Configuration d'un adaptateur USB Sans−Fil (chipset Atmel)

par Arnaud Fontaine

Juillet 2003

Table of Contents

1 Présentation

Les réseaux Sans−Fil ont connu un succès fulgurant. Le standard international IEEE 802.11, élaboré dès 1997 et à l'origine de ces réseaux, permettent de mettre en place des réseaux hertziens à hauts débits et dont les principaux avantages restent la mobilité et le faible coût. De plus, ce standard a donné naissance à de nombreuses autres normes proposant un débit plus élevé, une meilleure intéropabilité ou une sécurité accrue. Ainsi est née la norme 802.11b avec un débit de transmission théorique maximum de 11 Mbps (seulement 1 ou 2 Mbps pour le standard initial). Nous n'irons pas plus loin dans la présentation du Wireless dans cet article mais si vous voulez en apprendre plus, je vous conseille de visiter le site de la Fédération France Wireless qui rassemble les efforts des différentes communautés partout en France.

L'adaptateur USB Sans−Fil de Belkin est donc destiné aux réseaux Sans−Fil supportant le standard 802.11b. Cet adaptateur repose sur un chipset Atmel AT76C503A utilisé sur de nombreux autres périphériques Sans−Fil. La configuration de l'adaptateur qui fait l'objet de cet article devrait donc s'adapter aux cartes présentes sur cette page.

Afin de bien assimiler le contenu de cet article, je vous conseille vivement de lire le Kernel HOWTO afin de mieux comprendre les termes *noyau*, *modules*... Étant donné que nous décrirons l'installation sur une distribution Debian GNU/Linux et consorts telle que Knoppix, il faudra que celle−ci soit installée correctement. La procédure décrite est valable aussi pour les versions *testing* et *unstable*. De plus vous devriez déjà connaître les commandes de bases d'un système UNIX ainsi que les commandes de gestion de paquets sous Debian.

2 Choix des pilotes

Il existe deux pilotes différents pour ce chipset. La version initiale des pilotes pour Atmel est disponible à l'adresse suivant : http://atmelwlandriver.sourceforge.net/. Une autre version proposant principalement des corrections de bogues est disponible à l'adresse suivante : http://at76c503a.berlios.de/. Pour ce document, nous allons utiliser la deuxième version de ces pilotes qui supporte parfaitement *iwconfig* et qui nous permettra un

déboguage plus aisé.

Malheureusement comme l'indique la page d'accueil des pilotes Berlios, ils possèdent quelques limites dues à l'adaptateur en lui même. En effet le firmware actuel ne supporte pas le mode promiscuité, monitor, ou station (hostap) pour la librairie libpcap. Vous ne pourrez donc pas utiliser kismet, airsnort et autres pour *« sniffer les airs »* et votre carte ne pourra pas non−plus se comporter comme un point d'accès. De plus, Les pilotes actuelles ne fonctionnent pas sur la série 2.5.x du noyau et encore moins sur la série 2.6.x, ce qui ne nous pose pas de problèmes car nous allons utiliser des versions stables du noyau (2.4.x).

Si votre noyau est déjà compilé avec le support de l'USB (comme dans la distribution Knoppix) alors passez directement à la section *Installation des pilotes*, la section suivante n'étant d'aucune utilité té dans ce cas à moins que vous vouliez mettre à jour votre version du noyau. Dans la section suivante, je vais considérer que vous venez juste d'installer votre distribution Debian en stable, c'est à dire que vous disposez (normalement) d'un noyau 2.2.20.

3 Configuration du noyau

Avant de commencer cette section, il faudra que vous ayez installé différents paquets, ce que nous allons faire :

```
 # apt−get install bzip2 libncurses5−dev kernel−package gcc−2.95 make
```

La distribution Debian GNU/Linux fournit différentes versions du noyau suivant l'état de votre distribution (*stable*, *testing* ou *unstable*). Nous n'allons pas utiliser la dernière version des sources du noyau disponible en paquet Debian sur la version stable de la distribution (2.4.18). il vaut mieux utiliser la dernière version qui corrige certains bogues et ajoute le support de périphériques en plus. (les changements apportés par rapport aux autres versions sont disponibles ici) Nous allons donc télécharger la dernière version du noyau disponible à l'adresse suivante : ftp://ftp.fr.kernel.org/pub/linux/kernel/v2.4/linux−2.4.21.tar.bz2. Une fois téléchargé, copiez le dans le répertoire `/usr/src/`.

