

# Mise en place d'un firewall avec FreeBSD

Jean-Marc LICHTLE

11 septembre 2005

## Table des matières

<b>1</b>	<b>Introduction</b>	<b>2</b>
1.1	Fonction d'un firewall	2
1.2	Les firewall sous FreeBSD	2
1.3	Le réseau test	2
<b>2</b>	<b>Activation d'IPFW2</b>	<b>3</b>
<b>3</b>	<b>Les règles de filtrage</b>	<b>3</b>
3.1	Ouvrir tout grand le firewall	3
3.2	Lire les règles actuelles	4
3.3	Syntaxe des commandes	4
3.3.1	Champ <cde>	4
3.3.2	Champ <num>	5
3.3.3	Champ <action>	5
3.3.4	Champ <proto>	5
3.3.5	Champs <src> et <dest>	5
3.3.6	Champ <port>	6
3.3.7	Champ <sens>	6
3.3.8	Champ <if>	6
3.3.9	Champ <type>	6
3.4	Un exemple de règles de filtrage	6
3.5	Test de l'efficacité du firewall	10
<b>4</b>	<b>Log des datagrammes bloqués</b>	<b>10</b>
<b>5</b>	<b>Automatisation</b>	<b>11</b>
<b>6</b>	<b>Contrôle du débit avec dummynet</b>	<b>11</b>
6.1	Mise en place manuelle	12
6.2	Mise en place automatique	12

**Préambule**

# **1 Introduction**

## **1.1 Fonction d'un firewall**

On se reportera utilement à l'abondante littérature disponible à ce sujet sur Internet ou dans les librairies spécialisées. Je simplifierais ici en disant que le rôle essentiel d'un programme firewall est de trier les paquets de données qui circulent via les interfaces réseau (sens large, y compris localhost) et de décider, selon un jeu de règles mis au point par l'administrateur du devenir de ce paquet, transmission, abandon en silence, abandon avec message à l'expéditeur, enregistrement du passage etc. Bien configuré il doit permettre de rejeter les paquets non désirés, susceptibles d'être le vecteur d'agressions.

## **1.2 Les firewall sous FreeBSD**

FreeBSD offre la possibilité d'utiliser au moins trois firewalls :

- ipfilter (ipf), le produit historique, à l'origine commun aux trois \*BSD et à d'autres UNIX et Unix-like dont LINUX,
- packetfilter (pf), un développement de la branche OpenBSD entre temps porté aussi sur FreeBSD,
- et enfin ipfw, le firewall qui va nous intéresser ici.

J'ai choisi de mettre en oeuvre IPFW, plus précisément IPFW2 qui est la version 2 d'ipfw apparue en 2002. Plusieurs raisons motivent ce choix, parmi lesquelles la simplicité de la mise en oeuvre de IPFW et la possibilité d'employer DUMMYNET, un module additionnel à IPFW et qui permet des traitements intéressants comme la réduction de bande passante pour certains utilisateurs.

## **1.3 Le réseau test**

La suite du document décrit le paramétrage de IPFW sur le PC jml1 du schéma 1. Ce schéma est une partie du réseau que j'ai installé à mon domicile, les machines de Madame et des jeunes raccordés via ethernet ou wifi n'y figurent pas (ne tournent pas sous un système d'exploitation libre ...).

Dans ce réseau jml1 est le noeud du système et prend en charge :

- La connexion ADSL (via PPPoE)
- La translation d'adresse (également PPPoE)
- La connexion du boîtier ADSL externe

