Linux Wireless LAN Howto

Jean Tourrilhes

25 July 07

*Linux & Wireless LANs : Un*x, with no string attached...*

1 Introduction

This document will explore the magical world of **Wireless LANs** and **Linux**. Wireless LAN is not a very widespread and well known technology, even in the Linux world, so we will try to gather here most of the available information. Despite the fact that it is very similar to common networking technologies, it is significantly different to justify this specific document covering the subject.

1.1 What is a Wireless LAN ?

It's a **networking** technology allowing the connection of computers without any wires and cables (apart from the mains), mostly using **radio** technology (and sometime **infrared**). It's called LAN (Local Area Network) because the range targeted is small (within an office, a building, a store, a small campus, a house...). This technology is slowly growing (I should say maturing), and despite a general lack of interest, Linux is able to take advantage of some of the wireless networks available.

1.2 Content of this document

My first task is to talk a bit about the different Wireless LANs options under Linux. What the **products** on the market are, their compatibility with Linux and where to find the necessary bits and pieces to make them work. This should help you to make your mind on the product of your dreams.

Once you've picked a Wireless LAN, you will have to live with it. The next chapter go through the main **differences** of Wireless LAN compared to other networking technologies. This includes the main steps of the installation and usage considerations.

Then, we will have a nice overview of the **Wireless Extensions**. The Wireless Extensions is a new standard interface to configure Wireless LAN devices and get wireless specific statistics from them. Of course, this is a Linux exclusivity !

At this point, you will find a long and dense section, talking mostly of the different **technologies** used in Wireless LANs and other boring related stuff. It is quite safe to skip that one.

1.3 Target and Assumptions

The main goal of this document is to reduce the traffic of unanswered questions related to wireless in the Linux newsgroups and mailing lists (and in my

mailbox). After that, you should have no more arguments for asking foolish questions around (but I know you will do anyway).

I hope that this document will help people to make the most of their Wireless LAN under a competent operating system and understand what is in the box. If I could convince people to give it a try, it would make me happy.

This document act mostly as a complement to the exhaustive documentation existing for Linux. Because of that, I might not explain every details of everything and target already quite knowledgeable people. Don't worry, there is a section on how to improve your culture at the beginning of the *section 3*.

1.4 Legal stuff

Strange world where everybody has to protect himself from sharks, lawyers and crazy people :

Any information in this document is purely fictious and any resemblance to real hardware, software or driver is purely coincidental...

I mean, if because you read this document your hardware burn, you get fired from your job or anything else bad happen, I'm not responsible, it can't be my fault, so please use your own brain. Writing this kind of documents is not part of my job at HP, so I don't expect them to claim any responsibility for its content.

Any brand mentioned in this document is trademark of its respective owner. For example Linux is a trademark of Linus Torvalds.

Then, this is my document, written by me (Jean Tourrilhes), therefore I own its copyright. So don't remove my name (and copyright notice) and pretend that you wrote it yourself. In matter of copy, distribution and modification, you should ask me politely and use common sense.

Having said that, this document is also licensed under the terms of the **Linux Documentation Project Copying License**.

1.5 This document

This document is only available in the format that are convenient to me (acrobat/pdf, html). It might be updated in the future (if I feel like it and if I have some time). I guess that it is pretty safe to assume that it will still be available for the time to come at these **web** addresses :

http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Wireless.html

http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Wavelan.html

I may be reached at the following **e-mail** address :

jt@hpl.hp.com

Constructive comments and interesting information are welcomed. I hope that you will help me to keep this document up to date and improve its content.

Comments about my english and my style will be answered in french (because I still curse better in french than in italian). Flames and spam will be processed through a Rayleig Fading channel with a -120 dB attenuation in order to reduce the noise :-)

2 The devices, the drivers - pre-802.11 and early 802.11

The following four sections describes the most common Wireless LAN products available on the market and their **compatibility** with Linux. Except in a few case, you need a **driver** to interface you wireless network device to the Linux kernel. The availability of a driver is as usual your main concern, especially with wireless devices because fewer people are using such hardware, and the wireless drivers are complex, so few of them are willing to develop, debug and maintain such a piece of code.

I will make a short description of each product and will mainly focus on the drivers. For each driver, I will list its *status* (stable, buggy...), the *maintainer*, the *version*, how to *get it* and the main *features*. If you hear about something new or if you have developed yourself a driver, please notify me.

Because of the large number of drivers, it has been divided in four sections, the first cover really old devices, the second 802.11b devices, the third one devices faster than 802.11b and the fourth one other stuff.

This section list devices predating the IEEE 802.11 standard or based on the early IEEE 802.11 standards (see *section 8*), either 802.11 Frequency Hopping or 802.11 Direct Sequence. Those devices are mostly obsolete and discontinued, so most of you want to skip directly to the next section (see *section 3*).

2.1 Lucent Wavelan & DEC RoamAbout DS

Driver status :	stable
Driver name :	ISA : wavelan.o
	Pcmcia : wavelan_cs.o
Version :	v19 (20/4/99), v20 (29/7/99) or v23 (10/10/00)
Where :	ISA : Linux kernel (2.0.37, 2.2.11 & 2.3.15)
	Pcmcia : Pcmcia package (3.0.11)
Creators :	Bruce Janson (ISA) and Anthony D. Joseph (Pcmcia)
Maintainer :	Jean Tourrilhes <jt@hpl.hp.com>
Web page :	http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Wavelan.html
Mailing list :	http://lists.samba.org/pipermail/wireless/
Documentation :	man pages, headers
Configuration :	Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Ad-Hoc
Security :	DES (only for hardware with the DES option)
Scanning :	No
Monitor :	Yes (nwid off + tcpdump)
Multi-devices :	isa : up to 4
	pcmcia : yes

Interoperability :	proprietary protocol, interoperate with Windows
Other features :	module, hardware multicast, Wireless Extensions, SMP
Non implemented :	roaming
Bugs :	see release notes on web page :-(
License :	GPL & OpenSource
Vendor web pages :	http://www.wavelan.com/
	http://www.networks.digital.com/dr/wireless/
	http://www.cabletron.com/dnpg/dr/npg/lanfm-mn.html

2.1.1 The device

The Wavelan has been around for quite a while now, and this product is now **discontinued** (and replaced by the Wavelan IEEE/Orinoco - see *section 3.1*). The Wavelan is a radio LAN, using the 900 MHz or 2.4 GHz ISM band (Direct Sequence). It is built by Lucent, formerly AT&T, formerly NCR, and there is a few OEM version (for example the DEC RoamAbout DS). The Wavelan comes in two flavours, an ISA card and a PCMCIA card (plus the access point).

The Wavelan appears to the PC as a standard network card and interfaces naturally with the networking stack. The configuration includes setting the frequency (10 different channels), Network ID (16 bits). Hardware encryption is optional (DES or AES - 64 bits key).

This product is built around a standard Ethernet controller (that may be found in some 3Com and Intel Ethernet cards), and the Ethernet physical layer is replaced by a radio modem. The ISA and Pcmcia cards share the same basic architecture, have the same modem, but have different Ethernet Controllers and bus interfaces (the pcmcia has only one transmit buffer). Because the Wavelan doesn't use a specific radio MAC (no MAC level retransmissions for example), it uses very efficiently the bandwidth, but is more sensitive to packet loss and collisions.

There is two versions of the modem, a 900 MHz and a 2.4 GHz version. Revision 2 of the 2.4 GHz modem allows the user to set the frequency (from a set of predefined channels - the availability of each channel depend on the regulation). The Wavelan is Direct Sequence Spread Spectrum (11 chips encoding), using a 2 Mb/s signalling rate (using effectively 22 MHz of bandwidth) and diversity antennas.

2.1.2 The driver

The ISA driver has also been around for quite a while now in the kernel and is pretty stable. The last set of modifications were to solve a few remaining small problems and add Wireless Extensions and some other features, so the driver is fairly complete now. The only things remaining to do is the implementation of the roaming protocol (but it might come, if I'm not too lazy...).

The Pcmcia driver has caught up with the ISA one to offer the same level of functionality and reliability. The only difference are the pcmcia specific functions (auto loading, auto unloading, crude power saving).

The latest releases of both drivers (v23) adds SMP support.

The drivers use the card EEprom to save the configuration changes for subsequent reboots. Wireless Extensions let you configure the NWID, the frequency, the sensitivity and the encryption key (optional). Statistics include the signal quality, signal level, noise level and the count of packet received with an invalid NWID (see Wavelan documentation). Private Wireless Extensions include the setting of the quality threshold.

2.2 Netwave AirSurfer & Xircom Netwave

Driver status :	fairly stable
Driver name :	netwave_cs.o
Version :	v 0.4.1
Where :	Linux kernel (2.4.0)
	Pcmcia package (2.9.12)
Maintainers :	John Markus Bjørndalen <johnm@cs.uit.no>
	Dag Brattli <dagb@cs.uit.no>
Web pages :	http://www.cs.uit.no/~johnm/
	http://www.cs.uit.no/~dagb/
Documentation :	man page
Configuration :	Module parameters & Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	8 bit scrambling
Scanning :	No
Monitor :	No
Multi-devices :	yes (except for module parameters setting)
Interoperability :	proprietary protocol, interoperate with Windows
Other features :	-
Non implemented :	hardware multicast, multiple transmit buffers
Bugs :	-
License :	OpenSource
Vendor web page :	http://www.netwave-wireless.com/

2.2.1 The device

The Netwave was also a quite common product, but nowadays this product is **discontinued**. This is a radio LAN operating in the 2.4 GHz ISM band. It was built by Netwave Technologies, formerly part of Xircom. The Netwave is Pcmcia only, and comes in a small form factor (everything is included on the Pcmcia card !).

The Netwave use a specific MAC protocol designed for radio (a pre 802.11 protocol, with fancy stuff such as RTS/CTS, virtual carrier sense and

fragmentation). It uses a 9 bits domain (Network ID), the highest bit of it used for the type of network (set for access point operation and unset for ad-hoc operation). The Netwave uses also a 16 bits scrambling key (encryption). The Modem offers a 1 Mb/s signalling rate and frequency hopping (100 ms hop period). On the bad side, the Netwave has no antenna diversity and a high overhead.

Note that the Netwave AirSurfer plus is a very different beast (see below).

2.2.2 The driver

The original author of the driver (*John*) has made a very good job for debugging it, and his good friend (*Dag*) has joined the project, and is fixing the remaining bugs and adding new features. The driver is quite simple and don't implement yet the full Wireless Extensions. The driver uses only one transmit buffer, which lower slightly the performance. The device configuration includes the domain and the scrambling key which can be set through Wireless Extensions or as module parameters (need to be set in /etc/pcmcia/config.opts - don't forget to restart *cardmgr* after a change).

It seems that the Netwave is quite picky with some pcmcia sockets and you might need to choose carefully the interrupt (try different ones) and set the memory speed correctly. In some cases, under high load (big ftp), the transmission sometime get stuck (I guess that some interrupt are lost) and the driver has to reset the card (you won't notice it, it just decreases the performance).

2.3 Netwave AirSurfer plus

Driver status :	fairly stable
Driver name :	asplus_cs.o
Version :	1.0.2
Where :	http://ipoint.vlsi.uiuc.edu/wireless/asplus.html ftp://projects.sourceforge.net/pub/pcmcia-cs/contrib/
Maintainer :	Jay Moorman <jrmoorma@uiuc.edu>
Documentation :	Readme, man page
Configuration :	Module parameters, Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	8 bit scrambling
Scanning :	No
Monitor :	No
Multi-devices :	yes (except for module parameters setting)
Interoperability :	proprietary protocol (same as Netwave), interoperate with Windows
Other features :	-
Non implemented :	802.11 mode, hardware multicast, multiple transmit buffers
Bugs :	-

License : OpenSource

Vendor web page : http://www.netwave-wireless.com/

2.3.1 The device

The Netwave AirSurfer plus is the second generation of Netwave card (this product is now also **discontinued**), still operate in the 2.4 GHz ISM band and is as well a small Pcmcia card. Netwave Technologies has now been acquired by BayNetwork, now a part of Nortel. The BayStack 650 is the new name of the hardware.

The AirSurfer plus has two modes of operation, compatible with the old generation of Netwave, or 802.11 compliant. The hardware is based on an AMD core, and a 1 Mb/s frequency hopping modem.

2.3.2 The driver

Jay took the code of the original Netwave driver and modified it to support the new AirSurfer plus, keeping most of the features with it. So, you still have the Wireless Extensions, and modules parameters (in /etc/pcmcia/config.opts).

The current driver support the AirSurfer plus only in Netwave compatible mode, and doesn't support the AirSurfer plus with the 802.11 firmware.

2.4 Harris Prism based cards : BayStack 660, ZoomAir, YDI and other...

Driver status :	stable
Driver name :	wlan_cs.o
Version :	0.2.7, 0.2.7a, 0.3.1.1 (beta version) and 0.3.4 (beta version)
Where :	http://www.linux-wlan.com/linux-wlan/
	http://www.astro.umd.edu/~teuben/linux/wireless.html
	http://www.cs.berkeley.edu/~jhill/linuxwlan/
	http://www.spesh.com/danny/wlan
Maintainers :	Mark S. Mathews <mark@linux-wlan.com>
	Peter Teuben <teuben@astro.umd.edu>
	Jason Hill <jhill@cs.berkeley.edu>
	Danny O'Brien <danny@spesh.com>
Mailing list :	http://www.linux-wlan.com/linux-wlan/
Documentation :	Readme, man page
FAQ :	http://linux.grmbl.be/wlan/
Configuration :	Module parameters & configuration tool
Statistics :	Statistic tool
Modes :	Managed, Ad-Hoc
Security :	?
Scanning :	?
Monitor :	Yes

Multi-devices :	yes
Interoperability :	802.11-DS, interoperate with Windows
Other features :	Quite exhaustive 802.11 support
Non implemented :	WEP
Bugs :	-
License :	MPL
Vendor web pages :	http://www.netwave-wireless.com/
	http://www.zoomtel.com/zoomair/
	http://www.ydi.com
	http://www.intalk.com/
	http://www.dbtel.com.tw/english.html
	http://www.gemtek.com.tw/
	http://www.sem.samsung.co.kr/
	http://www.intersil.com/prism/
	http://www.amd.com/products/npd/npd.html

2.4.1 The device

The Harris Prism chipset and the AMD AM930 controller are some highly integrated parts designed to ease the process of building 802.11 products. Harris has done a Pcmcia reference design based on their chipset and the AMD core, which explain the high number of vendors building variants of this card (the Harris website has a longer list than mine ;-). A special mention for YDI (Young Design Inc) which openly support Linux (see below). Note that all those products are now **discontinued** (and replaced with PrismII design - see *section 3.6*)...

The AMD core integrates a generic microcontroller and the hardware baseband (ASIC) to do the time critical functions of 802.11. AMD has developed the 802.11 firmware with all the usual basic 802.11 features (MAC level ACK, RTS/ CTS, Fragmentation...). The Prism chipset is a 2.4 GHz Direct Sequence modem offering 1 and 2 Mb/s. The Prism chipset can also be extended to supports the new 802.11 HR standard, with 5.5 and 11 Mb/s bit-rate (either MBOK or CCK modulation).

The Pcmcia cards are mostly similar from vendor to vendor. Some vendors offer ISA cards, and the Access Points are where vendors are making their difference (ZoomTelephonics uses a software AP on a PC, others have hardware AP). Each vendor also has to provide the high level 802.11 in their drivers (authentication, WEP, Roaming...), so those might be different (not that it does matter much under Linux).

The BayStack 650 and Netwave AirSurfer plus use the same AMD controller, but a different physical layer (Frequency Hopping), so are not compatible with this driver.

Harris has just become Intersil and released the Prism II chipset, successor of the PrismI chipset, this time including the MAC controller (so they won't use any

more the AMD part in their reference design). I'll detail it in the next section (see *section 3.6*).

2.4.2 The driver

Mark and the people at AVS have developed a full 802.11 driver for the Prism reference design card (AMD controller + Prism chipset), and this driver work for the many other implementations as well (see web page).

The driver is well written and very complete : it's currently the only driver where most of the higher layer 802.11 functionality is implemented. There is also many initialisation parameters and a tool to configure the card. Because the 802.11 standard is very complex, not everything is totally finished and a few features like WEP (RC4 40 bits encryption) are missing.

There is currently two branches maintained by *Mark*, 0.2.X which is stable and 0.3.X which is experimental.

Peter (with help from YDI) has created a alternative version of *Mark*'s package to add ISA support, fix a few bugs and with explicit support of cards from YDI. In the long run, those changes should find their way in *Mark*'s package...

Jason has created a version of the 0.3.1 beta driver with support for the BayStack 660, by porting bits from 0.2.6 (this allow support for both the BayStack 660 and infrastructure).

Danny has a patch to make the driver compile and work with kernel 2.4.X.

I believe that this driver doesn't support the BayStack 650 and Netwave AirSurfer plus cards (which don't use the Prism chipset but Frequency Hopping), but the changes for that might not be that hard to implement.

2.5 Z-Com LANEscape, ELSA MC2, Siemens I-Gate

Driver status :	?
Driver name :	wl24_cs.o
	wl3501_cs.o
Version :	1.3 (stable), 2.03 (unstable) and 1.53beta2 (Elsa full source)
Where :	http://developer.berlios.de/projects/mc2drv/
	Linux kernel (2.6.0-test2)
	http://www.boerde.de/~matthias/airnet/zcom/
	http://www.boerde.de/~tobias/
	http://www.elsa.de
	http://pcmcia-cs.sourceforge.net/ftp/contrib/
Maintainers :	Jörg Albert <joerg dot albert at gmx dot de>
	Arnaldo Carvalho de Melo <acme@conectiva.com.br>
	Gustavo Niemeyer <niemeyer@conectiva.com>
	Matthias Weingart <matthias@penthouse.boerde.de>
	Alfred Arnold <alfred.arnold@lancom-systems.de>

	Heiko Kirschke <Heiko.Kirschke@acm.org>
	Tobias Hintze <th@hbsn.de>
Documentation :	README file
Configuration :	Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	?
Scanning :	No
Monitor :	No
Multi-devices :	unknown
Interoperability :	802.11-DS (with firmware 2.0), interoperate with Windows
Other features :	-
Non implemented :	-
Bugs :	Must have the correct firmware revision (or driver version).
License :	Binary only (1.3, 2.03) or Open source (1.20), no license info
Vendor web page :	http://www.zcom.com.tw/
	http://www.elsa.de

2.5.1 The device

Z-Com is based in Taiwan, and the WL2400 family is based on the classic AMD+Prism design. The family includes the usual ISA and Pcmcia cards, the Access Point, and also a PC104 version (that's interesting)...

Z-Com claims that the WL2400 is firmware upgradable to 802.11, but I've been told that some old cards have an hardware bug preventing it. Anyway, the card has all the usual 802.11 features, and the modem is classsical Direct Sequence at 2.4 GHz, supporting 1 and 2 Mb/s. Those cards are based on the classical Prism design (AMD controller + Prism chipset - see *section 2.4*), but use a **firmware** written by Z-Com which is very different from the regular AMD firmware. This firmware has more features, is more performant, but has more bugs.

Z-Com also offers the XI family, which support 5.5. and 11 Mb/s (probably using a Prism II chipset). Those are not supported by this driver.

Elsa is a German company selling various hardware component and started to sell Z-Com cards as Elsa AirLancer MC2. Those cards were quite popular in Germany.

Elsa also sell a new AirLancer MC11 which has nothing to do with Z-Com and is the Wavelan-IEEE (see *section 3.1*).

2.5.2 The driver

The driver has been written by the manufacturer, and *Matthias* put it on its web site. The driver only contains the object files (no source) and seem to have been designed for kernel 1.3.X and working in 2.0.X kernels (but, as the driver interfaces in the kernel have changed since, this driver might not work in 2.2.X). The driver

only work with old firmware revisions (1.2/1.3), and doesn't work with the 802.11 compliant firmware (2.0).

Matthias seems to now have access to the driver source code and is investigating compatibility with 2.2.X and new firmware revisions.

Then, *Elsa* has released the full source code of this driver for their card, including configuration utility. *Elsa* has made the setup easier and seem to have also fixed a few bugs, because it is now working with kernel 2.2.X... This version works with 802.11 compliant firmware (2.0).

More recently, *Heiko* ported the version of the driver from Elsa to 2.4.X. This version works with the Pcmcia package (in the Pcmcia contrib directory) and with **firmware 2.0**. And then, *Tobias* has released a full source code version of this driver for kernel 2.4.X, likely based on the latest version on *Matthias* web page. This version is a patch to the kernel (on his web page) and works with **firmware 1.2/1.3**.

Jörg took over the maintenance of Elsa driver (for firmware 2.0) and did a partial rewrite of it. He fixed many bugs preventing interoperability with more modern 802.11b hardware, improved compatibility with kernel 2.2.X and 2.4.X, implemented Wireless Extensions and many other fixes and polishing.

Arnaldo and *Gustavo* took over the maintenance of the driver from *Matthias* (for firmware 1.2/1.3) and did a partial rewrite of it. They updated the driver to kernel 2.6.X, added full support for Wireless Extensions, and this driver is now included in Kernel 2.6.0-test2. This driver requires Wireless Extension v16 and will be soon ported to 2.4.23.

2.6 Proxim RangeLan2, Proxim Symphony, DEC RoamAbout FH, AMP Wireless, Intel AnyPoint and Compaq Symphony

Driver status :	stable
Driver name :	rlmod.o
Version :	1.7.1
Where :	http://www.komacke.com/distribution.html
Creator :	Paul Chinn <loomer@1000klub.com>
Maintainer :	Dave Koberstein <davek@komacke.com>
Mailing list :	http://www.komacke.com/archive/rl2-library/
Documentation :	Readme file
Configuration :	Specific tool, partial implementation of Wireless Extensions
Statistics :	none
Modes :	Managed, Ad-Hoc, Master
Security :	No
Scanning :	Yes (specific tool)
Monitor :	No
Multi-devices :	No (“insmod -o” multiple modules)

Interoperability :	proprietary protocol or HomeRF, interoperate with Windows
Other features :	Uses Proxim source code
Non implemented :	-
Bugs :	-
License :	Binary only for the core + OpenSource Linux wrapper
Vendor web pages :	http://www.proxim.com/
	http://www.wlif.com/
	http://www.homerf.org/
	http://www.networks.digital.com/dr/wireless/
	http://www.intel.com/anypoint/

2.6.1 The device

The **RangeLan2** is a classical product using the 2.4 GHz band, made by Proxim, a small californian company. The products are certified and sold in approximately 50 countries. The RangeLan2 is based on Proxim proprietary protocol, **OpenAir**, that Proxim is trying to push as an alternative to 802.11. Of course, you will find many OEM version (like the DEC and AMP versions). It comes as ISA cards, Pcmcia cards, design-in modules, and access point.

The RangeLan2 implements a specific MAC protocol designed for radio (OpenAir, another pre 802.11) implemented on a generic microcontroller. It uses a 4 bits domain, 4 bits channel and 4 bits subchannel, and also a station type (primary master, secondary, slave - this is used for network synchronisation). There is no encryption, instead it uses a technique called Security ID (which is a simple password used to derive the network ID). The OpenAir protocol is heavily based on RTS/CTS, offer a good robustness but some overhead. It offers as well a modulable contention window size, contention free access for the master, packet fragmentation and power saving.

The Modem uses frequency hopping, and 2 levels of modulations (2FSK/ 4FSK) : it runs a 1.6 Mb/s signalling rate for good channel condition (short to medium distances) and falls back to 0.8 Mb/s otherwise.

The **Symphony** line of product (home networking) offered by Proxim uses the MAC protocol of the RangeLan2 (OpenAir) with a lower cost radio, and the main difference is the software bundle and the price. On the other hand, the Proxim **RangeLan802** line is very different from OpenAir products, using the 802.11-FH protocol and a different interface, so the Linux driver won't work with it.

Recently, Proxim has released its first Symphony products compatible with the **HomeRF SWAP** standard. These are also sold as **Intel AnyPoint** and **Compaq Symphony-HRF**. The ISA, PCI and Pcmcia versions are still offered, and a USB version has been added. Those products use the same physical layer as the original Symphony, but the MAC protocol can either operate in OpenAir mode or SWAP mode. The main advantage of SWAP is the support for cordless telephony.

Proxim has also various **802.11-b** products, named *Harmony*, *Skyline* or *RangeLan-DS* which are PrismII cards (see *section 3.6*). Proxim has also released some **802.11-a** products which are Atheros cards (see *section 4.2*).

2.6.2 The driver

Dave uses the Proxim driver source code to build a library (distributed as object only), so we should expect a good quality code. *Paul* wrote the part to interface with the Linux kernel and *Dave* maintains it. He has written as well a small utility to set the configuration in the driver (through ioctl). The driver supports the Proxim Rangelan2, the Proxim Symphony, the DEC RoamAbout FH and the AMP Wireless products. The driver support both ISA PnP and Pcmcia cards, both with the RangeLan2 and Symphony labels...

