

Configuración Básica de nodo con MadWifi en Debian

Álvaro López Hornero
bartlevi83@madridwireless.net
V1.0 01/26/04

COPYRIGHT

Copyright (c) 2003 Álvaro López Hornero. Se otorga permiso para copiar, distribuir y/o modificar este documento según los términos de la Licencia GNU Para Documentación Libre (GNU Free Documentation License), versión 1.2 o cualquier versión posterior publicada por la Free Software Foundation. Esta licencia está disponible en <http://gnu.org/copyleft/fdl.html>.

RESUMEN

Breve documento acerca de la configuración de un AP con Madwifi para dummies, con todos los pasos que seguí para configurar mi AP, que espero os ayuden a configurar el vuestro y hacer un nodo más unido a los múltiples proyectos GNU.

Importante: ¿ Por que me decidí por esta tarjeta ? Pues la razón es simple, desde luego no elegí la tarjeta, sino el chipset, la tarjeta es lo de menos, el tema es que había dos chipsets en el mercado con soporte para linux en 802.11g y pci, que era lo que buscábamos para los nodos; el atheros, que es del que estamos tratando, y el prismGT, pero visto que la mayoría de los nodos, son maquinas antiguas, y que los prismGT mas accesibles, es decir, la tarjeta SMC2802, no es compatible con los pcs antiguos, por seguir la norma PCI2.2, pues decidí ir por la otra rama, que además, resultó mucho mas barata. Vuelvo a decir que la razón mas contundente, fue el tema de que las conceptronic, si se pudieran instalar en pcs antiguos, mientras que las SMC, no.

Para nuevas versiones de este documento: <http://www.alvarolopez.net/WIFI/> y en <http://www.madridwireless.net>

DESCARGO DE RESPONSABILIDAD

No me hago responsable en ningún caso de los posibles daños o pérdidas de garantía que pueda ocasionar el uso, debido o indebido, de la información existente en este documento. Si decides hacerme caso es tu problema. Acepto información diciendo: "me lo he cargado", pero no del tipo: "me lo he cargado por tu culpa".

CONFIGURACION

Como es un documento pequeño, no vamos a hacer un índice, ya que este micro howto no se lo merece

Vamos allá:

HARDWARE NECESARIO

Obviamente necesitaras un pc, con un p75 ya tendrás un nodo, pero es recomendable algo como un p200, porque sino, morirás mientras esperas a que compile el kernel, así mismo, con 32 de memoria, o 16, te ira bien.

Con 400 megas de hdd tendría que llegarte, si haces una instalación mínima, pero conviene meter mas, ya que siempre acabas metiéndole un apache, y haciendo tontadas con el ap.

Tarjetas de Red, por supuesto, una ethernet, y la wireless, en pci, o adaptador + pcmcia, lo que prefieras, pero que obviamente lleve chip **ATHEROS** que es de lo que estamos hablando aquí. Yo he probado con conceptronic, C54i, y C54c, (pci y pcmcia), y no me falla con ninguno de ellos. Luego ya si quieres poner más ethernet, para usar tu AP con firewall o proxy, y más wireless, para hacer enlaces pto-ptu, es cosa tuya, cuantas más cosas, más divertido ;)

En un principio no te hace falta nada mas, todo lo demás que le pongas al pc, es accesorio.

INSTALACIÓN

Instalaremos el sistema base como de costumbre, sin X, por supuesto, ya que este pc, intentara ser lo mas sencillo posible, y además se trata de una maquina la cual administraremos por ssh/telnet, así que no merece la pena meterse a consumir memoria y hdd por un sistema de ventanas, que no usaremos.

