faites un '`make modules`'. Cela va alors compiler tous les modules que vous n'avez pas spécifié dans le noyau, et installer les liens dans `/usr/src/linux/modules`. Vous pouvez alors les installer en faisant '`make modules_install`', qui les installera dans `/lib/modules/x.y.z`, ou `x.y.z` est le numéro de version.

Cela peut être particulièrement utile pour les filesystems. Vous n'utilisez peut-être pas fréquemment les filesystems minix ou msdos. Par exemple, si je rencontre une disquette msdos, je devrai faire un `insmod /usr/src/linux/modules/msdos.o`, puis `rmmod msdos` lorsque j'ai terminé, Ce système économise près de 50 Ko de RAM dans l'utilisation du noyau. Remarque : il est très fortement conseillé de *toujours* le configurer directement dans le noyau en cas de récupération de disque.

10 Autres options de configuration

Cette section contient une description d'options particulières du noyau (`make config`) qui ne sont pas décrites dans la paragraphe de configuration Pas mal de gestionnaires de périphériques ne sont pas listés ici.

10.1 Configuration générale

`Normal floppy disk support` : utilisation normale des lecteurs de disquettes. Il vaut mieux lire le fichier `drivers/block/README.fd`; Ceci est très important pour les utilisateurs d'IBM Thinkpad.

`XT harddisk support` : Si vous voulez utiliser les contrôleurs XT 8bits.

`PCI bios support` : Si vous avez un PCI, vous voudrez sûrement activer cette option. Attention : certaines vieilles cartes mères PCI risquent de ne pas supporter cette option. Des informations supplémentaires sont détaillées dans le PCI-HOWTO.

`Kernel support for ELF binaries`: ELF est un système pour permettre aux binaires de franchir les architectures et les systèmes d'exploitation. **Linux** semble se diriger dans cette direction.

`Set version information on all symbols for modules`: dans le passé, les modules du noyau étaient recompilés avec tout nouveau noyau. Si vous dites `y`, il sera alors possible d'utiliser les modules avec un numéro de version différent. Lisez `README.modules` pour plus de détails.

10.2 Options de réseau

Ces options sont décrites dans le NET-3-HOWTO.

11 Trucs et astuces

11.1 Rediriger la sortie standard des commandes make ou patch

Si vous voulez voir ce que font les commandes « `make` » ou « `patch` », vous pouvez rediriger la sortie des messages. En premier, regardez quel shell vous utilisez. « `grep root /etc/passwd` » et cherchez quelque chose comme « `/bin/csh` ».

Si vous utilisez sh ou bash,

```
(commande) 2>&1 | tee  (Fichier de sortie)
```

placera une copie de la sortie dans le fichier

`(Fichier de sortie)`

.

Pour csh ou tcsh, la syntaxe est :

`(commande) |& tee (Fichier de sortie).`

Pour rc (Note : vous n'utilisez probablement pas rc) c'est :

```
(commande) > [2=1]| tee (Fichier de sortie).
```

11.2 Installations diverses du noyau

Autrement qu'en utilisant des disquettes, il y a d'autres méthodes pour tester un nouveau noyau sans détruire les précédents. A la différence d'autres systèmes Unix, LILO est capable de booter sur un noyau situé n'importe où sur le disque. (Si vous avez un gros disque (500 Mo ou plus), lisez la documentation de LILO car cela risque de poser quelques problèmes). Donc, ajoutez cela :

```
image = /usr/src/linux/arch/i386/boot/zImage
label = new_kernel
```

à la fin de votre fichier de configuration LILO, vous pouvez alors choisir de booter sur le nouveau noyau sans avoir à détruire l'ancien `/vmlinuz` (après avoir lancé `lilo`, bien sûr). La manière la plus simple d'indiquer à LILO qu'il doit booter sur un nouveau noyau est de presser la touche shift au boot (lorsqu'il vous indique `LILO`), ce qui vous donne un prompt. A ce niveau, tapez '`new_kernel`' pour qu'il boote sur le nouveau noyau.

Si vous désirez conserver plusieurs sources du noyau de **Linux** (cela peut prendre *beaucoup* de place; faites attention), la manière la plus simple est le nommer: `/usr/src/linux-x.y.z`, ou `x.y.z` est la version du noyau. Vous pouvez alors choisir le source à utiliser en faisant un lien symbolique; par exemple, '`ln -sf linux-1.2.2 /usr/src/linux`' indiquera la version 1.2.2 comme version courante. Avant de créer ce type de lien symbolique, soyez certain que le dernier argument de `ln` n'est pas un répertoire existant. Le résultat ne serait pas celui que vous espériez.

11.3 Mises à jours des noyaux

Russell Nelson (`nelson@crynwr.com`) résume les changements dans les nouvelles versions des noyaux. Ils sont courts et vous devriez les regarder avant de réaliser la mise à jour. Ils sont disponibles par ftp anonyme `ftp.emlist.com` dans `pub/kchanges` ou via WEB:

`http://www.crynwr.com/kchanges`

12 D'autres HOWTO qui peuvent être très utiles

- Sound-HOWTO: cartes sons et utilitaires,
- SCSI-HOWTO: tout sur les contrôleurs SCSI ainsi que les périphériques,
- NET-2-HOWTO: réseaux,
- PPP-HOWTO: réseau avec PPP,
- PCMCIA-HOWTO: à propos des périphériques pour votre portable,
- ELF-HOWTO: ELF: qu'est-ce que c'est, installation, etc,
- Hardware-HOWTO: survol du matériel supporté.
- Module-HOWTO: plus de détails sur les modules du noyau
- Kerneld mini-HOWTO: dédié à kerneld
- BogoMips mini-HOWTO: pour information

13 Divers

13.1 Auteur

L'auteur du Kernel-HOWTO **Linux** est Brian Ward (`ward@blah.math.tu-graz.ac.at`). S'il vous plaît, envoyez-lui des commentaires, corrections, rajouts. Les corrections sont en particuliers les plus importantes pour lui.