Starting with version 1.7.0, the driver also support the SWAP protocol and SWAP compliant devices from Proxim, Intel and Compaq (in both OpenAir and SWAP mode). Both the driver and the configuration tools have been extended for this support. Also, some primitive support for USB hardware has been added.

What I like about this driver is that after all those years, *Dave* is still strongly supporting the driver, fixing bugs, adding new features and adding support for the newer cards. It's impressive to see such consistency and dedication...

2.7 Symbol Spectrum24 (FH)

Driver status :	Beta (Pcmcia only)
Driver name :	spectrum24_cs.o
Version :	Beta 4
Where :	http://sourceforge.net/projects/spectrum24
	ftp://projects.sourceforge.net/pub/pcmcia-cs/contrib/
Maintainer :	Lee John Keyser-Allen <frozbiz@hotmail.com>
Discussion forums :	http://sourceforge.net/forum/?group_id=11099
Documentation :	Readme file
Configuration :	module parameters, partial support of Wireless Extensions
Statistics :	None
Modes :	Managed, Ad-Hoc
Security :	No
Scanning :	No
Monitor :	No
Multi-devices :	-
Interoperability :	802.11-FH, interoperate with Windows
Other features :	Support of micro-AP, multicast, statistics...
Non implemented :	-
Bugs :	-
License :	GPL

Vendor web page : http://www.symbol.com/products/wireless/wireless.html

2.7.1 The device

Symbol is one of the other major player for Frequency Hopping devices in the 2.4 GHz band and has been selling its Spectrum24 line of products for ages. Symbol sells mostly to vertical market (in their bar-code readers, in warehouses, in supermarket), so their products are not usually found in retailers. The Spectrum24 family include an Access Point, a ISA card, a Pcmcia card and a Pcmcia card with micro-AP functionality. However, the main strength of Symbol is their “all-in-one” products, including a Palm or a WinCE device with a bar code reader and a 802.11 card, all neatly integrated.

The Spectrum24 products were designed from the start to be compliant with the 802.11 standard, way before the standard was eventually adopted. The first generation (1 Mb/s only) was compatible and interoperable with other 802.11 products (but not compliant), and the second generation of Spectrum24 (1 and 2 Mb/s) is officially 802.11 compliant.

Symbol is also very active in developing Voice over IP solutions for their wireless LANs, and that’s why they are also selling some Spectrum24 phones. They are using the H.323 codec, compression and call setup (raw 64 kb/s, compressed 10 times) and a 30 ms packet rate (but I fail to see what they have done to overcome overhead and latency issues at the MAC level).

The MAC has all the usual features of the 802.11 standard, like MAC level retransmission, RTS/CTS, fragmentation, auto bit-rate selection, power saving and roaming. A nice feature of the MAC is the support of the micro-AP functionality, which allows to turn a PC into an Access Point (I would like more vendors to start doing that). However, their products don’t seem to support ad-hoc mode.

The physical layer is Frequency Hopping supporting 1 and 2 Mb/s, with 100 mW or 500 mW output power and 100 ms dwell size.

2.7.2 The driver

Lee has written the driver as a student project for Symbol, so with active help from Symbol. He plans to continue supporting it, and Symbol may get more active in distributing the driver.

The driver is designed for the Pcmcia card (LA2400 and micro-AP version), and the new 2 Mb/s version of the card. It is possible to use older cards (1 Mb/s) by updating the firmware for 802.11 compliance, and to use ISA card by configuring properly the Pcmcia package (those cards use a regular ISA to Pcmcia bridge).

Despite being beta, the driver is stable, well written and supports most features of the card (like micro-AP, shared memory access...).

2.8 Aironet ARLAN

Driver status : stable (ISA only)

Driver name : arlan.o

Version : 2.0 & 2.1b

Where :	Linux kernel (2.3.10 & 2.2.7-acX), web-page for 2.0.X version
Maintainers :	Elmer Joandi <Elmer.Joandi@ut.ee>
	Cullen Jennings <c.jennings@ieee.org>
Web pages :	http://www.ylenurme.ee/~elmer/655/
	http://www.cs.ubc.ca/spider/jennings/
Documentation :	README file + web page
Configuration :	/proc interface (2.1.X kernels and up only)
Statistics :	?
Modes :	Managed, Ad-Hoc
Security :	?
Scanning :	?
Monitor :	?
Multi-devices :	?
Interoperability :	proprietary protocol, interoperate with Windows
Other features :	-
Non implemented :	Multicast (driver is point to point ?)
Bugs :	-
License :	GPL
Vendor web page :	http://www.aironet.com/products/2200fam/2200fams.html

2.8.1 The device

The Arlan was a radio LAN, built by Aironet, using the 900MHz or 2.4GHz ISM band (Direct Sequence). This product has been **discontinued** and replaced by the 4500 series (see *section 3.14*). The Arlan comes in 3 flavour, an ISA (655), an MCA (670) and a pcmcia (690) card (plus the access point). Later, they renamed the ISA card IC2200 and the Pcmcia card PC2200 (still the same hardware).

The configuration include setting the frequency and Network ID (24 bits ?). The MAC protocol is implemented on a generic microcontroler. There is two versions of the modem, a 900 MHz and a 2.4 GHz version. Both use Direct Sequence Spread Spectrum. The 900 MHz modem allow signalling rate up to 860 kb/s (fall back to 215 kb/s) and 12 channels. The 2.4 GHz version allow signalling rate up to 2 Mb/s (fall back to 1 Mb/s) and 5 channels.

2.8.2 The driver

Russell Nelson told me a while ago that he was trying to convince Aironet to release the specifications of the Arlan to develop a Linux driver. *Cullen Jennings* started the development of a point to point driver, *Elmer Joandi* rewrote some parts and added a lot of features to be compatible with the Access Point, released the whole under GPL, and here is the result.

The driver support only the ISA version of the card (655 or IC 2200). The driver have been fully tested and optimised by *Elmer Joandi*, includes a complete /proc interface and should be soon included in the kernel.

2.9 Raytheon Raylink, WebGear Aviator2.4 & Aviator Pro and BUSlink wireless LAN

Driver status :	stable
Driver name :	ray_cs.o
Version :	1.67 (stable) and 1.70 (experimental)
Where :	Pcmcia package (3.1.9)
	Linux kernel (2.3.18 & 2.3.24)
Maintainer :	Corey Thomas <coreythomas@charter.net>
Web page :	http://webpages.charter.net/corey/index.html
	http://world.std.com/~corey/raylink.html
Documentation :	README file + headers
Configuration :	modules parameters and Wireless Extensions (init only)
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	No
Scanning :	No
Monitor :	No
Multi-devices :	yes (configuration via Wireless Extensions)
Interoperability :	802.11-FH (need updated firmware), interoperate with Windows (need to set the correct parameters)
Other features :	hardware multicast, MTU selection
Non implemented :	A few high level 802.11 functionalities.
Bugs :	SMP not fully tested, changing parameters through Wireless Extensions doesn't work right yet.
License :	GPL
Vendor web page :	http://www.raylink.com/micro/raylink/
	http://www.webgear.com/
	http://www.buslink.com/Net1.htm

2.9.1 The device

The Raylink is a IEEE 802.11 FH device build by Raytheon for the 2.4 GHz ISM band. Raytheon build only a Pcmcia card and an Access Point. I've been told that some version of the BreezeCom BreezeNet Pro Pcmcia card was an OEM version of the Raylink.

You are more likely to buy the Raylink as a WebGear product, either as Aviator2.4 or Aviator2.4 pro (which have nothing in common with their old Aviator 900 MHz line). The Aviator2.4 and Aviator2.4 pro are in fact the same product as the Raylink, the Aviator2.4 driver comes pre-configured in ad-hoc mode and offer only the Pcmcia card, whereas the Aviator2.4 pro driver comes preconfigured in managed mode and offer both the Pcmcia card and the Access Point (translation

seems also to be different in each driver). Of course, it is possible to change the mode in the driver and all these products are fully interoperable. WebGear also offers a ISA to Pcmcia bridge to install the Pcmcia card in desktops.

Lately, WebGear has stop selling those cards, but recently BUSlink has started selling them again (same card, different sticker).

The Raylink delivers all the features expected from a 802.11 compliant device, with ad-hoc networking, access point operation, authentication and roaming. The MAC protocol is as defined in 802.11 : CSMA/CA with MAC level retransmissions. Configuration includes mostly the ESSID (network name).

The modem is 2.4 GHz Frequency Hopping, with 1 Mb/s and 2 Mb/s bit rate, and includes antenna diversity.

2.9.2 The driver

Corey has implemented a very complete driver supporting most of the feature of the hardware and some 802.11 functionality (it should be able to talk to some 802.11 nodes). There is an exhaustive list of configuration parameters, a /proc interface for more parameters, and a tool to dump 802.11 frames. Good work !

The new version of the driver adds Alpha support, authentication, and compatibility with the Windows driver. SMP is slowly being tested. I've added to the driver quite complete support for Wireless Extension (changing parameters still doesn't work right - therefore wireless.opts do not work).

The driver has been developed for the Raytheon Raylink and has also been successfully tested with the WebGear Aviator2.4 and the BUSlink.

I recently improved the support for **Wireless Extensions**. It is now possible to configure the card before the card is brought up (for example in the various configuration files). However, it is still not possible to change most parameters at run time (but they can always be read).

2.10 Diamond Multimedia HomeFree

Driver status :	stable
Driver name :	tir2000.o
Version :	06/02/2000
Where :	http://david.poda.cz/homefree
Maintainer :	Pavel Machek <pavel@suse.cz>
Documentation :	README file
Configuration :	Module parameters
Statistics :	None
Modes :	Ad-Hoc
Security :	No
Scanning :	No
Monitor :	No
Multi-devices :	yes (except for module parameters setting)

Interoperability :	proprietary protocol, do not interoperate with Windows
Other features :	Act as a tty device (not a network driver)
Non implemented :	Windows compatibility
Bugs :	**May not be legal** in all locales...
License :	GPL
Vendor web page :	http://www.diamondmm.com/
	http://www.alation.com/

2.10.1 The device

The HomeFree was one of the first affordable home networking solution. It is sold by Diamond Multimedia and designed by a small company, Alation. The card comes in ISA, PCI and Pcmcia form factor.

To reduce the cost, Alation has used the same solution as IrDA : to implement the MAC protocol in the driver instead of on the card. In fact, they are using a IrDA chip as the baseband, and instead of connecting it to an Ir transceiver, they use a classical 1 Mb/s Frequency Hopping modem at 2.4 GHz.

This solution save the cost of an embedded microcontroller on the card and allow to build a cheaper product (and to develop it faster). The downside is that building the MAC protocol in the driver tend to increase the protocol overhead (the MAC need more time to react to events - this reduce throughput and increase latency) and use more resources on the host (processor cycles and memory). In fact, this is an effect similar to win-modems and win-printers. Also, because there is a lot more code (which is more tightly integrated in the OS and performance critical), the driver is more difficult to port to other OSes (and that's why the driver below doesn't implement the HomeFree MAC protocol).

Personally, I'm not a fan of this design, but it seems to do the job cheaply.

2.10.2 The driver

Pavel has developed a very simple and nice driver for the HomeFree. The development was sponsored by PODA s.r.o., a Czech company, which allowed Pavel to release the driver as GPL after some time...

This driver is both very different from a standard network driver (as the other driver I present on this page) and very different from the HomeFree Windows driver. This driver is a straight tty interface to the hardware (like a serial port), and doesn't implement any MAC protocol. Therefore, it can't be interfaced directly to the standard Linux networking stack, and is not compatible with the Windows driver.

Therefore, to use this driver, a MAC protocol of some sort is needed (to arbitrate access to the medium, multiplex connection and ensure reliability). *Pavel* recommend to use either some Ham protocols such as *Scarab*, or to use the *Linux-IrDA* stack. You can also develop you own application directly on top of this half duplex interface (most serial applications will assume full duplex).

The advantage of that is that those protocols are very lightweight, so usually perform much better (in term of raw throughput) than the original HomeFree

protocol, and even better than some other WLAN products. However, those protocol (Scarab, IrDA) are not designed for the specifics of the 2.4 GHz band and don't include all the goodies found in 802.11. For example, IrDA allow only two nodes to be exchanging data at one time (only one IrLAP connection active) and deal poorly with multi nodes network. I also don't know how they deal with co-located networks and radio interferences.

However, the most critical missing feature is regulations compliance. The 802.11 protocol include some feature to insure compliance with all the various regulations in the 802.11 band (such as Frequency Hopping - usage of Radio Frequency tend to be highly regulated). As the driver of *Pavel* doesn't include all these features, this driver **may not be legal** in your country (note : this doesn't apply to the Windows driver, the Windows driver is legal because Diamond has certified it with the FCC and ETSI), and usage of this driver may bring you big troubles (same as setting up a illegal transmitter in the FM band). So, if you care about legislation, I advise you to check with *Pavel* about your specific case, otherwise use at your own risks...

2.11 BreezeCom BreezeNet PRO Pcmcia

Driver status :	stable
Driver name :	brzcom_cs.o
Version :	1.0, 1.1-Beta
Where :	http://www.alvarion.com/RunTime/Support_10010.asp?tNodeParam=30 http://www.breezecom.com/Support_10010.asp?tNodeParam=30 ftp://projects.sourceforge.net/pub/pcmcia-cs/contrib/
Creator :	Christian Olrog
Maintainer :	Alfred Cohen <alfred@breezecom.co.il>
Documentation :	Readme file
Configuration :	module parameters
Statistics :	/proc interface
Modes :	Managed, Ad-Hoc
Security :	No
Scanning :	No
Monitor :	No
Multi-devices :	no
Interoperability :	802.11-FH (only pro.11), interoperate with Windows
Other features :	-
Non implemented :	security (WEP), power saving
Bugs :	-
License :	GPL
Vendor web page :	http://www.breezecom.com/

http://www.alvarion.com/

2.11.1 The device

The BreezeNet is a Radio LAN using the 2.4 GHz ISM band (Frequency Hopping). The earlier versions of the Pcmcia cards were OEM of other vendors, the old one was an OEM version of the **Netwave**, then it was an OEM version of the **Raylink**, but their latest pro.11 Pcmcia card is **100 % BreezeCom** (the one with two little antenna sticks). BreezeCom has also a 802.11-b line, called DS.11, and the Pcmcia card is a **NWN** card.

In term of protocol and modem, the Pcmcia cards are very similar to the other BreezeCom products (see *section 2.12*). The first two Pcmcia cards were limited in term of bit-rate (only 1 Mb/s), and have lower transmit power.

2.11.2 The drivers

The driver presented here apply only to the latest pro.11 Pcmcia card. For the old Pcmcia card (not pro), you may use the **netwave_cs** driver (see *section 2.2*). For the first pro.11 Pcmcia card, you may use the **ray_cs** driver (see *section 2.9*). For the DS.11 card (802.11-b compliant), you may use the **swallow_cs** driver (see *section 3.17*).

BreezeCom has also release a Linux driver for their latest pro.11 card. I've been informed of the existence of this driver since October 99, and many people have been using it since by getting it directly from BreezeCom, but BreezeCom did release this driver to the wide public only 6 months later. Let's not complain, because the driver contains the full source and is now GPL, so it was worth the wait !

The driver was written by *Christian Olrog*, an employee of Ericsson, based on the original Windows driver source, and it seems that the maintenance has been taken over by *Alfred Cohen* of BreezeCom. The source code looks very nice and complete, with only a few features missing. One interesting feature is that the driver can show the signal strength for Access Points in the area. However, the initial configuration could be simpler...

The driver has been in use by many Linux users since its original development and there doesn't seem to have been much complains about it, which is good ;-)

The original version of the driver, 1.0, is only for Linux kernel 2.2.X. There is a beta version of the driver for Linux kernel 2.4.X.

2.12 BreezeCom BreezeNet (not Pcmcia)

Driver status : not needed (for Pcmcia, see above)

Driver name : -

Version : -

Where : -

Maintainer : none

Documentation : none

Configuration :	none
Statistics :	none
Modes :	-
Security :	-
Scanning :	-
Monitor :	-
Multi-devices :	yes
Interoperability :	802.11-FH (only pro.11), 802.11-DS and 802.11-b (only DS.11), interoperate with Windows
Other features :	-
Non implemented :	configuration & statistics
Vendor web page :	http://www.breezecom.com/
	http://www.alvarion.com/

2.12.1 The device

The BreezeNet is a Radio LAN using the 2.4 GHz ISM band (Frequency Hopping). It is built by BreezeCom, a small company from Israel, and some OEM version might be available. The BreezeNet doesn't connect to any of the usual PC bus but instead uses an Ethernet network card to interface to the host computer, and so require no driver to work (they have also some real access points). For the Pcmcia hardware, see above.

There is three versions of the BreezeNet, the old one, somewhat Netwave compatible, then the first pro.11 version (flash upgradable to 802.11) and the new pro.11 version, which is 802.11 compliant, so with all the usual MAC features expected from 802.11 devices. In all cases the modem includes Frequency Hopping Spread Spectrum (20 ms hop period), a 3 Mb/s signalling rate (fall back to 2 and 1 Mb/s) and antenna diversity. Note that the 3 Mb/s bit rate is not 802.11 compliant.

BreezeCom now offers a DS.11 series of adapters which is 802.11-b compliant, with usual 802.11-b features (and up to 11 Mb/s) and still using the Ethernet interface.

In 2001, BreezeCom and Floware Wireless Systems merged together to form Alvarion. The new combined company seems to put less emphasis on wireless LANs and more on point-to-point links and wireless distributions systems, even though the old BreezeNet product lines are still available.

2.12.2 The drivers

No driver is needed, this product use an Ethernet connection. You need to have an dedicated Ethernet 10baseT card configured under Linux to plug it into. For the device configuration and statistics, unless someone write the necessary tools for Linux, I guess that you must return to DOS/Windows.

3 The devices, the drivers - 802.11b

This section list devices based on the IEEE 802.11b standard (see *section 8*), operating in the 2.4 GHz band up to 11 Mb/s.

3.1 Lucent Wavelan IEEE, Lucent Orinoco, Enterasys RoamAbout 802, Elsa AirLancer 11 and Melco/Buffalo 802.11b

Driver status :	obsolete (see *section 3.2*)
Driver name :	wvlan_cs.o
Version :	v1.0.7
Where :	Pcmcia package (3.1.25)
Maintainers :	Anton Blanchard <anton@samba.org>
	Andreas Neuhaus <andy@fasta.fh-dortmund.de>
	Harald Roelle <harald@roelle.com>
	Moustafa A. Youssef <moustafa@cs.umd.edu>
Web pages :	http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Wavelan-IEEE.html
	http://www.fasta.fh-dortmund.de/users/andy/wvlan/
	http://www.roelle.com/wvlanPPC/index.html
	http://www.cs.umd.edu/~moustafa/mwvlan/mwvlan.html
Mailing list :	http://lists.samba.org/pipermail/wireless/
Documentation :	man page, headers
Configuration :	Wireless Extensions & module parameters
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc and Ad-Hoc-demo
Security :	WEP (based on hardware support)
Scanning :	No
Monitor :	No
Multi-devices :	Yes
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	MTU selection, multicast, promiscuous mode, power management, WEP hardware encryption, SMP, multi-firmware and PPC support.
Non implemented :	Some optimisations... Does not support HermesII.
Bugs :	May have some performance issues
License :	GPL
Vendor web page :	http://www.wavelan.com/
	http://www.proxim.com/
	http://www.enterasys.com/wireless/
	http://www.elsa.com/

http://www.hp.com/notebooks/us/eng/products/wireless/
http://www.buffalotech.com/
http://www.1stwave.de/
http://www.artem.de/

3.1.1 The device

Even if it uses the same name, the **Wavelan IEEE** product is completely different from the old Wavelan (see *section 2.1*), and totally incompatible in term of protocol and hardware interface. It is still built by Lucent, and it still operate in the 2.4 GHz ISM band (Direct Sequence), but the new hardware fully support the IEEE 802.11 protocol (and 802.11-b for the more recent versions) and is no longer based on a Ethernet MAC chip. There is only a Pcmcia version (the ISA version uses a ISA to Pcmcia bridge) and the different access points. Recently, Lucent has added a USB adapter and mini PCI version of the card for laptop (this one is based on a PCI-Pcmcia bridge).

To confuse the issue, Lucent has renamed the Wavelan IEEE as **Orinoco** (Wavelan was better IMHO), and this division was part of Lucent spin-off into a new company called **Agere**. **Avaya** (another Lucent spin-off) is also selling the Orinoco. Enterasys is also selling the Wavelan IEEE as **RoamAbout 802** (this company was formerly known as Cabletron, which was the former DEC networking division). Elsa is selling it in Europe as **AirLancer 11** (on the other hand, the 2 Mb/s version is quite different). In Japan (and maybe also in Europe), Melco is selling it as **Buffalo WLI-PCM-L11**. Lately, more vendors have been joining the club, such as HP (**HP 802.11b Wireless LAN**), IBM (**IBM High Rate Wireless LAN**), Dell (**Dell TrueMobile 1150** - on the other hand, the 1100 is an *Aironet* card), Compaq (**Compaq WL 110, WL 210 and WL 215** - the WL100 and WL200 are *PrismII* based), 1stWave (**1stWave PC-Card**) and ARtem (**ARtem ComCard**). The Apple **Airport** is also derived from the Wavelan IEEE (see *section 3.5*).

The Wavelan IEEE saga never ends. **Proxim** bought the card and access point business of Agere (Agere kept the chipset and radio part), so now the same Orinoco cards are sold by Proxim under the name **Orinoco Classic** or **Orinoco World** (841X - with the big square antenna and using the same Agere Hermes chipset). In a bold marketing move, Proxim renamed all it's other lines of wireless cards as Orinoco, however those cards are not based on the Agere chipset but on **Atheros chipset** (846X, 847X and 848X). The **Orinoco 11b** (842X - 802.11b only with a short antenna) are based on the Agere HermesII chipset, which is different from the old chipset (and therefore not compatible with the usual Orinoco drivers). So, if the Proxim Orinoco card doesn't have a big square antenna and do support 802.11a or 802.11g, you can be sure it's not a **true Orinoco**.

The Wavelan IEEE appears to the PC as a standard network card and interfaces naturally with the networking stack. The configuration includes only setting the network name (ESSID), the rest is automatic (finding the equivalent BSSID and channel). As usual for Lucent, the documentation and website are rich.

The Lucent Wavelan IEEE is based on the Lucent Hermes chipset. As with all IEEE 802.11 products, the Hermes offer a fully featured MAC protocol, including MAC level acknowledgement (good news for all of us having dealt with the old Wavelan card), optional RTS/CTS, fragmentation, automatic rate selection, roaming. This seems exhaustive, but is mandatory for IEEE 802.11 compliance. Different version of the card include different levels of security (bronze is basic, silver is with WEP (RC4-40 bits) and gold is with RC4-128 bit encryption).

The MAC support both Managed and **Ad-Hoc modes**. However, the initial firmware for those cards did support only a non-compliant Ad-Hoc mode (called Ad-Hoc demo mode - which interoperate with most PrismII cards). In order to gain WiFi compliance, Lucent added in recent firmware (6.06 and greater) a second Ad-Hoc mode which is fully 802.11 compliant (called Peer to Peer mode or IBSS Ad-Hoc mode - and which interoperate with Aironet cards). Of course, the two Ad-Hoc modes are not interoperable.

The 2.4 GHz modem is an enhanced version of the previous generation, Direct Sequence Spread Spectrum (11 chips encoding), using both 1 and 2 Mb/s signalling rate (using effectively 22 MHz of bandwidth) and 5.5 and 11 Mb/s in second generation cards, diversity antennas and with 13 different frequencies (depending on the regulations).

Initially, the Wavelan was only offering 1 and 2 Mb/s bit rates (basic IEEE 802.11 DS standard). For a while, Lucent was also selling a “turbo” version of the card, which was adding 5 and 10 Mb/s bit-rates for shorter range using Lucent proprietary modulations (so, not compatible with 802.11-b).

Later, Lucent introduced the second generation of the Wavelan IEEE, still based on the same Hermes chipset, which is much cheaper and fully compliant with the new **802.11-b** standard, supporting 1, 2, 5.5 and 11 Mb/s bit-rate (compatible with other 11 Mb/s products).

