Le HOWTO du noyau Linux (Kernel HOWTO)

par Brian Ward, ward@blah.tu-graz.ac.at Version 1.0, 5 juin 1999

(5 juillet 1999. Adaptation française par Eric Dumas dumas@freenix.fr jusqu'à la version 0.80, Christophe Deleuze christophe.deleuze@lip6.fr à partir de la version 1.0). Il s'agit d'un guide détaillé sur la configuration du noyau, sa compilation et ses mises à jour pour les plates-formes x86.

Contents

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 2.8.193 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, ça 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 standards, et ne devraient pas vous poser trop de problèmes. On suppose également que vous connaissez la structure de votre disque (partitions). 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 repartitionniez votre disque, en ajoutiez un nouveau, réinstalliez votre système, ou quelque chose de similaire.

La dernière version stable du noyau au moment où j'écris ces mots est la 2.2.9, 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 pairs : 1.2.x, 2.0.x sont des versions stables... tout comme la 2.2.x. Ces noyaux sont considérés comme étant les plus stables et sans erreurs. Les noyaux de développement (1.3.x, 2.1.x, etc) sont des noyaux de test, pour les développeurs, les testeurs... mais ces noyaux peuvent être très bogués. Vous êtes prévenus.

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, 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).

2 Quelques questions... avec leurs réponses

2.1 Mais quel est le rôle du noyau ?

Le noyau Unix joue le rôle d'intermédiaire 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).

Bien sûr c'est un petit peu plus compliqué que ça, 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 plus de matériels (c'est-à-dire qu'ils gèrent 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 stupides des versions précédentes. Beaucoup de gens 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 essayer "`make config`", ce qui vous permet de voir tous les matériels supportés par le 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` fourni avec le 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, 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 et les retirer du noyau à n'importe quel moment. En raison de cette souplesse, c'est devenu la méthode préférée pour coder certaines fonctionnalités 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** fait presque 16 Mo pour la version 2.2.9. Vous voulez généralement le garder dans un coin après l'avoir décompressé. Décompressé et compilé avec une configuration moyenne, il occupe 67 Mo de plus.

2.7 Combien de temps ça prend ?

Sur les machines récentes la compilation demande beaucoup beaucoup moins de temps que sur les anciennes. Un AMD K6-2/300 avec un disque rapide peut compiler un noyau 2.2.x en à peu près quatre minutes. Avec de vieux Pentium, 486 ou 386 soyez prêts à patienter, éventuellement des heures, jours...

Si cela vous pose problème et que vous avez accès à une machine rapide, vous pouvez compiler sur celle-ci (fournissez les bons paramètres, assurez-vous que vos utilitaires sont à jour, etc.) et transférer l'image du noyau sur la machine lente.

3 Comment configurer le noyau ?

3.1 Récupérer les sources

Vous pouvez récupérer les sources via ftp depuis `ftp.kernel.org` dans le répertoire `/pub/linux/kernel/vx.y`, où `x.y` est la version (par ex. 2.2). Comme expliqué plus haut les versions se terminant par un chiffre impair sont les versions de développement et peuvent être instables. Le fichier est typiquement `linux-x.y.z.tar.gz`, où `x.y.z` est le numéro de version. Une version avec suffixe en `.bz2` (compressé avec bzip2) est aussi généralement disponible (la compression bzip2 est plus performante donc le transfert sera plus rapide).

Il vaut mieux utiliser `ftp.xx.kernel.org` où `xx` est votre code de pays, par exemple `ftp.fr.kernel.org` pour la France, `ftp.be.kernel.org` pour la Belgique.

3.2 Installer les sources

Faites un `su` (super utilisateur) ou bien loggez-vous en tant que “`root`”, et allez dans le répertoire `/usr/src`. Si vous avez déjà installé les sources du noyau, il y aura déjà 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 tous les 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 xpvf 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 instructions lorsque nécessaire - prenez garde à ce que les liens soient en place et que les fichiers .o soient détruits, etc.

Si votre fichier est en .bz2, il vous faudra utiliser le programme bzip2 (voir http://www.muraroa.demon.co.uk/). Faites:

```
bz2cat linux-x.y.z.tar.bz2 | tar xvf -
```

3.3 Configurer le noyau

Note : il s’agit d’une répétition, clarification de la section correspondante 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 de possibilités autres que make config et il est possible que vous les trouviez plus simples 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 devoir tout relancer.

