

## 0.1 Thème I (noyau 2.6.x) : Compilation facile du noyau

Dans ce didacticiel, je vais expliquer comment installer facilement un noyau 2.6.X. La plus part des cas de figures courants vont être abordés. C'est à dire que je prends en compte, les personnes dont la distribution fourni déjà un noyau 2.6.x, pour eux il s'agit d'une simple mise à jour. Je traite également le cas des personnes utilisant une distribution avec noyau 2.4.x mais dont l'ensemble des logiciels est près pour le 2.6.x.

### 0.1.1 1. Introduction

#### 0.1.2 1.1 Les nouveautés du 2.6.x

Voici un petit récapitulatif des avancées technologiques du noyau Linux 2.6 :

- Un nouvel ordonnanceur de tâches, nommé O(1) a fait son apparition. Celui-ci tient beaucoup mieux la charge quand de nombreuses tâches concurrentes s'exécutent, et privilégie une répartition équitable du temps de calcul entre celles-ci.
- Le noyau 2.6 est capable de gérer plus de 4 Go de mémoire physique sur des machines x86 32 bits.
- Ce noyau apporte NPTL, une bibliothèque optimisée pour la gestion des threads POSIX et Futexes des sémaphores optimisées pour les processus.
- L'interactivité du nouveau noyau a été largement améliorée. Ainsi, des modifications visant à diminuer le temps d'exécution des appels systèmes (patches low-latency et preempt) ont été intégrés.
- Simplification de devfs en udev.
- La granularité de l'horloge est passée de 10 ms à 1 ms.
- L'architecture ALSA, qui fournit un système avancé de gestion des cartes sons, a été intégrée au noyau, en lieu et place d'OSS.
- Concernant les systèmes de fichiers, de nombreuses optimisations sont disponibles pour ext2/ext3 : EA, ACL, répertoires indexés et allocateur Orlov.
- Côté périphériques, plus besoin d'émuler un graveur IDE en SCSI pour graver. Notez également le support amélioré de l'USB 2 et de l'ACPI.
- Une dernière amélioration très importante est l'incorporation d'un ordonnanceur intelligent des accès aux disques durs. Celui-ci limite très largement les déplacements de la tête de lecture du disque, et apporte des débits élevés en cas d'accès concurrents.

#### 0.1.3 1.2 Pré-requis

Pour installer le noyau 2.6.x, assurez-vous d'avoir les paquets suivants :

- la librairie ncurses-5, certaines distributions l'appellent libncurses5 et libncurses5-dev (ou libncurses5-devel)
- l'utilitaire bzip2
- l'utilitaire gzip

vérifiez également les versions des logiciels suivants :

- gcc 2.95.3 (commande : gcc -version)
- Gnu make 3.78 (commande : make -version)
- binutils 2.12 (commande : ld -v)
- util-linux 2.10 (commande : fdformat -version)
- module-init-tools 0.9.10 (commande : depmod -V)

- e2fsprogs 1.29 (commande : tune2fs )
- jfsutils 1.1.3 (commande : fsck.jfs -V)
- reiserfsprogs 3.6.3 (commande : reiserfsck -V 2>&1|grep reiserfsprogs)
- xfsprogs 2.1.0 (commande : xfs\_db -V)
- pcmcia-cs 3.1.21 (commande : cardmgr -V)
- quota-tools 3.09 (commande : quota -V)
- PPP 2.4.0 (commande : pppd -version)
- isdn4k-utils 3.1pre1 (commande : isdnctrl 2>&1|grep version)
- nfs-utils 1.0.5 (commande : showmount -version)
- procps 3.1.13 (commande : ps -version)
- oprofile 0.5.3 (commande : oprofiled -version)

## **0.1.4 2. Installer les sources**

### **0.1.5 2.1 Avec les paquetages de sa distribution**

De nombreuses distributions comme Mandriva, Fedora, Debian et Slackware peuvent s'installer d'office avec un noyau 2.6.x déjà patché. Il est tout à fait légitime de vouloir mettre à jour ce noyau avec un noyau provenant de kernel.org<sup>1</sup>, mais cela vous ferait perdre tous les patches déjà inclus. C'est pourquoi je vous conseille, pour ces distributions, de simplement recompiler votre noyau, en rajoutant les options qui vous poussent à compiler.