Ensuite vous décompressez l'archive copiée et créer un lien :

```
 # cd /usr/src
# tar xvjf linux−2.4.21.tar.bz2
# ln −s linux−2.4.21 linux
```

Nous allons ensuite configurer notre noyau. Au préalable vous devriez disposer d'informations sur votre matériel afin que vous puissiez le sélectionner dans les menus de configuration du noyau. Le plus important est de savoir de quel processeur vous disposez car vous devrez en sélectionner un dans les menus de configuration du noyau. Tapez ensuite la commande suivante :

```
# make menuconfig
```

Je décrirais seulement les options minimum à modifier pour que votre noyau soit fonctionnel (pour ajouter le support de vos périphériques, je vous invite à lire cette documentation très complète : http://via.ecp.fr/~alexis/formation−linux/config−noyau.html).

Vous arrivez désormais sur un système de menus, activez les options suivantes pour prendre en charge votre adaptateur USB :

Code maturity level options
    [*] Prompt for development and/or incomplete code/drivers

Network device support
  Wireless LAN (non−hamradio)
    [*] Wireless LAN (non−hamradio)

USB support
    <M> Support for USB
    [*] USB verbose debug messages
    <M> UHCI Alternate Driver (JE) support

Une fois que vous activez ces options, il vous suffit de taper ceci pour compiler votre noyau (vous pouvez également consulter l'article sur la compilation d'un noyau à la sauce Debian) :

```
 # make−kpkg clean
# make−kpkg −−revision 2.4.21 kernel−image
```

Note : J'ai rencontré quelques problèmes en compilant le noyau avec gcc 3.x, j'ai donc utilisé gcc 2.95 qui a fonctionné sans problème. Si vous avez gcc 3.x installé, je vous conseille d'utiliser, pour la compilation, gcc 2.95 :

```
 # rm /usr/bin/gcc
# ln −s /usr/bin/gcc−2.95 /usr/bin/gcc
```

La compilation peut être plus ou moins longue selon le processeur dont vous disposez, avec un Athlon à 1200 Mhz, cela me prend environ 13 minutes mais sur des processeurs beaucoup moins puissant comme un i386 ou un i486 cela peut prendre quelques heures voire quelques jours, dans ce cas configurer donc votre noyau avec les options adaptées à votre vieille machine et copiez le paquet obtenu sur celle ci. Une fois que la compilation de votre nouveau noyau est terminé, il ne vous reste plus qu'à installer le paquet créé :

```
 # dpkg −i /usr/src/kernel−image−2.4.21_2.4.21_i386.deb
(Lecture de la base de données... 87141 fichiers et répertoires déjà
installés.)
Dépaquetage de kernel−image−2.4.21 (à partir de
kernel−image−2.4.21_2.4.21_i386.deb) ...
Paramétrage de kernel−image−2.4.21 (2.4.21) ...
Not updating image symbolic links since we are being updated (2.4.21)
```

Une fois le paquet installé, il ne vous reste plus qu'à configurer votre gestionnaire d'amorçage (*lilo* ou *grub*). Notez qu'un article sur la migration de Lilo vers Grub est disponible sur Andesi. Je vais donc décrire ce qu'il est nécessaire d'ajouter pour que vous puissiez démarrer sur votre prochain noyau. Il vous suffira seulement d'adapter selon le partitionnement de votre disque dur (prenez exemple sur la configuration que vous utilisez actuellement) :

 # Pour Lilo, ajoutez ceci au fichier de configuration /etc/lilo.conf
image=/boot/vmlinuz−2.4.21
  label=Linux−2.4.21
  root=/dev/hda7
  read−only
 # Pour que vos changements sur le fichier de configuration de Lilo soient pris en compte il # faut taper ceci dans une console :
 # lilo

 # Pour Grub, ajoutez ceci au fichier de configuration /boot/grub/menu.lst (usuellement)

```
title Debian GNU/Linux, 2.4.21
root (hd0,0)
kernel /vmlinuz−2.4.21 root=/dev/hda7
```

Enfin redémarrez sur votre nouveau noyau. Nous allons maintenant passer à l'installation des pilotes Berlios pour notre chipset Atmel.