Avec make menuconfig et make xconfig les options de configuration apparaîtront hiérarchisées.

Vous êtes alors prêt à répondre aux questions, la plupart du temps par “y” ou par “n”. Les pilotes de périphériques acceptent aussi généralement “m”, pour “module”. Le pilote sera compilé non pas directement dans le noyau mais en tant que module chargeable. Une interprétation plus amusante est que le “m” est pour “maybe” (peut-être). Certaines options non critiques ou évidentes ne seront pas décrites ici. Consultez le paragraphe “Autres options de configuration” pour une courte descriptions de quelques options non détaillées ici. Avec make menuconfig vous pouvez utiliser la barre d’espace.

À partir des versions 2.0.x, une option “?” est présente : elle permet d’obtenir une aide pour l’option désirée qui correspond sûrement aux informations les plus à jour. Voici une liste décrivant brièvement les options les plus importantes avec leur place dans la hiérarchie (NdT : et la traduction de l’intitulé).

3.3.1 Kernel math emulation (Precessor type and features)

(Émulation 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 vous en avez un et que vous répondez “y”, ce n’est pas grave : le coprocesseur sera utilisé et l’émulation ignorée. Pour toute machine pas trop vieille, la réponse sera non.

3.3.2 Enhanced (MFM/RLL) disk and IDE disk/cdrom support (Block Devices)

(Gestion améliorée 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. Il est supposé "amélioré" ce qui signifie qu'à part avoir un nombre différent de bogues il peut améliorer les performances de votre disque, notamment si vous possédez du matériel récent (comme de l'EIDE).

3.3.3 Networking support (General Setup)

(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) ont besoin de ce support réseau même si votre machine n'y est pas reliée ; 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 System V IPC (General Setup)

L'une des meilleures définition des IPC (communication inter-processus) se trouve dans le glossaire du livre de Perl. Ces mécanismes sont utilisés par certains développeurs Perl et quelques 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.

3.3.5 Processor family (Processor type and features)

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

Cette option sélectionnait 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.

3.3.6 SCSI support

(Gestion SCSI) Si vous avez au moins un périphérique SCSI, répondez "y". On vous demandera alors de plus amples renseignements sur la gestion des CD-ROM, des disques, et quel type de contrôleur SCSI vous possédez. Consultez le SCSI-HOWTO pour plus de détails.

3.3.7 Network device support

(Gestion de périphériques réseau) Si vous avez une carte réseau, ou si vous voulez utiliser SLIP, PPP, ou un adaptateur sur port parallèle, répondez "y". Le script de configuration vous demandera alors quel type de carte vous possédez et quels protocoles vous voulez inclure.

3.3.8 Filesystems

(Systèmes de fichiers) On vous demandera quels types de systèmes de fichiers vous souhaitez gérer :

Standard (minix) - Les distributions récentes ne créent pas de systèmes de fichiers de type minix, et bon nombre de personnes ne s'en servent pas, mais cela peut être malgré tout une bonne idée de le configurer. Quelques programmes de "sauvetage de disques" l'utilisent, et encore aujourd'hui, un bon nombre de disquettes risquent d'avoir ce système de fichiers, car il est assez pratique pour les disquettes.

Second extended - C'est le système de fichiers standard de Linux. Vous en aurez très certainement besoin, donc "y". (NdT : et puis c'est un Français qui l'a fait alors... !)

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

Des systèmes de fichiers originaires d'autres systèmes d'exploitation sont également disponibles.

/proc - Un des systèmes de fichiers les plus puissants (idée effrontément chipée aux Laboratoires Bell, je pense). Ça n'a rien à voir avec vos partitions disques, mais c'est une interface sous forme de système 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 Linux standards en ont besoin.

NFS - Si vous êtes en réseau et que vous voulez utiliser des volumes présents sur d'autres machines, répondez "y".

ISO9660 - Format rencontré sur la plupart des CD-ROM. Si vous possédez un CD-ROM et que vous comptez vous en servir sous Linux, répondez "y".