All Wavelan IEEE cards do not offer the exact same set of features, because Lucent keep changing the **firmware**. From firmware **1.00** to **4.52**, Lucent was mostly adding features (encryption, power saving) and keeping it backward compatible, but firmware **6.04** and later created a major incompatibility. Firmware **6.06** and later implement a fully 802.11 compliant IBSS Ad-Hoc mode (on top of the Ad-Hoc demo mode). Firmware **6.04** dropped Fragmentation Threshold setting in favor of microwave oven robustness (an automatic fragmentation scheme). Firmware **6.16** did fix a few bugs with the IBSS Ad-Hoc mode (security, ESSID=”any”).

Agere has recently released a new **HermesII** chipset, derived from the venerable Hermes chipset. The most notable improvements are a higher integration (smaller & cheaper), a PCI interface with DMA support and a USB interface. The chip interface and firmware is not compatible with the old Hermes chipset, requiring specific driver support for HermesII. To my knowledge, this chipset is only used in the Proxim 842X cards.

3.1.2 The driver

Andreas Neuhaus is no longer working to improve this driver, therefore it's now discontinued in favor of the new Orinoco driver (see *section 3.2*). The driver is based on Lucent source code, which is a cut down version of their full driver. So, it lacks all the part about handling natively 802.11 frames and Lucent proprietary API, and initially it lacked some of the more fancy features of Lucent's driver, but *Andreas* is adding them slowly. Of course, the driver support all version of the card (bronze, silver, gold - basic, turbo, turbo 11 Mb/s) and is fully interoperable with Access Points and Windows nodes.

Andreas has done a very good job into providing features like Wireless Extensions (I must admit that I did help him quite a bit ;-) and many configuration parameters (station name, channel, mtu size). The new version adds Power management and encryption setting, change of the operating mode via Wireless Extensions, promiscuous and multicast support...

Andreas has done a lot of debugging of the driver and it seems now much more stable. Lastly, the ISA to Pcmcia and PCI to Pcmcia bridges may be a source troubles under Linux. The latest version of the driver fixes SMP support, multi-cards configuration, improve wireless.opts support, add IBSS Ad-Hoc mode support and support properly and sanely the various firmware releases.

Harald Roelle has developped a patch for this driver in order to fully support the PPC architecture. This patch mostly contain some bit order fixes. This patch should help other architecture with endianess issues. His patch was eventually integrated (with major changes) by *David Hinds* in version 1.0.6 of the driver. I added firmware detection support in 1.0.6 to properly handle all the various firmware releases and their variations (in particular the two Ad-Hoc modes), and fixed the remaining SMP bugs.

The driver does not support the USB and Mini-PCI version of the Wavelan.

Nowadays, *Anton Blanchard* is the official maintainer of the driver, with the help of *David Gibson*. *David* has done a complete rewrite of the driver (see *section 3.2*), so this driver won't be maintained anymore...

Moustafa has released a version of this driver with scanning support.

Note that Lucent has also released a binary library driver (see *section 3.3*) which is maybe more solid and performant than the driver of *Andreas*, but lack complete support for Wireless Extensions.

3.2 Wavelan IEEE/Orinoco, PrismII and Symbol cards

Driver status :	stable
Driver name :	Pcmcia : orinoco_cs.o
	PLX : orinoco_plx.o
	PCI : orinoco_pci.o
Version :	v0.15, CVS
Where :	Linux kernel (2.4.21 ; 2.6.18)
	Pcmcia package (3.1.34)

	http://www.ozlabs.org/people/dgibson/dldwd
	http://savannah.nongnu.org/projects/orinoco/
Maintainers :	David Gibson <hermes@gibson.dropbear.id.au>
	Pavel Roskin <proski@gnu.org>
Web page :	http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Orinoco.html
Mailing list :	http://sourceforge.net/mail/?group_id=44338
Documentation :	man page, headers
Configuration :	Wireless Extensions only
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc and Ad-Hoc-demo
Security :	WEP (based on hardware support), 802.1x
Scanning :	Wireless Extensions (v0.14 and later)
Monitor :	Yes (v0.14 and later)
Multi-devices :	Yes
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	MTU selection, multicast, promiscuous mode, power management, SMP, multi-firmware, multi vendors, PPC & ARM support, PLX and PCI support.
Non implemented :	Does not support HermesII.
Bugs :	WEP not functional on old Prism2 firmwares, some older driver versions don't handle properly some Symbol cards.
License :	MPL and GPL
Vendor web pages :	[Too many to list here]

3.2.1 The devices

As explained in various sections, Lucent Wavelan-IEEE/Orinoco devices (see *section 3.1*), Intersil PrismII devices (see *section 3.6*) and Symbol High Rate devices (see *section 3.10*) are basically using the **same MAC controller**. This driver attempt to support all those devices, which are described in details in their own sections.

However, even though those devices use the same MAC controller and the same driver, those devices are not the same. Each vendor has its own **firmware**, so the set of features of those cards vary. Some differences are visible to the user (for example 128 bits key support and multicast), some are more related to performance and robustness tuning of the MAC.

Moreover, those devices don't use the same **radio modem** (mostly Lucent or Intersil) and same antennas. For PrismII cards, even the actual layout of the radio components on the card can make a huge difference. This will mostly translate into difference of **coverage** between the various cards (range and resistance to interference). The range between some cards may vary by a factor 2 in some conditions.

3.2.2 The driver

Anton Blanchard and *David Gibson* became official maintainers of the **wvlan_cs** driver (see *section 3.1*) in the end of 2000. *David* was not very happy about the state of wvlan_cs.

The HCF (the low level library provided by Lucent) hadn't been maintained since the initial release of the driver and was quite difficult to read and understand. While the higher layer of the driver had gone a long way and were robust and fully featured, the HCF was a mess and the cause of many problems (TxTimeout, driver corruption/crashes and else).

Rather than put up with that, *David* looked deeply in the low level of the **wlan-ng** driver from *Mark* (see *section 3.6*) and the FreeBSD driver and wrote a totally new driver combining a new low level core and the high level features of wvlan_cs. The end result was a driver much more readable, robust and well behaved than wvlan_cs. In the process, David added support for PrismII cards. Then, I fixed a few Wireless Extensions bugs, added some support for Symbol cards, and we pushed the driver in the kernel. The driver was initially named **dldwd_cs** and was renamed **orinoco_cs** at this point. Later on, *David Hinds* backported this driver to the Pcmcia package for users of earlier kernels.

The main goal of the driver is to support **Wavelan IEEE/Orinoco** cards and OEM. The driver support all the firmwares and features of those cards properly and fully (Ad-Hoc demo mode, IBSS mode, bit-rate, encryption keys...), and support all the features available in wvlan_cs (except module parameters) with less bugs.

Ben has added **Airport** support to this driver (see *section 3.5*), and the support of those cards is similar to Orinoco cards (i.e. most features supported properly).

Starting in release v0.6d, the support of **Symbol cards** and OEM is complete, at least for firmware 1.5 and 1.7. Bit rate, mode of operation (managed, ad-hoc IBSS and ad-hoc demo), encryption and power management are fully working. The release v0.8 added full support for later Symbol firmware 2.00 and 2.20. On firmware 2.20 and later, Power Management is disabled. Version of the driver from **v0.10 to v0.12b don't work** properly with Symbol cards due to a bug, so avoid those releases. **Symbol CF cards** are very different and supported in their own driver (see *section 3.13*).

The support of **PrismII cards** and clones is still in progress. More debugging and testing need to be done, but the driver can set most features to some degree (Ad-Hoc demo mode, IBSS mode, bit-rate, encryption keys have been seen to work). It seems the upgrading firmware fixes problems related to encryption. However, the *wlan-ng* and *HostAP* drivers still have more features and are more tested...

Starting in release v0.8, the orinoco driver collection also support **PLX adapters** that are sold with some PrismII cards (via the *orinoco_plx* driver). Those adapters are not real Pcmcia adapters and the card looks to the system like a PCI card. The driver also support Pcmcia cards in regular ISA to Pcmcia or PCI to Pcmcia adapters, as long the **Pcmcia adapter** is recognised and configured properly by the Pcmcia package (which might be tricky).

Starting in release v0.11a, the orinoco driver collection also support **PCI cards** (all of them being PrismII cards - via the *orinoco_pci* driver). The standard driver does not support the various **USB** versions of the cards. There is various kind of **MiniPCI** implementation of the card, the driver support some of them (Pcmcia based - Lucent ; PCI based - PrismII) but not most (USB based - PrismII).

The latest version (v0.13b) seems to have fix most of the hardware reset problems of previous versions and seems to have fixed problems with Symbol firmwares. *Pavel* has integrated in v0.14 many previously external patches, such as support for **Wireless Scanning** and Wireless Events, support for **Monitor mode** and support for **802.1x**. *Pavel* also fixed support for kernel 2.6.X, dramatically improved Symbol firmware support and fixed a tons of bugs in v0.14.

Version 0.15-rc2 of the driver was merged into kernel 2.6.12. This brought all the features and bugfix above in the kernel. Since then, *Pavel* is continuing to fix and update the driver, pushing those changes into recent kernels, but he is no longer using version number (so I guess we are still at 0.15).

3.3 Lucent Wavelan IEEE, Enterasys Roamabout and Proxim Orinoco 8420

Driver status :	stable, beta
Driver name :	wavelan2_cs.o, roamabout_cs.o and wlags49_cs.o
Version :	v6.16, v7.18 and v7.22
Where :	http://greenblaze.com/proxim.html
	http://www.agere.com/mobility/wireless_lan_drivers.html
	http://www.agere.com/support/drivers/index.html
	http://www.cs.umd.edu/~moustafa/mwavelan/mwavelan.html
Contact :	Lucent support <usasupport@wavelan.com>
	EnteraSys support <support@enterasys.com>
Maintainers :	Richard van Leeuwen <rleeuwen@lucent.com>
	Dean W. Gehnert <deang@tpi.com>
	Moustafa A. Youssef <moustafa@cs.umd.edu>
Documentation :	Extensive readme
Configuration :	Module parameters, Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc and Ad-Hoc-demo
Security :	WEP (based on hardware support), WPA
Scanning :	Wireless Extensions
Monitor :	No
Multi-devices :	yes, but the ISA to Pcmcia bridge must be reconfigured
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows

Other features :	Power management and microwave oven robustness. Support HermesII cards (v7.08). WPA support.
Non implemented :	Do not support all firmware releases
Bugs :	?
License :	Binary only for the core + OpenSource Linux wrapper (up to v6.16), GPL (v7.08)
Vendor web page :	http://www.wavelan.com/
	http://www.enterasys.com/wireless/

3.3.1 The device

This is the same device as the previous entry (*section 3.1*).

3.3.2 The driver

Lucent has decided to not put all its eggs in the same basket and developed a bold strategy for the support of the Wavelan IEEE under Linux. Not only they have released some source code to allow the source driver mentioned above, but they have as well contracted *Dean* to release a driver based on a binary library. This gives Linux users the choice, a GPL full source driver to hack with and a stable full featured binary driver (the official term from Lucent is “Linux Driver Source/ Library”).

Dean has written the code interfacing between Linux and the library, and had put together a nice package easy to install and with documentation. As expected, the binary driver is probably more stable and than the full source driver mentioned above, with a slightly different set of feature, and offers all the features of Lucent Window drivers, plus a nice integration with Linux. This driver supports both the basic version of the card and the “turbo”. The major drawback is the binary core, preventing the use on other architectures (PPC, Arm...).

Now, the driver is supported by Lucent, and they keep adding in it the same features they add to the Windows drivers (such as microwave oven robustness). Their also have added support for the IBSS Ad-Hoc mode (see discussion above). The latest version adds support for 2.4 kernel and many common Wireless Extensions. Note that Enterasys/Cabletron is also distributing a slightly modified version of this driver (usually an older one).

Moustafa has released a version of this driver with scanning support.

Recently, *Agere* has released a new version of this driver which is fully Open Source. This new version has support for the **HermesII** chipset found in the Proxim Orinoco 842X cards (but it seems it no longer support the old Orinoco cards - use the older driver version). It has also improved Wireless Extension support. *Agere* is still working hard on the driver and has recently added support for Wireless Scanning and WPA.

3.4 Orinoco USB cards and HP/Compaq multiport

Driver status :	beta
Driver name :	orinoco_usb.o

Version :	0.1.4
Where :	http://www.nongnu.org/orinoco/
	http://savannah.nongnu.org/projects/orinoco/
Maintainers :	Manuel Estrada Sainz <ranty@debian.org>
	Ramon Rey Vicente <ramon.rey at hispalinux.es>
Mailing list :	http://sourceforge.net/mail/?group_id=44338
Documentation :	headers
Configuration :	Wireless Extensions only
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc and Ad-Hoc-demo
Security :	WEP (based on hardware support)
Scanning :	Wireless Extensions (with optional patch)
Monitor :	With optional patch
Multi-devices :	Yes
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	MTU selection, multicast, promiscuous mode, power management, multi-firmware.
Non implemented :	-
Bugs :	-
License :	MPL and GPL
Vendor web pages :	http://www.wavelan.com/
	http://www.proxim.com/
	http://www.hp.com/

3.4.1 The devices

The Orinoco-USB is of course related to other **Wavelan-IEEE/Orinoco** devices (see *section 3.1*). As most first generation USB designs, it is really a standard Orinoco Pcmcia card plugged into a **Cypress EZ-USB** USB-Pcmcia bridge. Such a setup of course brings some performance degradation due to USB high latency. The Orinoco card inside the device has the same exact features as other Orinoco Pcmcia cards and is 802.11b compliant.

Lucent, Agere and Proxim are directly selling this **Orinoco** device. HP/ Compaq sells it as **WL215** (standalone) and **W200** (multiport option for Compaq laptops). Other vendors such as Melco are also selling this hardware. One of the particularity of this hardware is that the USB-Pcmcia bridge doesn't contain a firmware, so the driver need to upload the firmware at power up. On the other hand, the Pcmcia card behind the USB-Pcmcia bridge already contains its own firmware.

Note that most USB 802.11b cards are based on either the Intersil PrismII chipset (see *section 3.6*) or the Atmel chipset (see *section 3.20*), and are quite different from this hardware.

3.4.2 The driver

Manuel has managed to reverse engineer the Orinoco USB hardware and is providing a modified version of the Orinoco driver for Orinoco USB adapters. This driver only support **Orinoco USB** hardware, and not other USB cards. Because it is based on the Orinoco driver (see *section 3.2*), this driver offer the same extended feature set, such as Wireless Extension support.

The big difference with the standard Orinoco driver is **firmware uploading**. You will need to extract the firmware for the USB-Pcmcia bridge from the Windows driver using the tools provided on the driver web page. The firmware uploading support in Linux needed for this driver is currently being finalised, so check the latest driver documentation. On the other hand, the driver offer no support for updating the firmware in the Pcmcia card.

This driver was merged into the CVS of the **Orinoco driver** (see *section 3.2*). However, because the Orinoco maintainers are not happy with the locking strategy of this driver, this driver was never included in any release of the Orinoco driver and is not included in the kernel. The latest version of the driver may be found in the Orinoco CVS (see *section 3.2*).

3.5 Apple Airport

Driver status :	stable
Driver name :	airport.o
Version :	0.15
Where :	Linux kernel (2.6.18) http://www.ozlabs.org/people/dgibson/dldwd http://ppclinux.apple.com/~benh/
Maintainer :	Benjamin Herrenschmidt <benh@kernel.crashing.org>
Documentation :	web page, headers
Configuration :	Wireless Extensions & module parameters
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc and Ad-Hoc-demo
Security :	WEP (based on hardware support)
Scanning :	Wireless Extensions
Monitor :	With optional patch
Multi-devices :	No
Interoperability :	802.11-DS and 802.11-b, interoperate with Mac-OS ;-)
Other features :	MTU selection, multicast, promiscuous mode, power management, SMP and multi-firmware.
Non implemented :	Some optimisations...

Bugs : -

License : GPL

Vendor web page : http://www.apple.com/airport/

3.5.1 The device

The Apple **AirPort** is in fact the Lucent **Wavelan IEEE** repackaged, so has the same characteristic as the Wavelan (see *section 3.1*). All Airport hardware is 802.11-b compliant (second generation of Wavelan IEEE) and support 11 Mb/s, and Apple seem to offer only the version with 40 bit encryption.

The AirPort card for the most Apple hardware is the OEM version of the Wavelan IEEE, but it uses a specific slot in those computers and the antennas are pre-integrated in the host. Most recent Apple machines offer this interface (iBook, PowerBook 2000 (aka Pismo), AGP G4s, recent iMacs (DV/SE)...). Note that this interface is **not Pcmcia compatible** even is the connector is the same, so this card can't be used in the normal PC-Card slot of other laptops. This is why this card work only in specific Apple hardware slot and only with a specific driver.

The Access Point (the famous flying saucer) is similar in functionality to the Lucent RG-1000 Residential Gateway, and is fully interoperable with other 802.11-b hardware.

3.5.2 The driver

The **first version** of the Airport driver was done by *Benjamin Herrenschmidt* by porting the driver of *Andreas Neuhaus* (see *section 3.1*) to support the Apple Aiport card. He has basically integrated the specific PPC patch of *Harald Roelle*, thrown away all the Pcmcia code and replaced it with the specific Apple initialisation code.

Apart from that, the driver is basically the same, with the same features and same bug ;-)

The **second version** of the driver was also done by *Benjamin Herrenschmidt* and is just a wrapper on top of the driver of *David Gibson* (see *section 3.2*), and was integrated in version 0.05 (kernel 2.4.5). This is a much cleaner solution, because both driver share the same source, so the feature set is identical and all improvements and bug fixes of the Orinoco driver are automatically in the Airport driver and vice-versa. For example, this driver gained both **Scanning** and **Monitor mode** support in version 0.14 and later.

3.6 Intersil PrismII based cards (the most common 802.11b cards)

Driver status : stable

Driver name : Pcmcia : prism2_cs.o
PLX : prism2_plx.o
PCI : prism2_pci.o
USB : prism2_usb.o

Version : 0.2.8

Where : http://www.linux-wlan.com/linux-wlan

Maintainers :	Mark S. Mathews <mark@linux-wlan.com>
	Solomon Peachy <solomon@linux-wlan.com>
Mailing list :	http://www.linux-wlan.com/linux-wlan/
	http://www.lifix.fi/extarchive/lwlan/
Documentation :	Readme
	ftp://ftp.linux-wlan.org/pub/linux-wlan-ng/FAQ
	http://www.linux-wlan.org/docs/linux-wlan-FAQ.html
Configuration :	Module parameters & configuration tool
Statistics :	Statistic tool & Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP (based on hardware support)
Scanning :	Specific tool & Wireless Extensions
Monitor :	Yes
Multi-devices :	Yes
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Quite exhaustive 802.11 support, Encryption, PPC support, PLX, PCI and USB support.
Non implemented :	?
Bugs :	?
License :	MPL
Vendor web pages :	http://www.compaq.com/products/wlan/index.html
	http://www.magiclan.com/
	http://www.dlink.com/products/
	http://www.linksys.com/products/
	http://www.zoomtel.com/zoomair/za11index.html
	http://www.nokia.com/corporate/wlan/card_c110.html
	http://www.addtron.com/
	http://www.gemtek.com.tw/
	http://www.smc.com/
	http://www.netgear.com/
	http://www.ambicom.com/
	http://www.teletronics.com/
	http://www.intersil.com/design/prism/
	http://www.conexant.com/

3.6.1 The device

The PrismII chipset is the successor of the PrismI chipset, described in the previous section (see *section 2.4*), and is build by Intersil (formerly Harris). Intersil

offer this chipset and some reference design to various OEM, allowing them to build various 802.11-b products (cards or integrated in their own products). I expect that all the people that were formerly using the PrismI chipset will switch sooner or later to the PrismII.

The first manufacturers to offer PrismII cards were **Samsung** and **Compaq** (WL100, WL200, rumored to be a rebadged Samsung card), with a Pcmcia card, a PCI card and an Access Point. Other Prism vendors like **ZoomAir**, **Nokia** and **GemTek** did release later their own version of the PrismII cards, as well as **Proxim** (RangeLAN DS, Harmony 802.11b...). Some big networking vendors like **D-Link**, **LinkSys**, **NetGear** and **SMC** were also quick to jump on this new opportunity for them, as well as many smaller vendors like **AddTron**, **Ambicom**, **Teletronics**, **Ampwave** and many other that I can't list... The rule of thumb is that if your card is not listed in another section of the Howto, it could be a PrismII card (or not, see below).

Some notable exceptions which are **not** PrismII cards : the **Compaq WL 110, WL 210 and WL 215** cards (which are *Orinoco* cards), the **D-Link 650H** (which is a *Symbol* card), most **D-Link/LinkSys/SMC USB** cards (which are *Atmel* cards), the **SMC 2632W-v2** (which is an *Atmel* card), all **22 Mb/s** cards such as the **D-Link 650+/520+** (based on the *TI chipset*), all **CardBus** cards such as the **new D-Link 650** (which is an *ADMtek* card), all **802.11a** cards (which are *Atheros* or *Intersil PrismDuette* cards) and all **802.11g** cards (which are *Broadcom*, *Atheros* or *Intersil PrismGT* cards). In fact, so many vendors seem to be moving away from the PrismII chipset (usually without warning and without changing the model name) and there is so many changes happening that it's impossible to keep track of who is using what.

Please note that everything that **looks like a PrismII card** may not be a PrismII card, and many people are quite confused about that. The cards described in this section use both a Intersil PrismII chipset and an Intersil firmware. Other vendors, such as Lucent (see *section 3.1*), Aironet (see *section 3.14*), Symbol (see *section 3.10*) and Atmel (see *section 3.20*) use part of the PrismII chipset but with their own firmware and therefore are **not compatible** (even if they sometime use the same device identification as PrismII cards and sort of work with PrismII drivers).

Most PrismII vendors offer regular *Pcmcia cards* for laptops. For desktop machines, the situation is a bit more messy, some vendors offer standard *PCI-Pcmcia cards* (where you can slot the Pcmcia card), dedicated *PLX cards* (that look like a regular PCI-Pcmcia bridge but is not) or some *fully integrated PCI cards* (Prism2.5). Some vendors also offer *USB adapters* (beware, some of them are *Amtel* cards, and all of them have performance issues). Lastly, some laptop include *MiniPCI cards* that may be either integrated PCI cards or USB adapters.

Like the initial PrismI design, the PrismII is fully compatible with 802.11 and include a 2.4 GHz Direct Sequence modem, with all the usual features (Roaming, WEP...).

The main differences between the PrismI and PrismII chipset are a higher integration, a higher performance modem and the replacement of the AMD

controller with Intersil own design. The higher integration (5 chips instead of 8) allows to reduce the price and the size of the product, and to simplify the integration. The new physical layer (modem) has a better performance (but a lower transmit power), increasing range, speed and battery life, and is fully compliant with the **802.11-b** standard (5.5 and 11 Mb/s). Finally, the new MAC controller handle most of the 802.11 functionality (instead of leaving it to the driver), which simplify driver development and help performance on slow devices (palmtop, embedded design).

The **Prism2.5** and **Prism3** chipsets are evolution of the PrismII chipset, offering even higher integration, lower cost and backward compatibility. With respect to the driver, these 3 chipset look the same, and therefore driver supporting PrismII hardware will also support Prism2.5 and Prism3 hardware.

Note that the PrismII **firmwares** are usually not of the highest quality and quite inconsistent from one release to another, both on the cards and on the Access Points, and you may have to try a few of them before finding the one that work for you. For example, encryption and IBSS ad-hoc mode seems to be working only in the latest firmwares (0.8.3 and later), and multicast is not working at all. It also usually takes a bit of time to get the workaround for the latest firmwares in the various Linux drivers. Latest firmware seem to have fixed most problems and have added the feature missing from earlier firmwares.

A few words about **Ad-Hoc modes** : like for Orinoco card, the firmware support two ad-hoc mode, the Ad-Hoc demo mode (not 802.11-b compliant, but reported to be Orinoco Ad-Hoc demo mode compatible) and the IBSS Ad-Hoc mode (802.11-b compliant). The IBSS Ad-Hoc mode is only available in firmwares 0.8.3 and later.

3.6.2 The driver

Who was more qualified to write this driver than *Mark*, from AVS, who already wrote the driver for the PrimsI cards ? In fact, Intersil did partner with *Mark* to get this driver written for us !

As usual with *Mark*, the driver is really complete and well written. It is currently only in beta stage, and *Mark* told me that he needs to add more documentation and debug some more features. The driver support both Pcmcia and PCI cards. This driver is compatible with Linux bridging software, includes a generic 802.11 interface, exposing the full 802.11 MIB to user space, and include hooks to build an Access Point. The driver also come with a configuration tools, an utility to dump 802.11 frames and a daemon responding to 802.11 events.

The release 0.1.10 fixes a number of long standing problems and include a number of patches and features that were floating around on the mailing lists. This version supports properly WEP encryption and Ad-Hoc mode. Note that the driver supports only **IBSS ad-hoc** mode (0.1.10 and later) and only for recent firmwares, whereas most cards also support the old ad-hoc demo mode.