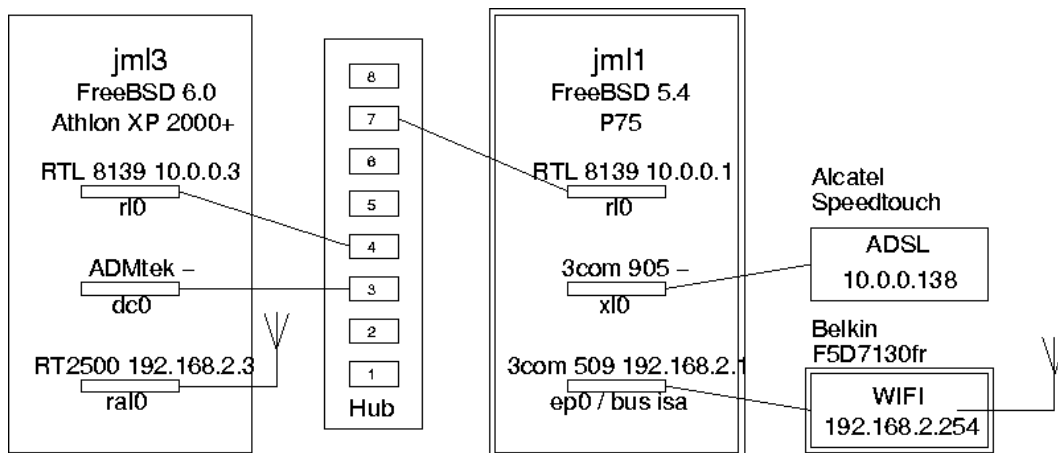


FIG. 1 – Réseau test

Nous allons maintenant, en plus de tout ça, le configurer en firewall. Je précise à cet endroit que pour un réseau domestique la puissance du P75 est très largement suffisante, nul besoin de mettre un matériel moderne à cet endroit.

Concernant la méthode pour configurer l'ADSL ou le WIFI sous FreeBSD je vous invite à vous reporter vers les articles que j'ai rédigé sur ces sujets accessibles, entre autres, sur le site du Mirabellug (<http://www.mirabellug.org>)

## 2 Activation d'IPFW2

Pour activer IPFW il suffit de charger le module correspondant :

```
# kldload ipfw
```

pour que le firewall soit activé immédiatement. ATTENTION : Par défaut ipfw bloque toute communication dès son activation. Ne chargez donc jamais ce module si vous accédez à la machine via telnet ou ssh faute de quoi vous allez vous retrouver immédiatement coupé de votre machine.

## 3 Les règles de filtrage

### 3.1 Ouvrir tout grand le firewall

Les règles de filtrage sont enregistrées dans une liste ordonnée et numérotée. La toute dernière ligne de cette liste, numérotée 65535 contient la règle qui bloque toute communication :

```
65535 deny ip from any to any
```

La toute première chose à faire si on vient de charger IPFW2 et qu'on se retrouve ennuyé du fait de la coupure de toutes les communications est donc de revenir à un état comparable à l'état antérieur en tapant :

## 3.2 Lire les règles actuelles

---

```
# ipfw add allow all from any to any
```

Cette commande conduit le firewall à ajouter (add) une règle disant d'accepter (allow) toutes (all) les trames quelque soit leur source (any) ou leur destination (any). Pour être tout à fait clair avant de charger IPFW votre machine se comportait comme une passoire, maintenant elle est équipée d'un firewall grand ouvert. L'informatique n'y a pas gagné grand chose mais vous venez de passer du stade d'utilisateur à celui d'administrateur de firewall et ça, ça vous classe son homme !

## 3.2 Lire les règles actuelles

Il peut s'avérer utile de "lire" les règles actuellement appliquées par IPFW. Pour cela taper sous compte root :

```
# ipfw list
```

ce qui devrait afficher, si vous avez suivi depuis le début de ce document :

```
00100 allow ip from any to any
65535 deny ip from any to any
```

## 3.3 Syntaxe des commandes

L'exemple ci-dessus donne un premier aperçu de la syntaxe des commandes. D'une façon plus générale les commandes suivent la syntaxe suivante :

```
ipfw <cde><num><action><proto>from<src>to<dest><port><sens>via<if><type>
```

dans laquelle il suffit de donner les valeurs ad-hoc au différents champs pour constituer par empilage de lignes successives le groupe de règles de filtrage.

La première ligne ajoutée plus haut est conforme à cette syntaxe avec, certes, des champs vides.