Mais je ne sais pas de quels systèmes de fichiers 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 côté de "`type`" est le type du système de fichiers. Dans cet exemple, mes partitions `/` et `/usr` sont du type ext2fs, j'utilise `/proc`, et il y a une disquette montée avec le système de fichiers MS-DOS (beuurrrk !).

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

Le support de systèmes de fichiers rarement utilisés et qui ne sont pas absolument nécessaires peut augmenter la taille du noyau. Voir la section sur les modules chargeables pour résoudre ce problème, ainsi que la section sur les pièges.

3.3.9 Character devices

(Périphériques caractère) Vous trouverez ici les pilotes pour gérer votre imprimante parallèle, souris bus, souris PS/2 (beaucoup de portables utilisent une souris constructeur de type PS/2), quelques lecteurs de bande et d'autres périphériques de type "caractère". Répondez en conséquence.

Note : "gpm" est un programme qui vous permet d'utiliser la souris à l'extérieur de X Window pour effectuer des opérations de copier-coller entre les consoles virtuelles. C'est assez utile si vous avez une souris série, car cela coexiste bien avec X Window, mais vous devrez faire quelques trucs pour que cela fonctionne avec les autres.

3.3.10 Sound

(Son) Si vous tenez à entendre `biff` aboyer, répondez "y", puis donnez au programme de configuration tout ce qu'il veut savoir sur votre carte son. À propos : lorsqu'il vous demande si vous voulez installer la version

entière du pilote, vous pouvez répondre '`n`' et donc économiser de la mémoire en insérant uniquement les caractéristiques dont vous avez besoin.

Si vous vous intéressez de près au support des cartes son, jetez un oeil sur les pilotes libres à `http://www.linux.org.uk/OSS/` et au Open Sound System (commercial) à `http://www.opensound.com/` .

3.3.11 Autres options de configuration

Toutes les options de configuration ne sont pas détaillées dans ce document car elles changent trop souvent ou bien sont vraiment évidentes (par exemple, le support de la carte 3Com 3C509 permet la compilation du pilote de cette carte). L'aide en ligne fournit une liste assez complète de ces options (plus une manière de les insérer dans le script `Configure`) mise à jour par Axel Boldt (`boldt@math.ucsb.edu`). La liste est dans le fichier `Documentation/Configure.help` des sources du noyau depuis la version 2.0.

3.3.12 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)

Quand la configuration est terminée, 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 pourrez également changer les options avec la commande "`rdev`" quand le noyau sera prêt. Si vous ne comprenez rien à ce fichier, passez à la suite.

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 (peut-être) "`make clean`". "`make dep`" assure que toutes les dépendances, comme les fichiers d'en-têtes (include) sont en place, et 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'autres fichiers que les anciennes versions laissent trainer. *N'oubliez pas* cette étape avant de recompiler un noyau.

4.2 La compilation

Après avoir cherché les dépendances et nettoyé l'arborescence, vous pouvez lancer la compilation par "`make bzImage`" ou "`make bzdisk`" (c'est cette partie qui prend du temps). "`make bzImage`" va compiler le noyau, et créer le fichier "`bzImage`" dans le répertoire `arch/i386/boot` qui correspond au nouveau noyau compressé. "`make bzdisk`" fait la même chose, mais copie le noyau sur une disquette que vous aurez insérée dans le lecteur "A:".

"`bzdisk`" est assez commode pour tester de nouveaux noyaux ; s'il ne fonctionne pas, enlevez juste la disquette et relancez la machine avec votre ancien noyau. C'est également une méthode pratique si vous avez détruit accidentellement votre noyau (ou quelque chose de ce genre). Vous pouvez également vous en servir pour installer de nouveaux systèmes, en copiant directement le contenu d'un disque sur un autre ("tout ça ? ouah !").

Tous les noyaux assez récents sont compressés, comme l'indique le "`bz`" au début des noms. Un noyau compressé est automatiquement décompacté lors de l'exécution.

Pour les noyaux plus anciens, `bzImage` n'existait pas et on utilisait `zImage`. Cette possibilité est toujours disponible mais vue la taille des noyaux récents il est à peu près obligatoire d'utiliser `bzImage` car l'ancienne méthode ne peut gérer que des noyaux de taille limitée.