The driver supports **Pcmcia, PLX and PCI cards**. The PLX card allow to add a Pcmcia card in a PCI slot, but does not support any of the Pcmcia functionality, so is not supported through the Pcmcia package but directly by the

driver. PCI support is for fully integrated PCI cards or MiniPCI cards. *Mark* has also added **USB support** (only for Intersil USB cards, not Atmel cards).

Reyk has developped a patch that adds basic Wireless Extension support to the driver, and that was included in version 0.1.13. He needs help for testing and improving it.

Since then, *Mark* is concentrating on a Intersil 802.11a driver (not Atheros) and has transferred the maintenance of the driver to *Solomon* (a new AVS employee). *Solomon* is making the driver SMP compliant, cleaning it up and keeping up with the new firmwares from Intersil, and keeping up with new kernels. *Solomon* has also added pretty complete support for **Wireless Extensions** in 0.2.1, including Scanning support.

Mark is also selling a Wireless Development kit and an Access Point, based on a PPC platform and this driver.

3.7 Intersil PrismII support in the Orinoco driver

The Orinoco driver (see *section 3.2*) may be used with most PrismII cards.

3.8 Intersil PrismII driver with HostAP mode

Driver status :	stable
Driver name :	Pcmcia : hostap_cs.o
	PLX : hostap_plx.o
	PCI : hostap_pci.o
Versions :	v0.4.9
Where :	Linux kernel (2.6.17)
	http://hostap.epitest.fi/
Maintainer :	Jouni Malinen <jkmaline@cc.hut.fi>
Mailing list :	http://lists.shmoo.com/pipermail/hostap/
Documentation :	Readme, web page
Configuration :	Module parameters and Wireless Extensions
Statistics :	Wireless Extensions and /proc interface
Modes :	Managed, Ad-Hoc, Master (HostAP), Repeater (WDS)
Security :	WEP (hardware or host based), 802.1x, WPA
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	Yes
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Host AP mode, bridging, access list, WDS, PLX and PCI support
Non implemented :	?
Bugs :	?

License : GPL

Vendor web pages : [Same as PrismII driver]

3.8.1 The device

This is the same device as the previous entry (*section 3.6*).

One of the most interesting feature of the standard PrismII firmware is that it can allow the host to act as an Access Point (**HostAP mode**). This allow to turn a regular PC with a Prism2 cards into an Access Point, allowing other nodes to connect to it. In HostAP mode, the card does only the critical part of the Access Point (sending beacons) and simply pass all the 802.11 management frames to the driver (which does 802.11 management itself).

Note that this HostAP mode doesn't exist or is not documented for other cards (non-PrismII firmwares). Also, it is possible to load special firmware in PrismII card which allows the card to perform the full Access Point functionality by itself (tertiary firmware).

3.8.2 The driver

Jouni has recently written this totally new driver for PrismII card. It is well written, it was probably inspired by the various other driver floating around and is much more simpler than the linux-wlan-ng driver (see *section 3.6*).

The driver has complete support for the various feature of the PrismII card (WEP, IBSS Ad-Hoc mode, scanning...), Monitor mode, very complete support for Wireless Extensions and offer various extra information in a /proc directory, making already an excellent choice for a standard wireless client.

What set this driver apart from the other driver is its support for HostAP mode. In this mode, the driver act as an Access Point on the air and does all the 802.11 management necessary. In this mode, the driver also allows bridging through the regular Ethernet bridge driver of Linux. This explain why this driver is use by most Linux Access-Point projects.

Jouni continues to refine his driver and has added PLX and PCI cards support, monitor mode, MAC address based access list and WDS support (to allow Access Point to communicate with each other). *Jouni* latest masterpiece is the addition of **WPA support** in the HostAP driver, and the associated user space wpa_supplicant.

In other words : impressive work...

This driver was included in the Linux kernel 2.6.14, and has been maintained in the kernel since without increasing the version number.

3.9 Samsung MagicLAN (binary library driver)

Driver status : beta

Driver name : swld11_cs.o

Version : 1.22

Where : http://www.magiclan.com/product/magiclan/download/mlist.jsp

Maintainer : Jae-Jun Lee <brucejr@samsung.co.kr>

Documentation :	Readme
Configuration :	Module parameters, Wireless Extensions and utility
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP
Scanning :	No
Monitor :	No
Multi-devices :	yes
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Encryption, Proprietary Samsung API
Non implemented :	?
Bugs :	?
License :	Binary only for the core + (?)source wrapper
Vendor web pages :	http://www.sem.samsung.co.kr/
	http://www.magiclan.com/

3.9.1 The device

The Samsung **MagicLAN** is one of the various products based on the Intersil PrismII chipset (see *section 3.6* for full details). It's a fully featured wireless lan compliant with 802.11-b. The Compaq products are rumored to be the Samsung one, with a new sticker...

3.9.2 The driver

Samsung has released their own version of a PrismII driver for their card. The driver seems complete and well written, the new releases fixes more bugs and I had report of people successfully using it (with Samsung cards and even some LinkSys and D-Link cards).

The main difference with the PrismII driver of *Mark* (see *section 3.6*) is that the Samsung driver is based on a binary library (so, only available on x86 platforms), offer encryption and Ad-Hoc mode and offer some support for Wireless Extensions.

3.10 Symbol Spectrum24 High Rate, 3Com AirConnect, Intel PRO/ Wireless and Socket Communication

Driver status :	Beta (Pcmcia only)
Driver name :	spectrum24t_cs.o
Version :	1.03 and 1.03-CF
Where :	http://sourceforge.net/projects/spectrum24
	ftp://ftp.symbol.com/pub/SOFTWARE/IEEE/PC_CARD/LINUX/
Contact :	Brad LeFore <blefore@sj.symbol.com>
Maintainer :	Lee John Keyser-Allen <frozbiz@hotmail.com>

Discussion forums :	http://sourceforge.net/forum/?group_id=11099
Documentation :	Readme file
Configuration :	module parameters
Statistics :	None
Modes :	Managed, Ad-Hoc
Security :	WEP
Scanning :	No
Monitor :	No
Multi-devices :	-
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Multicast, WEP encryption and support for CF cards
Non implemented :	-
Bugs :	-
License :	GPL or BSD
Vendor web page :	http://www.symbol.com/products/wireless/wireless.html http://support.intel.com/support/network/wireless/ http://www.3Com.com/ http://www.socketcom.com/

3.10.1 The device

Despite being a long time proponent of Frequency Hopping, Symbol couldn't ignore the success of 802.11-b. After a strategic agreement with Intel, Symbol is back with a complete line of 802.11-b products, that are called **Spectrum24 High Rate** (to better confuse them with their old FH products). Symbol still sell mostly to vertical markets through VAR, but both 3Com and Intel are repackaging Symbol cards, as **Intel PRO/Wireless** and **3Com AirConnect**. The Symbol CF card is also sold by **Socket Communications**.

Of course, there are exceptions : the **Symbol/Socket CF** cards (Compact Flash) and the Intel **2011B** card don't have a built in firmware and require a specific version of the driver (called CF). On the other hand, the **3Com/Intel PCI cards** are *PrismII* cards, and the latest 3Com 802.11b cards include various chipsets (see *section 3.18*).

The card is mostly sold in the Pcmcia form factor, along with the Access Point. There is a PCI version that looks like a Pcmcia card in a regular PCI to Pcmcia slot. The main originality of Symbol is that it offer those famous "all-in-one" products (PDA + barcode + wireless) with 802.11-b (beware, they share the same model numbers as the non-802.11b devices). Recently Symbol released a **Compact Flash** (CF) version of their card called *Wireless Networker* which has an amazing form factor.

The Symbol product is composed of the Intersil PrismII chipset (see *section 3.6*) with Symbol own MAC controller (which is originally derived from the same

core as the MAC from Lucent, Aironet and Intersil). From Symbol, we can expect a design giving good quality and performance.

The MAC has all the usual features of the 802.11 standard, like MAC level retransmission, RTS/CTS, fragmentation, auto bit-rate selection, power saving, WEP encryption and roaming, which extensive configurability. The physical layer has the classic PrismII feature, supporting 1, 2, 5.5 and 11 Mb/s.

3.10.2 The driver

The driver was initially written by TriplePoint, and *Lee* has taken over the maintenance. Not surprisingly, the driver is very similar to the Wavelan-IEEE binary driver (except for being full source), to the point of mentioning “Turbo” cards (what Symbol calls “High Rate”).

The driver is well written, has an extensive collection of module parameters and has been tested successfully with Symbol, 3Com and Intel cards. *Lee* plans to add Wireless Extensions and fix the few remaining bugs...

The version 1.01 of the driver fixes some bugs related to higher bit rate (11 Mb/s) and encryption. The version 1.02, adds support for kernel 2.4.X and disable power management (doesn’t work on latest firmwares).

Symbol has recently release a separate version of this driver to support **Compact Flash** cards. Compact Flash cards need a specific driver because they don’t have the firmware stored on the card and therefore the driver has to download the firmware to the card after each reset.

3.11 Ericsson WLAN 11 Mb/s

Driver status :	First shot
Driver name :	eriwlan_cs.o
Version :	1.0 (2000-10-11)
Where :	http://www.ericsson.com/wlan/su-downloads11.asp
Maintainer :	Christian Olrog <Christian.Olrog@ericsson.com>
Documentation :	Readme file
Configuration :	module parameters and /proc interface
Statistics :	/proc interface
Modes :	Managed, Ad-Hoc
Security :	WEP
Scanning :	No
Monitor :	No
Multi-devices :	-
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Power management
Non implemented :	-
Bugs :	-

License : GPL

Vendor web page : http://www.ericsson.com/wlan/

3.11.1 The device

After their success with wide area communications (GSM and co.), Ericsson decided to expand in new markets and started looking seriously at local connectivity. Ericsson is of course the main driving force behind **BlueTooth** (see *section 8*), but they realised pretty quickly the BlueTooth would not fulfil the need of the Wireless LAN market. Ericsson is also pushing hard **HiperLAN II** (see *section 8*), a high performance system (54 Mb/s) in the 5 GHz band with strong quality of service support.

The initial Ericsson Wireless LAN products were OEM of **BreezeCom pro.11** products (Frequency Hopping, 3 Mb/s - see *section 2.11*). Due to the success of 802.11-b, their second product line are fully 802.11-b compliant, and are in fact OEM of the **Symbol** cards (see *section 3.10*). As such, this product has all the usual 802.11-b features...

3.11.2 The driver

This driver apply only to the 11 Mb/s version of the Ericsson card. This is only the second driver written by *Christian* from scratch, after the BreezeCom driver (see *section 2.11* - this other driver applies to Ericsson 3 Mb/s cards). And as usual for him, the source code is well written, concise and clean. Impressive job !

This driver is very new, so I don't have yet report of its use. The driver seems to support only a minimal set of configuration and statistics for now. *Christian* told me that it should work with other Symbol cards with minor changes, and that the driver has been tested with IPsec and MobileIP. I hope to have more info about it at a later date...

3.12 Symbol High Rate support in the Orinoco driver

The Orinoco driver (see *section 3.2*) may be used with most Symbol HR cards.

3.13 Symbol CF driver based on the Orinoco driver

Driver status : Beta

Driver name : spectrum_cs.o

Version : v0.15

Where : Linux kernel (2.6.18)
http://www.red-bean.com/~proski/symbol/
http://savannah.nongnu.org/projects/orinoco/

Maintainer : Pavel Roskin <proski@gnu.org>

Mailing list : http://lists.samba.org/pipermail/wireless/

Documentation : Readme file

Configuration : Wireless Extensions only

Statistics : Wireless Extensions

Modes :	Managed, Ad-Hoc and Ad-Hoc-demo
Security :	WEP (based on hardware support)
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	Yes
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Similar to Orinoco driver (including ARM support)
Non implemented :	-
Bugs :	-
License :	MPL and GPL
Vendor web pages :	[Same as Symbol HR driver]

3.13.1 The device

This is the same device as the previous entry (*section 3.10*).

Note that this driver is specific to the **Compact Flash** (CF) version of the card and the **Intel 2011B**.

3.13.2 The driver

The Orinoco driver (see *section 3.2*) already includes support for regular Symbol HR cards. However, the Compact Flash cards don't work with the standard Orinoco driver because they lack built in firmware. This is the same reason why there is two different Spectrum24 drivers (see *section 3.10*).

Pavel has created a new driver based on the Orinoco driver and Spectrum24-CF driver for those cards. It is similar to the regular Orinoco Pcmcia driver, but add the **firmware download** at each reset necessary for those cards. As the core of the driver is common with the **Orinoco driver**, this driver has the exact same feature set (which is quite extensive - see *section 3.2*). This driver is now integrated in the Orinoco driver collection (v0.14 and later), and no longer distributed separately. It is now part of the Linux kernel (2.6.14 and later).

Note that for cards that don't require the firmware download (regular Pcmcia cards), it is recommended to use the regular Orinoco driver instead of this one.

3.14 Aironet ARLAN 4500, 4800, Cisco 340 and Cisco 350 series

Driver status :	stable
Driver name :	ISA, PCI : airo.o
	Pcmcia : airo_cs.o
Version :	1.4
Where :	Linux kernel (2.6.17)
	Pcmcia package (3.1.26)
Maintainers :	Benjamin Reed <breed@almaden.ibm.com>
	Javier Achirica <achirica@gmail.com>

	Dan Williams <dcbw@redhat.com>
	Matthieu Castet <castet.matthieu@free.fr>
Web pages :	http://sourceforge.net/projects/airo-linux/
Mailing list :	http://sourceforge.net/mail/?group_id=24926
Documentation :	README file
Configuration :	/proc interface and Wireless Extensions
Statistics :	/proc interface and Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP (hardware), AES (host), MIC, 802.1x, WPA
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	N/A
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Support MPI cards (Mini PCI)
Non implemented :	-
Bugs :	-
License :	MPL & GPL
Vendor web page :	http://www.aironet.com/

3.14.1 The device

Aironet has been the producer of some of the most performant wireless LANs for a long time. Aironet was a division of Telxon, and was spun-off when Symbol, one of their competitor, did acquire Telxon. After a short independent life, Aironet was acquired by Cisco.

The previous section was dealing with Aironet old pre-802.11 products (see *section 2.8*), this section deals with their more recent 802.11 compliant products. Their first 802.11 products were the 3500 family, Frequency Hopping (1 and 2 Mb/s), and 4500, Direct Sequence (1 and 2 Mb/s).

The Arlan **4500** family is 802.11 compliant wireless LANs in the 2.4 GHz ISM band, and is Direct Sequence. It includes an ISA, PCI, Pcmcia, serial, Ethernet and multi-Ethernet versions, plus the Access Point.

These cards are based on the Harris **Prism** chipset, like many other cards (see *section 2.4*), but Aironet are using their own MAC controller. The 4500 offer standard 1 and 2 Mb/s bit rate. The MAC includes all the standard 802.11 features, with Power Saving, WEP, Ad-Hoc mode and roaming, plus a lot of Aironet extensions (short headers, variable base rate...). Conform to their reputation, their MAC is one of the richest in term of features, and one of the most performant.

The **3500** family (Frequency Hopping) eventually died, and I won't talk about it here.

The 4500 family was quickly followed by the **4800** family, still based on the Prism chipset, adding 5.5 and 11 Mb/s bit rate, either in MBOK (proprietary) or

CCK, which is 802.11-b compliant. The 4800 can do encryption only at 1 and 2 Mb/s (this limitation was removed in the 4800B).

With introduction of the **PrismII** chipset, Aironet did release the **4800B** family. It is functionally equivalent to the 4800, except that the new PrismII chipset allows lower price, greater sensitivity but force a lower transmit power (30 mW). Aironet still use their own MAC controller in the 4800B (and not the new PrismII MAC - see *section 3.6*).

After the acquisition by Cisco, the Aironet 4800B was renamed **Cisco 340** series (exact same hardware, new name). Dell also sell the same hardware under its own brand as **Dell TrueMobile 1100** (on the other hand, the TrueMobile 1150 is a Wavelan IEEE).

Like Lucent, Cisco offer different cards with different level of encryption. The cards labelled **340** feature no encryption, the cards labelled **341** feature 40 bits encryption and the cards labelled **342** feature 128 bits encryption. Moreover, some versions of the Pcmcia card are sold with antenna but others without antennas.

Cisco has now released the **Cisco 350**, a new family of 802.11b cards. From the information I did gather, it seems to be equivalent to the 340 series with a greater transmit power (100 mW instead of 30 mW). The Cisco 350 also improves the performance of the AP and introduce greater security (Radius authentication and co).

Cisco has also released a **Mini-PCI (MPI)** version of the Aironet 350, to be added in laptops that support a Mini-PCI slot. For some strange reason, this hardware is slightly different from the regular Aironet 350 PCI.

Cisco has also a wide range of **IEEE 802.11g** products, those are completely different from this hardware, and most often they are Atheros cards (see *section 4.2*).

3.14.2 The driver

Ben has produced a solid driver for the Aironet card, The driver supports the ISA, PCI and Pcmcia cards (4500, 4800 & 4800B versions), it looks fairly complete and debugged, with a nice /proc interface. The driver also has very complete WEP support.

Ben also told me that the driver was able to recognise the PC3500 cards, but more work would be needed there to get it fully working.

Recently, I've started adding Wireless Extension to this driver. *Ben* was kind enough to integrate properly my work in his driver. Then, *Javier Achirica* did an amazing job of completing Wireless Extension support (power management, spy and co), and this driver has one of the most complete Wireless Extension support of all.

Then, *Javier* added to the driver the Cisco proprietary API, which allow communication with Cisco utilities (see *section 3.16*) and, amongst other things, flashing new firmware on the card. All this amazing work is in the latest release from *Ben* (1.5). He also wrote a couple of open source utilities allowing to dump all the register of the card and to flash new firmwares through this API.

Later, the driver has been integrated in the Linux kernel (2.4.6 and later) and moved to SourceForge. *Javier* has also added the ability to dump raw 802.11 frames. Then *Javier* did extensive work to fix locking (SMP support), add monitor mode and Wireless Scanning support (in version 1.4).

Ben attempted to add support to MPI card and added code for those cards in the CVS. The work on MPI card was completed and now MPI cards are properly supported.

Ben and *Javier* are no longer active. *Dan* has fixed various bug and kept the driver up to date in the kernel, and *Matthieu* has added WPA support.

3.15 Aironet ARLAN 802.11 (alternate driver)

Driver status :	stable
Driver name :	ISA, PCI : aironet4500_card.o
	Pcmcia : aironet4500_cs.o
Version :	0.1
Where :	Linux kernel 2.3.31 to 2.5.X
Maintainer :	Elmer Joandi <elmer@linking.ee>
Documentation :	Configure.help file
Configuration :	/proc interface
Statistics :	/proc interface
Modes :	Managed, Ad-Hoc
Security :	WEP
Scanning :	?
Monitor :	No
Multi-devices :	-
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	-
Non implemented :	Pcmcia interface
Bugs :	Buggy SMP support.
License :	GPL
Vendor web page :	http://www.aironet.com/

3.15.1 The device

This is the same device as the previous entry (*section 3.14*).

3.15.2 The driver

To some, it may seem that this is a totally new driver that has just popped up in the kernel with little warning. In fact, *Elmer* had developed this driver for a commercial company (SpectrumWireless) a while back and they agreed to let him release it in GPL form after some month.

The code is very complete, especially the /proc interface. It comes as four modules, the generic core, the /proc interface, the PCI/ISA interface and the Pcmcia interface. The driver support both the 4500 and 4800 families. Unfortunately, the Pcmcia interface is incompatible with the Linux Pcmcia support and doesn't work well.

Elmer told me that compared to *Ben* driver, his driver was probably more robust and featured but much less friendly. In essence, the focus was slightly different, so each driver has it own strength.

This driver was **removed** from the Linux kernel during 2.5.X, so it is no longer available in kernel 2.6.X.

3.16 Cisco/Aironet 802.11 (Cisco driver)

Driver status :	stable (rock solid)
Driver name :	ISA, PCI : airo.o
	Pcmcia : airo_cs.o
Version :	2.1
Where :	http://www.cisco.com/public/sw-center/sw-wireless.shtml
Maintainer :	Cisco
Documentation :	Text files
Configuration :	Cisco utilities, Wireless Extensions
Statistics :	Cisco utilities, Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP (hardware), AES (host), MIC
Scanning :	No
Monitor :	No
Multi-devices :	-
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	-
Non implemented :	-
Bugs :	-
License :	Cisco open source license
Vendor web page :	http://www.aironet.com/

3.16.1 The device

This is the same device as the two previous entries (*section 3.14*).

3.16.2 The driver

Recently, *Cisco* decided to get more involved with supporting their Wireless LAN cards under Linux. Rather than developing an entirely new driver, they decided to base their work on *Ben*'s driver (*section 3.14*), which is a good idea. One of the key person behind this operation was *Jim Veneskey*.

The main contribution of Cisco is a proprietary API, which allow communication with Cisco utilities and, amongst other things, flashing new firmware on the card, and of course a set of utilities which are mostly identical to the Windows utilities. They also provided nice installation scripts and did lot's of testing of the driver to guarantee its stability (*Cisco* usually do some pretty intensive testing of their products).

However, even if *Cisco* regularly synchronise with *Ben*'s driver (*section 3.14*), this one continues to improve. As they are derived from the same base, it's easy to compare the two drivers. In term of features, I guess that *Ben*'s driver is winning, because it now has the Cisco API of this driver and more complete Wireless Extensions support. However, I believe that *Cisco* has an edge in term of stability.

I hope that the two drivers will merge rather than diverge, and that changes will be propagated from one to the other, so that we have a driver with both features and rock solid stability, but only time will tell... *Cisco* told me that they were going to try to catch up with *Ben*'s driver.

Note that *Cisco* changed their web site and their driver is no longer available as a public download. I can not check if the driver still exist and what changed has been made to it.

3.17 No Wires Needed Swallow (and BreezeCom DS11)

Driver status :	stable
Driver name :	swallow_cs.o
Version :	0.7.0 (kernel 2.4.0) and 0.4.0 (kernel 2.2.16 - older version)
Where :	http://www.xs4all.nl/~bvermeul/swallow/
Maintainer :	Bas Vermeulen <bvermeul@blackstar.xs4all.nl>
Documentation :	README file
Configuration :	Module parameters, Wireless Extensions, /proc interface
Statistics :	no
Modes :	Managed
Security :	WEP, AES
Scanning :	Wireless Extensions
Monitor :	No
Multi-devices :	unknown
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Security (very complete), roaming table
Non implemented :	Multicast
Bugs :	-
License :	GPL
Vendor web page :	http://www.nwn.com/
	http://www.breezecom.com/

3.17.1 The device

No Wires Needed is a small company in the Netherlands building a range of 802.11 DS devices, including a Pcmcia card (Swallow), an Access Point and a Hub. They also offer a ISA version using a ISA to Pcmcia bridge. They have two version, the **550** (5.5 Mb/s) and the more recent **1100** (11 Mb/s). BreezeCom also EOM this card for their DS.11 range (the **PC-DS.11**).

The Swallow delivers all the features expected from a 802.11 compliant device, with ad-hoc networking, authentication and roaming. The main difference with other 802.11 devices is that NWN offers some strong link layer encryption and a key management and distribution system.

The modem is the famous Prism chipset used in many other cards (see *section 2.4*), which is 2.4 GHz Direct Sequence, with 1 Mb/s, 2 Mb/s, 5.5 Mb/s and 11 Mb/s bit rate. No Wires Needed use their own MAC design on an embedded ARM processor, and not the AMD or PrismII MAC controller. This give them more performance and flexibility. Now that Intersil has acquired No Wires Needed, Intersil can offer 2 different 802.11 MAC controller !

3.17.2 The driver

Bas has implemented a quite complete driver for the Swallow 550 and 1100 card (Pcmcia). He has patiently debugged the driver to fix races, timeouts and increase the performance. The driver is working for both the NWN and the BreezeCom cards.

Bas has also implemented Wireless Extension support for the security support, and support the full range of security features in the driver. You can also configure the ESSID on the fly with Wireless Extensions...

Lately, *Bas* has been doing lot of work on roaming support. The driver export the roaming tables in a /proc interface, allowing the implementation of a user space roaming daemon. This interface also contains some other configuration parameters.