#### **Pour les utilisateurs de Mandriva**

il suffit de taper :

```
# urpmi kernel-headers kernel-source
```

#### **Pour les utilisateurs de Fedora**

il suffit de taper :

```
# yum install kernel-source
```

#### **Pour les utilisateurs de Debian**

Deux cas possibles, votre distribution est installée d'office avec le noyau 2.6, tapez :

```
# apt-get install kernel-headers-$(uname -r) kernel-source-$(uname -r)
```

Pour tous les autres cas, passez au 2.2

#### **Pour les utilisateurs de Slackware**

Deux cas possibles, votre distribution est installée d'office avec le noyau 2.6, il suffit d'installer les paquetages `k/kernel-source-2.6.x.tgz` (et `k/kernel-headers-2.6.x.tgz` s'il y'en a) par :

```
# installpkg /où_est/kernel-source-2.6.x.tgz /où_est/kernel-headers-2.6.x.tgz
```

Pour tous les autres cas, passez au 2.2

---

<sup>1</sup><http://www.kernel.org/>

### 0.1.6 2.2 Avec les sources officielles

D'autres distributions, sont prêtes pour le 2.6. Pourtant, elles continuent à utiliser le 2.4 par défaut. Pour ces distributions, téléchargez les sources du noyau le plus récent sur [kernel.org](http://kernel.org)<sup>2</sup>. Dans la suite, je vais supposer que vous avez téléchargé le fichier `linux-2.6.12.4.tar.bz2`. Avant de les installer je vous propose d'en vérifier l'intégrité, grâce à l'outil `gpg`. Pour le noyau `linux-2.6.12.4.tar.bz2` téléchargez également le fichier `linux-2.6.12.4.tar.bz2.sign` qui se trouve dans le même dossier. Copiez les deux Dans `/usr/src/`. Quand vous êtes prêt, téléchargez la clé publique de "Linux Kernel Archives" par :

```
# gpg --keyserver wwwkeys.pgp.net --recv-keys 0x517D0F0E
```

Un répertoire `~/gnupg` avec un fichier `options` sera créé. Si c'est la première fois, re-exécutez la même commande. Vérifiez maintenant l'authenticité de cette clé grâce à l'option `fingerprint` :

```
# gpg --fingerprint
```

Comparez le résultat avec le site principal du noyau Linux à cette adresse<sup>3</sup>. Si tout vous semble correct, passez à la vérification des sources :

```
# cd /usr/src/  
# gpg --verify linux-2.6.12.4.tar.bz2.sign linux-2.6.12.4.tar.bz2
```

Qui devrait vous retourner quelque chose comme :

```
gpg : Signature made Thu 12 May 2005 01 :13 :10 AM CEST using DSA key ID  
517D0F0E  
gpg : Good signature from "Linux Kernel Archives Verification Key <ftpadm@kernel.org>"  
gpg : WARNING : This key is not certified with a trusted signature !  
gpg : There is no indication that the signature belongs to the owner.  
Primary key fingerprint : C75D C40A 11D7 AF88 9981 ED5B C86B A06A 517D  
0F0E
```

Il ne vous reste plus qu'à installer les sources comme suite :

```
# cd /usr/src  
# tar xjvf linux-2.6.12.4.tar.bz2  
# rm linux  
# ln -s linux-2.6.12.4 linux  
# cd linux  
# make mrproper
```

### 0.1.7 3. Patcher son noyau :

Cette étape, concerne les personnes qui souhaitent rajouter un patch à leur noyau, si vous n'avez pas de patch à appliquer, passer au 4. Sinon, commencez par copier le patch dans le répertoire des sources du noyau : `cp /où_est/lePatch.gz /usr/src/linux/` Si vous avez téléchargé ce patch sur [kernel.org](http://kernel.org)<sup>4</sup>, vous pouvez en vérifier l'authenticité exactement comme pour les sources du noyau, ci-dessus. Quand vous êtes prêt appliquez-le ainsi :

<sup>2</sup><http://www.kernel.org/>

<sup>3</sup><http://www.kernel.org/signature.html>

<sup>4</sup><http://www.kernel.org/>

```
cd /usr/src/linux/
gunzip lePatch.gz
patch -p1 < lePatch.patch
```

Pour annuler ce patch, il suffira de rappeler patch. Ceci suppose bien-sûr que le patch soit compressé avec gzip.

### 0.1.8 4. Le ./configure

Rappelons que cette étape a pour but, la génération du fichier de configuration, qui indiquera au compilateur les parties à inclure dans l'exécutable. Heureusement pour nous, de nombreuses distributions sont livrées avec le fichier de config du noyau 2.6.x qu'elles utilisent. Ce fichier est souvent localisé dans /boot/, il s'appelle /boot/config-ide-2.6.xx ou /boot/config-2.6.xx, peu importe, copiez-le, dans le dossier des sources du noyau : `cp /boot/config-ide-2.6.xx /usr/src/linux/.config` Si vous n'avez pas de fichier de config pour le noyau 2.6, je fournis un fichier prêt à l'emploi ICI<sup>5</sup>. Ce fichier est adapté à un noyau 2.6.12.4. Vous ne pouvez l'utiliser qu'avec un noyau de version au moins égale 2.6.12.4 : `cp /où/est/config-trust-2.6 /usr/src/linux/.config` Ceux qui ont l'expérience des commandes Unix, ont naturellement compris que le fichier de configuration du noyau n'est pas config.h mais .config, peu importe, ils jouent le même rôle. C'est à dire que la modification de ce fichier n'influe en rien sur l'exécutable déjà présent sur votre système. Pour changer de noyau, il faut le re-compiler et le re-installer exactement comme avec les autres logiciels. A ce stade, si vous décidez de lancer la compilation, vous obtiendrez exactement l'exécutable que vous avez déjà sur votre système ou exactement le système que mon fichier de configuration génère par défaut. Pour le customiser lisez le point 5. sinon, passez au 6.