### 3.3.1 Champ <cde>

Les valeurs les plus courantes sont :

add qui permet d'ajouter une ligne supplémentaire

flush , la commande qui purge toutes les règles actuellement valides et permet de remettre le firewall dans un état connu, en général bloqué.

#### 3.3.2 Champ <num>

Il n'est pas obligatoire de renseigner ce champ. Toutefois il est important de se souvenir que :

- Les règles sont lues et évaluées dans l'ordre déterminé par leur numéro
- A défaut d'indication explicite ipfw numérote les règles par incréments de 100, la première aura le num 00100, la suivante 00200 etc.
- La première règle "pertinente" met fin à l'examen de la liste.
- Deux règles portant le même numéro seront examinées dans l'ordre dans lequel elles ont été mises en place.

#### 3.3.3 Champ <action>

Ce champ peut prendre de très nombreuses valeurs parmi lesquelles :

allow qui "autorise" le paquet,

log qui enregistre le passage du paquet dans un fichier log et qui peut se combiner avec une autre action par exemple "allow log" qui laisse passer mais en enregistrant.

check-state qui demande un contrôle du paquet à l'aune des règles dynamiques en vigueur (voir plus loin).

deny , l'inverse de allow, on refuse le paquet.

A noter que log fait partie d'un tout petit nombre d'actions qui font exception à la règle de la fin de traitement à la première règle pertinente. Une correspondance uniquement sur un log ne suffit pas à clore le traitement, celui-ci continuera à la ligne suivante.

#### 3.3.4 Champ <proto>

Les datagrammes ou paquets correspondent à des protocoles évidemment connus et codifiés :

all tout type de protocole connu

ip idem

tcp "transport control protocol", la base des transferts sur Internet, web, ftp, telnet etc.

udp , un protocole \*\*\*\*

icmp le protocole des "pings"

#### 3.3.5 Champs <src> et <dest>

Les champs source et destination ont une syntaxe identique. Différentes formes courantes :

any qui signifie "tout"

80.10.246.1 qui précise de façon tout à fait univoque l'adresse visée

me qui signifie toute interface de cette machine.

192.168.0.0/16 qui définit une plage d'adresse ip, celles du réseau 192.168.0.0 avec masque de sous réseau 255.255.0.0.

### 3.3.6 Champ <port>

Une adresse IP ne définit pas à elle seule l'endroit ou doit être délivré le message (ou d'où il provient). Par analogie on pourrait dire que l'adresse IP donne l'adresse de l'immeuble, reste à préciser l'étage. C'est là qu'intervient le port. La consultation du contenu du fichier /etc/services vous donnera un aperçu très exhaustif de la gamme des ports, numérotés de 1 à 65535. Exemple : le port 80 est, sauf réglage particulier, le port sur lequel vous devriez pouvoir communiquer avec le serveur web. Dans le même ordre d'idées et dans les ports utilisés couramment le 631 est affecté au dialogue avec CUPS, le programme de gestion des imprimantes.

### 3.3.7 Champ <sens>

Deux valeurs triviales pour ce champ, "in" et "out" qui définissent les sens entrants et sortants des messages.

### 3.3.8 Champ <if>

Une règle peut très bien être spécifique à une seule interface. Dans ce cas "via <if>" permet de le préciser, par exemple "via rl0".

### 3.3.9 Champ <type>

C'est certainement le champ le plus complexe à définir. On y trouve en effet des syntaxes qui peuvent se combiner :

- setup
- keep-state
- limit src-addr 2

## 3.4 Un exemple de règles de filtrage

Pour cet exemple j'avoue que je ne me suis pas vraiment foulé les neurones, j'ai simplement pris la liste qui figure dans le Handboock et adapté celle-ci à mon usage ce qui donne, après traduction des commentaires :

```
##### Début du fichier des paramètres de IPFW #####
# Vider toute la liste avant d'en installer une nouvelle.
ipfw -q -f flush

# Variables communes à beaucoup de règles, spécifiques à la machine

cmd="ipfw -q add" # Commande de base
pif="tun0"        # Interface réseau côté "public"
                  # c'est à dire en principe Internet (voir nota)
```