3.18 No Wires Needed 1148, old 3Com Wireless LAN XJack

Driver status :	stable
Driver name :	poldhu_cs.o
Version :	0.2.13 (for kernel 2.4.X), 0.3.1 (for kernel 2.6.X)
Where :	http://www.xs4all.nl/~bvermeul/swallow/
Maintainer :	Bas Vermeulen <bvermeul@blackstar.xs4all.nl>
Documentation :	README file
Configuration :	Module parameters, Wireless Extensions, /proc interface
Statistics :	no
Modes :	Managed, Ad-Hoc
Security :	WEP, AES
Scanning :	Wireless Extensions
Monitor :	No

Multi-devices :	unknown
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Security (very complete).
Non implemented :	Multicast
Bugs :	-
License :	GPL
Vendor web page :	http://www.nwn.com/
	http://www.3Com.com/

3.18.1 The device

Recently, No Wires Needed has replaced the Swallow 550 and 1100 with a new 802.11-b card, the **1148**. The design and the feature of this card seems very similar to their previous one (see *section 3.17*) : The MAC is the same ARM core and they still offer AirLock encryption. The main difference seems that they are now using a PrismII modem (see *section 3.6*).

3Com has also released a quite rare **old Wireless LAN XJack** card (3CRWE62092A) which is a clone of the NWN 1148 (this card). On the other hand, the newly released **OfficeConnect** Wireless LAN cards (3CRSHPW196/696 - with or without the XJack antenna) and **new Wireless LAN XJack** card (3CRWE62092B) are *Atmel* cards (see *section 3.20*). The even newer **OfficeConnect CardBus** card (3CRSHPW796) is an *ADMTek* card (see *section 3.23*). The older **AirConnect Pcmcia** cards (3CRW737A/B) are clone of the *Symbol HR* cards (see *section 3.10*), and the **AirConnect PCI** card (3CRW777A) is a *PrismII PLX* card (see *section 3.6*). The **OfficeConnect 11g** card (3CRWE154G72) is an *Intersil PrismGT* card (see *section 4.3*), and the **11a/b/g** cards (3CRPAG175/3CRDAG675) are *Atheros* cards (see *section 4.2*).

Coming back to the NWN 1148, the main difference between the 3Com 3CRWE62092A and NWN 1148 cards is the removal of AirLock encryption (WEP is still available), the addition of Ad-Hoc mode, and of course the famous pop-up antenna ;-)

3.18.2 The driver

Because the card is so similar to the Swallow, it was natural that a driver for this card would be derived from the Swallow driver. In fact, No Wires Needed contacted *Bas* to implement a driver for the new card. *Bas* modified his Swallow driver and created this driver which offer very similar functionality and feature as the Swallow driver (see *section 3.17*).

So, the driver includes complete security support, Wireless Extensions and roaming support. The driver also include read only support for most low level commands (SNWNMP).

Bas has also added in the driver the necessary support for the 3Com WLAN XJack cards, including its specific features. All features of the card (ad-hoc mode, encryption) are configurable through Wireless Extensions.

3.19 Nokia C110/C111

Driver status :	???
Driver name :	nokia_c110.o
Version :	2.05
Where :	http://www.nokia.com/nokia/0,5184,2718,00.html
Maintainer :	Nokia
Documentation :	Readme
Configuration :	Specific Pcmcia scripts
Statistics :	/proc file
Modes :	Managed, Ad-Hoc
Security :	WEP
Scanning :	No
Monitor :	No
Multi-devices :	yes
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Encryption, multicast, user profiles
Non implemented :	?
Bugs :	?
License :	Binary only for the core + Nokia OpenSource Linux wrapper
Vendor web pages :	http://www.nokia.com/corporate/wlan/

3.19.1 The device

In the past years, Nokia has slowly moved into the Wireless LAN market. They started by buying *Intalk*, a company producing PrismI clones (see *section 2.4*), and Nokia continued selling those cards, renamed Nokia **C020** and **C021**. While being busy working on BlueTooth and HiperLanII, Nokia didn't forget 802.11-b and released a new set of card, the **C110** and **C111**.

As can be expected, the Nokia C110/C111 is another PrismII clone (see *section 3.6*). On the other hand, it seem that Nokia has changed quite a few things compared to the original PrismII design, for example they have added a Smart Card reader on the Pcmcia card (for security settings).

3.19.2 The driver

Nokia initially quietly made this driver available in the registered only part of their web site and didn't mention it anywhere, fortunately I have my spies to inform me ;-). Later on, they moved this driver to their official driver page on their public web site. This driver supports only the C110/C111 cards, on the other hand the C020 is supposed to work with the linux-wlan package (see *section 2.4*).

The driver contains a very thin source wrapper on top of the binary part (one version for kernel 2.2.X, one for 2.4.X). On the other hand, the package come with exhaustive set of complex Pcmcia scripts to configure the card and enable profiles.

The driver works in infrastructure and Ad-Hoc mode, and support WEP.

3.20 Atmel AT76C502A/AT76C503A cards (802.11b USB and Pcmcia)

Driver status :	stable
Driver name :	Pcmcia : fastvnet_cs.o
	USB : vnetusbX.o
Version :	v3.4.1.0 ; 2002-12-09
Where :	http://atmelwlandriver.sourceforge.net/
Maintainers :	Stavros Markou <smarkou@patras.atmel.com>
	Titos Mpetsos <tmpetsos@patras.atmel.com>
	Ron Smith (Wireless Extensions)
Web pages :	http://atmelwlandriver.sourceforge.net/howto/howto.html
	http://www.fuw.edu.pl/~pliszka/hints/wireless.html
	http://www.gemtek.com.tw/faq_download.htm
	http://www.houseofcraig.net/belkin_howto.php
Mailing lists :	http://sourceforge.net/mail/?group_id=59001
	http://iprserv.jura.uni-leipzig.de/mailman/listinfo/atmel-wlan-usb
Documentation :	Readme files
Configuration :	Specific tools, Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP, WPA
Scanning :	No
Monitor :	No
Multi-devices :	yes
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	-
Non implemented :	Intersil radio support
Bugs :	-
License :	GPL
Vendor web pages :	http://www.atmel.com/atmel/products/prod32a.htm
	http://www.3com.com/
	http://www.gemtek.com.tw/
	http://www.dlink.com/products/
	http://www.linksys.com/products/
	http://www.smc.com/

3.20.1 The device

Atmel has decided to join the selected club of company selling 802.11b chipsets. Their design is based on an ARM7 processor, and offer a Pcmcia and a USB version. The USB version is by far the most popular, because it was one of the first USB chipset to market and is cheap. Various 802.11b radios can be used with this chipset (such as the Intersil PrismII radio and Atmel RFMD). Very little public information is available about those products, but as they are 802.11 compliant we can expect the usual set of features.

Of course, with any USB cards the main issue is performance. The streaming abstraction of USB doesn't work well with register/memory based chipset designs and slow down operations. Also, USB add a noticeable latency. I currently don't have any data on how well those cards performs in the respect.

Linksys and D-Link are selling various USB products based on this chipset, such as Gemtek **WL-280**, D-Link **DWL-120**, Linksys **WUSB11**... It seems that the SMC **2632W** is a Pcmcia card also based on the Atmel chipset.

The newly released **3Com OfficeConnect** Wireless LAN cards (3CRSHPW196/696 - with or without the XJack antenna) and new **3Com Wireless LAN XJack** (3CRWE62092B) are also Atmel Pcmcia cards.

3.20.2 The driver

Atmel decided to release themselves a Linux driver. This driver is in fact a set of driver for the different combination of controllers and radios, and was initially based on a binary library. However, *Atmel* changed their mind and eventually released the driver as full source GPL. The USB versions work with both uhci and ohci USB drivers. The driver support WEP and Ad-Hoc mode.

The package from *Atmel* include specific configuration tools (command line and X-Window). Just after the driver was GPL'ed, the support for Wireless Extensions was fixed and greatly enhanced by *Ron* (Wireless Tools can now be used to configure the card). Lately, *Atmel* has improved and enhanced the USB driver, but also removed support for Intersil radio, so you will need to use older version of the driver for products using Intersil radio. On the other hand, the Pcmcia driver seems to be still based on an older version of the code.

Lately, the driver has gained support for kernel 2.6.X, better support for Wireless Extensions and support for WPA.

3.21 Atmel USB alternate driver

Driver status :	in development
Driver name :	USB : at76_usb.o
Versions :	0.16 (for kernel 2.6.X), 0.12 (for kernel 2.4.X)
Where :	http://at76c503a.berlios.de/
	http://git.80211libre.org/at76_usb.git/
Maintainers :	Jörg Albert <joerg dot albert at gmx dot de>
	Oliver Kurth <oku at masqmail dot cx>

	Guido Guenther <agx@sigxcpu.org>
	Pavel Roskin <proski@gnu.org>
Web pages :	http://at76c503a.berlios.de/
	http://www.wireless.org.au/~jhecker/atmeldrv/
Mailing lists :	http://lists.berlios.de/pipermail/at76c503a-user/
	https://lists.berlios.de/pipermail/at76c503a-develop/
Documentation :	Readme files
Configuration :	Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP
Scanning :	Wireless Extensions (0.12 and later)
Monitor :	No
Multi-devices :	?
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Intersil radio support
Non implemented :	-
Bugs :	-
License :	GPL
Vendor web pages :	http://www.atmel.com/atmel/products/prod32a.htm
	http://www.gemtek.com.tw/
	http://www.dlink.com/products/
	http://www.linksys.com/products/

3.21.1 The device

This is the same device as the previous entry (*section 3.20*).

3.21.2 The driver

Oliver and a group of people were not happy about the official Atmel driver and its development (*section 3.20*), and therefore he started to write an alternate driver from scratch, based on the information available in the official driver, and soon those other people started to help him.

The driver support only USB devices, and it support both Intersil and RFMD radios. At this time, the new driver is still in development, and therefore still has some limitations, but it already support both infrastructure and ad-hoc mode, and has some Wireless Extension support.

As *Olivier* was no longer active, *Jörg* took over the driver and is now fixing many bugs, making it more robust and adding a few missing features. For example, version 0.12 adds Wireless Scanning support, and is the last version to support kernel 2.4.X.

When *Jörg* became inactive, *Guido* and *Pavel* took over the driver. Most of their work in subsequent releases (0.13 to 0.16) was all the driver cleanup necessary to be able to integrate the driver in the kernel (this integration has not happened yet).

3.22 Atmel PCI and Pcmcia alternate driver

Driver status :	in development
Driver name :	PCI : atmel_pci.o
	Pcmcia : atmel_cs.o
Version :	0.9
Where :	Kernel 2.6.20
Maintainer :	Simon Kelley <simon@thekelleys.org.uk>
Mailing list :	http://lists.berlios.de/pipermail/at76c503a-user/
Documentation :	Readme files
Configuration :	Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP
Scanning :	Wireless Extensions
Monitor :	No
Multi-devices :	?
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Firmware loading via HotPlug
Non implemented :	-
Bugs :	-
License :	GPL
Vendor web pages :	http://www.atmel.com/atmel/products/prod32a.htm
	http://www.3Com.com/
	http://www.smc.com/

3.22.1 The device

This is the same device as the previous entry (*section 3.20*).

3.22.2 The driver

Simon also was not happy with the original Atmel driver. He decided to rewrite the Pcmcia/PCI driver based on the original code, with the main target being integration in **kernel 2.6.X**, which was done on 2.6.5. *Simon* did an extensive amount of work on the driver, for example his driver has very complete Wireless Extension support including Wireless Scanning. However, this driver is not compliant with kernel 2.4.X and earlier.

This driver was mostly tested with Pcmcia cards, however *Simon* also managed to get a PCI card for testing and added PCI support in the driver. Some of those cards don't have firmwares and therefore need to use the new **firmware uploading** facility of Linux (via HotPlug).

3.23 ADMtek ADM8211 based cards

Driver status :	beta
Driver name :	PCI & Cardbus : 8211.o
Version :	v1.05
Where :	http://www.admtek.com.tw/index/index/Download.htm
	http://www.espina.info/papers/officeconnect/
Contact :	ADMtek support <NIC_tech_support@admtek.com.tw>
Maintainers :	?
Documentation :	Readme files
	http://www.houseofcraig.net/belkin_howto.php
Configuration :	Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP
Scanning :	No
Monitor :	No
Multi-devices :	?
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	PCI busmaster
Non implemented :	-
Bugs :	?
License :	Binary only for the core + OpenSource Linux wrapper
Vendor web pages :	http://www.admtek.com.tw
	http://www.infineon.com/
	http://www.dlink.com/products/

3.23.1 The device

ADMtek is a small taiwanese company that designed a single chip 802.11b MAC solution. Very little is known about it (for example which radio is used with the MAC), but we can expect compliance with the 802.11b specification and the usual feature set. One of the main feature of this chipset is that is uses a PCI busmaster interface (as opposed to the Intersil PrismII that uses ISA PIO - *section 3.6*), which result in much lower CPU utilisation (and potentially higher performance).

Most **Cardbus 802.11b** cards are based on this chipset. The chipset is starting to appear in products, such as some D-Link 802.11b Cardbus and PCI cards, Belkin PCI cards, the SMC 2602W V2 and the 3Com 3CRSHPW796.

ADMtek was bought by **Infineon** in 2004. Since then, their web site disappeared, and the Infineon web site has no information on their wireless LAN products. They seem to have shifted their focus towards 'system on a chip' and don't seem to make standalone 802.11 modules anymore.

3.23.2 The driver

ADMtek has released a Linux driver based on a binary library, that can be compile for different distributions. The driver has complete support for Wireless Extensions, and many people have reported success using it.

Eduardo made a patch to add support for the 3Com 3CRSHPW796 to this driver.

3.24 ADMtek ADM8211 full source driver

Driver status :	pre-release
Driver name :	PCI & Cardbus : adm8211.o
Version :	20060111
Where :	http://aluminum.sourmilk.net/adm8211/
Maintainers :	Michael Wu <flamingice@sourmilk.net>
	Jouni Malinen <jkmaline@cc.hut.fi>
Documentation :	-
Configuration :	Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	?
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	PCI busmaster
Non implemented :	-
Bugs :	?
License :	GPL
Vendor web pages :	http://www.admtek.com.tw
	http://www.dlink.com/products/

3.24.1 The device

This is the same device as the previous entry (*section 3.23*).

3.24.2 The driver

Michael took a driver originally started by *Jouni* (of HostAp fame - *section 3.8*), completed it, debugged it and finally released it. The main difference with ADMtek driver is that this driver is fully OpenSource and support only kernel 2.6.X.

The driver is still young, but its gaining feature and stability with each release, and is well written. Recently it has gained support for monitor and Ad-Hoc mode, and many bugfixes and cleanups. You should contact *Michael* if you want to help.

The initial version of the driver uses standalone 802.11 code. *Michael* is porting the driver to the new **mac80211** kernel stack (see *section 4.9*). This alternate version is only available in the wireless-dev GIT repository, and should appear in a Linux kernel in the future.

3.25 Realtek RTL8180L cards

Driver status :	?
Driver name :	PCI : rtl8180_24x.o
Version :	1.5
Where :	http://www.realtek.com.tw/downloads/
Maintainers :	ShuChen <shuchen@realtek.com.tw>
Documentation :	Readme files
	http://www.alumni.caltech.edu/~rbell/Realtek8180.html
	http://www.linuxquestions.org/questions/showthread.php?s=&threadid=61832&perpage=15&pagenumber=1
Configuration :	Private Extensions
Statistics :	?
Modes :	Managed, Ad-Hoc, Master
Security :	WEP
Scanning :	No
Monitor :	No
Multi-devices :	?
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	-
Non implemented :	-
Bugs :	?
License :	Binary only driver + OpenSource Linux wrapper
Vendor web pages :	http://www.realtek.com.tw/

3.25.1 The device

Realtek is a taiwanese company known for its Ethernet cards, which has recently released a 802.11 chipset. This chip seems to have all the usual 802.11 features, and offer only a PCI/CardBus interface (for lower CPU utilisation).

3.25.2 The driver

Realtek has released a binary driver for their device, with precompiled version for Red-Hat and SuSE. The driver doesn't use Wireless Extension (but some private extensions), and has a strange setup procedure (manual enable, require proper commands in the proper sequence).

I didn't try this driver, but I got some feedback from many users. The driver version before 1.2 were difficult to install and prone to crash the kernel. The version 1.2 had many troubles (such as reassociation), but was working. Version 1.3 is more stable, but a little too talkative.

Version 1.4 of the driver also adds support for making the card an Access Point. Version 1.5 supports newer version of gcc and newer kernels.

3.26 Realtek RTL8180L full source driver

Driver status :	Beta
Driver name :	r8180.o
Version :	0.22
Where :	http://rtl-wifi.sourceforge.net/
	http://rtl8180-sa2400.sourceforge.net/
Maintainers :	Andrea Merello <andreamrl *at* tiscali.it>
	Hauke Mehrtens <hauke@hauke-m.de>
Discussion forums :	https://sourceforge.net/forum/?group_id=186406
Documentation :	Readme files
Configuration :	Wireless Extensions
Statistics :	?
Modes :	Managed, Ad-Hoc
Security :	WEP
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	?
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Philips and Maxim radio support
Non implemented :	SMP, non-i386
Bugs :	-
License :	GPL
Vendor web pages :	http://www.realtek.com.tw/

3.26.1 The device

This is the same device as the previous entry (*section 3.25*).

3.26.2 The driver

The *Realtek* driver is not the most user friendly, and many people can't manage to make it work on their configuration. This lead *Andrea* to write his own driver for the Realtek chipset. After much hard work by *Andrea*, his driver is now functional, and evolving quickly.

This driver is fully OpenSource and the source is much closer to the standard of other Linux drivers. It support PCI and Pcmcia card with the Philips radio (the most common), and has experimental support for the Maxim radio. It reuses the WEP implementation of the Centrino driver (*section 3.28*), which is well tested and featured. It supports the latest kernel 2.6.X and 2.4.X, has complete support for Wireless Extensions, and support monitor mode, so you can see how much *Andrea* has been busy lately...

During early 2005, *Andrea* decided to reorganise the driver, and created a new branch of the driver called **rtl818x-newstack**. This branch is based on the **Intel ieee80211 stack** from the Centrino driver (see *section 3.28*). The goal of this branch is better integration in the kernel, and also support for the newer Realtek 802.11g cards (see *section 4.16*). It seems that *Realtek* is also taking a more active role in supporting *Andrea* and this driver.

Mid 2005, *Andrea* stopped updating his project, leaving his work on the new branch unfinished. Beginning 2007, a group of users lead by *Hauke* restarted working on the driver, with a new project page and new repository. Their main work so far has been updating the driver to work with the latest kernel.

3.27 Ralink RT2400 cards (Minitar driver)

Driver status :	Stable
Driver name :	stable : rt2400.o
	alpha : rt2x00.o
Version :	1.2.2 (stable) and 2.0.4 (pre-alpha)
Where :	http://rt2x00.serialmonkey.com/
	http://rt2400.sourceforge.net/
	http://minitar.com/index.php?maincat=download
	http://flavio.stanchina.net/debian/rt2400.html
Maintainers :	Paul Lin
	Mark Wallis <markwallis at users.sourceforge.net>
	Flavio Stanchina <flavio_AT_stanchina.net>
	Ivo van Doorn <ivd at euronet.nl>
Mailing list :	http://sourceforge.net/mail/?group_id=107832
Documentation :	Text files, Howtos
Configuration :	Wireless Extensions and specific graphical tool

Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP
Scanning :	Yes (specific tool)
Monitor :	Yes
Multi-devices :	?
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Promisc/bridge mode support
Non implemented :	-
Bugs :	SMP problems
License :	GPL
Vendor web pages :	http://www.ralinktech.com/
	http://minitar.com/
	http://rt2x00.serialmonkey.com/wiki/index.php/Hardware

3.27.1 The device

Ralink is yet another taiwanese company having recently released a 802.11b chipset. This chip seems to have all the usual 802.11b features, and offer only a PCI/CardBus interface (for lower CPU utilisation).

3.27.2 The driver

Ralink created the original Linux driver for this card, and *Minitar* released it as fully OpenSource and started helping driver users on their forums. This driver support the Wireless Extensions, and also comes with a dedicated graphical utility. *Flavio* created a Debian package for this driver, and documented how to install and use this driver.

Then, *Mark* created a SourceForge project for the driver and started to maintain it, helped by many users of the driver. Many patches have been integrated to improve the stability and functionality of the driver.

Ivo has started a rewrite of the driver, called **rt2x00**, his goal is to have a source code easier to integrate in the Linux kernel and to maintain. His rewrite targets both the RT2400 and the new RT2500 (*section 4.11*). The initial version of this new driver was using the **Intel ieee80211 stack** from the Centrino driver (see *section 3.28*).

Then, *Ivo* ported the rt2x00 driver to the new **mac80211** stack (see *section 4.9*), and lots of development has happened on that version of the driver. This alternate version is available in the wireless-dev GIT repository and in the CVS, and should appear in a Linux kernel in the future.

3.28 Intel PRO/Wireless 2100 802.11b (Centrino)

Driver status :	Beta
Driver name :	ipw2100.o

Version :	1.2.2
Where :	Linux kernel (2.6.19)
	http://ipw2100.sourceforge.net/
	http://ieee80211.sourceforge.net/
Maintainer :	James P. Ketrenos <ipw2100-admin@linux.intel.com>
Mailing list :	http://lists.sourceforge.net/lists/listinfo/ipw2100-devel/
Documentation :	Readme
Configuration :	Wireless Extensions
Statistics :	/proc interface and Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP, 802.1x, WPA
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	?
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Firmware loading via HotPlug
Non implemented :	-
Bugs :	?
License :	GPL
Vendor web page :	http://www.intel.com/

3.28.1 The device

The Intel PRO/Wireless 2100 is the second generation wireless LAN product from Intel. Intel initially sold various wireless LAN cards and Access Points from Symbol as Intel **PRO/Wireless 2011** (*section 3.10*). However, Intel has a long experience with Ethernet MAC controllers, so it was natural for them to create their own wireless LAN chips. The first MAC controller created by Intel is the PRO/Wireless 2100, and we can expect many more products to come.

The **PRO/Wireless 2100** MAC controller is fully IEEE 802.11b compliant and has all the usual 802.11b features. It's a PCI Busmaster design, for low CPU utilisation. The Radio most likely is done in collaboration with Symbol (Intel and Symbol have one of those strategical alliances).

The main difference between this chipset and the vast number of other 802.11b chipset is Intel's marketing. The PRO/Wireless 2100 is not sold as a standalone product, but only as part of the **Centrino** package. Centrino is just a marketing exercise, only laptops equipped with a Intel processor, an Intel chipset and a Intel PRO/Wireless card can carry the Centrino logo. The PRO/Wireless 2100 is a regular MiniPCI card (it's not integrated on the motherboard as some rumors suggests), and you can replace the PRO/Wireless 2100 of Centrino laptops by any other MiniPCI wireless LAN of your choice.

Intel has been quite successful, and a large number of recent laptops includes a PRO/Wireless 2100 MiniPCI card. In fact, if you buy a **laptop with integrated 802.11b** support (without 802.11g or 802.11a support), it most likely has a PRO/ Wireless 2100 card. On the other hand, the PRO/Wireless 2100 is not available as a separate Pcmcia, Cardbus, PCI or USB card.

3.28.2 The driver

The Linux driver for the PRO/Wireless 2100 was released a long time after the card themselves. For this reason, Intel did receive a lot of harsh criticism from people buying laptops with this card and finding that there was no driver or documentation for it. This was a bit unfair, as Intel has always been supportive of Linux, for example most drivers for Intel Ethernet chipset are directly maintained by Intel, and producing good driver takes time. Eventually, Intel released a fully Open Source driver, which they support directly.

By the time *James* released the driver, it was already functional, but limited. Since them, many people have contributed fixes and enhancements, and development is happening quickly. Wireless Scanning was added. WEP support was added by borrowing code from HostAP. The driver require firmware loading, and the firmware loading procedure was converted to use the standard HotPlug firmware loading support. Support is also being added for the wireless on/off button of many specific laptops.

The Linux driver require a **specific firmware** which Intel says is identical to the Windows firmware but packaged differently. This firmware mostly make sure that Linux users don't do things breaking various radio regulations. Intel must be applauded for taking the effort of releasing this specific Linux firmware (and their updates), this is what enables the driver to be fully OpenSource and forced other driver to be binary, such as the Atheros driver (*section 4.2*).

James and various other contributors have been very hard at work on the driver and many bugs have been fixed. After that, most of the work has been in the area of ad-hoc and monitor mode, WPA support and power management.

Around summer 2005, *James* decided to spin-off the code he had borrowed from the HostAP driver in a separate **Intel ieee80211 stack**, shared by all Intel drivers (see *section 4.6* and *section 4.8*). The Intel ieee80211 stack was included in **kernel 2.6.14**, alongside this driver, and has seen continuous improvements after this. This stack only handles 802.11 framing and 802.11 encryption, but not the SoftMAC function (802.11 management), as those drivers don't require it. A separate SoftMAC layer was created for this stack (see *section 4.13*), but none of the Intel driver uses it.

