



## Madwifi on Debian Sarge

### Introduzione

Le schede Wireless supportate da questo driver di primo acchito possono creare qualche problema di funzionamento, ma sono comunque da tempo supportate su piattaforma GNU/Linux.

Avevo già affrontato il loro utilizzo su Slackware, in questo articolo vediamo invece come installarle e configurarle sotto Debian Sarge.

Nel mio caso ho utilizzato una scheda Wireless PCI D-Link DWL-G520 in versione 802.11 b/g da 54 Mb, anche se attualmente sul mercato è presente quella da 108 Mbit.



Vediamo se il sistema la riconosce una volta installata.

```
# lspci -v
0000:00:0b.0 Ethernet controller: Atheros Communications, Inc. AR5212 802.11abg NIC (rev 01)
    Subsystem: D-Link System Inc DWL-G520 Wireless PCI Adapter
    Flags: bus master, medium devsel, latency 168, IRQ 11
    Memory at dbfe0000 (32-bit, non-prefetchable) [size=64K]
    Capabilities: [44] Power Management version 2
```

La scheda viene perfettamente riconosciuta. Come noto questo tipo di Chipset viene supportato dal Driver madwifi (Atheros).

- <http://sourceforge.net/projects/madwifi/>

Vediamo come farla riconoscere al sistema e configurarla per poter utilizzare la nostra Linux Box in una rete wireless.

Il nostro punto di partenza è una Debian Sarge con Kernel standard 2.6.8.

### Installazione

Usando il kernel standard 2.6.8 si può aggiungere il modulo necessario (ath\_pci) seguendo questa procedura. Ovviamente occorre aggiornare i repository:

```
# vi /etc/apt/sources.list
deb ftp://debian.marlow.dk/ sid madwifi
deb-src ftp://debian.marlow.dk/ sid madwifi
```

Installiamo i pacchetti necessari:



```
# apt-get update
# apt-get install kernel-source-2.6.8
```

Configuriamo il Kernel copiando il file config necessario nel directory dei sorgenti.

```
# cd /usr/src
# tar -xjvf kernel-source-2.6.8.tar.bz2
# cd kernel-source-2.6.8
# cp /boot/config-2.6.8-2-686 .config
# make-kpkg --append-to-version "-2-686" --revision 2.6.8 --config old configure
```

Adesso possiamo installare i sorgenti del driver ed i tool necessari alla sua gestione.

```
# apt-get install madwifi-source madwifi-tools
# apt-get source madwifi
# cd madwifi-1.7
# apt-get install fakeroot
# apt-get install cvs
# fakeroot dpkg-buildpackage
```

A questo punto vengono creati due file .deb che possono essere installati sul sistema:

```
# cd ..
# dpkg -i madwifi-source_20050212_all.deb
# dpkg -i madwifi-tools_20050212_i386.deb
```

Non resta che scompattare i sorgenti del modulo e compilarlo nel kernel, sempre seguendo la Debian way:

```
# cd /usr/src/
# tar -xzvf madwifi.tar.gz
# cd kernel-source-2.6.8
# fakeroot make-kpkg --append-to-version "-2-686" --revision 2.6.8 --added-modules madwifi
modules_image
# dpkg -i /usr/src/madwifi-module-2.6.8-2-686_20050212-1-one+2.6.8_i386.deb
# update-modules
```

Adesso possiamo provare a caricare il modulo necessario a far funzionare la scheda:

```
# modprobe ath_pci
```

