

# Compilation du noyau FreeBSD, application à la mise en place d'un firewall

Jean-Marc LICHTLE \*

30 septembre 2004

## Table des matières

<b>1 Introduction</b>	<b>1</b>
<b>2 Mettons toutes les chances de notre côté</b>	<b>1</b>
<b>3 Première compilation</b>	<b>2</b>
3.1 On remonte les manches . . . . .	2
3.2 Config . . . . .	2
3.3 Make depend . . . . .	2
3.4 Make . . . . .	3
3.5 Make install . . . . .	3
<b>4 Vérification</b>	<b>3</b>
<b>5 Application à la mise en place d'un firewall</b>	<b>3</b>
5.1 Modification du fichier de configuration du noyau . . . . .	3
5.2 Compilation . . . . .	4
5.3 Test du firewall . . . . .	4

### Résumé

Les informations de base permettant d'envisager une recompilation du noyau FreeBSD.

## 1 Introduction

La recompilation du noyau est nécessaire lorsqu'on souhaite modifier les caractéristiques de celui-ci.

## 2 Mettons toutes les chances de notre côté

Vous venez de passer plusieurs heures (jours, semaines) à peaufiner le réglage de votre installation de FreeBSD et craignez, peut-être à juste titre, de tout casser en vous lançant dans une opération hasardeuse ?

Raison de plus pour commencer l'étude de ce dossier par l'examen de la procédure d'urgence qui doit vous permettre, le cas échéant, de relancer FreeBSD sur l'ancien noyau (celui qui fonctionnait avant que vous fassiez cette [censuré] de compilation !

Une compilation se termine, forcément, par l'installation du nouveau noyau. Et l'installation d'un nouveau noyau s'accompagne automatiquement du transfert de l'ancien noyau dans un sous répertoire de sauvegarde. En clair, et pour une version 5.2.1 de FreeBSD, le noyau qui occupait initialement le sous répertoire /boot/kernel, se trouve, après une nouvelle compilation, transféré avec tous ses modules dans /boot/kernel.old.

En cas de tuile vous avez donc toujours la possibilité de reprendre la main et de rebooter sur l'ancien noyau. Pour cela :

\* Ingénieur Arts et Métiers promotion CH73

- Laissez booter la machine jusqu'à l'écran "Welcome to FreeBSD", celui qui est orné d'un petit diable dont la fourche menace d'embrocher le menu.
  - A l'apparition de ce menu tapez le touche 6 pour valider l'option 6 "Escape to loader".
  - A cet endroit vous êtes invités à vous documenter, "Type '?' for a list of command 'help' for more detailed help. (ouais et moi je sers à quoi alors!)
  - Tapez calmement boot /boot/kernel.old/kernel pour lancer l'ancien noyau.
- Quand je dis calmement c'est à cause du clavier qui est un peu passé de l'autre côté de l'atlantique à ce moment. Il en découle que vous obtenez / en tapant!, . et tapant : et, éventuellement, ? en tapant \$.

### 3 Première compilation

Maintenant que nous savons comment nous sortir de l'ornière en cas de problème sur notre nouveau noyau il est temps de passer au stade suivant, compiler un nouveau noyau !

Je suppose que vous avez installé les sources du noyau, sinon c'est le moment de le faire. Appelez donc sysinstall sous compte root et suivez le chemin Configure -> Distributions -> src -> sys. Vous installez ainsi le code source du noyau, le strict mini donc pour envisager une recompilation.

#### 3.1 On remonte les manches

Rendez-vous, sous compte root, dans le sous répertoire /usr/src/sys/i386/conf. L'accès à ce sous répertoire est certe possible sous compte utilisateur mais à partir de maintenant il va falloir disposer des droits d'administrateur qui vont permettre de copier et d'éditer les fichiers.

Ce sous répertoire contient, entre autre, un fichier nommé GENERIC. Il s'agit du fichier de configuration du noyau. Notez, c'est assez curieux, que le nom est en majuscules. Il paraît qu'il s'agit d'une tradition, respectons la donc... La première précaution à prendre est de copier le fichier GENERIC d'origine pour en conserver un exemplaire au cas ou, un jour, tout irait vraiment mal. L'emploi d'un support externe, clef USB ou autre peut être une bonne solution. A ce stade une nouvelle copie, de travail cette fois, est souhaitable, par exemple :

```
# cp GENERIC TEST
```

crée un fichier de configuration nommé TEST.

#### 3.2 Config

La première chose à faire est certainement de faire une compilation "à blanc". Le fichier TEST est la copie du fichier qui a servi à obtenir votre noyau, la compilation d'un nouveau noyau avec ce fichier ne peut donc pas présenter de risque. Par contre cette opération peut servir à nous entraîner.

Lancer :

```
# config TEST
```

crée un nouveau sous répertoire ../compile/TEST et affiche :

```
Kernel build directory is ../compile/TEST
Don't forget to do a ``make depend``
```

Le sous répertoire TEST contient maintenant toute une série de fichiers se terminant en .h (header = entête), un makefile et un fichier config.c.

