

Comment recompiler son kernel sous linux Debian

par [Ashgenesis](#)

Date de publication : 19.07.2006

Dernière mise à jour :

Cet article montre comment récupérer le dernier kernel disponible et comment le recompiler à la sauce Debian.

- I - Introduction
- II - Acquérir le dernier noyau disponible
 - II-A - Modification du fichier sources.list
 - II-B - Le fichier preferences
- III - Recompilation de son kernel
 - III-A - Installation des dépendances
 - III-B - Décompression et Configuration
 - III-C - Ajustement de la configuration et compilation
 - III-D - Installation
 - III-E - Résumé des Commandes & Scripts
- IV - Remerciement

I - Introduction

La recompilation de kernel à la sauce Debian, n'est pas une chose très compliquée à faire en respectant certaines conditions que nous verrons un peu plus tard. A travers ce tuto, je vais donc présenter une manière de recompiler son noyau pour Debian. Nous allons donc voir comment acquérir le dernier kernel disponible, ensuite comment le recompiler.

II - Acquérir le dernier noyau disponible

II-A - Modification du fichier sources.list

Les nouveaux noyaux sont d'abord testés en unstable et ne sont donc pas directement accessibles en stable ou testing. Sid étant la version unstable de Debian, il peut être difficile de l'utiliser à cause des nombreuses mises à jour disponibles régulièrement (risque de bug). Elle possède cependant l'avantage d'avoir les dernières versions des logiciels. Les nouveaux noyaux étant d'abord intégrés sous sid (unstable), il n'est pas obligatoire d'y passer pour avoir les derniers noyaux sortis.

En première étape, nous allons voir qu'il faut modifier le fichier sources.list se trouvant dans le répertoire **/etc/apt** pour obtenir le dernier noyau. Ensuite, nous regarderons les préférences de récupération des paquets pour ne prendre que ce qui nous intéresse dans une distribution ou une autre. Et finalement, il ne restera plus qu'à télécharger le dernier noyau.

II-A - Modification du fichier sources.list

En premier lieu, nous allons modifier le fichier sources.list de manière à ce qu'il puisse télécharger les paquets de la distribution SID. Il faut donc lui rajouter les sources unstable.

```
deb http://ftp.fr.debian.org/debian unstable main contrib non-free
deb-src http://ftp.fr.debian.org/debian unstable main contrib non-free
```

Nous obtenons ainsi un fichier sources assez complet comme le montre le listing suivant. Ne pas oublier de passer une ligne à la fin du fichier sans quoi vous aurez un **warning**.

```
# Sources.list by ashgenesis
# L'ordre de ces lignes est important dans le cas de packages ayant la même priorité

# Stable
deb http://security.debian.org/stable/updates main contrib non-free
deb-src http://security.debian.org/stable/updates main contrib non-free
deb http://ftp.fr.debian.org/debian stable main contrib non-free
deb-src http://ftp.fr.debian.org/debian stable main contrib non-free

# Testing
deb http://security.debian.org/etch/updates main contrib non-free
deb-src http://security.debian.org/etch/updates main contrib non-free
deb http://ftp.fr.debian.org/debian etch main contrib non-free
deb-src http://ftp.fr.debian.org/debian etch main contrib non-free

# Unstable
deb http://ftp.fr.debian.org/debian unstable main contrib non-free
deb-src http://ftp.fr.debian.org/debian unstable main contrib non-free

# Mirroir Media http://debian.video.free.fr/
deb ftp://ftp.nerim.net/debian-marillat/sarge main
deb ftp://ftp.nerim.net/debian-marillat/etch main

# Package non officiel
deb http://ftp.debian-unofficial.org/debian sarge main contrib non-free restricted
deb-src http://ftp.debian-unofficial.org/debian sarge main contrib non-free restricted
```

Le fait d'avoir ajouté les sources SID nous permet d'accéder à tous les paquets unstable mais le problème qu'il se pose ici est que l'on obtient tout les paquets de cette distribution alors que l'on en désire qu'une partie. Pour cela, nous allons ajouter un fichier **preferences** qui va définir quel paquet télécharger sur quelle distribution.