3.29 ZyDAS ZD1201 driver (USB dongles)

Driver status :	Stable
Driver name :	zd1201.o
Version :	0.15
Where :	Kernel 2.6.12
	http://linux-lc100020.sourceforge.net/

Maintainer :	Jeroen Vreeken <pe1rxq@amsat.org>
Forums :	http://sourceforge.net/forum/?group_id=94356
Documentation :	Web page
Configuration :	Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	?
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	-
Non implemented :	-
Bugs :	-
License :	GPL
Vendor web pages :	http://www.zydas.com.tw/
	http://www.sweex.com/

3.29.1 The device

ZyDAS is yet another taiwanese company having recently released a 802.11b chipset. This chip seems to have all the usual 802.11b features, and offer a PCI. Pcmcia and USB interface. For some reasons, it seems to have only been used in various USB dongles.

3.29.2 The driver

Originally *Sweex* distributed a Linux driver for their ZyDAS based USB dongles, but suddenly they retired their Linux drivers from their support site. This driver was based on the linux-wlan-ng driver (see *section 3.6*). The driver was then improved by the Linux users.

Jeroen rewrote the driver to get rid of its dependency on linux-wlan-ng, and created a much smaller and simpler driver. This driver was further improved and included in kernel 2.6.12. Despite its very small size, the driver support a large number of features, such as Wireless Extensions, Scanning and Monitor mode.

3.30 SiS 160/162

Driver status :	Beta
Driver name :	PCI: sis160.o, sis162.o
	USB: sis162u.o
Version :	R093.02.24
Where :	http://driver.sis.com/
	http://driver.sis.com/linux/wlan/wlan_162_linux.tgz

Maintainer :	?
Mailing list :	?
Documentation :	Readme file
Configuration :	Specific tool, Wireless Extensions
Statistics :	-
Modes :	Managed, Ad-Hoc
Security :	?
Scanning :	Specific tool
Monitor :	No
Multi-devices :	?
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	-
Non implemented :	-
Bugs :	-
License :	Binary only for the core + OpenSource Linux wrapper
Vendor web pages :	http://www.sis.com/

3.30.1 The device

SiS is a taiwanese company better known for their CPU chipsets, but they do other products, and amongst them 802.11b chips. The SiS 160 was their first product, offers all the standard 802.11 features and has a PCI interface. The SiS 162 is their second product, adds WPA support and has both a PCI and a USB interface. Those chips are low cost, and are typically found in cheap USB dongles.

3.30.2 The driver

SiS released two drivers, a driver for the SiS 160 and another driver for the SiS 162. Those driver are based on a large binary part and a very small source wrapper, and are designed for kernel 2.4.X (and only on x86 platforms). They also contains their own binary configuration tool, and some basic support for Wireless Extensions.

Users reports are that these drivers can be problematic and only work for kernel 2.4.0 to 2.4.18, and do not work in any 2.6.X kernel. From a quick look at the source code, most of the functionality is in the binary part and it might not be possible to port it to more recent kernels.

4 The devices, the drivers - 802.11a, 802.11g and 802.11n

This section list devices going beyond the IEEE 802.11b standard, such as those based on the IEEE 802.11a, 802.11g and 802.11n standards (see *section 8*), offering bitrates higher than 11 Mb/s in the 2.4 GHz and 5.2 GHz bands.

4.1 TI ACX100 OpenSource driver (All 802.11b+ cards)

Driver status : beta

Driver name :	PCI : acx100_pci.o
Version :	0.3.36 [20070101]
Where :	http://acx100.sourceforge.net/ http://lisas.de/~andi/acx100/ http://rhlx01.fht-esslingen.de/~andi/acx100/ http://www.cmartin.tk/acx/
Contact :	ACX100 OSS driver project team <acx100-users@lists.sf.net>
Maintainers :	Denis Vlasenko <vda at port.imtp.ilyichevsk.odessa.ua> Andreas Mohr <andim2 at users.sourceforge.net> Jeff Williams <angelbane at users.sourceforge.net> Carlos Martin Nieto <carlos@cmartin.tk> Benjamin Schrauwen <bschrauw at users.sourceforge.net> Moritz Angermann <sfoi at users.sourceforge.net> Frederic Deletang <ziga at users.sourceforge.net>
Mailing list :	http://sourceforge.net/mail/?group_id=75380
Documentation :	Readme file, Wiki http://www.houseofcraig.net/acx100_howto.php
Configuration :	Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	Yes
Interoperability :	802.11b and PBCC, 802.11g depending on hardware
Other features :	Experimental support for HostAP mode
Non implemented :	"4X" mode, USB support (WIP), CF support, packet fragmentation, AutoRate
Bugs :	some configurations may lockup or loose packets
License :	MPL
Vendor web pages :	http://www.ti.com/ http://www.dlink.com/products/ http://www.linksys.com/products/ http://www.gemtek.com.tw/ http://www.smc.com/ http://www.netgear.com/

4.1.1 The device

In 2001, TI (Texas Instruments) decided to make a big push on the 802.11 market. Because there was already many other 802.11 vendors, TI needed to differentiate themselves. Their solution was to introduce a chip with faster data rate. TI introduced a new modulation, **PBCC** (Packet Binary Convolutional) at 5.5, 11 and 22 Mb/s, which offers higher speed and longer range than 802.11b modulations (CCK).

However, to get the market to accept PBCC, TI needed the stamp of approval of the IEEE standard group. Other vendors within the IEEE were not keen to give TI any advantage, and decided to base 802.11g on the OFDM modulation rather than PBCC. After much political fight, the final **802.11g** standard include both OFDM (mandatory) and PBCC (optional). Because PBCC is only optional and slower, I don't expect anybody else than TI to implement it.

The **ACX100** was one of the first second generation 802.11b chipset. It is based on a ARM7 core, and offer both a PCI/Cardbus and USB interface. The use of PCI busmaster allow to reduce CPU utilisation compared to 16 bit cards. The MAC includes all the common 802.11 features, the Phy implement the regular 802.11b modulations and PBCC. The design goal seems to have been performance and features.

As soon as TI released its ACX100 chipset, many usual vendors started selling PCI, Cardbus and USB cards based on it, for example the D-Link 520+ and 650+ cards based on it. All the cards that support up to 22 Mb/s are based on this chipset.

The ACX100 is known for its requirement for PCI2.2 mainboards (lack thereof or some other reason may lead to conflicts with other cards such as SB Live! and crashes). This is not specific to the ACX100, other modern wireless chipsets also tend to require PCI2.2.

Note that TI has released newer chipsets, the **TNETW1100b/1130/1230** chipsets, also called **ACX111**, which add support for 802.11g and 802.11a, and don't seem to be driver compatible with the ACX100. All cards supporting both **22 Mb/s** and 802.11g/a are based on those chipsets. Note that for 802.11g, a name including "plus" does not imply a TI chipset, D-Link has a range of 802.11g "plus" cards including Atheros chipsets.

4.1.2 The driver

From the start, TI has refused to give any help towards a Linux driver and have decided to totally ignore the Linux community. Toward the end of 2002, a binary Linux driver was leaked on the net. Because this binary driver is not officially released by any vendor, its legal status is unclear (and therefore it is not listed here). Moreover, this driver was compiled for a very specific kernel and was overall very problematic to install and use.

A group of people using this driver got fed up with the situation and decided to reverse engineer the binary driver by disassembling it. This is a huge task, as this hardware and driver are fairly complex, and most other reverse engineering project I've heard off eventually failed. After a few month of work, they eventually released their first version of the ACX100 opensource driver.

The initial driver was very experimental, but it evolved fairly rapidly, as *Andy* was working hard on the driver fixing bugs. Most basic features are now working (Ad-Hoc and Infrastructure mode, WEP, Wireless Extensions, statistics...), and some advanced features are implemented (monitor mode, 2.6.X support...). However, there are still lockups or eventual traffic loss on some machines/ networks, power management/hotplug is experimental. Preliminary support for USB and ACX111 devices has been added, but this is not yet complete. *Denis* joined and fixed support for ACX111, as well as ton of other bugs.

Around mid 2005, *Denis* created a branch of the driver targeted at inclusion in kernel 2.6.X. He dropped support for 2.4.X and some non standard features such as the old firmware loader, and fixed many bugs. For example, support for USB dongles, 64 bits, SMP is now functional. This driver was included in *Andrew's* kernels (-mm version), and meanwhile can be made to compile for standard kernels. This version got almost included in the Linux kernel but was eventually rejected, and also dropped from -mm kernels.

The initial ACX100 driver contains its own 802.11 code. When the **SoftMAC** 802.11 stack became available in 2005 (see *section 4.13*), *Andi* and *Denis* ported the driver to it. This SoftMAC driver is currently in alpha stage. This driver was included in wireless-2.6 GIT repository, in preparation for inclusion in the kernel, and later dropped. The concern is that there was no clean separation between the people that did the reverse engineering and those who wrote the driver, unlike what was done for the Broadcom driver (see *section 4.13*).

In 2006, *Jeff* and *Carlos* started porting the driver to the new **mac80211** kernel stack (see *section 4.9*). Their intention is to initially replace the SoftMAC driver, and eventually replace the initial driver. The driver is currently in experimental stage.

In parallel, the branches of the initial driver from *Andi* and *Denis* were merged. *Andi* has still been busy fixing various bug and keeping this driver up to date with respect to kernel changes. This is the version of the driver recommended for maximum stability.

4.2 Atheros MADWiFi driver (most cards with 802.11a or 108 Mb/s)

Driver status :	stable
Driver name :	ath_pci.o
Version :	0.9.3.1
Where :	http://madwifi.org/
	http://sourceforge.net/projects/madwifi
Maintainers :	Sam Leffler <sam@errno.com>
	Michael Renzmann <netdev@nospam.otaku42.de>
	Greg Chesson <greg@atheros.com>
Mailing list :	http://sourceforge.net/mail/?group_id=82936
Documentation :	Readme file
	http://www.mattfoster.clara.co.uk/madwifi-faq.htm

Configuration :	Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc, Master (HostAP)
Security :	WEP, 802.1x, WPA
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	?
Interoperability :	802.11b, 802.11g, 802.11a depending on hardware
Other features :	HostAP mode
Non implemented :	Ad-Hoc mode
Bugs :	-
License :	Binary only for the HAL + BSD driver
Vendor web pages :	http://www.atheros.com/
	http://actiontec.com.tw/
	http://www.proxim.com/
	http://www.dlink.com/products/
	http://www.linksys.com/products/
	http://www.gemtek.com.tw/
	http://www.smc.com/
	http://www.netgear.com/

4.2.1 The device

Atheros was the first company to release a complete 802.11a solution, and therefore most existing products are based on it. **Proxim** was the first company releasing products based on the Atheros chipset, but other familiar names such as **Linksys, D-Link** and **SMC** did follow quickly.

Most vendor offer a CardBus card (32 bits Pcmcia) and Access Points, and a few offer PCI solutions.

The first Atheros chipset, **ar5210**, implement the full **802.11a** standard. The MAC protocol is identical to 802.11b, the Atheros chipset offer both infrastructure and ad-hoc mode, enhanced WEP with per link keys (for 802.1x) The radio modem use OFDM in the 5 GHz band (8 separate channels), with speeds from 6 Mb/s to 54 Mb/s (depending on range).

The Atheros chipset has a few proprietary features, such as its X2 mode (turbo) that allow to double the rate by using two channels in parallel (unfortunately, it also increases the sensitivity to interferences, decrease the number of channels and is incompatible with 802.11a).

The Atheros cards offer a busmaster host interface, allowing high performance and low CPU utilisation (almost all 802.11b cards are limited to PIO, which keeps the CPU busy). The card deals only with the low level of the MAC

protocol, and the driver has to deal with all the 802.11 management (making driver development more complicated). The other main characteristics of the Atheros chipset is that it's a fully CMOS solution (including the radio), which reduces the cost.

Atheros has released its second generation chipset, derived from the initial ar5210, that have better performance and are no longer limited to only 802.11a. The **ar5211** support both *802.11a* and *802.11b*, whereas the **ar5212** support *802.11a*, *802.11b* and *802.11g*. The following chipsets from Atheros (**ar5213, ar5214, ar5413, ar5414, ar5513**) kept the basic same design and features and mostly added a wide range of proposed enhancements to the 802.11g/a standard, add 802.11e QoS or proprietary extensions, such as Turbo mode (bound two channels), bursting and proprietary MIMO.

After that, Atheros released a number of more specialised chipsets, such as single-chip solutions (integrating both MAC and radio on the same chip) and PCI-Express chipsets. Their USB chipset AR5007UG is the former ZyDAS ZD1211 (see *section 4.14*). Their new generation chipset, such as the **ar5416**, support draft-802.11n MIMO (up to 300 Mb/s), and obviously adds new proprietary extensions.

There is many products on the market using Atheros chipsets. In fact, the vast majority of **802.11a** cards are based Atheros chipset. Most all the 802.11g cards that advertise **108 Mb/s** are also based on Atheros chipsets.

4.2.2 The driver

Sam, with the help of Atheros, had created a BSD driver for those cards some time ago. Unfortunately, he was unable to release it because of the FCC regulations. The Atheros hardware is basic and doesn't enforce that valid operating parameters are set (such as frequency and power level) and does not ensure compliance to the various **radio regulations**, however the regulatory organisations, such as the FCC and ETSI, mandate that end user should not be able to set invalid operating parameters. Eventually, *Atheros* managed to find a solution that was acceptable : they created a **HAL**, a binary layer that sits between the hardware and the driver and enforces that radio regulations are respected. The downside is that the HAL is available only for selected architectures (i386, PPC, Arm, Mips, SH4, Alpha and Sparc64).

In the mean time, *Sam* had ported is original BSD driver to Linux, and integrated into Linux. Like the BSD driver, the initial MADWiFi driver is based on the HAL, however it supports Wireless Extensions and Wireless Scanning (which are Linux specific). This driver requires a recent 2.4.X kernel or 2.6.X kernel, and requires Wireless Extensions V14 or later.

As the hardware is basic, a lot of the 802.11 functionality is handled in the driver (as opposed to firmware). To handle this, *Sam* has created a 802.11 stack for this driver that became the de-facto 802.11 stack of BSD operating systems. This is a lot of code, and for the first release there was still some bugs and many features and optimisations missing (for example, Ad-Hoc mode is not yet functional). The good news is that over time, the performance and functionality of this driver increased. Lately, WPA support has been added to the driver (which is a lot of work).

In early 2005, the project went into hibernation, no new release was done and activity slowed down. Then, in mid 2005 the project was re-energised, and *Michael* took over the driver maintenance. He is coordinating the work on a new branch of the driver called **madwifi-ng** created by Atheros to support more recent cards and their features. This is now the main branch of the driver, seeing active development, and the old branch has been deprecated.

This driver has a vast number of **features** and good hardware compatibility, thanks to the support from Atheros and the close integration with the 802.11 stack. Nearly all features of the hardware are supported (QoS and performance enhancements). The driver also support virtual network interfaces (called VAP), which allow one card to act as multiple 802.11 devices.

The MadWiFi project has recently added two other branches of the driver. The first branch is called **DadWiFi** and is a port of the driver to the new **mac80211** kernel stack (see *section 4.9*). Currently, the mac80211 stack does not offer much more than the 802.11 stack in the madwifi-ng driver, but with time it may change.

The other branch is called **OpenHAL**, it is a replacement of the binary only HAL with OpenSource code. The OpenHAL is mostly a port from the OpenBSD Atheros driver called ar5k from *Reyk Floeter*, which itself is derived from the vt_ar5k driver for Linux written by *Reyk*, which was listed in this document a few years ago. The work on OpenHal and the port to mac80211 would make this driver more likely to go in the Linux kernel at some point in the future.

4.3 Intersil Prism54 FullMac (802.11a/802.11g)

Driver status :	stable
Driver name :	Cardbus : prism54.o
	PCI : prism54.o
Version :	1.2
Where :	Linux kernel (2.6.19)
	http://prism54.org/fullmac.html
Authors :	W. Termorshuizen
	R. Bastings
Maintainers :	Luis R. Rodriguez <mcgrof@ruslug.rutgers.edu>
	Frederik Kunz <frederik.kunz@web.de>
	Sebastian Schoeps <sebastian@schoeps.org>
	Aurelien Alleaume <slts@free.fr>
	Herbert Valerio Riedel <hvr@gnu.org>
Mailing lists :	http://prism54.org/mailing.html
Documentation :	Readme, web pages
Configuration :	Wireless Extensions & configuration tool
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc, Master (FirmwareAP)

Security :	WEP, AES, WPA
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	Yes
Interoperability :	802.11b, 802.11g, 802.11a depending on hardware
Other features :	Encryption, Access Point mode, WDS, Firmware loading via HotPlug.
Non implemented :	SoftMAC firmware support
Bugs :	?
License :	GPL
Vendor web pages :	http://www.dlink.com/products/
	http://www.smc.com/
	http://www.netgear.com/
	http://www.gemtek.com.tw/
	http://www.conexant.com/

4.3.1 The device

The Prism Indigo, Prism GT and Prism Duette are the third generation of chipsets from Intersil, logical follower of the second generation PrismII and Prism3 chipsets (see *section 3.6*). The **Prism Indigo** was released first, and support 802.11a (5 GHz band). The **Prism GT** support 802.11g (and 802.11b - 2.4 GHz band). The **Prism Duette** support both 802.11a and 802.11g (in both the 2.4 GHz and 5 GHz band).

Because of the limitations of the previous PrismII design, those new chipsets feature a totally new design, much cleaner and much more efficient, and doesn't share much in common with the earlier Prism designs, but is more likely to be related to the NWN MAC that Intersil acquired (see *section 3.17*).

The chipset is highly integrated (2 or 3 chip solutions). Like previous Prism chipsets, most of the 802.11 management is handed in firmware. However, the bus interface is now PCI DMA only (less CPU overhead, faster). The MAC support the usual 802.11 features, and hardware WEP and AES encryption, and the firmware fully act as an Access Point. The Radio implement the OFDM modulation necessary to support 802.11a and 802.11g.

There doesn't seem to be any products using the Prism Indigo chipset (most card offering 802.11a are based on Atheros chipset - see *section 4.2*), and I don't expect to see any in the future. A few of the usual Intersil partners, such as **Netgear** and **SMC**, have started selling 802.11g cards based on the Prism GT and Prism Duette chipsets.

Note that there are two types of firmware for this card, the FullMAC firmware, that can handle most of the 802.11 functionality by itself, and the SoftMAC firmware, that delegates it to the driver. Older revision of the hardware can handle both types of firmware, whereas newer revisions can only handle the

SoftMAC firmware. Cards that can support the older FullMAC firmware are now very difficult to find.

After releasing this chipset, the wireless division of **Intersil** (formerly part of Harris) was bought by **GlobespanVirata**, and after a few month, GlobespanVirata merged with **Conexant**. Therefore, the Prism chipsets are now sold by Conexant.

4.3.2 The driver

This driver was written and released as GPL by *Intersil* themselves ! This is a much stronger commitment to OpenSource than some other vendors have made. Unfortunately, after the aquisition by Conexant, they stopped contributing to the OpenSource community.

Luis, *Frederik* and *Sebastian* are taking care of this driver after the initial release from Intersil and helping people with the driver. They are fixing bugs and improving compatibility with the various kernels and configurations. *Aurelien* has done a tremendous work on Wireless Extensions support.

After an initial frenetic pace of changes and reorganisation, the driver has stabilised and matured, and is now included in the **Linux 2.6.X kernel**. The driver requires firmware download (through the standard HotPlug support) and has full support for Wireless Extensions (including Wireless Scanning) and monitor mode. The driver also can act as an Access Point through a FirmwareAP mode (all the AP management is handled by the firmware). Full support for WPA has been added recently to the driver.

One of the limitation of the driver is that it does not support the new SoftMAC devices, which means that it only works with older Prism54 devices. For SoftMAC devices, you will need the SoftMAC driver (see *section 4.4*).

4.4 Intersil Prism54 SoftMac driver (islsm)

Driver status :	beta
Driver name :	islsm.o
Version :	0.7
Where :	http://prism54.org/newdrivers.html
	http://jbnote.free.fr/
	http://islsm.org/~jb/islsm/islsm.git/
Maintainer :	Jean-Baptiste Note <jean-baptiste.note@m4x.org>
	Jean-Baptiste Note <jbnote@gmail.com>
Mailing lists :	?
Documentation :	?
Configuration :	?
Statistics :	?
Modes :	?
Security :	?

Scanning :	?
Monitor :	?
Multi-devices :	?
Interoperability :	802.11b, 802.11g, 802.11a depending on hardware
Other features :	?
Non implemented :	?
Bugs :	?
License :	GPL
Vendor web pages :	http://www.dlink.com/products/
	http://www.smc.com/
	http://www.netgear.com/
	http://www.gemtek.com.tw/
	http://www.conexant.com/

4.4.1 The device

After Conexant acquired the Prism54 family of chipsets (see *section 4.3*), the Prism54 chipsets changed their mode of operation from **FullMAC**, where the firmware handle all the 802.11 protocol, to **SoftMAC**, where the driver has to handle all the 802.11 protocol. This allows to reduce the cost of the cards by decreasing the on-board memory and flash. However, the chipset is no longer compatible at the driver level, because the firmware is different. Older devices can support both types of firmwares, newer card with less memory can not accommodate the FullMAC firmwares. All the Prism54 USB devices and most recent PCI/Cardbus cards can only support SoftMAC operation.

Conexant also released the **Prism WorldRadio**, an enhanced version of the Prism Duette with 802.11a and 802.11g support with proprietary bit rate enhancement. Then, they released the CX85510, which adds QoS and more security features, and the CX3110x, a single chip solution.

4.4.2 The driver

Up to that point, *Intersil* had a long tradition of being one of the most friendly vendor towards Linux, and they did help the many Linux drivers and even released themselves the Prism54 FullMAC driver (see *section 4.3*). However, after being acquired by Conexant and with the release of SoftMAC, they decided to ignore the Linux community. The original Prism54 team tried for months to negotiate the release of specs from Conexant in vain.

Jean-Baptiste has led the effort to reverse engineer the Windows driver and create a driver for the Prism54 SoftMAC devices. He reversed engineered the driver by loading it with ndiswrapper and spying on the USB traffic, as opposed to other effort that actually disassemble a driver.

Jean-Baptiste is mostly concentrating on support for USB devices, however support for PCI devices was added as well to the Prism54 SoftMAC driver. The driver is based on the 802.11 stack from the **MadWifi** project (see *section 4.2*). The

driver is claimed to work, but at the time of writing not tarball or release seems to be available.

4.5 Intersil Prism54 mac80211 driver (p54)

Driver status :	alpha
Driver name :	PCI/Cardbus : prism54pci.o
	USB : prism54usb.o
Version :	?
Where :	wireless-dev GIT
Maintainers :	Michael Wu <flamingice@sourmilk.net>
	Christian Lamparter <chunkeey@web.de>
	Jean-Baptiste Note <jbnote@gmail.com>
Mailing lists :	-
Documentation :	-
Configuration :	Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP, 802.1x, WPA
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	Yes
Interoperability :	802.11b, 802.11g, 802.11a depending on hardware
Other features :	mac80211 features, Firmware loading via HotPlug
Non implemented :	-
Bugs :	?
License :	GPL
Vendor web pages :	http://www.conexant.com/

4.5.1 The device

This is the same device as the previous entry (*section 4.4*).

4.5.2 The driver

When *Jean-Baptiste* decided to write the SoftMAC Prism54 driver (see *section 4.4*), he decided to use the 802.11 stack from the **MadWifi** project (see *section 4.2*), which is one of the most fully featured 802.11 stack available. Since then, the Linux kernel developers decided to standardise on the **mac80211** stack (see *section 4.9*), and therefore a driver using another stack can not be included in the kernel.

Michael ported the work of *Jean-Baptiste* to the mac80211 stack, with the goal of merging it in the Linux kernel in the near future. The driver has two versions, one for PCI/Cardbus devices and one for USB devices. This driver is only

available in the wireless-dev GIT repository, and not much documentation is available about it.

4.6 Intel PRO/Wireless 2200 802.11g and 2915 802.11ag (Centrino)

Driver status :	Stable
Driver name :	ipw2200.o
Version :	1.2.2
Where :	Linux kernel (2.6.20) http://ipw2200.sourceforge.net/ http://ieee80211.sourceforge.net/
Maintainer :	James P. Ketrenos <ipw2100-admin@linux.intel.com>
Mailing list :	http://lists.sourceforge.net/lists/listinfo/ipw2100-devel/
Documentation :	Readme
Configuration :	Wireless Extensions
Statistics :	/proc interface and Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP, 802.1x, WPA
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	?
Interoperability :	802.11b, 802.11g, 802.11a depending on hardware
Other features :	Firmware loading via HotPlug, 802.11e (QoS)
Non implemented :	-
Bugs :	?
License :	GPL
Vendor web pages :	http://www.intel.com/ http://www.passys.nl/tips/tip40_en.htm

4.6.1 The device

The Intel **PRO/Wireless 2200** is the second 802.11 product designed by Intel, following the previous **PRO/Wireless 2100** product (*section 3.28*). It mostly adds support for 802.11g (i.e. faster data rates). The **PRO/Wireless 2915** is their third 802.11 product, and adds support for 802.11a.