### 0.1.9 5. Les options les plus courantes

Pour rajouter une option au fichier de configuration du noyau, lancez l'utilitaire **menuconfig**, par :

```
cd /usr/src/linux/
make menuconfig
```

C'est une interface ncurses, rudimentaire, mais qui conviendra pour ce qu'on souhaite faire. Elle s'organise en menus, vous pouvez naviguer grâce aux touches directionnelles. Pour valider une action, positionnez-vous sur <Select> et pressez la touche **[enter]**, pour changer une option (**[N]** pour non, **[M]** pour module, **[\*]** pour yes) utilisez la touche **[espace]**.

### 0.1.10 5.1 Customiser le nom du noyau :

Pour modifier le nom de votre noyau, il vous suffit d'éditer le fichier /usr/src/linux/Makefile, qui ressemble à ceci :

```
VERSION = 2
PATCHLEVEL = 6
SUBLEVEL = 14
EXTRAVERSION = -DBZ
```

<sup>5</sup><http://file.trustonme.net/data/config-trust-2.6>

Votre noyau s'appelle en fait Linux-\$(VERSION).\$(PATCHLEVEL).\$(SUBLEVEL)\$(EXTRAVERSION) (Linux-2.6.14-DBZ chez moi). Si vous souhaitez disposer d'un noyau 5.4.6-BigBoss avant tout le monde, c'est le moment d'en profiter. Pour cela, il suffit de modifier /usr/src/linux/Makefile comme suite :

```
VERSION = 5
PATCHLEVEL = 4
SUBLEVEL = 6
EXTRAVERSION = -BigBoss
```

Je vous conseille, cependant, de ne modifier que la variable \$(EXTRAVERSION). Notez enfin, qu'après compilation, vos modules s'installent dans /lib/modules/\$(VERSION).\$(PATCHLEVEL).\$(SUBLEVEL) (/lib/modules/2.6.14-DBZ chez moi).

### 0.1.11 5.2 Processor Type and features

L'option « *Subarchitecture Type* » permet d'accéder à un sous-menu, pour indiquer le type de machine pour l'architecture du processeur, pour lequel le noyau doit être défini. Pour la plupart des machines basées sur des processeurs x86 ou x86-64 d'AMD, le type d'architecture utilisé est « *PC-Compatible* ». L'option « *Processor family* » vous permet de spécifier le type de processeur sur lequel le noyau fonctionnera. L'option « *Preemptible Kernel* » permet d'autoriser le multitâche préemptif au sein même des appels systèmes, à activer pour les PC Desktop.

### 0.1.12 5.3 Device Drivers

Cette partie sert à gérer les périphériques. C'est notamment dans « *Sound* » que vous pourrez mettre le driver de votre carte son en module. Les drivers ALSA seront choisis de préférence en activant l'option « *Advanced Linux Sound Architecture* ». Cependant, l'interface de programmation des drivers OSS étant très utilisée, on veillera à activer les options de compatibilité « *OSS Mixer API* », « *OSS PCM (digital audio) API* », et « *OSS Sequencer API* ».

### 0.1.13 5.4 File systems

Assurez-vous que le système de fichier utilisé par votre partition « / » est bien compilé en dur (option Yes).

### 0.1.14 6. make

Comme annoncé dans l'introduction, le make sert à générer un exécutable, pour ce faire il suit les indications contenues dans le .config. Rappelons que la compilation est une traduction, au même titre que le passage de l'anglais au français. On passe d'un fichier ascii, contenant du code source C à un fichier binaire (contenant des 0 et des 1) appelé exécutable. Pour la plus part des logiciels, la commande make suffit à lancer la compilation, pour le noyau aussi, alors tapez :

```
# cd /usr/src/linux/
# make
```

### 0.1.15 7. make install

Rien de bien compliqué ici non plus, puisqu'il s'agit d'installer le noyau et les modules. On commence par le noyau :

```
# cd /usr/src/linux/
# cp arch/i386/boot/bzImage /boot/vmlinuz-2.6.12.4
# cp System.map /boot/System.map-2.6.12.4
```