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. Il efface également votre fichier de configuration donc il faudra éventuellement en faire une copie avant (`.config`).

"`make oldconfig`" essaiera de configurer le noyau à partir d'un vieux fichier de configation, il choisira toutes les options de "`make config`" pour vous. Si c'est la première fois que vous compilez un noyau ou si vous n'avez pas un ancien fichier de configuration n'utilisez pas ceci, vous devrez sans doute changer la configuration par défaut.

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 semble fonctionner comme vous le voulez, il est grand temps de l'installer. La plupart des gens utilisent LILO (LInux LOader). "`make zlilo`" installera votre noyau, lancera LILO, et laissera votre machine prête à rebooter, mais SEULEMENT si lilo est configuré de la manière suivante : le noyau s'appelle `/vmlinuz`, lilo se trouve dans `/sbin`, et votre fichier de configuration pour LILO (`/etc/lilo.conf`) correspond à ces données.

Sinon, vous devrez utiliser LILO directement. C'est un produit assez facile à installer, mais dont le fichier de configuration peut être déroutant. Regardez le fichier de configuration (`/etc/lilo/config` pour les versions les plus anciennes, `/etc/lilo.conf` pour les récentes), et regardez son contenu. Le fichier de configuration doit ressembler à ceci :

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

La ligne "`image =`" sert à indiquer le nom du noyau utilisé. Beaucoup de gens utilisent `/vmlinuz`. "`label`" servira à indiquer à LILO sur quel noyau ou sur quel système d'exploitation vous allez booter (nom logique), et "`root`" est la partition racine de ce système d'expoitation. Faites une sauvegarde de votre vieux noyau (quel que soit son nom), et copiez le nouveau noyau (bzImage) à sa place : "`cp bzImage /vmlinuz`" si vous utilisez "`/vmlinuz`", puis relancez lilo. Sur les systèmes récents, vous pouvez simplement 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 méthode de secours quand le noyau plante), 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`. Si elle n'y est pas, vous devrez 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 label de l'image du noyau de sauvegarde (au cas où 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 du noyau ("`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 n'est pas 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 lire, et on ne sait pas trop si ça a marché. Vous pouvez utiliser l'option `-s` de `patch` 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 quelque chose...!"). Pour vérifier que tout s'est passé sans encombre, allez dans `/usr/src/linux` et cherchez les fichiers ayant pour extension `.rej`. Quelques versions de patch (vieilles versions) utilisent `#` pour les fichiers rejetés. 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.

La commande `patch` possède quelques options. 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 présentait était lorsqu'un patch modifiait le fichier "`config.in`" et que vous aviez 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 ("`no fails`"), 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 peut-être installé 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 de détruire votre source et il vous faudra récupérer un source complet (vous auriez peut-être dû commencer par là).

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 du fichier `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 plus d'un demi 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 en utilisant 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 d'autres patches (je les appellerai "non-standards") que ceux distribués par 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 anciennes 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, vous pouvez revenir à l'ancienne version, ou essayer de modifier le patch pour le faire fonctionner, ou encore attendre qu'un nouveau patch arrive.

Vous entendrez probablement parler de ces patches non standards. J'utilisais 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). Les pilotes de périphériques étant de plus développés sous la forme de modules chargeables, le nombre de patches "non standards" décroît.

6 Paquetages supplémentaires

Votre noyau 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 de nombreuses caractéristiques. Elle peut changer les polices de caractères, remapper le clavier, 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). Il est disponible sur les mêmes sites qui proposent les sources du noyau.

6.2 util-linux

Rik Faith (`faith@cs.unc.edu`) a rassemblé une grande quantité d'outils Linux dans le paquetage util-linux. Ils est désormais maintenu par Andries Brouwer (`util-linux@math.uio.no`). Disponible via ftp anonyme sur sunsite.unc.edu dans `/pub/Linux/system/misc`, il contient des programmes tels que `setterm`, `rdev`, et `ctrlaltdel`, qui concernent le 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

Comme beaucoup de paquetages, c'était au départ un patch pour le noyau et un ensemble de programmes. Le patch a été intégré dans le noyau standard 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 consoles 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 plantage de votre système, des problèmes d'entrées-sorties étranges, une chute des performances, des reboot aléatoires... Vérifiez que vous 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 986DX6/440, c'est que vous avez configuré un tas de choses (pilotes de périphériques, systèmes de fichiers) dont vous n'avez pas besoin. Si vous ne les utilisez pas, ne les configurez pas car cela prend beaucoup de place en mémoire. 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 ces vieux disques Fujitsu Eagles qui font le bruit d'un avion lors de son atterrissage 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`”.