Vediamo se è stata riconosciuta correttamente dal sistema:

```
# dmesg
ath_hal: no version for "struct_module" found: kernel tainted.
ath_hal: module license 'Proprietary' taints kernel.
ath_hal: 0.9.14.9 (AR5210, AR5211, AR5212, RF5111, RF5112, RF2413)
wlan: 0.8.6.0 (EXPERIMENTAL)
ath_rate_onoe: 1.0
ath_pci: 0.9.6.0 (EXPERIMENTAL)
ACPI: PCI interrupt 0000:00:0b.0[A] -> GSI 11 (level, low) -> IRQ 11
Build date: Apr 17 2006
Debugging version (IEEE80211)
ath0: 11b rates: 1Mbps 2Mbps 5.5Mbps 11Mbps
```



```
ath0: 11g rates: 1Mbps 2Mbps 5.5Mbps 11Mbps 6Mbps 9Mbps 12Mbps 18Mbps 24Mbps 36Mbps 48Mbps 54Mbps
ath0: H/W encryption support: WEP AES AES_CCM TKIP
ath0: mac 5.6 phy 4.1 radio 1.7
ath0: Use hw queue 1 for WME_AC_BE traffic
ath0: Use hw queue 0 for WME_AC_BK traffic
ath0: Use hw queue 2 for WME_AC_VI traffic
ath0: Use hw queue 3 for WME_AC_VO traffic
ath0: Use hw queue 8 for CAB traffic
ath0: Use hw queue 9 for beacons
Debugging version (ATH)
ath0: Atheros 5212: mem=0xdbfe0000, irq=11
```

La scheda è stata perfettamente riconosciuta ed il device creato è ath0.

```
# iwconfig ath0
```

```
ath0 IEEE 802.11g ESSID:"WiFi Netlink" Nickname:"Neo CLient"
Mode:Managed Frequency:2.422 GHz Access Point: 00:09:5B:40:85:DE
Bit Rate:11 Mb/s Tx-Power:18 dBm Sensitivity=0/3
Retry:off RTS thr:off Fragment thr:off
Encryption key:0596-9112-70 Security mode:open
Power Management:off
Link Quality=44/94 Signal level=-51 dBm Noise level=-95 dBm
Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0
Tx excessive retries:0 Invalid misc:0 Missed beacon:0
```

Proviamo a configurarlo utilizzando i wireless tools:

```
# iwconfig ath0 mode Managed
# iwconfig ath0 essid "WiFi LAN"
# iwconfig ath0 nickname "Dlink Client"
# iwconfig ath0 key open 333444333
```

A questo punto non resta che assegnargli un indirizzo IP:

```
# ifconfig ath0 192.168.0.100 up
```

Proviamo a pingare il nostro Access Point

```
# ping 192.168.0.10
PING 192.168.4.10 (192.168.4.10) 56(84) bytes of data.
64 bytes from 192.168.4.10: icmp_seq=1 ttl=64 time=3.54 ms
64 bytes from 192.168.4.10: icmp_seq=2 ttl=64 time=2.22 ms
64 bytes from 192.168.4.10: icmp_seq=3 ttl=64 time=12.1 ms
```

## Note sull'impostazione della chiave WEP

La sicurezza delle LAN wireless è garantita di un protocollo di base il WEP (wired equivalent protocol) che prevede una procedura di autenticazione associata ad una procedura di crittografia. In pratica attraverso il WEP viene utilizzata una chiave segreta che deve essere scambiata tra le parti (AP e client) per poter cifrare e decifrare i frame trasmessi.

E' possibile specificare una chiave personale che deve essere nota a priori oppure può far generare la chiave in automatico. La chiave può essere lunga 40 bit oppure 104 bit, si appoggia su un algoritmo RC4 ed è di tipo simmetrico.



La chiave a 128 bit rende più complesse le operazioni necessarie a ricostruire la chiave sulla base di un certo numero di pacchetti sniffati, ma può rallentare le connessioni.

Ovviamente la chiave scelta deve esserci fornita o comunque deve esser al stessa di quella del nostro Access Point.

Attenzione comunque, le chiavi WEP possono venire agevolmente decifrate, è sufficiente sniffare una grossa quantità di traffico Wireless per poi procedere alla decrittazione della chiave.

Se i vostri dispositivi le supportano e meglio rivolgersi a moderni sistemi di crittografia del traffico (WPA) oppure a sistemi di autenticazione combinati, come tunnel cifrati o radium.