### 3.4 Un exemple de règles de filtrage

---

```
# Pas de restrictions sur les interfaces avec le réseau privé
# Inutile si vous n'avez pas de réseau privé.
# Changer xl0 pour adapter à votre cas (et décommenter si nécessaire!)

$cmd 00005 allow all from any to any via rl0
$cmd 00006 allow all from any to any via ep0

# Pas de restrictions sur l'interface loopback

$cmd 00010 allow all from any to any via lo0

# Autorise la circulation des paquets s'ils correspondent aux règles
# dynamiques créées par la commande keep-state.

$cmd 00015 check-state

#####
# Interface vers l'extérieur c'est à dire Internet (Section sorties)
# Concerne le démarrage de session émanant du réseau privé ou du PC
# lui-même et destinées au réseau extérieur
# Autorise l'accès au DNS du fournisseur d'accès Internet
# Dupliquer cette ligne en cas de DNS secondaire (cas ici)
# Pour les adresses voir /etc/resolv.conf
# Dans cet exemple les DNS sont ceux de WANADOO

$cmd 00110 allow tcp from any to 80.10.246.1 53 out via $pif setup keep-state
$cmd 00111 allow udp from any to 80.10.246.1 53 out via $pif keep-state
$cmd 00112 allow tcp from any to 80.10.246.132 53 out via $pif setup keep-state
$cmd 00113 allow udp from any to 80.10.246.132 53 out via $pif keep-state

# Autorise les connexions sortantes non sécurisées standard www
$cmd 00200 allow tcp from any to any 80 out via $pif setup keep-state

# Idem mais connexions sécurisées https via TLS SSH
$cmd 00220 allow tcp from any to any 443 out via $pif setup keep-state

# Connexions relatives à l'envoi et à la réception de mails
$cmd 00230 allow tcp from any to any 25 out via $pif setup keep-state
$cmd 00231 allow tcp from any to any 110 out via $pif setup keep-state

# Autorise les sorties FreeBSD relatives à make install et CVSUP
$cmd 00240 allow tcp from me to any out via $pif setup keep-state uid root
```

### 3.4 Un exemple de règles de filtrage

---

```
# Autorise les sorties de pings
$cmd 00250 allow icmp from any to any out via $pif keep-state

# Autorise les sorties vers serveurs de temps
$cmd 00260 allow tcp from any to any 37 out via $pif setup keep-state

# Autorise les sorties nntp news (par ex. news groups)
$cmd 00270 allow tcp from any to any 119 out via $pif setup keep-state

# Autorise les sorties sécurisée FTP, Telnet et SCP
# Ces applications utilisent SSH (secure shell)
$cmd 00280 allow tcp from any to any 22 out via $pif setup keep-state

# Autorise les sorties whois
$cmd 00290 allow tcp from any to any 43 out via $pif setup keep-state

# Bloque et enregistre toute tentative de sortie
# ne correspondant pas aux critères précédents
$cmd 00299 deny log all from any to any out via $pif

#####
# Interface vers l'extérieur c'est à dire Internet (Section entrées)
# Concerne le démarrage de session émanant du réseau public et destinés
# au réseau interne ou au PC lui-même.

# Cette série bloque tout trafic venant de l'extérieur avec comme
# adresse source une adresse non routable (réservée réseau privé)

$cmd 00300 deny all from 192.168.0.0/16 to any in via $pif #RFC 1918 private IP
$cmd 00301 deny all from 172.16.0.0/12 to any in via $pif #RFC 1918 private IP
$cmd 00302 deny all from 10.0.0.0/8 to any in via $pif #RFC 1918 private IP
$cmd 00303 deny all from 127.0.0.0/8 to any in via $pif #loopback
$cmd 00304 deny all from 0.0.0.0/8 to any in via $pif #loopback
$cmd 00305 deny all from 169.254.0.0/16 to any in via $pif #DHCP auto-config
$cmd 00306 deny all from 192.0.2.0/24 to any in via $pif #reserved for docs
$cmd 00307 deny all from 204.152.64.0/23 to any in via $pif#Sun cluster interconnect
$cmd 00308 deny all from 224.0.0.0/3 to any in via $pif #Class D & E multicast

# Fermeture accès ssh et telnet depuis Internet
$cmd 00309 deny all from any to any 22 in via $pif
$cmd 00309 deny all from any to any 23 in via $pif
```