7.3 Le port parallèle ne fonctionne pas/mon imprimante ne fonctionne pas

Les options de configuration pour les PC sont : premièrement, dans la catégorie "General Setup" sélectionnez "Parallel port support" et "PC-style hardware". Puis dans "Character devices", sélectionnez "Parallel printer support".

Il y a ensuite le problème des noms de périphérique des imprimantes qui ont changé dans Linux 2.2. Si vous aviez une imprimante `lp1` avec votre noyau précédent, elle s'appelle probablement `lp0` maintenant. Utilisez "`dmesg`" ou cherchez dans les logs dans `/var/log` pour le vérifier.

7.4 Le noyau ne compile pas

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

Dans des cas relativement rares, gcc peut échouer en raison de problèmes de matériel. Le message d'erreur ressemble à un truc assez mystérieux "xxx exited with signal 15". Je n'en n'aurais probablement pas parlé si cela ne m'était arrivé une fois. J'avais un cache mémoire défectueux et le compilateur fonctionnait de manière plutôt aléatoire. Essayez dans un premier temps de reinstaller gcc si vous avez des problèmes. Si votre noyau compile très bien avec les caches externes vidés ou une mémoire réduite, alors vous pourrez commencer à soupçonner votre matériel.

Certaines personnes ont tendance à ne pas aimer que je mette en doute leur matériel. Je n'invente rien. Il existe une FAQ dédiée à ce sujet : `http://www.bitwizard.nl/sig11/` (NdT : traduite en français à `http://www.linux-france.org/article/sig11-fr/sig11-fr.html`).

7.5 La nouvelle version du noyau ne boote plus !

Soit LILO ne fonctionne pas, soit il n'est pas configuré correctement. Une fois, un problème dans le fichier de configuration m'a posé pas mal de soucis : j'avais mis "`boot = /dev/hda1`" à la place de "`boot = /dev/hda`" (ce genre d'erreurs n'est pas facile à trouver, mais une fois que vous avez un fichier de configuration qui fonctionne, il n'y a pas de raison d'y toucher).

7.6 Vous avez oublié de lancer LILO, ou le 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 elle est montée.

Dans cet exemple,la racine `/` est `/dev/hda1`, la partition qui supporte `/usr/src/linux` est `/dev/hda3`, normalement montée sur `/usr`. Toutes les deux ont un système de fichiers de type ext2fs. L'image du noyau se trouve dans `/usr/src/linux/arch/i386/boot/` et elle s'appelle `bzImage`.

L'idée est que s'il existe un noyau `bzImage` qui fonctionne il est possible de l'utiliser pour la nouvelle disquette. Une autre possibilité qui peut être meilleure ou pas est présentée après cet exemple (cela dépend de la façon dont vous avez planté votre système).

Commencez par booter à partir d'une disquette d'installation (boot/root) ou d'une disquette de secours et 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=bzImage of=/dev/fd0
rdev /dev/fd0 /dev/hda1
```

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