Una vez instalado el sistema base (no hay que preocuparse por el núcleo, puesto que vamos a recompilarlo mas tarde) instalamos todos los paquetes que nos harán falta al principio, así no nos olvidamos de hacerlo mas tarde uno a uno. Ya sabéis, con el clásico apt-get, para el que ande un poco perdido del tema, le dejo una fuente deb, que es la que yo uso siempre:

```
ftp://ftp.fr.debian.org/debian stable main contrib non-free
```

Es francesa, pero es que la española, da lastima, y esta suele funcionar bastante bien

```
apt-get install iptables → necesario para las rutas de enmascaramiento
apt-get install openvpn / apt-get install vtun → túneles contra otros nodos
apt-get install zebra → protocolos de red tales como ospf, y bgp
A estas alturas de la película, convendría instalar Quagga, que es un fork de zebra, y esta mucho mas desarrollado.
apt-get install wireless-tools → paquete conocido por todos, control de las ifaces wifi
apt-get install bridge-utils → por si decides hacer bridging en vez de NAT
apt-get install dhcpd → servidor dhcp
```

Creo que no me olvido ningún paquete, de todos modos, esto es un documento abierto, el que tenga algo que cambiar, añadir, matizar, es libre de hacerlo, pero agradecería que me mandara una copia al e-mail que dejo en el documento.

COMPILANDO EL KERNEL

Como ya dije antes, vamos a recompilar el núcleo, yo he usado el 2.4.22, obviamente se puede usar el que os venga en gana, pero yo he usado este, porque junto con el 2.4.20, son los dos que mas me gustan, para gustos están los colores, así que, la elección es tuya.

Aquí están las configuraciones necesarias para el sistema del AP, si le quieres meter sonido o algo así, es cosa tuya, pero si no lo vas a hacer, desactiva todo el sonido, SCSI y el soporte USB, así te ocupara menos el kernel, y cargara en menos tiempo.

También tenemos la opción de compilar el kernel sin soporte pcmcia, y meterlo mas tarde con pcmcia-cs, compilado a parte, esto tiene como ventaja, que por lo visto, van mejor los controladores del pcmcia-cs compilados por separado, que los incluidos en el kernel, y que además, podremos compilar el pcmcia-cs parcheado con "juguetes" para el airtight, o parches para poner Orinoco en modo monitor, y cosas así. Quizás sea un buen momento para decir que el Madwifi también se puede usar como un simple driver para tu tarjeta, puesto que soporta los modos managed y ad-hoc perfectamente, solo tienes que retocar el `/etc/network/interfaces` para poner la tarjeta en modo manager, o lo que quieras, así mismo también soporta el modo monitor, para escanear redes.

Una vez dicho esto, puedes elegir entre seguir esta configuración, o compilar el pcmcia-cs a tu aire.

```
make mrproper
make menuconfig
```

Una vez aquí, metemos estos parámetros en el núcleo:

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

Processor type and features
(Pentium-MMX) Processor family
(en mi caso un p200 mmx, tu pon tu procesador)

General setup
PCMCIA/CardBus support --->
<*> PCMCIA/CardBus support
[*] CardBus support

Networking options
<*> Packet socket
[*] Network packet filtering
[*] Socket Filtering
[*] TCP/IP Networking
[*] IP: multicasting (enrutamiento dinamico; ospf)
[*] IP: advanced router (políticas de trafico)
activando todas las sub-opciones
<*> IP: tunneling

Dentro de IP: Netfilter Configuration
Todas las opciones como modulos

<*> 802.1d Ethernet Bridging --> bridge-utils

Network device support

<*> Universal TUN/TAP device driver support
<*> Ethertap network tap (OBSOLETE) (NEW)

Ethernet (10 or 100Mbit)
<*> RealTek RTL-8139 PCI Fast Ethernet Adapter support
(en mi caso una realtek, tu mete tu tarjeta Ethernet)

Wireless LAN (non-hamradio)
[*] Wireless LAN (non-hamradio)

PCMCIA network device support
[] PCMCIA network device support **SIN ACTIVAR!!!**

Con esto acabamos de configurar el núcleo, ahora lo compilamos así:

```
make dep
make clean
make bzImage
make modules
make modules_install
cp -i arch/i386/boot/bzImage /boot/vmlinuz-2.4.22
cp -I System.map /boot/System.map-2.4.22
```