II-B - Le fichier preferences

Ce fichier va déterminer dans quel distribution les paquets seront téléchargés en priorité c'est ce que l'on appelle le pinning. Il est placé dans le même répertoire que le fichier sources.list, c'est à dire le répertoire **/etc/apt/**. Les priorités sont représentées par des nombres dont voici leurs significations :

1001 : Le package ne sera jamais remplacé par APT.

1000 : idem, mais APT refusera d'installer le package si une autre version est déjà présente.

990 : Le package ne pourra être remplacé que si une version supérieure est disponible dans la distribution utilisée.

500 : Toute version du package supérieure à celle présente sera installée.

100 : Toute version du package, supérieure ou inférieure, remplacera la version en place.

-1 : On empêche un package (ou une version spécifique) d'être installé.

Dans l'exemple, la distribution testing est préférée aux autres mais toutes les nouvelles sources du noyau linux-source-2.6.* seront pris dans les paquets prévus pour Sid. Ainsi, nous restons en testing tout en ayant les dernières sources du kernel. Il ne reste plus qu'à recompiler son noyau et le tour est joué.

```
Package: *
Pin:release a=testing
Pin-Priority: 550

Package: *
Pin:release a=apt-build
Pin-Priority: 990

Package: *
Pin:release a=stable
Pin-Priority: 500

Package: *
Pin:release a=unstable
Pin-Priority: 33

Package: *
Pin:release a=experimental
Pin-Priority: 15

Package: linux-source-2.6.*
Pin:release a=unstable
Pin-Priority: 550
```

Il suffit, une fois tout cela copié dans **/etc/apt/**, de faire une mise à jour des paquets disponibles et d'installer les sources du dernier noyau (en ce moment 2.6.15)

```
ashgenesis@debian:~$ sudo apt-get install linux-source-2.6.15
```

Cette méthode ne se restreint pas à seulement télécharger les sources du dernier noyau. Elle peut-être aussi utilisée afin d'acquérir la dernière image plutôt que les sources afin de ne pas avoir à recompiler son noyau. Mais ici, le but n'est pas de faire dans la simplicité, nous allons donc recompiler notre kernel.

III - Recompilation de son kernel

Maintenant que nous avons téléchargé le dernier noyau, il faut le décompresser, le configurer, le compiler et passer enfin à l'installation. Mais avant toutes choses, il faut installer certaines dépendances si elle ne le sont pas déjà permettant la configuration du kernel.

III-A - Installation des dépendances

Afin de pouvoir correctement installer et configurer notre nouveau noyau, nous avons besoin de certains packages qu'il suffit donc d'installer de cette manière :

```
ashgenesis@debian:~$ su
password *****
debian:/home/ashgenesis# apt-get install debconf-utils dpkg-dev debhelper
debian:/home/ashgenesis# apt-get install build-essential kernel-package
debian:/home/ashgenesis# apt-get install libncurses5-dev
```

Il est inutile de les installer s'ils le sont déjà et dans ce cas on passe directement à l'étape suivante :)

III-B - Décompression et Configuration

Passons maintenant à l'étape de la décompression :

```
debian:/home/ashgenesis# cd /usr/src
debian:/usr/src# tar -xvjf linux-source-2.6.15.tar.bz2
```

Nous avons donc maintenant tout le nécessaire pour configurer notre noyau, Nous rajoutons un petit lien vers le répertoire créé et on copie notre configuration actuelle afin de ne pas avoir à tout refaire. **ATTENTION**, les quotes entourant `uname -r` sont des quotes inversés (Alt Gr + 7)

```
debian:/usr/src# ln -s linux-source-2.6.15 linux
debian:/usr/src# cp /boot/config-`uname-r` /usr/src/linux/.config
debian:/usr/src# cd linux
```

III-C - Ajustement de la configuration et compilation

La configuration actuelle étant recopiée pour le nouveau noyau, elle ne correspond pas forcément avec les nouvelles options. Les nouvelles options ou celles obsolètes possèdent des valeurs par défaut. Lors de la configuration avec `menuconfig` ou `xconfig`, le fichier `.config` est analysé et seules les valeurs comprises sont retenues les autres sont fixées à leurs valeurs par défaut. Ces nouvelles valeurs n'étant pas forcément optimales pour votre matériel il est possible d'ajuster la configuration pour cela :

```
debian:/usr/src/linux# make oldconfig
```

Ce qui permet de valider et de définir correctement toutes les nouvelles options contenues dans la nouvelle version du kernel.