Sul nostro sistema utilizziamo (almeno) la classica autenticazione WEP:

```
# iwconfig wlan0 key s:password → occorre conoscere la password impostata sull'HostAP
```

oppure restringendo il campo utilizzando una chiave su entrambi i sistemi

```
# iwconfig wlan0 key restricted 0123456789

wlan0 IEEE 802.11b ESSID:"WiFi Netlink" Nickname:"Neo Client"
Mode:Ad-Hoc Frequency:2.462GHz Cell: 02:09:C9:89:85:DD
Bit Rate:11Mb/s Sensitivity=1/3
Retry min limit:8 RTS thr:off Fragment thr:off
Encryption key:0123-4567-89 Security mode:restricted
Power Management:off
Link Quality:84/92 Signal level:-46 dBm Noise level:-99 dBm
Rx invalid nwid:0 Rx invalid crypt:11 Rx invalid frag:0
Tx excessive retries:2 Invalid misc:42 Missed beacon:0
```

Ricapitolando si può impostare:

una chiave numerica (esadecimale) a 40 bits

```
# iwconfig wlan0 key 0123456789
```

una chiave numerica (esadecimale) a 104 bits

```
# iwconfig wlan0 key 1234-1234-5678-5678-1234-1234-99
```

una chiave testuale (stringa ASCII) come una sorta di password, utilizzando il prefisso s:

```
# iwconfig wlan0 key s:passwordwlan
```

Inoltre si possono specificare fino ad un massimo di 4 chiavi che possono essere poi selezionate tramite iwconfig, specificando il numero della chiave:

```
iwconfig wlan0 key [1] 0123456789
iwconfig wlan0 key [2] 5656852123
iwconfig wlan0 key [3] 1234-1234-5678-5678-1234-1234-99
iwconfig wlan0 key [4] s:passwordwlan
```

Con il comando iwlist key possiamo vedere lo stato delle nostre WEP key

```
# iwlist wlan0 key
wlan0 2 key sizes : 40, 104bits
4 keys available :
[1]: 0123-4567-89 (40 bits)
[2]: 5656-8521-23 (40 bits)
[3]: 1234-1234-5678-5678-1234-1234-99 (104 bits)
[4]: 7061-7373-776F-7264-0000-0000-00 (104 bits)
```



```
Current Transmit Key: [1]  
Security mode:restricted
```

a questo punto si può assegnare la WEP key scelta semplicemente con il comando

```
# iwconfig wlan0 key [3]
```

## Note

Questa scheda PCI si è dimostrata davvero valida è perfettamente funzionante in una rete Wireless standard 802.11b, dialogando con dispositivi Wireless diversi.

Anche con chiavi crittografiche da 128 bit non ha presentato problemi di sorta, a differenza della KRAUN PCI Wireless con driver RaLink, che non era invece riuscita a lavorare con la crittografia WEP abilitata.

## Risorse

MADWiFi 'First Time User' HOWTO

<http://madwifi.org/wiki/UserDocs/FirstTimeHowTo>

Multiband Atheros Driver for WiFi

<http://www.thinkwiki.org/wiki/Madwifi>

ISG.EE Debian Packages

<http://debian.isg.ee.ethz.ch/public/>

Install per SARGE DI MADWIFI

<http://questier.com/howto.html>

Multiband Atheros Driver for WiFi (madwifi) package for Debian

<http://www.marlow.dk/site.php/tech/madwifi>

Getting MADWiFi

<http://madwifi.org/wiki/UserDocs/GettingMadwifi>

*Doc: madwifi\_sarge.pdf*

**Dott. Paolo PAVAN [Netlink Sas]**- [pavan@netlink.it](mailto:pavan@netlink.it)

**Data:** Aprile 2006

## Note finali

- Il presente documento è a semplice scopo divulgativo
- L'autore non si assume la responsabilità di eventuali danni diretti o indiretti derivanti dall'uso dei programmi, o dall'applicazione delle configurazioni menzionate nel seguente articolo
- I marchi citati sono di proprietà dei rispettivi proprietari e sono stati utilizzati solo a scopo didattico o divulgativo.
- I contenuti di questo documento vengono rilasciati sotto Licenza Creative Commons.  
<http://creativecommons.org/licenses/by-nc-sa/2.0/>
- Sono possibili errori o imprecisioni, segnalatemele a [pavan@netlink.it](mailto:pavan@netlink.it)
- Chi volesse integrare il presente documento, può scrivere a [pavan@netlink.it](mailto:pavan@netlink.it).