Y después de esto, editamos el lilo, lilo es el programa cargador del sistema (LInus LOader) y se configura desde el archivo de configuración; /etc/lilo.conf Hay que meter los siguientes parámetros:

```
image=/boot/vmlinuz-2.4.22
    label=2.4.22
    read-only
```

```
Y cambiar la etiqueta
default=Linux
Por
default=2.4.22
```

Con eso lilo ya está preparado para arrancar, pero ahora hay que ejecutarlo:

```
lilo
reboot
```

Así en el siguiente arranque, el arrancará con el nuevo kernel, preparado para nuestro MadWifi, y nuestra atheros

Esta es la manera en la que yo compilo el kernel, cada uno tiene la suya, y yo solo quiero indicar, que de esta forma, a mi me funciona, cada uno es libre de hacer lo que quiera, eso esta claro.

Con esto tenemos la maquina preparada para el MadWifi

Vamos con la instalación de lo que seria el sistema para el AP propiamente dicho

INSTALACIÓN DEL MADWIFI

Bajamos el madwifi de la pagina del proyecto: <http://sourceforge.net/projects/madwifi/>
Bueno, como podeis observar en esta página, han quitado las releases del servidor, pero por suerte, algunos guardamos el source. La versión que he usado, no se decir la versión, esta es la verion del README; v 1.17 2003/07/31 pero no tiene nada que ver con la versión del driver, no se la versión, porque lo baje del CVS, y no recuerdo por donde andaban, pero el tema, es que funciona.

He colocado el driver en esta página: <http://www.alvarolopez.net/WIFI/madwifi-howto/madwifi-cvs.tar.gz>

Una vez descargado, pasamos a descomprimirlo:

```
tar zxvf madwifi-cvs.tar.gz
cd madwifi-cvs
```

Si vas a instalar pcmcia, deberás instalar también el pcmcia-cs, esto es, bajarlo de <http://pcmcia-cs.sourceforge.net/>, descomprimirlo, y:

```
./Configure
make all
make install
```

Y ahora a compilar e instalar el MadWifi:

Simplete:

```
make
make install
insmod ath_pci ( en el caso de una tarjeta pci )
```

Esta es la salida que da mi lsmod (En una maquina con una tarjeta conceptronic C54i):

```
ap:~# lsmod
Module                Size  Used by    Tainted: P
ath_pci                29376  1
wlan                   41896  1 [ath_pci]
ath_hal                108208  1 [ath_pci]
```

Con esto ya tendremos el Madwifi instalado en nuestra maquina.

Por lo pronto tenemos la maquina con el madwifi instalado, pero sin configurar, en este paso, por supuesto, no aparece ip ni essid ni nada por el estilo, ya que aun no lo habéis configurado, a eso vamos ahora.

CONFIGURACIÓN DE IFACES

Ya tenemos nuestro AP, ahora vamos a configurar sus parámetros:

Nos vamos al archivo de configuración de ifaces, que en debian es:

```
ap:~# cat /etc/network/interfaces
```

```
auto lo
iface lo inet loopback

iface eth0 inet static
    address *.*.*.*
    netmask 255.255.255.0
    network *.*.*.*
    broadcast *.*.*.*
    gateway *.*.*.*
```

(no pongo los datos de mi red cableada, porque no son necesarios)

```
auto ath0
iface ath0 inet static
    address 10.64.7.1
    netmask 255.255.255.224
    network 10.64.7.0
    broadcast 10.64.7.31
    wireless_essid madridwireless
    wireless_mode master
    wireless_nick estrella001
    wireless_channel 10
```