4 Installation des pilotes

Une fois que avez copié l'archive des pilotes disponibles à http://download.berlios.de/at76c503a/at76c503−0.10.tar.gz dans le répertoire `/usr/src/` :

```
# cd /usr/src/
# tar zxvf at76c503−0.10.tar.gz
# cd at76c503a−0.10
```

L'installation des pilotes sous forme de modules du noyau s'avèrent très simple car il vous suffit de taper la commande qui suit pour les installer sans recompiler votre noyau :

```
# make
# make install
```

Les modules sont placés dans `/lib/modules/2.4.21/kernel/drivers/usb/` :

```
# ls −l /lib/modules/2.4.21/kernel/drivers/usb/at*
−rw−r−−r−− 1 root root 31876 2003−07−12 11:58 at76c503−i3861.o
−rw−r−−r−− 1 root root 31620 2003−07−12 11:58 at76c503−i3863.o
−rw−r−−r−− 1 root root 66332 2003−07−12 11:58 at76c503.o
−rw−r−−r−− 1 root root 41720 2003−07−12 11:58 at76c503−rfmd.o
−rw−r−−r−− 1 root root 39096 2003−07−12 11:58 at76c505−rfmd.o
```

Chargez maintenant les modules pour l'USB et votre adaptateur USB (voir les logs dans une autre console : `# tail −v −f /var/log/syslog`) :

```
# modprobe uhci
uhci.c: USB Universal Host Controller Interface driver v1.1 [...]
hub.c: 2 ports detected [...]
usb.c: USB device 2 (vend/prod 0xd5c/0xa002) is not claimed by any active
driver.

# modprobe at76c503−rfmd
at76c503−rfmd.c: Atmel at76c503 (RFMD) Wireless LAN Driver v0.10
usb.c: registered new driver at76c503−rfmd
at76c503.c: $Id: at76c503.c,v 1.25 2003/06/01 19:42:28 jal2 Exp $ compiled
Jul 12 2003 11:58:44
at76c503.c: firmware version 1.101.2 #84
at76c503.c: device's MAC 00:30:bd:62:a1:52
at76c503.c: registered wlan0
```

Normalement l'installation et le chargement de ces modules ne devraient pas du tout vous poser de problèmes. Vous pouvez également vérifier que tout a bien fonctionné :

```
# lsmod

at76c503−rfmd 39424 0 (unused)
```

```
uhci           24112 0 (unused)
usbdfu         7540  0 [at76c503−rfmd]
at76c503       48192 0 [at76c503−rfmd]
usbcore        36288 0 [at76c503−rfmd uhci usbdfu at76c503]
```

Pour que vos modules soient lancés à chaque démarrage de votre système, il vous suffit d'éditer le fichier `/etc/modules` en ajoutant ces entrées :

```
 uhci
at76c503−rfmd
```

5 Configuration

Maintenant que vous avez installé les pilotes de la carte, vous vous demandez certainement comment configurer votre adaptateur pour le mettre en réseau. Pour cela commencez par installer le paquet `wireless−tools` :

```
# apt−get install wireless−tools
```

Notre réseau est centré autour d'un point d'accès qui porte le *ssid default* sur le *canal 10*. Le réseau est donc de type Infrastructure (*Managed*) car les clients sont connectés au réseau Sans−Fil par un point d'accès, contrairement au type Ad−Hoc où les clients sont connectés les uns aux autres. Nous allons donc configurer notre périphérique portant le nom *wlan0* par défaut afin qu'il se connecte à notre point d'accès puis nous allons assigner l'adresse IP 192.168.0.2 à l'adaptateur :

```
 # iwconfig wlan0 essid default channel 10 mode Managed
# ifconfig wlan0 192.168.0.2 netmask 255.255.255.0 up
```

