**Comment transformer un vieille UC NEC powermate 223 Mo de RAM (par exemple, une vieille uc, quoi) en serveur pour l'autohébergement derrière une livebox.**

Première remarque : cet article est un aide mémoire et ne prétend pas être la meilleure méthode.
C'est une solution simple, qui fonctionne bien et qui me permet, en plus, de conserver un pc de secours (quand mon grand fiston me taxe mon laptop pour jouer à Battle for Wesnoth par exemple).

Ce PDF est une sorte de première ébauche afin de permettre à tous ceux comme moi qui n'ont pas de connaissance particulière en informatique de pouvoir profiter d'internet sans qu'on leur dise bêtement « RTFM » à chaque fois qu'il ne comprennent pas la documentation pour informaticiens. Il est sûrement farci de bêtises, mais il est fonctionnel. Soyez indulgents si vous vous y connaissez plus que moi, et participez à l'amélioration de ce mini tuto plutôt que de le dénigrer.

**1) Installer un OS** (pas forcément une version serveur)

Ceci est une étape extrêmement simple : J'ai téléchargé une image du liveCD de la distribution GNU/Linux Ubuntu (version desktop pour garder l'avantage d'un gnome pour pc de secours ou de jeu pour les petits). Je sais, les puristes vont hurler car je gaspille des ressources système avec une interface graphique, et une lourde de plus. Peu importe.

Ensuite j'ai gravé l'image sur un cd (avec brasero).

Enfin, j'ai installé l'OS sur l'UC. (boot sur le cd, installer)

Ceci est simplissime et prend moins d'une heure.

**2) Régler le firewall**, car il est activé au démarrage mais configuré pour tout laisser passer dans tous les sens. (merci Diodio13)

Pour cela il faut connaitre le nom de ton interface réseau et faire en sorte que les règles soient sauvée lors des arrêts et rechargées lors des démarrage (ce qui n'est pas le cas d'origine).

Ceci se fait en console avec la commande « iptable ».

D'abord je bloque tout le trafic en entrée, en sortie et relais avec les commandes suivantes dans une console :

```
sudo iptables -P INPUT DROP
```

```
sudo iptables -P OUTPUT DROP
```

```
sudo iptables -P FORWARD DROP
```

Ensuite, j'accepte le trafic sur la boucle locale :

```
sudo iptables -A INPUT -i lo -s 127.0.0.1/24 -j ACCEPT
```

```
sudo iptables -A OUTPUT -o lo -s 127.0.0.1/24 -j ACCEPT
```

Ensuite, j'accepte le suivi de connexion en entrée, c'est-à-dire j'autorise les connexions entrantes si elles sont une réponse à une demande initiée par moi (ou plutôt mon serveur) :

```
sudo iptables -A INPUT -i eth0 -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Enfin, j'accepte tout le traffic en sortie (ce qui rend dérisoire la ligne « iptables -P OUTPUT DROP » je sais) :

```
sudo iptables -A OUTPUT -o eth0 -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
```

Mettre le système à jour :

`sudo apt-get update` ainsi que `sudo apt-get upgrade`

Comme c'est un serveur web il faut ouvrir le port 80 en entrée :

```
sudo iptables -I INPUT -i eth0 -p tcp -m tcp --dport 80 -j ACCEPT
```

Comme je veux pouvoir gérer avec webmin, je dois ouvrir le port 10000 (en effet, on accède à webmin avec https://192.168.x.xx:10000/, les x,xx dependent de l'IP locale de l'UC qui sert de serveur. Cette adresse commence par 192.168.x.xx. Pour la connaître, taper ceci sur la console du serveur : ifconfig . À la seconde ligne après eth0, cette adresse est écrite à la suite de « inet adr : ».)

```
sudo iptables -A INPUT -p tcp --dport 10000 -j ACCEPT
```

NB : à chaque fois qu'on rajoute un service en écoute, comme à l'instant pour webmin, il faut penser à ouvrir les ports qui vont bien, avec iptables.

Il faut également sauvegarder ces règles afin qu'elles s'appliquent à chaque fois que le serveur est redémarré (sinon, il faudrait tout recommencer manuellement :/)

Pour cela, il faut créer un fichier avec gedit par exemple, fichier sans extension particulières qu'on appelera comme on veut (gestion-des-ports par exemple).

La première ligne du fichier sera `#!/bin/sh`
On y recopiera (à la suite de cette entête) toutes les lignes en gris, les unes en dessous des autres et elles s'appliqueront à chaque démarrage.