Si tu tarjeta fuera una pcmcia, ocurre que a menudo no se puede configurar como es debido desde el /etc/network/interfaces, y tendras que configurarla desde /etc/pcmcia/network.opts y desde /etc/pcmcia/wireless.opts, de esta manera:

```
ap:~# cat /etc/pcmcia/network.opts
*,*,*,*)
    INFO=""
    IF_PORT=""
    BOOTP="n"
    DHCP="n"
    IPADDR="10.10.10.1"
    NETMASK="255.255.255.0"
    NETWORK="10.10.10.0"
    BROADCAST="10.10.10.255"
    GATEWAY="10.10.10.253"
    DNS_1="212.106.192.250"
    DNS_2="212.106.192.251"
;;
esac
```

```

ap:~# cat /etc/pcmcia/wireless.opts
*,*,*,*)
    INFO="NOMBRE-DEL-NODO"
    ESSID="madrdiwireless"
    MODE="Master"
    RATE="auto"
#    KEY="s:84d70"
    ;;
esac

```

Tiene que quedaros algo parecido a eso.

Si alguien quiere montar WEP, es sencillo, solo tiene que poner una línea mas:

```
wireless_key "s:tu_clave"
```

Quiero recordar, que el WEP lleva cifrado de 40, o 104 bits, porque a los 64 o 128, le tenemos que quitar la parte del vector de inicialización(1), así que la clave "s:tu_clave" ha de ser de 5 letras ("s:cinco") para encriptación de 64 (40bits), o de 11 letras ("s:once_letras") para la encriptación de 128. Si queremos nuestra clave en exadecimal, y no en ascii, solo tendremos que omitir "s:" y escribir caracteres exadecimales (desde 0 hasta F)

(1) La clave de encriptación que utiliza WEP con el protocolo RC4 se genera en base a concatenar dos cosas: la clave introducida por el usuario (o el resultado de un proceso concreto si lo que introduce el usuario es una passphrase) y el vector de inicialización. El vector de inicialización es un simple contador que se incrementa con cada trama emitida, y que también figura en claro en la cabecera de cualquier trama encriptada con WEP.

Aunque el estándar 802.11 define que el vector de inicialización puede incrementarse o no, afortunadamente los fabricantes sí que lo hacen porque de otro modo sería aún más fácil reventar el WEP. Gran parte de los ataques que existen en la actualidad sobre WEP se basan precisamente en la inclusión del vector de inicialización dentro de la propia clave antes de aplicar el RC4, además del hecho que representa disponer de un espacio tan reducido (2^{24} bits para los distintos vectores de inicialización que pueden usarse.

(Gracias ed0 por esta aclaración)

Importante: esta tarjeta, soporta WPA, pero aun no me ha dado tiempo a meterme con ello, para la configuración WPA y el modo TURBO de estas tarjetas, tendréis que esperar un poco.

El modo turbo se basa en la tecnología de ensanchamiento de espectro, y consigue duplicar las velocidades; el tema de esto, es que aun esa en proceso, y el driver no lo soporta, cuando lo soporte, actualizaré el manual, y e incluiré el WPA.

Aquí vemos, como el driver si da soporte, pero aun no funciona como es debido el modo "turbo"

```

ap:~# iwpriv ath0
ath0      Available private ioctl :
          setparam (8BE0) : set   2 int  & get   0
          getparam (8BE1) : set   1 int  & get   1 int
          (8BE0) : set   1 int  & get   0
          (8BE1) : set   0      & get   1 int
          turbo (1) : set   1 int  & get   0
          get_turbo (1) : set   0      & get   1 int
          mode (2) : set   1 int  & get   0
          get_mode (2) : set   0      & get   1 int
          reset (3) : set   1 int  & get   0

```

Volviendo a la config:

Una vez cambiado el /etc/network/interfaces, hacemos un

```
/etc/init.d/networking restart
```

Y observamos con el ifconfig e iwconfig, como se ha levantado la iface ath0 como es debido.

```
ap:~# ifconfig ath0
ath0      Link encap:Ethernet  HWaddr 00:90:D1:08:0E:54
          inet addr:10.64.7.1  Bcast:10.64.7.31  Mask:255.255.255.224
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:22 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:0 (0.0 b)  TX bytes:3422 (3.3 KiB)
          Interrupt:10 Base address:0x6400
```