The design of the 2200 and 2915 are based on the original 2100, the main improvement are the new radio modems and better support for the new security protocols (WPA). The MAC core still support the same vast array of standard 802.11 features. The radio are obviously different, adding the OFDM bit-rates, up to 54 Mb/s, and support for 5 GHz frequencies for the 2915.

The marketing program is still the same, and the PRO/Wireless 2200 and 2915 Mini-PCI cards are exclusively sold as part of the **Centrino** package,

therefore you will mostly find them **integrated in various laptops**, and not as a separate add-on card.

If you really want a separate add-on card, Intel also sells the Intel **PRO/Wireless 2225**, which is basically the PRO/Wireless 2200 in a half-height PCI card and sold separately. And obviously, it's not called Centrino. And this card is very hard to find and buy, and is not listed on Intel web site.

Some people, such as **Passys**, have built PCI to Mini-PCI adapters for the PRO/Wireless 2200. This allows to put a standard Mini-PCI PRO/Wireless 2200 in any regular desktop, and it is supported by Linux.

4.6.2 The driver

Intel is actively supporting the driver for both the PRO/Wireless 2200 and 2915, like they do for the PRO/Wireless 2100 (*section 3.28*). *James* is still leading the development of this driver. The driver is fully OpenSource, and like for the 2100 require a specific **firmware**.

Obviously, the driver share a lot of code with the 2100 driver, it supports both the **2200** and the **2915** and offer a similar rich feature set (WEP, WPA, Scanning). The development is very active and many bug are being fixed and features added.

The 802.11 code originally taken from the HostAP driver (see *section 3.8*) was eventually spin-off into the separate **Intel ieee80211 stack** (see *section 3.28*). This driver and the 802.11 stack was included in Linux kernel 2.6.14.

4.7 Intel PRO/Wireless 2200/2915 with HostAP mode

Driver status :	Beta
Driver name :	ipw2200.o
Version :	0.3
Where :	http://sourceforge.net/projects/ipw2200-ap
Maintainer :	York Liu <york_liu@linux.intel.com>
Mailing list :	http://sourceforge.net/mail/?group_id=149502
Documentation :	Readme
Configuration :	Wireless Extensions
Statistics :	/proc interface and Wireless Extensions
Modes :	Managed, Ad-Hoc, Master
Security :	WEP, 802.1x, WPA
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	?
Interoperability :	802.11b, 802.11g, 802.11a depending on hardware
Other features :	Firmware loading via HotPlug
Non implemented :	-
Bugs :	?

License : GPL

Vendor web pages : http://www.intel.com/

4.7.1 The device

This is the same device as the previous entry (*section 4.6*).

4.7.2 The driver

Intel is one of the most active company when it comes to Linux wireless drivers. After releasing the standard driver for the PRO/Wireless 2200 and 2915 (see *section 4.6*), they released an alternate driver with **Access Point mode** (HostAP mode), which let you use your card as an Access Point.

The driver is still based on the 802.11 code originally taken from the HostAP driver (see *section 3.8*), it is a variant of the Intel ieee80211 stack (see *section 3.28*) that includes the HostAP functionality. This driver requires a specific firmware for the card, provided by Intel. This driver seems to be updated less often than the standard drivers.

4.8 Intel PRO/Wireless 3945ABG 802.11g/802.11ag (Centrino)

Driver status : Stable

Driver name : ipw3945.o

Version : 1.2.1

Where : http://ipw3945.sourceforge.net/
http://ieee80211.sourceforge.net/

Maintainer : James P. Ketrenos <ipw2100-admin@linux.intel.com>
Zhu Yi <yi.zhu@intel.com>

Mailing list : https://lists.sourceforge.net/lists/listinfo/ipw3945-devel

Documentation : Readme

Configuration : Wireless Extensions

Statistics : /proc interface and Wireless Extensions

Modes : Managed, Ad-Hoc

Security : WEP, 802.1x, WPA

Scanning : Wireless Extensions

Monitor : Yes

Multi-devices : ?

Interoperability : 802.11b, 802.11g, 802.11a depending on hardware

Other features : Firmware loading via HotPlug, 802.11e, binary daemon

Non implemented : -

Bugs : ?

License : GPL, except the binary daemon

Vendor web pages : http://www.intel.com/

4.8.1 The device

The Intel **PRO/Wireless 3945** is the third 802.11 product designed by Intel, following the previous **PRO/Wireless 2200** product (see *section 4.6*). The main goal was to reduce the cost and the size of the previous chipset, however Intel has added a few novel features such as WakeOnWLAN and better VoIP support.

Like their previous chipset, the 3945 is only available as a **MiniPCI card** for Intel Centrino processor, integrated in various laptops, and not available as a separate add-on card.

4.8.2 The driver

Like for all their previous chipset, *Intel* decided to release a driver for this chipset. However, they were faced with a problem. To achieve reduced cost, a lot of functionality was moved from the hardware/firmware into the driver. The task of insuring compliance with various **radio regulations** was moved out of the hardware, which is an issue because regulatory organisations, such as the FCC and ETSI, mandate that end user should not be able to set invalid operating parameters. This is the reason why the Atheros drivers contains a binary HAL (see *section 4.2*).

Intel opted for a different approach, they created a fully OpenSource driver and moved regulation compliance to a **binary userspace daemon** interacting with the driver. This is technically better than using a binary driver, as the driver could be included in the kernel and the user just need to add the binary daemon to his system, which is simpler than adding a kernel driver. However, this has similar downside as a binary driver, the platform support is limited (currently only to x86 and x86-64).

The driver is similar to the ipw2200 driver (see *section 4.6*), it share a lot of code with it, has a similar set of features and uses the same **Intel ieee80211** stack (see *section 3.28*).

Note that Intel has a second driver for this hardware using the new **mac80211** kernel stack (see *section 4.9*).

4.9 Intel PRO/Wireless 3945ABG mac802.11 driver

Driver status :	Experimental
Driver name :	iwl3945.o
Version :	0.1.3
Where :	http://intellinuxwireless.org/?p=iwlwifi
	http://intellinuxwireless.org/?p=mac80211
Maintainer :	James P. Ketrenos <ipw2100-admin@linux.intel.com>
	Zhu Yi <yi.zhu@intel.com>
Mailing list :	https://lists.sourceforge.net/lists/listinfo/ipw3945-devel
Documentation :	Readme
Configuration :	Wireless Extensions
Statistics :	Wireless Extensions

Modes :	Managed, Ad-Hoc
Security :	WEP, 802.1x, WPA
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	?
Interoperability :	802.11b, 802.11g, 802.11a depending on hardware
Other features :	Firmware loading via HotPlug, 802.11e (QoS)
Non implemented :	-
Bugs :	?
License :	GPL
Vendor web pages :	http://www.intel.com/

4.9.1 The device

This is the same device as the previous entry (*section 4.8*).

4.9.2 The driver

With all their previous drivers, *Intel* were using their **Intel ieee80211** stack (see *section 3.28*), derived from the HostAP driver (see *section 3.8*). This stack is a good match for the previous Intel chipsets, however it is not well suited to the 3945 and other modern wireless chipset from other vendors due to the fact that it does not handle the MAC management functionality. The SoftMAC addition to Intel ieee80211 stack adds such function (see *section 4.13*), but is very limited in functionality compared to modern 802.11 stacks such as the MadWiFi stack (see *section 4.2*).

For these reasons, various driver maintainers decided to standardised on a new stack, the **mac80211 stack**. This stack was initially released in middle 2006 by DeviceScape, a Linux consulting company, and was mostly designed by *Jouni Malinen*, author of the HostAP driver (see *section 3.8*). After this release, the DeviceScape stack was very heavily reworked by *Jiri Benc*, *Michael Wu* and *Zhu Yi* and the rest of the Linux wireless community, to become mac80211. The mac80211 stack was included in kernel 2.6.22, but it is still experimental and a lot of work remains. For example, early version of mac80211 have deadlocks and are not SMP safe.

Intel was obviously interested in this new mac80211 stack, as in the long term it makes their driver better and easier to maintain. Intel is contributing to the mac80211 stack, and they also have their own branch of mac80211 with features needed by their driver and not found yet in the kernel stack (but planed to be merged later).

Intel also ported their driver for the 3945 chipset to the new mac80211 stack, as part of the **iwlwifi** package. The driver is still experimental and under heavy development.

The previous driver for the 3945 requires a binary daemon for regulation compliance, but this turned out to be not satisfactory. Intel decided to rewrite the

firmware for the 3945 to ensure regulation compliance. This means that the iwlwifi Linux driver requires its **own specific firmware**, and that the driver can be fully OpenSource without binary daemon. This represents an impressive commitment to the Linux community.

4.10 Intel PRO/Wireless 4965AGN draft-802.11n (Centrino)

Driver status :	Experimental
Driver name :	iwl4965.o
Version :	0.1.3
Where :	http://intellinuxwireless.org/?p=iwlwifi
	http://intellinuxwireless.org/?p=mac80211
Maintainer :	James P. Ketrenos <ipw2100-admin@linux.intel.com>
	Zhu Yi <yi.zhu@intel.com>
Mailing list :	https://lists.sourceforge.net/lists/listinfo/ipw3945-devel
Documentation :	Readme
Configuration :	Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP, 802.1x, WPA
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	?
Interoperability :	802.11b, 802.11g, 802.11a depending on hardware
Other features :	Firmware loading via HotPlug, 802.11e (QoS)
Non implemented :	-
Bugs :	?
License :	GPL
Vendor web pages :	http://www.intel.com/

4.10.1 The device

The Intel **PRO/Wireless 4965** is the fourth 802.11 product designed by Intel, following the previous **PRO/Wireless 3945** product (*section 4.8*). It mostly adds support for **draft-802.11n** support, i.e. MIMO.

MIMO uses multiple antennas to simultaneously transmit or receive signal, and using some clever processing can achieve much greater range or speed than regular transmissions. The 802.11n standard is not yet finalised, which is why those products are called draft-802.11n compliant. It offer bitrate up to 300 Mb/s (most users won't see that in practice), and may double the range indoors.

Like their previous chipset, the 4965 is only available as a **MiniPCI card** for Intel Centrino processor, integrated in various laptops, and not available as a separate add-on card.

4.10.2 The driver

For the 4965, *Intel* decided to directly based their driver on the new **mac80211** kernel stack (see *section 4.9*). This driver is part of the **iwlwifi** package, and currently requires a version of mac80211 with Intel changes to support draft-802.11n. This driver also require a specific firmware the ensure regulation compliance.

4.11 Ralink RT2500 and 2570 cards

Driver status :	Beta
Driver name :	rt2500.o
	PCI : rt61.o
	USB : rt73.o
	rt2x00.o
Version :	1.1.0 (beta) and 2.0.4 (alpha)
Where :	http://rt2x00.serialmonkey.com/
	http://rt2400.sourceforge.net/
	http://www.ralinktech.com/supp-1.htm
Maintainers :	Paul Lin
	Mark Wallis <markwallis at users.sourceforge.net>
	Ivo van Doorn <ivd at euronet.nl>
Mailing list :	http://sourceforge.net/mail/?group_id=107832
Documentation :	Text files, Howtos
	http://www.bb-zone.com/misc/rt2500/
Configuration :	Wireless Extensions and specific graphical tool
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP, WPA
Scanning :	Yes
Monitor :	Yes
Multi-devices :	?
Interoperability :	802.11b, 802.11g and 802.11a depending on hardware
Other features :	-
Non implemented :	-
Bugs :	-
License :	GPL
Vendor web pages :	http://www.ralinktech.com/
	http://minitar.com/
	http://rt2x00.serialmonkey.com/wiki/index.php/Hardware

4.11.1 The device

Ralink may have been late in releasing a 802.11b chipset, the RT2400 (*section 3.27*), however it was fairly quickly followed by the **RT2500**, a 802.11g chipset. The RT2500 is a evolution of the RT2400, and share many characteristics with it.

Like other 802.11g chipsets, it supports standard OFDM bitrates up to 54 Mb/s and all modern 802.11 features such as WPA and QoS (802.11e). On top of that, it support a proprietary 72 Mb/s bitrate. The RT2500 is designed for MiniPCI and CardBus interfaces, and the **RT2570** is designed with a USB 2.0 interface.

As usual, this chipset is sold by a wide variety of vendors under different model names, and some of those use the same model name for different chipsets. The project pages includes a long list of cards including this chipset. A special mention to Minitar which has a dedicated Linux support forum.

The RT2500 was quickly followed by other 802.11g chipsets. The **RT61** family includes the rt2561 and rt2661 PCI chipsets and adds support for turbo bit rate (100 Mb/s) and for the rt2661 only proprietary MIMO (not 802.11n compatible). The **RT73** family includes the RT2573 and RT2571 USB chipsets, and is the equivalent of the RT61 (but without the MIMO chipset). Those chipset are available with two radio options, 2.4 GHz only, or 2.4 GHz plus 5 GHz, for additional 802.11a support.

Ralink has now released the **RT2800** family. The main feature of those chipset is the addition of draft-802.11n MIMO support.

4.11.2 The driver

Like for the RT2400, *Ralink* wrote a Linux driver for the **RT2500** and **RT2750**, but this time they decided to release it themselves as GPL. Moreover, the driver is functional, full of features and with a graphical utility, so this represent a very generous contribution to the OpenSource community. The driver supports WEP, WPA, Scanning and Monitor mode...

Mark integrated this driver in the existing SourceForge project for the RT2400 driver and started to maintain it. Many patches have been integrated to improve the stability and functionality of the driver.

After that, *Ralink* released the **RT61** driver and the **RT73** driver, to support the new 802.11g chipset. Those drivers are GPL and have the same features as the RT2500 chipset. *Mark* has integrated this driver in the existing SourceForge project, and the community is provided various improvement to those drivers.

Ivo has started a rewrite of those drivers, called **rt2x00**, his goal is to have a source code easier to integrate in the Linux kernel and to maintain. His rewrite targets both the RT2400 (see *section 3.27*), the RT2500, the RT61 and RT73. The initial version of this new driver was using the **Intel ieee802.11** stack from the Centrino driver (see *section 3.28*).

Then, *Ivo* ported the driver to the new **mac80211** kernel stack (see *section 4.9*), and lots of development has happened on that version of the driver. This alternate version is available in the wireless-dev GIT repository and in the CVS, and should appear in a Linux kernel in the future.

4.12 Ralink ural-linux driver (BSD driver)

Driver status :	Alpha
Driver name :	ural.o
Version :	0.8.2
Where :	http://etudiants.insia.org/~jbobbio/ural-linux/
Maintainers :	Jeremy Bobbio <jeremy.bobbio@etu.upmc.fr>
Mailing list :	-
Documentation :	-
Configuration :	Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP, WPA
Scanning :	Wireless Extensions
Monitor :	No
Multi-devices :	?
Interoperability :	802.11-b and 802.11g
Other features :	-
Non implemented :	-
Bugs :	-
License :	BSD
Vendor web pages :	http://www.ralinktech.com/

4.12.1 The device

This is the same device as the previous entry (*section 4.11*).

4.12.2 The driver

Jeremy has ported the **BSD driver** for the RT2750 to Linux. This port is based on the 802.11 stack from the **MadWifi** project (see *section 4.2*) and support only USB devices. This driver is experimental and has not seen much work since 2005.

4.13 Broadcom 43xx cards

Driver status :	Beta
Driver name :	bcm43xx.o
Version :	?
Where :	Linux kernel (2.6.22)
	http://bcm43xx.berlios.de/
	http://linux-bcom4301.sourceforge.net/
Maintainers :	Michael Buesch <mbuesch@freenet.de>
	Martin Langer <martin-langer@gmx.de>

	Stefano Brivio <st3@riseup.net>
	Danny van Dyk <kugelfang@gentoo.org>
	Andreas Jaggi <andreas.jaggi@waterwave.ch>
	Larry Finger <Larry.Finger@lwfinger.net>
Mailing list :	https://lists.berlios.de/mailman/listinfo/bcm43xx-dev
Documentation :	README files
Configuration :	Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP, 802.1x, WPA
Scanning :	Wireless Tools
Monitor :	Yes
Multi-devices :	?
Interoperability :	802.11b, 802.11g, 802.11a depending on hardware
Other features :	-
Non implemented :	802.11a
Bugs :	-
License :	GPL
Vendor web pages :	http://www.broadcom.com/
	http://www.linksys.com/
	http://www.usr.com/

4.13.1 The device

Broadcom is a company that is developing all king of networking solutions, and well known for its Ethernet chipsets. When it decided to enter the Wireless LAN market, they decided a bold strategy. Instead of releasing a 802.11b product, which was the only standard at that time, they decided to release a product based on the upcoming 802.11g specification that was still one year from completion. Most competitors were taken by surprise, consumer were attracted by the higher speed and Broadcom quickly gained a good market share. When the 802.11g standard was eventually released, Broadcom quickly upgraded to it, and for a while the vast majority of 802.11g card were based on Broadcom chipsets. Many laptops of that period also included Broadcom chipsets. The Apple **Airport Express** is also based on the Broadcom chipset.

Eventually, the competition caught up, started offering more features than Broadcom or cheaper prices, and a lot of vendor using Broadcom chipsets migrated to other chipsets. **Linksys** is still a Broadcom stronghold, and a large portion of their 802.11g products are based on Broadcom chipsets (but not exclusively). A lot of laptops today that offer 802.11 includes either an Intel Centrino or a Broadcom chipset.

Like most 802.11g chipsets, the BCM43xx chipsets support standard OFDM bitrates up to 54 Mb/s and all modern 802.11 features such as WPA and QoS (802.11e). Broadcom has a complete family of chipsets, each iteration lowering the price or adding features. Their newer chipsets add support for 802.11a and pre-802.11n MIMO, and also add USB and PCI-Express interfaces.

Broadcom first proprietary extension is SpeedBooster, Broadcom's name for Packet Bursting, which is part of the 802.11e standard. They have few proprietary extension, their newer chipsets include a proprietary 125 High Speed Mode : this is not a enhanced modulation at 125 Mb/s, the card still use 54 Mb/s and a combination of protocol enhancement and compression. In fact, if a card advertise **125 Mb/s** (which is misleading), it's a good clue that it's most likely a Broadcom card. Broadcom also advertise BroadRange, which is mostly a higher radio sensitivity.

Broadcom has two main firmware version, V3 and V4. Most cards can run both firmware, however older cards do not support V3 (such as the BCM4301 chipset, which is 802.11b only). Firmware V4 is necessary for most of the more advanced features.

4.13.2 The driver

Like TI (see *section 4.1*), Broadcom quickly gained a reputation as being very unfriendly to Linux. It was known that Broadcom internally had a fully functional Linux driver, but it was unwilling to release it in any form (even binary), and was not answering call for chipset specifications. However, soon people realised that the **Linksys WRT54G**, a router/access point with a Broadcom chipset, was based on Linux. Linksys was made aware that, because Linux is GPL, they had to release the full source code of the WRT54G firmware, which they promptly did. This source code contained a binary Linux driver for the Broadcom chipset.

A team of people led by *Brett Wooldridge*, *Joe Jezak* and *Johannes Berg* started to reverse engineer this driver by disassembling it, like it was done for the TI driver (see *section 4.1*). They did not write directly a driver, they produced a set of documentation so that the driver development would be legally separate from the reverse engineering effort.

A second team of people led by *Michael* started to write a driver based on those documents. They based their driver on the **Intel ieee80211 stack** from the Centrino driver (see *section 3.28*), and added a **SoftMAC layer**. The SoftMAC layer add all the MAC management functionality missing from the Intel ieee80211 stack. Both the driver and the SoftMAC stack were merged in the Linux kernel 2.6.17.

The driver and SoftMAC have seen tremendous development and reached a point where they are usable and stable. Support for more variations of the Broadcom chipset was added, such as PCI-E, support basic and advanced encryption was added. This driver only supports **firmware V3**.

In 2006, *Michael* started porting the driver to the new **mac80211** kernel stack (see *section 4.9*). Along the way, the driver was significantly changed, this version only supports **firmware V4** (so dropped support for older cards), and should offer

support for the advanced features not supported in the SoftMAC driver, such as 802.11a. The mac80211 version is only available in the wireless-dev GIT repository, and should appear in a Linux kernel in the future.

In parallel, *Larry* has reused the SoftMAC driver and the mac80211 driver above to create an alternate driver for the mac80211 kernel stack that support only **firmware V3**. This driver supports cards that are not supported by the other mac80211 driver. Note that the names of all those drivers is currently in flux.

4.14 ZyDAS ZD1211 driver (USB dongles)

Driver status :	Stable
Driver name :	zd1211.o
Version :	2.18.0.0
Where :	http://sourceforge.net/projects/zd1211/
	http://zd1211.ath.cx/zd1211
	http://www.zydas.com.tw/downloads/download-1211.asp
Maintainers :	Markus Karg <markus-karg@users.sourceforge.net>
	ZyDAS
Mailing list :	http://sourceforge.net/mail/?group_id=129083
Documentation :	Web pages
Configuration :	Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc
Security :	WEP, WEP256, WPA
Scanning :	Wireless Extensions
Monitor :	No (need external patch)
Multi-devices :	?
Interoperability :	802.11b, 802.11g, 802.11a depending on hardware
Other features :	64 bit support
Non implemented :	-
Bugs :	-
License :	GPL
Vendor web pages :	http://www.atheros.com/

4.14.1 The device

After the release of the ZD1201 (see *section 3.29*), the logical step for ZyDAs was to release a 802.11g chipset, the **ZD1211**. This chipset has similar features as other 802.11g chipset, such as advanced security features, and it seems that some version also support 802.11a.

The main feature of ZyDAS chipsets is their small size and low power consumption, making them especially suitable for **USB dongles**. Actually, it

seems that this chipset is only found in USB dongles, even though it support other interfaces (PCI and Pcmcia). USB 2.0 is needed for full 802.11g operation.

ZyDAS was acquired by **Atheros**, and the ZD1211 is now known as Atheros **AR5007UG**.

4.14.2 The driver

Like for the ZD1201 (see *section 3.29*), *ZyDAS* themselves released a driver to support the ZD1211 under the GPL license.

At this point, a community organised around this driver : many people contributed patches and created branches of the driver. Most of the branches, which were aiming at simplifying the driver code or integrating it in the kernel, seems to have died. Instead, a new driver was written (see *section 4.15*).

Meanwhile, *ZyDAS* continues to release new version of the driver including bug fixes and patches from the community. *Markus* is hosting this driver and many patches on his web site.

4.15 ZyDAS ZD1211 SoftMAC driver

Driver status :	Stable
Driver name :	zd1211rw.o
Version :	-
Where :	Linux kernel (2.6.23)
	http://zd1211.wiki.sourceforge.net/
	http://www.linuxwireless.org/en/users/Drivers/zd1211rw
Maintainers :	Daniel Drake <dsd@gentoo.org>
	Ulrich Kunitz <kune@deine-taler.de>
Mailing list :	http://sourceforge.net/mail/?group_id=129083
Documentation :	Web pages
Configuration :	Wireless Extensions
Statistics :	Wireless Extensions
Modes :	Managed
Security :	WEP, 802.1x, WPA
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	?
Interoperability :	802.11b, 802.11g, 802.11a depending on hardware
Other features :	-
Non implemented :	Automatic bitrate selection, Ad-Hoc mode
Bugs :	-
License :	GPL
Vendor web pages :	http://www.atheros.com/

4.15.1 The device

This is the same device as the previous entry (*section 4.14*).

4.15.2 The driver

Various effort to cleanup and simplify the ZyDAS driver failed (see *section 4.14*), so *Ulrich* and *Daniel* decided to write a new driver for the ZD1211 USB.

The driver was developped based on the **SoftMAC** 802.11 stack (see *section 4.13*), saw a rapid development and was included in kernel 2.6.18. After that, it was gradually improved with new features and more hardware support. However, some important features, such as automatic bitrate selection, are missing from this driver, and users have to manually select a bitrate. The performance is also not as good as the official driver, but it is very stable.

Many of the shortcomings of the SoftMAC driver are due to function not implemented in the SoftMAC 802.11 stack. Rather than implement those features in their driver, *Daniel* and *Ulrich* decided to port their driver to the new **mac80211** kernel stack, which include those features (see *section 4.9*). This new driver currently does not have the same level of features and stability as the SoftMAC driver. This alternate version is currently only available in the wireless-dev GIT repository.