Une fois que vous avez tapé ces deux commandes, l'adaptateur devrait avoir récupéré l'adresse MAC du point d'accès en *italique* ci dessous :

```
 # iwconfig wlan0
wlan0 IEEE 802.11−DS ESSID:"default"
        Mode:Managed Channel:10 Access Point: 00:80:C8:03:40:33
        Bit Rate:11Mb/s
        RTS thr=1536 B Fragment thr=1536 B
        Encryption key:off
        Power Management:off
        Link Quality:0 Signal level:95 Noise level:0
        Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0
        Tx excessive retries:0 Invalid misc:0 Missed beacon:0
```

Les informations importantes ont été mises en **gras**. Elles concernent respectivement le mode de fonctionnement (*Mode*), le canal (*Channel*), l'adresse MAC du point d'accès (*Access Point*), la vitesse du lien (*Bit Rate*), la qualité du lien (*Link Quality*), le volume du signal (très important : *Signal level*, plus il est important meilleur sera la réception) et enfin le volume du bruit (*Noise level*).

6 Automatiser la configuration

Afin d'éviter de taper à chaque démarrage les commandes données dans la section précédente, vous pouvez *automatiser* la configuration de votre adaptateur à chaque démarrage de votre système. C'est ce que nous allons aborder dans cette section (merci à Christophe Nowicki pour la partie concernant le fichier

`/etc/network/interfaces`).

Nous allons tout d'abord ajouter les modules USB et de l'adaptateur Belkin dans le fichier `/etc/modules` afin qu'ils soient chargés au démarrage. Ajoutez donc les entrées suivantes dans ce fichier à l'aide de votre éditeur favori :

```
 uhci
at76c503−rfmd
```

Puis nous allons ajouter les entrées au fichier `/etc/network/interfaces` permettant d'assigner une adresse IP à l'adaptateur mais aussi de configurer la partie radio de la carte :

```
 auto wlan0
iface wlan0 inet static
    address 192.168.0.2
    netmask 255.255.255.0
    pre−up iwconfig wlan0 essid default channel 10 mode Managed
```

Désormais votre adaptateur USB Sans−Fil sera configuré à chaque démarrage de votre système, vous pouvez tout de suite vérifier que cela fonctionne bien en tapant :

```
 # /etc/init.d/network stop
Deconfiguring network interfaces... done.
# ifconfig
# /etc/init.d/networking start
Configuring network interfaces... done.
# ifconfig wlan0
wlan0 Lien encap:Ethernet HWaddr 00:30:BD:62:A1:52
    inet adr:192.168.0.2 Bcast:192.168.0.255 Masque:255.255.255.0
    BROADCAST MTU:1500 Metric:1
    RX packets:0 errors:0 dropped:0 overruns:0 frame:0
    TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 lg file transmission:100
    RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)
```

7 Conclusion

L'installation des pilotes s'avèrent finalement très simple. Le plus difficile reste tout de même la compilation du noyau pour le débutant, mais aussi réussir le lien. De plus ces adaptateurs sont beaucoup moins sensibles et puissants qu'une carte orinoco mais leur prix d'environ 50 euros reste très attractif pour monter un réseau Sans−Fil économique.

J'espère que cela vous aura donné envie de vous mettre au Wireless qui représente une technologie très intéressante. Nous n'avons pas aborder la sécurité (WEP, IPSEC, ...) car ceci sortirait du cadre de cet article mais je vous conseille de sécuriser un minimum votre réseau pour qu'il ne soit pas accessible à tout le monde, à moins que cela ne constitue le but de votre réseau :).

Modifications du document :
− Mises à jour effectuées le 28 Juillet 2003 : ajout de la section « Automatiser la configuration » en suivant la suggestion de Christophe Nowicki concernant le fichier `/etc/network/interfaces`.
− Mises à jour effectuées le 20 Septembre 2003 : correction du terme « Wireless » en « Sans−Fil » un peu plus français et ajout d'un lien vers l'article sur la compilation d'un noyau à la sauce Debian.

− *Mises à jour effectuées le 08 Novembre 2003 : relecture effectuée par Iglinux.*

This document was translated from LaTeX by HeVeA.