Y comprobamos el mode master en:

```
ap:~# iwconfig ath0
ath0      IEEE 802.11-b  ESSID:"madridwireless"  Nickname:"estrella001"
          Mode:Master  Frequency:2.457GHz  Access Point: 00:90:D1:08:0E:54
          Bit Rate:11Mb/s  Tx-Power:-8 dBm  Sensitivity=1/3
          Retry min limit:8  RTS thr:off  Fragment thr:off
          Encryption key:off
          Power Management:off
          Link Quality:0  Signal level:0  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
```

Con eso ya tendrías las ifaces configuradas

Este es el momento de traer un portátil, y hacer una prueba, si todo ha ido bien, debéis tener en este momento un AP muy bonito ;)

Si algo falla, mirad en que habéis fallado, pues este manual esta muy testado, de todos modos, mi correo también esta para las consultas. (bartlevi83@madridwireless.net)

DCHP

Ahora tenemos un AP que es muy bonito, pero que no sirve apenas para nada, porque no esta compartiendo conexión, ni rutando nada en absoluto, y además, ni siquiera nos da IP.

Empezaremos por ahí, configuraremos el dhcpd.

El archivo de conf del dhcp es /etc/dhcpd.conf

Este es mi archivo de configuración, por supuesto, no es el mismo que el vuestro, tendréis que cambiar las ips a lustra medida

```
ap:~# cat /etc/dhcpd.conf
        subnet 10.64.7.0 netmask 255.255.255.224 {
            # ---default gateway
            option routers                10.64.7.1;
            option subnet-mask            255.255.255.224;

            option domain-name-servers    212.106.192.250;
```

```

        range dynamic-bootp    10.64.7.2 10.64.7.30;
        default-lease-time     3600;
        max-lease-time         7200;
    }

    subnet 192.168.0.0 netmask 255.255.255.0 {
    }

```

El archivo es sencillísimo, no queda la opción de la duda. Sustituís los parámetros de la subred y demás por los vuestros.

Es importante comentar, que todas las subredes definidas en vuestra maquina, TIENEN que ser declaradas en el este archivo, es decir, si tienes un tunel, deberás declarar su interface, pero sin configurar nada, de este modo:

```
Subnet 172.64.*.* netmask 255.255.255.252 { }
```

Una línea de ese estilo por cada interface.

Ahora hacemos

```
ap:~# /etc/init.d/dhcp restart
Stopping DHCP server: dhcp.
Starting DHCP server: dhcp.

```

Y tendría que reiniciarse el servidor dhcp

Este seria otro buen momento para probar el AP, con el dhcp, lo veréis muy fácil con el wavemon, que es una herramienta para comprobar los link quality con el AP

Ya vamos teniendo un servidor en condiciones.

RUTADO DE INFORMACIÓN

Nos queda la parte de compartir Internet, tenemos dos formas de hacerlo, con NAT, o bridging, por supuesto solo hace falta una de ellas.

NAT

Lo vamos a hacer a través de iptables, es muy sencillo, introducimos estas dos reglas:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE

```

Una vez escrito esto, probamos con el portátil, el típico ping a google.com, y si funciona, felicidades, ya tenéis un AP con MadWifi ;)

Estas reglas iptables, no se conservan, así que tendremos que configurar el pc para arrancar con ellas.

Seguramente haya una forma mas ortodoxa, pero como yo soy un profano de las iptables (y de muchas otras cosas) y no tengo tiempo para ponerme a aprender, que no por falta de ganas, pues meto las reglas en un "script" que carga al inicio.

De esta forma:

```
vi /etc/init.d/iptables-rules
```

Dentro de ese archivo, metemos las dos líneas.

```
#!/bin/sh
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

Para que los cargue al arranque hacemos lo siguiente:

```
cp iptables-rules /etc/init.d/
update-rc.d -n iptables-rules defaults
```

Si todo va bien, haremos lo mismo, pero sin `-n`

```
update-rc.d iptables-rules defaults
```

Este script que hemos creado, no tiene permisos de ejecución mas que para el usuario que lo creo, así que tendremos que hacer un:

```
chmod 755 /etc/init.d/iptables-rules
```

Ahora ya tenemos un AP que nos hace NAT.