### 3.4 Un exemple de règles de filtrage

---

```
# Impose le silence sur le port 80
$cmd 00309 deny all from any to any 80 in via $pif

# Bloque les pings venant de l'extérieur
$cmd 00310 deny icmp from any to any in via $pif

# Bloque les demandes d'indentification
$cmd 00315 deny tcp from any to any 113 in via $pif

# Bloque les services Netbios. 137=name, 138=datagram, 139=session
# Netbios = service de partage MS/Windows.
# Bloque les requêtes MS/Windows hosts2 port 81
$cmd 00320 deny tcp from any to any 137 in via $pif
$cmd 00321 deny tcp from any to any 138 in via $pif
$cmd 00322 deny tcp from any to any 139 in via $pif
$cmd 00323 deny tcp from any to any 81 in via $pif

# Bloque tout paquet incomplet
$cmd 00330 deny all from any to any frag in via $pif

# Bloque les paquets ACK non autorisés par une règle dynamique
$cmd 00332 deny tcp from any to any established in via $pif

# Autorise les fonctions std www sur le serveur apache
$cmd 00400 allow tcp from any to me 80 in via $pif setup limit src-addr 2

# Autorise les paquets entrants proto. FTP, Telnet et SCP.
$cmd 00410 allow tcp from any to me 22 in via $pif setup limit src-addr 2

# Allow in non-secure Telnet session from public Internet
# Autorise les accès Telnet non sécurisés venant du reseau public
# considéré comme dangereux puisque login et passwd circulent en clair
# Supprimer si vous n'avez pas de serveur telnet
$cmd 00420 allow tcp from any to me 23 in via $pif setup limit src-addr 2

# Bloque et enregistre tous les autres paquets rentrants
$cmd 00499 deny log all from any to any in via $pif

# Tout le reste est bloqué par défaut. Bloque et enregistre.
$cmd 00999 deny log all from any to any
##### Fin du fichier des paramètres de IPFW #####
```

### 3.5 Test de l'efficacité du firewall

---

Nota : Notion d'interface "publique". Il convient ici de désigner la carte réseau qui ouvre sur l'Internet par le réseau. Dans le cas d'une connexion ADSL initiée par la machine elle-même l'interface "publique" est tun0 et non la carte réseau sur laquelle est connecté le modem ADSL/Ethernet.

Très schématiquement cet ensemble de règles est une copie des règles figurant dans le manuel. Quelques commentaires toutefois sur certaines modifications apportées :

- La ligne pif="tun0" et les règles 00005 et 00006 sont spécifiques au réseau décrit plus haut. A adapter à votre cas.
- Les règles 00309 sont des ajouts perso qui rendent le firewall silencieux aux scans de ports venant de l'Internet.
- Ces lignes illustrent un point de syntaxe, les numéros de règles n'ont pas besoin d'être distincts, les règles portant le même numéro sont examinées dans l'ordre dans lequel elles ont été "saisies", soit lues depuis le fichier de configuration, soit tapées au clavier.

### 3.5 Test de l'efficacité du firewall

Pour effectuer ce test deux types de solutions s'offrent à vous :

- Vous disposez d'un second accès à Internet, au travail, chez un voisin etc., vous pouvez alors lancer un scan de port avec un logiciel tel que nmap.
- Vous n'avez pas d'autre accès, la solution consiste à utiliser les services d'un serveur web spécialisé qui va effectuer le travail pour votre compte.