```
cd /
umount /mnt
```

Maintenant, vous devriez être capable de rebooter votre système normalement à partir de cette disquette. N’oubliez pas de lancer lilo (ou ce que vous aviez oublié) avant de rebooter !

Comme mentionné ci-dessus, il y a une autre manière très pratique. S’il se trouve que vous avez un noyau opérationnel 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 comme ci-dessus en changeant `/dev/hda3` en `/dev/hda1` (la partition `/`), `/mnt/src/linux` en `/mnt`, et `if=bzImage` 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 problèmes. Consultez le mini-Howto LILO ou la documentation.

7.7 Il me dit “warning: bdflush not running”

Cela peut être un problème assez grave. Avec les noyaux ayant une version supérieure à 1.0 (aux alentours du 20 avril 1994), le programme “`update`” qui vide périodiquement les tampons disque a été remplacé par “`bdflush`”. 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 nouveau noyau devrait ensuite fonctionner correctement.

7.8 Je n’arrive pas à faire marcher mon CD-ROM IDE/ATAPI

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 problèmes potentiels.

Si votre CD-ROM est le seul disque d’une interface IDE particulière il doit être configuré en “maître (master)” ou “seul (single)”. C’est l’erreur la plus fréquemment rencontrée.

Creative Labs (par exemple) a mis des interfaces IDE sur ses cartes sons. Toutefois, cela pose un problème pour les gens qui ont déjà deux interfaces IDE sur leur carte mère (IRQ15 généralement). Une pratique commune est de faire de l’interface soundblaster un troisième port IDE (IRQ11 je pense).

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 possibilités.

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 (vous économisez un IRQ) et connectez le disque sur votre seconde interface.

Si vous n'avez pas une seconde interface, mettez interface IDE (pas la partie son) de la carte son sur l'IRQ 15. Cela devrait fonctionner.

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

Récupérez des versions récentes du progamme `route` et de tous les autres programmes manipulant les routes : `/usr/include/linux/route.h` (qui est en fait un fichier dans `/usr/src/linux`) a changé.

7.10 Le Firewall ne fonctionne pas dans la version 1.2.0

Passez 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 mais `[..]/arch/i386/boot/bzImage`.

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`. Il faudra peut-être aussi ajouter une entrée `terminfo`.

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

Le source du noyau contient un certain nombre de fichiers d'en-têtes (les fichiers se terminant par `.h`) qui se trouvent dans le répertoire `/usr/include`. Ils sont référencés ainsi (où `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 la racine des sources du noyau (`/usr/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 à voir ses fichiers par défaut, et que vous désarchiviez les sources du noyau sans l'option `p` (conserve le mode), les utilisateurs ne pourront pas utiliser le compilateur C. Vous pouvez alors utiliser la commande `chmod` pour résoudre le problème mais il est probablement plus facile de réinstaller les fichiers include. Vous pouvez procéder de la même manière que lors de l'installation des sources au début, en ajoutant un argument pour n'extraire que les includes :

```
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 Augmenter les limites

Ces quelques *exemples* de commandes 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.x, 2.2.x

Les noyaux 2.0.x et 2.2.x ont introduit pas mal de modifications dans l’installation du noyau. Le fichier `Documentation/Changes` situé dans l’arborescence des sources 2.0.x contient des informations que vous devez connaître lorsque vous effectuez la mise à jour de votre système. Il est fort probable que vous deviez mettre à jour plusieurs paquetages tels que gcc, libc et SysVInit ainsi que quelques fichiers systèmes. Mais ne paniquez pas !

9 Modules

Les modules chargeables du noyau peuvent faire économiser de la mémoire et faciliter la configuration. L’utilisation des modules s’est accrue jusqu’à inclure les systèmes de fichiers, les pilotes de cartes ethernet, de lecteurs de cartouches, d’imprimantes, etc.

9.1 Installer les utilitaires pour gérer les modules

Ces utilitaires sont disponibles sur n’importe quel site où vous avez récupéré les sources du noyau : `modutils-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 modutils-x.y.z.tar.gz`’, `cd` dans le répertoire créé (`modutils-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`. Vous pouvez les tester avec le pilote 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 exemple est appelé `drv_hello.o`, donc, pour l’insérer, `insmod drv_hello.o`’. Pour voir les modules que le noyau utilise actuellement, 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 (4ko), et aucun autre module du noyau ne dépend actuellement de lui. Pour le retirer, lancez ‘`rmmod drv_hello`’. Notez que `rmmod` prend comme argument *le nom du module,* (celui donné par `lsmod`) pas un nom de fichier. 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, presque tout est disponible sous forme de 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 compiler tous les modules que vous n'avez pas spécifiés dans le noyau et installer les liens dans `/usr/src/linux/modules`. Vous pouvez alors les utiliser depuis ce répertoire ou faire '`make modules_install`' qui les installera dans `/lib/modules/x.y.z`, ou `x.y.z` est le numéro de version du noyau.