#### 3.3 Make depend

Suivons le bon conseil donné plus haut et tapons make depend dans le sous répertoire TEST. Cette opération va créer un gros fichier .depend, créer des fichiers .c en correspondance des .h et créer une arborescence partant d'un sous répertoire modules.

### 3.4 Make

Là nous entrons dans le vif du sujet. Tapez make et allez vous préparer un café, une tisane ou allez vous faire une coupe de glace, vous avez un peu de temps devant vous.

### 3.5 Make install

La commande make install renomme /boot/kernel en /boot/kernel.old puis crée un nouveau /boot/kernel avec les fruits de votre compilation.

## 4 Vérification

Vous venez de compiler un noyau copie strict du noyau d'origine. Malin allez vous me dire, en principe rien ne le distingue de l'ancien, comment vérifier même que c'est le nouveau noyau qui tourne ? La réponse est donnée par la commande uname.

```
$ uname -a
```

produit l'affichage de la version du noyau :

```
FreeBSD jml3.rezo 5.2-RELEASE FreeBSD 5.2-RELEASE #0:  
Sun Jan 11 04:21:45 GMT 2004  
root@wvlu.btc.adaptec.com:/usr/obj/usr/src/sys/GENERIC i386
```

On lit aisément la date de compilation, ici celle du noyau d'origine. Il suffit à présent de rebooter et de relancer uname pour vérifier que, désormais, vous faites tourner FreeBSD avec un noyau que vous avez compilé personnellement. Et bien entendu si la machine ne veut pas repartir proprement c'est le moment d'appliquer les conseils du début du document !

## 5 Application à la mise en place d'un firewall

Dans le monde LINUX les noyaux fournis avec les distributions majeures sont, très généralement, compilés avec des paramètres permettant l'emploi comme firewall. Malheureusement ce n'est pas le cas pour FreeBSD qui ne peut servir de firewall que si le noyau est recompilé. A la réflexion le terme "malheureusement" est inapproprié dans la mesure où cette opération ne présente, comme nous l'avons vu plus haut, aucune difficulté majeure et qu'il s'agit simplement d'une technique supplémentaire à apprendre.

### 5.1 Modification du fichier de configuration du noyau

Commencez par faire une copie de travail de votre fichier GENERIC, appelez d'un nom explicite, ce nom se retrouvera en effet dans la sortie de uname. Mettez en oeuvre votre éditeur de texte préféré et ajoutez les lignes suivantes, par exemple à la fin de votre fichier de configuration.

```
options IPFIREWALL  
options IPFIREWALL_DEFAULT_TO_ACCEPT  
options IPFIREWALL_VERBOSE
```

Commentaires concernant ces lignes :

- IPFIREWALL est vraiment la syntaxe mini pour demander au noyau de prendre en charge les fonctions de firewalling c'est à dire d'examiner chaque paquet circulant pour en vérifier la validité en fonction des règles que vous aurez choisies.
- IPFIREWALL\_VERBOSE permet d'inclure le support de logging c'est à dire d'enregistrer le trafic correspondant à des critères que vous aurez choisis.
- IPFIREWALL\_DEFAULT\_TO\_ACCEPT fixe une règle d'acceptation par défaut des paquets. Ce point est très discutable si vous travaillez comme informaticien chez un grand constructeur d'avions ou d'armes, nous accepterons ce compromis ici du moins dans un premier temps, le temps que vous fassiez le tour du sujet et que vous vous familiarisiez avec le sujet.

## 5.2 Compilation

Vous savez faire maintenant, allez-y et rendez-vous après le reboot de votre machine sur son nouveau noyau.

## 5.3 Test du firewall

FreeBSD peut utiliser aussi bien ipf (ipfilter) que ipfw (ipfirewall). C'est cette dernière option que nous utiliserons dans ce qui suit.

Commencer par taper la commande :

```
# ipfw list
```

L'affichage doit ressembler à ceci :

```
65535 allow ip from any to any
```

Ipfw list affiche la liste des règles actives. A ce stade on ne sera pas étonné de trouver une seule règles, de plus en accord avec l'option permissive choisie délibérement lors de la compilation du noyau. Essayez maintenant d'ajouter une règle, par exemple :

```
# ipfw add deny ip from any to any
```

Vous aurez deviné que la règle, très brutale puisqu'elle dit simplement "refuse tout trafic ip venant de n'importe ou pour aller n'importe ou" ne va pas vous laisser beaucoup de latitude pour accéder à Internet, à votre messagerie, au site du Mirabellug etc. En clait votre machine est bloquée, un vrai coffre fort.

Tapez maintenant :

```
# ipfw -f flush
```

et vous voilà revenu à la situation précédente, vous venez d'effacer toutes les règles (dans notre cas une seule), la seule règle conservée est la règle par défaut compilée en dur dans le noyau.

Il est évident que nous n'avons fait ici que soulever un tout petit coin de la carpe. Il reste énormément de choses à découvrir mais ces éléments dépasseraient nettement le cadre de ce document consacré à la compilation du noyau de FreeBSD.