Maintenant nous pouvons finir la configuration et lancer la compilation directement et tout ça en une seule commande.

```
debian:/usr/src/linux# make-kpkg --append-to-version "-version_perso" --initrd buildpackage --config
```

```
menuconfig
```

La configuration de chaque kernel étant bien spécifique à l'architecture du matériel sur lequel vous compilez votre noyau, je vous laisse choisir les différentes options pour votre matériel. Vous pouvez tout de même avoir une aide sur le forum [debian-fr](#) ou voir une configuration minimale sur le site [andesi](#)

III-D - Installation

La compilation finie, il nous reste à installer notre image personnalisée.

```
debian:/usr/src/linux# cd ..
debian:/usr/src# dpkg -i linux-image-2.6.15-version_perso.deb
```

Je vous déconseille de désinstaller à ce moment là votre ancienne version, essayez d'en garder toujours deux. Ainsi, dans le cas d'un problème vous pourrez toujours revenir à une version fonctionnelle et corriger ainsi les erreurs. Une fois installée, un reboot sera nécessaire pour accéder à la nouvelle version. Après s'être identifié, et pour vérifier, on peut lancer la commande `uname -r` qui nous renverra la version du kernel actuel.

```
ashgenesis@debian:~$ uname -r
2.6.15-version_perso
```

Et vous voila avec le dernier kernel configuré et installé.

III-E - Résumé des Commandes & Scripts

Voici, un résumé des commandes à exécuter pour la recompilation

```
ashgenesis@debian:~$ su
password *****
debian:/home/ashgenesis# apt-get install debconf-utils dpkg-dev debhelper
debian:/home/ashgenesis# apt-get install build-essential kernel-package
debian:/home/ashgenesis# apt-get install libncurses5-dev
debian:/home/ashgenesis# cd /usr/src
debian:/usr/src# tar -xvzf linux-source-2.6.15.tar.bz2
debian:/usr/src# ln -s linux-source-2.6.15 linux
debian:/usr/src# cp /boot/config-#uname -r# /usr/src/linux/.config
debian:/usr/src# cd linux
debian:/usr/src/linux# make-kpkg --append-to-version "-version_perso" --initrd buildpackage --config
menuconfig
debian:/usr/src/linux# cd ..
debian:/usr/src# dpkg -i linux-image-2.6.15-version_perso.deb
```

Je vous mets aussi à disposition un petit script tout simple qui facilite la compilation d'un nouveau kernel

Script de compilation

```
#!/bin/bash
#####
# @author : Ashgenesis
# @version : 0.1
# @contact : ashgenesis@free.fr
# Utilisation
# kernel.sh [nom_noyau] [nom_personnalisation]
# Exemple
# kernel.sh linux-source-2.6.15 seal
#####
echo "Mise à jour de la config actuelle"
apt-get update && apt-get upgrade
cd /usr/src/
echo "Installation des sources"
sleep 3
```

Script de compilation

```
#install les sources
apt-get install $1
sleep 3
echo "Decompression des sources"
tar -xvjf $1.tar.bz2
rm linux
#créé un lien symbolique
ln -s $1 linux
echo "Copie de la config"
cp /boot/config-`uname -r` /usr/src/linux/.config
cd linux
sleep 3
echo "Lancement de la compilation"
make-kpkg --append-to-version "-$2" --initrd buildpackage --config menuconfig
echo "Compilation Terminée !\nInstallation en cours\nPlease wait !!"
```

IV - Remerciement

Je tiens à remercier MattOTop et Ricardo pour leur aide dans la conception de cet article, ainsi que developpez.com pour la publication.