Cela peut être particulièrement utile pour les systèmes de fichiers. Vous n'utilisez peut-être pas fréquemment les systèmes de fichiers minix ou msdos. Par exemple, si je rencontre une disquette msdos (beuh), je devrai faire un `insmod /usr/src/linux/modules/msdos.o`, puis `rmmod msdos` lorsque j'aurai terminé, Cette façon de procéder économise près de 50 ko de RAM dans l'utilisation du noyau. Remarque sur le système de fichiers minix : il est très fortement conseillé de *toujours* le configurer directement dans le noyau pour l'utilisation en disques de secours.

10 Trucs et astuces

10.1 Rediriger la sortie standard des commandes make ou patch

Si vous voulez voir ce que font les commandes "`make`" ou "`patch`", vous pouvez rediriger les messages. Regardez d'abord 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).
```

10.2 Installations diverses du noyau

En plus des disquettes, d'autres méthodes existent pour tester un nouveau noyau sans détruire les précédents. À 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/bzImage
label = nouveau_noyau
```

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. À ce moment, tapez '`nouveau_noyau`' pour qu'il boote sur le nouveau noyau.

Si vous désirez conserver plusieurs sources du noyau (cela peut prendre *beaucoup* de place ; faites attention), la manière la plus simple est de 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.

10.3 Mises à jour des noyaux

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

11 D'autres HOWTO qui peuvent être utiles

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

12 Divers

12.1 Auteur

L'auteur du Kernel-HOWTO Linux est Brian Ward (`bri@cs.uchicago.edu`). S'il vous plaît, envoyez-moi des commentaires, corrections, rajouts. Les corrections sont en particulier très importantes.

Vous pouvez jeter un coup d'oeil à ma page web *ici* `http://www.math.psu.edu/bri/` ou *là* `http://blah.math.tu-graz.ac.at/~bri/` .

Bien que j'essaye d'être attentif à mon courrier, rappelez-vous que j'en ai *beaucoup* tous les jours, donc ça peut prendre un petit peu de temps pour vous répondre. Si le mail est une question, essayez d'être le plus clair et détaillé possible. Si vous m'écrivez à propos de matériel qui ne fonctionne pas (ou quelques chose du genre), j'ai besoin de savoir quelle est votre configuration matérielle. Si vous reportez une erreur, ne vous limitez pas à "J'ai essayé ceci et ça n'a pas marché", j'ai besoin de savoir quelle était l'erreur, et quelles versions du noyau, de gcc et de la libc vous utilisez. Me donner le nom de la distribution n'est pas suffisant. Ne craignez pas de poser des questions simples, si vous ne demandez pas vous n'aurez pas de réponse ! Merci à tous ceux qui m'ont envoyé des commentaires.

Si votre question ne concerne pas le noyau ou est écrite dans un langage que je ne comprends pas, il se peut que je ne réponde pas.

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

J'ai beaucoup de mail sur des problèmes et questions de matériel. Ce n'est pas un problème mais merci de noter que que je ne suis pas familier avec tout le matériel du monde. J'utilise des processeurs AMD, des controlleurs SCSI Adaptec et Symbios et des disques SCSI IBM.

La version 0.1 de ce document a été écrite le 3 octobre 1994. Il est disponible aux formats SGML, PostScript, TeX, roff et texte.

12.2 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 débogage/sauvetage lors de crash seraient les bienvenues.

12.3 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 Makarévitch) : mrproper, tar -p, et plein d'autres choses
- `boldt@math.ucsb.edu` (Axel Boldt) : a rassemblé les descriptions des options de configuration du noyau et m'a fourni la liste
- `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@ab11.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 "compétition du pain en tranches"

- `rms@gnu.ai.mit.edu` (Richard Stallman) : notice de distribution/concept de documentation libre
- `dak@Pool.Informatik.RWTH-Aachen.DE` (David Kastrup) : NFS
- `esr@snark.thyrsus.com` (Eric Raymond) : quelques trucs et astuces

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

12.4 Copyright, traduction, licence et autres

Cette version française peut être trouvée sur ftp.lip6.fr dans le répertoire /pub/linux/french/HOWTO.

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 des données (vous pourriez aussi m'envoyer un exemplaire de ce que vous faites). Idem pour les versions traduites, contactez le traducteur. Il est conseillé aux traducteurs de contacter l'auteur avant de commencer la traduction. La version imprimée est bien plus belle. Recyclez.