```
#!/bin/sh
sudo iptables -P INPUT DROP
sudo iptables -P OUTPUT DROP
sudo iptables -P FORWARD DROP
sudo iptables -A INPUT -i lo -s 127.0.0.1/24 -j ACCEPT
sudo iptables -A OUTPUT -o lo -s 127.0.0.1/24 -j ACCEPT
sudo iptables -A INPUT -i eth0 -m state --state ESTABLISHED,RELATED -j ACCEPT
sudo iptables -A OUTPUT -o eth0 -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
sudo apt-get update
sudo apt-get upgrade
sudo iptables -I INPUT -i eth0 -p tcp -m tcp --dport 80 -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 10000 -j ACCEPT
```

On enregistre ce fichier dans le répertoire /etc/network/if-up.d

il faut également le rendre exécutable :

```
sudo chmod +x /etc/network/if-up.d/gestion-des-ports
```

Remarque 1 : pensez à naviguer avec les droits admin (sudo nautilus) ou à lancer gedit avec les droits admin (sudo gedit)
Remarque 2 : cette gestion des ports est extrêmement restrictive, mais cela était bien ma volonté pour ce serveur.

**3) Installer les autres éléments de "LAMP"**

Au départ, j'avais choisi de simplement passer par tasksel (à installer en console avec un simple "sudo apt-get install tasksel" puis à lancer, toujours en console avec un très compliqué "tasksel" et sélectionné Server Lamp. Merci tshirtman

Ça fonctionne, mais j'ai voulu obtenir les dernières versions d'apache, de MySQL et PHP, alors j'ai aussi tapé (merci Diodio13) :

`sudo apt-get remove postgresql` (dans le cas où il était installé)

```
sudo apt-get install apache2 apache2-doc libapache2-mod-php5 mysql-server php5 php5-mysql proftpd php5-gd
```

Il « faut » également installer webmin (voir la doc ubuntu http://doc.ubuntu-fr.org/webmin )

Webmin n'est pas obligatoire, mais c'est un outil vraiment simple pour un débutant comme moi.

**4) Créer le (ou les) noms de domaines sur dyndns.com**

Il suffit de ce rendre sur le site http://www.dyndns.com , d'ouvrir un compte et de créer un nom de domaine Free Dynamic DNS du genre : le-nom-que-je-veux.is-a-geek.net (il y a 85 extensions possibles, dont « .is-a-geek.net », « .kicks-ass.org », « .homelinux.net », « .doesntexist.com », etc.

Puis il faut associer l'adresse ip actuelle de la livebox (que l'on peut connaître sur http://www.mon-ip.com/ par exemple) avec cette adresse dyndns et activer ce service sur le site de dyndns.com (c'est un clicodrome, ce n'est pas bien compliqué)

**5) Paramétrer la livebox**

Il suffit d'accéder à votre livebox en tapant l'adresse http://192.168.1.1/ dans votre navigateur. Entrer en vous identifiant (si vous avez eu l'intelligence de changer les login et password). Par défaut, le login est « admin » et le password est « admin » (pas bon de garder ça).

Ensuite, cliquez sur « Serveur LAN » un tableau apparaît. Une des ligne s'appelle « Nouvelle entrée ». Dans la colonne « Action » de cette ligne, cliquez sur la petite icône « nouveau ».
Une fiche de renseignement s'ouvre. Donner un nom (genre serveur-web ou mon-serveur ...), Accès activé : Oui, Du port : 80, Au port : 80 et précisez l'IP locale de l'UC qui sert de serveur. Cette adresse commence par 192.168.x.xx. Pour la connaître, taper ceci sur la console du serveur : ifconfig
À la seconde ligne après eth0, cette adresse est écrite à la suite de « inet adr : ».

Puis appliquer, et OK.

**6) Utiliser webmin pour associer chaque adresse choisie avec son contenu.**

En tapant https://192.168.x.xx:10000/ (en remplaçant les x.xx pour obtenir l'IP locale du serveur) dans la barre d'un navigateur d'un ordinateur connecté à la même livebox (en wifi par exemple), j'accède à l'interface de webmin.

Remarque : http**s** et **« : »** avant le 10000
Une page vous demandant votre nom d'utilisateur et votre mot de passe apparaît. Entrez « root » comme login et le password que vous utilisez pour le sudo de votre serveur, puis validez. Vous devriez être connecté.

Nous allons nous servir de webmin pour dire au navigateurs où chercher le fichier « index.htm(l) » de chaque adresse crée sur dyndns.

Pour cela cliquez sur « server->apache webserver » dans le menu à gauche, puis sur l'onglet « create virtual host »

Précisez :
adresse : « any »
port :80
document root : là ou se trouvera le fichier index.htm(l) de votre site, par exemple /var/www
server name : l'adresse choisie chez dyndns, par exemple trucbidule.is-a-geek.org

On peut faire autant de site que l'on veut, avec chacun son adresse (préalablement créée sur www.dyndns.com et chacun son dossier de rangement (on peut créer /var/www2, /var/www3, etc. par exemple)

**7) paramétrer les noms de domaines /IP dynamique.**

À chaque fois que votre livebox est reconnectée, rallumée ou mise à jour, son adresse IP est changée. Ceci est embêtant, car les adresses choisies chez dyndns et les adresses IP ne colleront plus et votre serveur sera inaccessible.

De façon étonnante, seule l'une des adresses voit son IP changer, les autres restent avec l'IP précédente...

Pour l'instant je ne sais pas comment faire, sinon me reloguer sur dyndns, accéder à mes adresses en cliquant sur Host Service, Cliquer sur le lien qui ne fonctionne plus et actualiser l'adresse IP en cliquant sur « Use auto detected IP address ». Mais une adresse dyndns est sensée s'actualiser seule, alors :/ ...