Le 2.6.12.4 c'est parce que j'installe un noyau 2.6.12.4, mais le nom n'a pas d'importance, évitez néanmoins de les appeler vmlinuz ou System.map tout court. A ce stade votre noyau est installé, mais pour être totalement en règle avec la syntaxe Linux, exécutez également les commandes suivantes :

```
# cd /boot
# mv vmlinuz vmlinuz.old
# ln -sf /boot/vmlinuz-2.6.12.4 vmlinuz
# rm System.map
# ln -s System.map-2.6.12.4 System.map
```

Il ne vous reste plus qu'à installer les modules :

```
# cd /usr/src/linux/
# make modules_install
```

Si votre distribution utilise un disque ram initial (initial ramdisk = initrd) pour démarrer, je pense à Mandriva, Fedora et Debian tapez également : `# mkinitrd -o /boot/initrd-2.6.14.4 2.6.14.4` Les « 2.6.14.4 » c'est parce que mon noyau s'appelle ainsi, adaptez à ce qu'il y a chez vous.

### 0.1.16 8. Mise à jour du chargeur d'OS

Maintenant que votre noyau est installé il vous reste à mettre à jour votre chargeur d'OS, il peut s'agir de lilo ou grub.

#### 0.1.17 8.1 Si vous utilisez Lilo

Editez votre fichier `/etc/lilo.conf`, focalisez sur la zone qui démarre Linux (label=linux), chez moi elle ressemble à ce qui suit, il peut y avoir quelques variations chez vous, cela n'a aucune importance :

```
image = /boot/vmlinuz
root = /dev/hda4
label = Linux
vga = 773
read-only
```

Vous devez la dédoubler et modifier chacun des deux blocs :

```
# Ancien noyau :
image = /boot/vmlinuz.old
root = /dev/hda4
label = mylinux-old
```

```
vga = 773
read-only
# Nouveau noyau :
image = /boot/vmlinuz
root = /dev/hda4
label = Linux
vga = 773
read-only
```

Le reste doit rester inchangé. Si vous utilisez une distribution avec `initrd`, votre `/etc/lilo.conf` ressemblera plutôt à ceci :

```
image=/boot/vmlinuz
label=linux
root=/dev/hdc6
initrd=/boot/initrd.img
append="quiet devfs=mount"
vga=788
read-only
```

Vous devez la dédoubler et modifier chacun des deux blocs :

```
# Ancien noyau :
image=/boot/vmlinuz.old
label = mylinux-old
root=/dev/hdc6
initrd=/boot/initrd.img
append="quiet devfs=mount"
vga=788
read-only
# Nouveau noyau
image=/boot/vmlinuz
label=Linux
root=/dev/hdc6
initrd=/boot/initrd-2.6.12.4
append="quiet devfs=mount"
vga=788
read-only
```

Le reste doit demeurer inchangé. Quand vous êtes satisfait, validez vos changements en tapant : `/sbin/lilo -v`

### 0.1.18 8.2 Si vous utilisez Grub

Pour `grub` c'est le fichier `/boot/grub/menu.lst`, qu'il faut éditer, la section `linux` (title `Linux`) ressemble à :

```
title Linux
root (hd0,3)
kernel /boot/vmlinuz root=/dev/hda4 vga=773
```

Vous devez la dédoubler et modifier chacun des deux blocs :

```
# Ancien noyau :
title mylinux-old
root (hd0,3)
kernel /boot/vmlinuz.old root=/dev/hda4 vga=773
# Nouveau noyau
title Linux
root (hd0,3)
kernel /boot/vmlinuz root=/dev/hda4 vga=773
```

Le reste doit rester inchangé. Si vous utilisez une distribution avec `initrd`, votre `/boot/grub/menu.lst` ressemblera plutôt à ceci :

```
title Linux
kernel (hd0,5)/boot/vmlinuz root=/dev/hdc6 quiet devfs=mount
vga=788
initrd (hd0,5)/boot/initrd.img
```

Vous devez la dédoubler et modifier chacun des deux blocs :

```
# Ancien noyau :
title mylinux-old
kernel (hd0,5)/boot/vmlinuz.old root=/dev/hdc6 quiet devfs=mount
vga=788
initrd (hd0,5)/boot/initrd.img
# Nouveau noyau
title Linux
kernel (hd0,5)/boot/vmlinuz root=/dev/hdc6 quiet devfs=mount
vga=788
initrd (hd0,5)/boot/initrd-2.6.12.4
```

### 0.1.19 9. On redémarre

Voilà vous pouvez redémarrer, au boot vous choisirez Linux, en cas de difficultés vous pourrez toujours revenir à l'ancienne version en choisissant `mylinux-old` et recommencer plus consciencieusement.