Parmi les sites qui proposent le scan j'utilise :

<https://grc.com/x/me.dll?bh0bkyd2>.

Ce site propose deux types de scans, une version rapide qui vérifie l'état des ports les plus utilisés, 21, 22, 23, 80 etc. et une version complète testant les 1056 premiers ports. Avantage de ce site, il est très rapide.

Il existe bien entendu d'autres sites, tapez "firewall test" dans votre navigateur et vous en obtiendrez une liste abondante.

L'intérêt de ces tests est évident. De façon tout à fait pratique c'est au moyen de ces tests que je me suis rendu compte que les ports 22, 23 et 80 étaient ouverts après que j'ai eu mis en place le firewall décrit dans le manuel. Ce constat a conduit aux trois règles 309 qui sont des compléments aux règles initiales.

Un autre site, francophone de plus, est :

<http://www.zebulon.fr/outils/>

Vous y trouverez un outil de scan mais aussi un outil de mesure du débit de votre connexion adsl.

## 4 Log des datagrammes bloqués

Les règles ci-dessus prévoient que certaines actions du firewall donnent lieu à des "log" c'est à dire à la création d'un enregistrement. Par défaut IPFW n'enregistre rien, même si les règles le prévoient. Pour déclencher l'enregistrement il est nécessaire de mettre à 1 la variable système

---

net.inet.ip.fw.verbose. En mode conversationnel cette opération se fait simplement en tapant :

```
# sysctl net.inet.ip.fw.verbose = 1
```

ce à quoi le système répond :

```
sysctl -w net.inet.ip.fw.verbose: 0 -> 1
```

confirmant ainsi que la variable vient bien de passer de 0 à 1. A compter de cet instant les messages de log vont s'ajouter à la fin du fichier /var/log/security.

## 5 Automatisation

Il n'est pas envisageable de refaire toutes les manipulations ci-dessus systématiquement à chaque démarrage. La solution est donc d'automatiser ce qui se fait en modifiant le fichier /etc/rc.conf pour ajouter les lignes suivantes :

```
firewall_enable="YES"  
firewall_script="/etc/ipfw.rules"  
firewall_logging="YES"
```

La première ligne est l'équivalent automatique de "kldload ipfw", la suivante indique le jeu de règles que doit appliquer ipfw, enfin la troisième conduit à enregistrer les logs, voir paragraphe 4.

## 6 Contrôle du débit avec dummynet

A l'origine l'objectif de dummynet était de fournir un outil permettant de simuler les problèmes classiques rencontrés sur une liaison réseau, délai de latence, routes multiples, paquets égarés etc. Nous nous intéresserons ici spécialement à l'option permettant de réduire la bande passante en fonction de certains critères comme l'adresse IP source ou destination. Formulés en des termes moins techniques, vous êtes à la tête du joli réseau décrit par le schéma 1 page 3. Fier de cette réalisation vous aimeriez, de temps en temps, pouvoir accéder à vos mails ou surfer sur le net, chose qui vous est bien entendu interdite en pratique puisque les post-ados dont les PC ne figurent pas sur le schéma, non contents d'utiliser un système d'exploitation non libre édité du côté de REDMOND, vous mangent la totalité de la bande passante en téléchargements divers. Vous êtes un homme prudent, le hub et le boîtier wifi sont à portée de main. La solution brutale consiste donc à déconnecter les importuns en débranchant le câble qui correspond. Cette solution binaire risque bien entendu de vous attirer des ennuis majeurs du genre porte qui s'ouvre et tête ébouriffée qui s'inquiète, généralement dans un français peu courtois, de la santé de ce (censuré !) de firewall !

Dummynet va vous offrir la possibilité de régler un débit maxi pour certains utilisateurs ce qui vous laissera un peu de bande passante disponible en permanence.