Vous pouvez jeter un coup d'oeil à ma page WWW :

`http://www.math.psu.edu/ward/`
`http://blah.math.tu-graz.ac.at/~bri/`

Même en étant aussi attentif que possible avec le courier, rappelez-vous que j'en ai *beaucoup* tous les jours, donc ça prendra un petit peu de temps pour vous répondre. Si le mail est une question, essayez d'être clair et détaillé dans la description du problème. Merci à toute personne qui me donnera ces informations. N'oubliez pas de m'indiquer les versions de gcc, de la libc et éventuellement la distribution que vous utilisez.

Si vous m'éccrivez et que vous n'obtenez aucune réponse dans un temps raisonable (trois semaines ou plus), alors il y a des chances pour que j'ai accidentellement détruit votre message, ou un autre problème (désolé). Rééssayez.

J'ai beaucoup de mail sur des problèmes de matériel ou de ce genre. S'il vous plait, souvenez-vos que que je ne suis pas familier avec tout le matériel et donc je peux être d'aucune aide dans certains cas. J'utilise normalement des machines avec des disques IDE et SCSI, des CD-ROM SCSI, des cartes éthernet 3Com et WD, des souris séries, des cartes mères PCI avec contrôleurs SCSI NCR 810, AMD 386DX40 w/Cyrix avec coprocesseur, AMD 5x86, AMD 486DX4, et des processurs Intel 486DX4.

La version 0.1 a été écrite le 3 Octobre 1994. Ce document est disponible aux formats : SGML, PostScript, TeX, roff et texte.

13.2 Reste à faire

Le paragraphe « Trucs et astuces » est un peu court. J'espère l'étoffer avec les suggestions venant d'autres personnes. De même pour « Autres paquetages ».

Des informations supplémentaires pour le déboguage/sauvetage lors de crash seraient les bienvenues.

13.3 Contributions

Une petite partie du README de Linus (cf. option kernel hacking) est incluse. (Merci, Linus !)

- `uc@brian.lunetix.de (Ulrich Callmeier)` : patch -s et xargs.
- `quinlan@yggdrasil.com (Daniel Quinlan)` : corrections et compléments dans un bon nombre de paragraphes.
- `nat@nataa.frmug.eu.org (Nat Makrévitch)` : mrproper, tar -p, et bon nombre d'autres choses

- boldt@math.ucsb.edu (Axel Boldt) : descriptions des options de configuration de noyau pour le réseau
- lembark@wrkhors.psyber.com (Steve Lembark) : suggestion pour le boot multiple.
- kbriggs@earwax.pd.uwa.edu.au (Keith Briggs) : quelques corrections et suggestions
- rmcguire@freenet.columbus.oh.us (Ryan McGuire): ajouts au niveau des options de compilation
- Eric.Dumas@freenix.fr (Eric Dumas) : Traduction française
- simazaki@yu-gate.yamanashi.ac.jp (Yasutada Shimazaki): Traduction japonaise
- jjamor@lml.ls.fi.upm.es (Juan Jose Amor Iglesias): Traduction espagnole
- mva@sbbs.se (Martin Wahlen) : traduction suédoise
- jzp1218@stud.u-szeged.hu (Zoltan Vamosi) : traduction hongroise
- bart@mat.uni.torun.pl (Bartosz Maruszewski) : traduction polonaise
- donahue@tiber.nist.gov (Michael J Donahue) : typos, gagnant de la « sliced bread competition »
- rms@gnu.ai.mit.edu (Richard Stallman) : Documentation sur la FSF.
- dak@Pool.Informatik.RWTH-Aachen.DE (David Kastrup) : NFS esr@snark.thyrsus.com (Eric Raymond) : quelques trucs et astuces

Les gens qui m'ont envoyé des mails avec des questions et des problèmes m'ont été également d'une aide précieuse.

13.4 Copyright, traduction

Copyright © Brian Ward, 1994-1997.

Version française Copyright © Eric Dumas, 1995-1997 dumas@freenix.fr
Cette version française peut être trouvée sur ftp.ibp.fr dans le répertoire /pub/linux/french/HOWTO.

Ce document peut être distribué sous n'importe quel format aussi longtemps qu'il n'est pas modifié, de même pour cette notice. La traduction est permise dans n'importe quelle langue, à condition que le nom du traducteur soit ajouté dans le document. Il n'y a **aucune garantie** concernant ce document ansi que sur son contenu. Personne ne peut être tenu pour responsable en cas de conséquence malheureuse de l'application de son contenu.

La distribution commerciale est autorisée et encouragée toutefois, il est fortement recommandé de contacter l'auteur avant la redistribution, dans l'intérêt de la mise à jour les données. Il en est de même pour les traductions.