BRIDGING

Si queremos hacer bridging, en vez de NAT, pues tendremos que configurar un bridge, puedes arrancarlo como antes; con un script, o directamente del `/etc/network/interfaces` pero a mi no me funciono bien lo de configurarlo en interfaces en mi portátil, así que prueba primero con interfaces, y si no funciona, no te comas la cabeza, tira de script, y todos tan felices.

Esto es lo que tienes que meter en `/etc/network/interfaces`

```
auto br0
iface br0 inet static
    address 10.64.7.1
    netmask 255.255.255.224
    network 10.64.7.0
    gateway 10.64.7.2
    bridge_ports eth0 ath0
    bridge_stp off
    bridge_maxwait 5
```

(Son unas ips aleatorias, por supuesto, tienes que poner las tuyas)

Si prefieres la opción del script, hacemos igual que antes con el NAT:

La formula es la misma:

```
vi /etc/init.d/bridge
```

y metemos esto en el script

```
#!/bin/sh
brctl addbr br0
brctl addif br0 eth0
brctl addif br0 ath0
ifconfig eth0 0.0.0.0
ifconfig ath0 0.0.0.0
ifconfig br0 la ip_que_quieras up
```

Lo mismo de:

```
cp bridge /etc/init.d/
```

```
update-rc.d -n bridge defaults
```

y si va bien:

```
update-rc.d bridge defaults
```

E igual que antes, daremos permisos de ejecución:

```
chmod 755 /etc/init.d/bridge
```

Elige la que quieras de las dos opciones, las dos son validas, tanto NAT, como bridge.

CONCLUSIÓN

Con esto ya tenemos nuestro AP funcionando, la parte de la securización es cosa tuya, aquí solo expongo lo básico de lo básico para que funcione un AP.

Puedes montar servicios como servidor DNS, o todos los que quieras, por encima de este AP.

Si esas dentro de una asociación, tipo madridwireless, además, tendrás que configurar los demonios de rutado del zebra, para pasar las tablas de rutas, y montar túneles contra los otros nodos, si tienes la suerte de poder montar un enlace directo con otro nodo cerca de el tuyo, no necesitaras estos túneles over-inet, con túneles ip-sec, todo ira mejor

Aquí acabamos el documento de configuración, espero que haya ayudado al mayor numero de gente posible, y que fomente la formación de nuevos nodos para las MAN, que creo son unos proyectos que de veras tienen futuro.

AGRADECIMIENTOS

Este documento esta hecho recopilando información de distintos sitios, en su mayoría de las listas de correo de madridwireless, y parte del README de Madwifi, como es normal.

Quiero dar las gracias a DMescal, por aguantar tantos mails de preguntas que siempre me ha contestado de muy buena gana, y a Simón de MadridWireless también, por su ayuda con las iptables y demás historias del SO, y en general a toda la gente de las listas de correo de MadridWireless y Kaslab, que siempre están dispuestos a echar una mano, así como a Ed0, que me ha corregido unas partes del documento, que yo no podría haber explicado mejor que él.

Agradecimiento a Syvic, por sus correcciones.

Si alguien leyó el documento "Configuración Básica de nodo con HostAP en Debian" Notara unos "sutiles parecidos", efectivamente, hay muchas cosas que son iguales, efectivamente solo cambia la instalación del paquete, y poca cosa más. Pero espero que al salir con mas antelación que el otro manual, puesto que el manual sobre hostap, salio cuando la cosa estaba mas que trillada, pueda ayudar a mucha mas gente. De hecho, me decidí a hacer este manual, porque en google no he encontrado ningún manual de este tipo para instalar el driver madwifi. Espero le sea de utilidad a un gran numero de personas.

Si hubiera alguna errata o fallo en el manual, me gustaría por favor, me la reportarais al corre que pongo al pie de cada página.