### 6.1 Mise en place manuelle

On admettra qu'il n'est pas très efficace de disposer d'une ligne ADSL et de l'étrangler en permanence pour un utilisateur donné à 1/2 voir 1/4 de son débit. Une première solution consiste donc à limiter le débit à la volée et à l'instant où le problème se pose. Pour cela procéder comme suit :

- Charger le module dumynet.
- Ajouter une règle qui enverra tout le trafic à contrôler vers un "pipe" (tunnel).
- Ajouter une règle qui va définir ce qui doit se passer dans ce tunnel.

Prenons un exemple très concret. Nous allons limiter le débit pour tout type de datagramme pour le PC dont l'adresse IP est 192.168.2.105. La succession des frappes est la suivante :

```
# kldload dumynet
# ipfw add 00003 pipe 1 ip from any to 192.168.2.105
# ipfw pipe 1 config bw 100Kbits/s
```

La première ligne charge le module dumynet. La suivante installe, en tête de la liste des règles, une règle mettant en place le tunnel 1 pour tous les messages à destination de 192.168.2.105. La dernière enfin limite de débit (bw = bandwidth) à 100 Kbits/s. Ainsi configuré le firewall limite la bande passante de l'utilisateur pour tous types de message, TCP, UDP, ICMP etc. Vous pourrez très facilement vérifier, par exemple sur [zebulon.fr](http://zebulon.fr), que la bande passante vue par 192.168.2.105 est effectivement limitée. Curieusement la valeur lue est en fait très proche de la moitié du chiffre entré en paramètre, probablement pas suite d'un double passage dans le tunnel (à vérifier).

### 6.2 Mise en place automatique

Le paramétrage automatique se heurte au problème du chargement du module dumynet. Je n'ai pas trouvé de syntaxe adéquate qui permettrait de le charger, par exemple à partir de `/etc/rc.conf`. Une solution, j'admets qu'elle est un peu "crade", consiste à charger ce module depuis `ipfw.rules`. Pour cela ajouter la ligne :

```
kldload dumynet
```

tout en haut du fichier, juste sous le titre. Au boot tout se passera bien, le module se chargera dès que les règles de firewall seront mises en place, par contre si vous relancer le firewall après avoir modifié les règles par :

```
# sh /etc/ipfw.rules
```

vous obtiendrez un message d'erreur (sans conséquence)

```
kldload: can't load dumynet: File exists
```

Les lignes suivantes :

---

```
$cmd 00003 pipe 1 ip from any to 192.168.2.105
ipfw pipe 1 config bw 2000Kbit/s
```

trouverons leur place juste après les lignes définissant les variables `cmd` et `pif`. Dans cette syntaxe je limite le débit du PC d'un de mes post-ados à 1 Mo/s (moitié de la consigne). En pratique puisque je dispose d'une ligne ADSL 1 Mo je laisse le tunnel grand ouvert mais je suis près à tout instant, en tapant simplement une ligne sur le modèle :

```
ipfw pipe 1 config bw 500Kbit/s
```

à réduire brutalement le débit mis à disposition d'un gamin trop gourmand.

Note : Il est impératif de fixer un débit initial dans le fichier `/etc/ipfw.conf` faute de quoi, le tunnel n'étant pas défini, les paquets seraient systématiquement tous perdus.

Il existe bien entendu une méthode "propre" pour mettre en place `dummynet`, celle qui consiste à recompiler le noyau après avoir fixé les bonnes options. Voir à ce sujet les manpages correspondantes. Dans mon cas le manque de place sur le petit disque dur du PC `jml1` m'interdit de recompiler.

## 7 Le mot de la fin

Ce document fait suite à une série d'articles écrits sur des sujets connexes dans lesquels je guide le lecteur pour lui permettre de faire ses premiers pas sur des sujets tels que :

- Le routage sous Linux ou FreeBSD
- La mise en place et le partage d'une connexion ADSL
- La mise en oeuvre du wifi sous FreeBSD

Ces documents sont consultables sur le site du Mirabellug à l'adresse <http://www.mirabellug.org>.  
Copyright ©jml 2005, document placé sous licence libre GNU FDL.

Auteur : [jean-marc.lichtle@gadz.org](mailto:jean-marc.lichtle@gadz.org)