4.16 Realtek RTL8185L & RTL8187L (802.11g)

Driver status :	Experimental
Driver name :	PCI : r8180.o
	USB : r8187.o
Version :	0.22
Where :	http://rtl-wifi.sourceforge.net/
	http://rtl8180-sa2400.sourceforge.net/
Maintainers :	Andrea Merello <andreamrl *at* tiscali.it>
	Hauke Mehrtens <hauke@hauke-m.de>
Discussion forums :	https://sourceforge.net/forum/?group_id=186406
Documentation :	Readme files
Configuration :	Wireless Extensions
Statistics :	?
Modes :	Managed, Ad-Hoc
Security :	WEP
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	?
Interoperability :	802.11b, 802.11g, 802.11a depending on hardware
Other features :	-
Non implemented :	SMP

Bugs : -

License : GPL

Vendor web pages : http://www.realtek.com.tw/

4.16.1 The device

Like all the other 802.11b chipset manufacturer, Realtek introduced a new 802.11g chipset based on their earlier design, the RTL8180L (see *section 3.25*). It mostly adds support for the OFDM bit rates (up to 54 Mb/s), and some version also support 802.11a. The **RTL8185L** has a PCI/Cardbus interface, whereas the **RTL8187L** has a USB 2.0 interface. Realtek offers a proprietary turbo mode with 72 Mb/s, but other chipsets do as well (see *section 4.11*).

4.16.2 The driver

With those new chipsets, *Realtek* changed their driver strategy. Instead of releasing a binary driver (see *section 3.25*), they decided to team up with *Andrea* who produced the GPL driver for their previous chipset (see *section 3.26*). They sent him a few reference design cards (prototypes) and specifications to help him.

Andrea has added preliminary support for those cards in the new branch of the RTL8180L driver, called **rtl8180-sa2400**. He also created a driver for the USB dongle called **rtl8187-dev**. Those branches are using their own 802.11 code.

To gain robust 802.11 support, he decided to create a new versions of the driver based on the **Intel ieee80211 stack** from the Centrino driver (see *section 3.28*), those branches are called respectively **rtl818x-newstack** and **rtl8187-newstack**.

Mid 2005, *Andrea* stopped this work, leaving his work on the new branches unfinished, and later started helping the mac80211 driver (see *section 4.17*). Beginning 2007, a group of users led by *Hauke* restarted working on the driver, with a new project page and new repository. Their main work so far has been updating the driver to work with the latest kernel.

4.17 Realtek RTL8187L mac80211 driver

Driver status : Experimental

Driver name : rtl8187.o

Version : -

Where : Linux kernel (2.6.23)

Maintainers : Michael Wu <flamingice@sourmilk.net>

Andrea Merello <andreamrl *at* tiscali.it>

Discussion forums : https://sourceforge.net/forum/?group_id=186406

Documentation : -

Configuration : Wireless Extensions

Statistics : Wireless Extensions

Modes : Managed, Ad-Hoc

Security : WEP, 802.1x, WPA

Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	?
Interoperability :	802.11b, 802.11g, 802.11a depending on hardware
Other features :	-
Non implemented :	-
Bugs :	-
License :	GPL
Vendor web pages :	http://www.realtek.com.tw/

4.17.1 The device

This is the same device as the previous entry (*section 4.16*).

4.17.2 The driver

Michael and *Andrea* decided to port the **USB driver** written by Andrea (see *section 4.16*)to new **mac80211** kernel stack (see *section 4.9*). *Michael* did most of the work, with *Andrea* helping with the tricky parts about the hardware.

This driver was included in Linux kernel 2.6.23. It will be the first driver included in the kernel to use the new **mac80211** kernel stack, and as such it may be a bit rough for a while. Because it uses the mac80211 stack, the driver offer a wide array of features, such as monitor mode, WPA and virtual interfaces, and as mac80211 evolves this set of features should increases.

4.18 Marvel Libertas (802.11g)

Driver status :	Stable
Driver name :	libertas.o
Version :	-
Where :	Linux kernel (2.6.22)
Maintainer :	Dan Williams <dcbw@redhat.com>
Mailing list :	http://lists.infradead.org/mailman/listinfo/libertas-dev
Documentation :	Readme
Configuration :	Wireless Extensions and debugfs
Statistics :	Wireless Extensions
Modes :	Managed, Ad-Hoc, Mesh
Security :	WEP, 802.1x, WPA
Scanning :	Wireless Extensions
Monitor :	Yes
Multi-devices :	?
Interoperability :	802.11b and 802.11g
Other features :	Firmware loading via HotPlug, 802.11e (QoS)

Non implemented : -

Bugs : ?

License : GPL

Vendor web pages : http://www.marvell.com/

4.18.1 The device

Marvell is a networking company mostly based in the Silicon Valley, they design various networking and storage chips. When Marvell decided to release a 802.11 product, the market was already very crowded. Obviously, they would have to include all the features of the latest standard, but they would also need a few differentiators.

The Libertas chipset was designed mostly for the **embedded** market and access point market. Embedded devices don't have much resources, and use all kind of weird software platforms, therefore Marvell decided to push as much functionality as possible in the chip, so that interfacing to the chip and writing driver would be simplified and use less resources. The Libertas chipset includes an ARM core and the whole 802.11 stack, including management and security, runs on that core. This is very similar to the architecture of early 802.11b chipsets (FullMAC).

The Libertas chipset has all the features of a modern 802.11g chipset, such as security (802.1x, WPA) and QoS (802.11e). Libertas also added proprietary extensions to increase bitrate and range.

One of the interesting thing of the architecture of this chipset is that some additional code can be run on the ARM core. For example, in the case of the Access Point version of this chipset, all the access point functionality is running on the ARM core and the chip doesn't need an external CPU. The One Laptop Per Child project contracted CozyBit to develop a **proprietary mesh** functionality running inside the Libertas chipset. The advantage of such an implementation is that packet in the mesh can be forwarded without having to wake up and go through the CPU.

4.18.2 The driver

The **One Laptop Per Child** (OLPC) project wanted integrate wireless mesh capability into their devices. Their desire was to be able to off-load the mesh functionality to the chipset, to save battery power, however nearly all modern 802.11 design are softmac, a simple hardware with all the 802.11 intelligence in the driver. One of the few candidate was the Marvell Libertas, however there was no Linux driver and Marvell was restrictive about sharing information. The OLPC project convinced Marvell to allow them to write a Linux driver and release it.

The driver was written for OLPC by Red-Hat, lead by *Dan* (who also wrote NetworkManager). The driver does not use any 802.11 stack, because such functionality is implemented in the firmware, and it offers all modern functionality such as Scanning and WPA support. The Linux driver also supports the **proprietary mesh** functionality when using the appropriate firmware. The driver has been developped and tested in the OLPC source tree for a long time, and after

various changes required for kernel inclusion, it was eventually included in kernel 2.6.22.

4.19 AirgoNetworks AGNX00 (TrueMIMO)

Driver status :	None yet
Driver name :	-
Version :	-
Where :	http://airgo.wdwconsulting.net/mymoin
Maintainer :	Jeff Williams <jeff AT SPAMFREE esawireless DOT com>
Mailing list :	-
Documentation :	-
Configuration :	-
Statistics :	-
Modes :	-
Security :	-
Scanning :	-
Monitor :	-
Multi-devices :	-
Interoperability :	-
Other features :	-
Non implemented :	-
Bugs :	-
License :	-
Vendor web pages :	http://www.airgonetworks.com/
	http://www.cdmatech.com/products/wlan.jsp

4.19.1 The device

AirgoNetworks was a small startup based in the Silicon Valley, and one of the pioneers in MIMO designs. AirgoNetworks decided to apply their expertise of MIMO to the successful 802.11 standard, and released in 2003 some 802.11g products with a proprietary MIMO extension. Their MIMO extension is branded **TrueMIMO** and initially delivered bitrate of 108 Mb/s, further version increased it to 240 Mb/s.

MIMO uses multiple antennas in the radio transmitter and in the receiver to exploit the physical diversity of the channel, especially indoor where you typically have multipath. The radio path between each pair or transmitting and receiving antenna is different, and therefore using clever techniques on both sides, it is possible for the receiver system to properly decode multiple simultaneous transmissions from the transmitter system. Using MIMO, it is possible to increase bandwidth by sending multiple data streams in parallel, the bandwidth increase

depends on the number of antenna used. By using more robust encoding, it is also possible to increase range (at lower speed).

Shorty after the release of AirgoNetwork products, the IEEE 802.11 committee decided to create the new **802.11n** standard, a 802.11 standard using MIMO technology. The new 802.11n standard was not planned to be compatible with TrueMIMO, and as of 2007 has not been yet finalised. Because there is not standard yet, any devices that is compliant with the evolving 802.11n draft is called pre-802.11n, and as the draft evolves future compatibility is unknown.

The strategy of AirgoNetworks was quite successful, and the vast majority of the early and non pre-802.11n devices were based on their chipset. However, the other major 802.11 vendor reacted quickly, first by releasing their own proprietary MIMO chipset, and then by releasing pre-802.11n products. The 802.11n standard is now gaining momentum, therefore the newer chipsets of AirgoNetworks are compliant with the 802.11n draft, with bitrate up to 315 Mb/s.

AirgoNetworks was acquired by **Qualcom**, most likely because of its patent portfolio on MIMO technology. Qualcom sells the former AirgoNetworks chipset under a new name but with the same TrueMIMO branding.

4.19.2 The driver

Like a lot of 802.11 vendors, AirgoNetworks does not want to release any driver of specification to the Linux community, even though they have internally a Linux driver (used on their Access Point platform).

Jeff Williams started reverse engineering the binary Linux driver of AirgoNetworks, and documented the interface for the AGN100 and AGN300 chipsets. *Jeff* previously led the reverse engineering on the TI ACX driver (see *section 4.1*), and because of the problems with the ACX driver, he is using the methodology of the team who wrote the Broadcom Linux driver (see *section 4.13*) : he is not implementing any driver and only publishing the specification of the chipsets.

Currently there is no Linux driver available, but hopefully some people will use the work of *Jeff* to produce a Linux driver.

5 The devices, the drivers - other

This section contains information about some devices which don't have a specific entry in one of the previous three sections.

5.1 Not supported (the hall of shame)

Netwave AirSurfer plus (in 802.11 mode), **BayStack 650** : Now that a driver for the BayStack 660 is available, it should be quite easy to make a driver for those cards, by reusing the physical layer parts in the AirSurfer plus driver. FreeBsd seems to have a driver for this device... This product has been discontinued.

RadioLan has a 10 Mb/s at 5 GHz product, rather very short range and no Linux drivers. This product has been discontinued.

WebGear Aviator 900 MHz : connect to the parallel port and offer cable replacement solution. No functional Linux driver. This product has been discontinued.

The **IBM Wireless LAN Entry** is a discontinued product that may be sometime found for a very very low price. Unfortunately, there is no working driver for those and information on the device is impossible to find. This product has been discontinued.

Both **TI** and **Broadcom** are very unfriendly toward Linux. They don't release Linux drivers or any information to help build Linux drivers for their cards despite using Linux drivers internally and in some products. Open Source drivers for those cards exist, based on reverse engineering, but because those drivers probably will never support all features/models of those cards and because of the overall attitude of those companies, you may want to avoid those cards.

Conexant (formerly Intersil) seems to have reverted their previously pro-Linux approach to ignoring the Linux community (which did help them so much).

5.2 Win32 drivers

Two packages enable the use of Win32 wireless LAN drivers under Linux :

DriverLoader : http://www.linuxant.com/driverloader/

NdisWrapper : http://ndiswrapper.sourceforge.net/

DriverLoader is commercial, whereas NdisWrapper is GPL. Both seems to work with a wide range of Win32 drivers and Linux features, for example both support Broadcom, TI and Intel drivers, and most Wireless Extensions including Wireless Scanning.

Some people have strong objections to using Win32 drivers under Linux. Obviously, such a solution is restricted to the i386 architecture, and make use of a lot of binary code. Also, some less used Wireless Extensions features are not available in the NDIS specification (such as link quality, noise level, iwspy, master and monitor mode).

5.3 A note on driver licenses

Donald Becker's web page alerted me on the **license and copyright** issues for networking drivers (see http://www.scyld.com/expert/modules.html#legal for details). If you just plan to use the driver in your Linux PC, there should be no problem, but if you plan other use of the drivers you should pay attention to the exact license the driver come in.

Most drivers are **GPL**, which prevent their use with non-GPL kernels (so commercial operating systems can't reuse the code) and prevent to use portions of the source in non-GPL drivers, except with the explicit authorisation from the author.

Some other drivers come with a **binary library**, which restrict its potential use (the driver can't be ported to other architectures).

This may be restrict what you can do with those drivers, but those people have spend long nights and week ends convincing the hardware manufacturer to release

information, writing and debugging the code, so please respect their copyrights and decisions.

5.4 More information on the devices, other Wireless LANs

You will notice that I don't give too much information on the different devices. The web page of each **vendors** usually contain the full specification of the products they sell.

They are many more products available than the ones that I've listed (which are the most common). If your favourite wireless LAN is not listed above, either there is no driver under Linux that I know of, or it is an **OEM** version of one of these (same hardware under a new brand).

5.5 Other Wireless technologies

5.5.1 Wireless bridges

Wireless bridges allow to connect different networks via radio, their goal is to replace a dedicated leased line (T1, for example). They usually offer longer distance through **directional antennas**, and are peer to peer.

These devices are a totally independent box (like other bridges, routers or gateways) and not a card to plug in your PC, so have no interactions with Linux.

5.5.2 Radio Amateur and AX25 (HAM)

These devices are quite specific and are described in their own *howto*.

5.5.3 Infrared

Apart from the remote control stuff, most infrared devices are **IrDA** compliant. IrDA defines a full lightweight protocol stack on top of very cheap and simple hardware, and is optimal for short ad-hoc transactions (using Obex for example). TCP/IP networking over IrDA can be done using **PPP over IrComm**, **IrLAN** or **IrNET** (all of them point-to-point solutions).

More information on IrDA for Linux is available at :

http://irda.sourceforge.net/

http://www.tuxmobil.com/howtos.html

http://www.hpl.hp.com/personal/Jean_Tourrilhes/IrDA/IrDA.html

There is also some real Wireless LANs using **diffuse infrared** (no more peer to peer), but I don't have much information on these.

5.5.4 BlueTooth

BlueTooth is a radio standard heavily influenced by IrDA and USB, and offers the functionality of a wireless USB and serial cable replacement (see *section 8* for a more complete description). BlueTooth defines its own protocol stack as well, and offers the possibility to create long term binding between devices (attach wirelessly peripherals to a phone or a PDA). TCP/IP networking over BlueTooth can be done using **PPP over RfComm** or **PAN** (BNEP).

More information on BlueTooth for Linux is available at :

http://bluez.sourceforge.net/

http://sourceforge.net/projects/bluetooth/

http://www.holtmann.org/linux/bluetooth/

http://www.hpl.hp.com/personal/Jean_Tourrilhes/bt/

5.5.5 Digital mobile phones and other radio WAN

Again, this is quite different from Wireless LANs. I don't know much about those devices, except the usual generalities.

Digital mobile phones (*GSM*, *TDMA*, *CDMA*, *PHS*) very often allow data connections (slow and expensive). Most of them offer a standard serial interface with an extended AT command set, so can be configured like a normal modem : ppp over the serial port.

Some Nokia GSM phones use a kind of half winmodem interface where the upper layer handling is done by the host. For these phones you need **gnokii** (http://gnokii.org). This package also provide tools to play with various extra features of the phone (SMS, address book...).

Of course, the ultimate geek challenge is to use IrDA (see *section 5.5.3*) to connect your mobile phone to your laptop. That's done with PPP over IrComm or gnokii.

In most cases, **Wireless WANs** such as *CDPD* cards, the *Metricom Ricochet* and *ARDIS* should use modem interface as well.

6 Wireless LANs in use

Installing and using a Wireless LAN is not such a big deal, and is not much different from other kind of networks. In this chapter, I will give you a few tricks on how to install those beast and will mostly redirect you to a lot of literature explaining the things much better than I would do.

Then, I will explain some of the difference of Wireless LANs compared to wired technology from the user point of view and why it reacts sometime differently. For more curious people, see the *section 5*.

6.1 Choice and selection of a Wireless LAN

There is far too many people buying a Wireless LAN and discovering only after that it is not supported under Linux. So, please, check that a driver is **available** for the hardware you plan to use.

Most Wireless LANs are designed to work well in most configurations, but my experience tells that some Wireless LANs or some environment may be capricious. Of course, the vendor won't advertise this, so it's your responsibility to check that the Wireless LAN is working with your particular setup. If you intend to cover a large range, **test** as many physical locations and combinations as possible to avoid surprises. Know the limits of your hardware.

The performance of different Wireless LANs may vary widely, depending on may factors. The throughput of two Wireless LANs advertising the same bit rate may vary by a factor 5 (I won't give the names). Range also can have wide

variations, even between similar cards. So, be warned and **benchmark** your Wireless LAN...

If you are not happy with your choice of Wireless LAN, don't hesitate and **return** it to where you bough it for a refund.

6.2 Features of the hardware

Obviously, I don't need to tell you to check the **price** of the Wireless LAN you plan to buy, but however consider checking the price in different places, especially on the Internet. Some hardware are still difficult to find and impossible to get directly from the manufacturer, and a bit of research always pay off...

Then, the second issue is the **form factor**, and the interface use to connect the card to your PC. Most cards nowadays are in *Cardbus* form factor and *USB* form factor, most vendors offer as well *PCI* or *Mini-PCI* versions of their cards. Older cards may come in *ISA*, *Pcmcia* (16 bits) and *CF* form factor. In some cases, it is possible to plug a Pcmcia or Cardbus card in a desktop using ISA-to-Pcmcia or PCI-to-Cardbus bridges, but those are not always properly supported. I've seen some old cards in *PC104* form factor (for embedded applications).

Often, people do wonder about **Access Points** and **residential gateways** and if they need some for their network. In most cases, the Access Point is a bridge and allow to connect the wireless network to an *Ethernet backbone*, whereas the Residential Gateway connect it to an *ISP* (via dialup/cable/DSL), and those two kind of products are usually not fully interchangeable. However, most Wireless LAN cards work in ad-hoc mode (so without Access Point), and a Linux box can offer connectivity to other networks (routing+proxy ARP, or masquerading).

For people who plan to establish point to point links, they will likely need the card to offer a **connector** for an **external antenna**. Quite often those connectors are small and not really standard. Of course, if the card doesn't offer such a connector, it's always possible to butcher it...

Then, you may care about all the usual performance characteristics, such as **speed**, **range**, and **power consumption**. However, be aware that it's quite difficult to compare products, for the speed read my various comments about *overhead* and *benchmarking*, and for range pay mostly attention to the *transmit power* and the *sensitivity*.

Finally, each implementation may offer more or less **wireless parameters**. Having those parameters will allow to tune the card for specific environments and configurations. With that, you probably want some deployment and diagnostic tools from the vendor...

Personally, I believe that the number one consideration in making your choice should be the range (coverage), and that raw speed is the least important of the things mentioned above. If you need speed, you should use Gigabit Ethernet. Wireless enables mobility, and you want to make sure to retain network connection wherever you are.

6.3 Interoperability

For people having dealt with Ethernet, it may seem absurd to see all the interoperability worries that we have to go through for Wireless LANs. But such is life, and product A may not communicate with product B. To their defense, Wireless MAC protocols are a few order of magnitude more complex than Ethernet and have not been around for as long.

For most of the old hardware, and for all the **proprietary** products, interoperability is none. In other word, those products communicate only with products from the same vendor. The risk for you is that you are locked with this vendor to upgrade your network. That is still OK if the vendor is strong and has been around in the business for a long time.

The only two standards that have been demonstrated to be interoperable across different implementations are **802.11-FH** and **802.11-DS** (other standards may be interoperable due to a single implementation). I must there applaud the various vendors which have gone through great pain to make sure that their hardware was playing nice with others and fixing their interoperability problems. Note that 802.11-FH products are not interoperable with 802.11-DS products, and vice versa, so you can only mix products of the same kind of 802.11.

As the **802.11-b** specification is just a simple extension of 802.11-DS (adding 5.5 and 11 Mb/s speeds), all products interoperate at least at 2 Mb/s (802.11-DS mode), and all products I have seen also interoperate at 11 Mb/s.

In general, all **proprietary extensions** of 802.11 have poor interoperability, they will only work if all devices are from the same vendor, however those devices will interoperate at standard speeds with other products. **802.11+** is a semi-proprietary extension of 802.11b, adding the 22 Mb/s speed. All products are based on the same chipset, and they also interoperate with 802.11b products.

The **802.11g** standard is a non straightforward extension of 802.11b. Early products from different vendors did **not** always interoperate properly with each other at higher speeds (but will interoperate at 802.11b speeds), and sometime they did **not** interoperate with 802.11b products and may actually disrupt 802.11b networks. Recent 802.11g products are much better, except for Ad-Hoc mode which is usually poorly supported.

Early **802.11a** products (5 GHz) are interoperable, because they were based on the exact same chipset, recent products have good interoperability. They do not interoperate with 802.11b products, because using a different frequency band, but some products doing both 802.11b and 802.11a are available.

Pre-802.11n products have a poor track record at interoperability so far, however it will get better as we get closer to the final standard.

Other standards (such as HiperLan, HiperLan2 and SWAP) are interoperable in theory. As there is very few vendors using those standards (or even none) and the products are often too recent, we can't say much about interoperability. Note that compliance doesn't mean interoperability and vice versa...

Note that in some cases, the **Linux driver** of a device doesn't implement all the features of the corresponding Windows driver (typically security), limiting the interoperability.

6.4 Which vendor to select

Buying a Wireless LAN card used to be relatively straightforward, because there was a limited choice, but has now become very complex and confusing.

The first problems is that there are now lot's of **different vendors and brands**, and most I've never heard off (and they tend to be the cheapest, so the most popular). It's almost impossible to track down all of them. Fortunately, there is only a limited number of chipset, so most of those vendors sell the same hardware (or minor derivation) and there is a good probability that it works with an existing Linux driver.

The other issue is that some vendors use **different chipsets** in their various products, so some of their cards might be supported and some might not be. They can switch their entire product line to a new chipset pretty quickly, so a "brand" is not a safe bet. Some vendor even sell different chipset under the **same model number** (example D-Link and Linksys), making it very difficult to know in advance if the card is supported or not.

In the past, most of the "generic" brands selling 802.11b cards were using the **Intersil PrismII** chipset, which is well supported under Linux (various drivers), so it was not a problem. A lot of them have recently moved to non-supported chipset. Plain 802.11g cards are nowadays usually supported, but in most case require work to get the driver going. Laptop with **Centrino** cards are a very good choice. It's usually easy to identify **Atheros** cards, they usually advertise SuperG and 108 Mb/s. Pre-802.11n cards are usually not supported.

There are various **hardware surveys** that attempt to capture the latest list of compatible cards (see Wireless Howto front page). You can also **search** the various mailing list archives and the web for information.

6.5 How to identify a card

Let assume you already have a wireless card plugged in your PC, and want to know which one it is and which driver you need. Linux has usually a way to display a **card identification**, but this depend on the type of card.

If the card is an **ISA** card, you are usually out of luck.

If the card is a **PCI** card, you need to use the command "*lspci*" to display the card identification strings.

If the hardware is a **USB** dongle, you need to use the command "*lsusb*" to display the dongle identification strings. In some case, "*lsusb*" doesn't work (for example if usbfs is not mounted), and you can get the identification strings from the kernel log using "*dmesg*" (or in /var/log/messages).

If the card is a **Cardbus** card (32 bits Pcmcia), and if you are using **kernel 2.6.X** or kernel 2.4.X with the kernel Pcmcia subsystem, you need to use the command "*lspci*" to display the card identification strings. If the card is a **Cardbus** card (32 bits Pcmcia), and if you are using an older kernel with the standalone

Pcmcia subsystem, you need to use the command "*cardctl ident*" display the card identification strings. Try both and see what comes out.

If the card is a true **Pcmcia** card (16 bits), and if you are using **kernel 2.6.14** or later, you need to use the command "*pccardctl ident*" to display the card identification strings. If the card is a true **Pcmcia** card (16 bits), and if you are using an older kernel, you need to use the command "*cardctl ident*" display the card identification strings. Note that cardmgr will also write some identification strings in the message logs (/var/log/daemon.log) that may be different from the real card identification strings.

The card identification usually helps to identify the **chipset** inside the hardware, and in some other cases it does not, because the vendor has changed the identity. Once you have identified the chipset, it is usually straightforward to check if the hardware is supported and which driver to use.

Most Linux drivers knows about some of those card identifications, and will automatically bind to the hardware. It is usually simple to add new identification to a driver.

Jacek Pliszka recommends to get the FCC-ID written at the back of the hardware and to run it through the FCC database. He also recommend to check the Windows driver (both identification and file name) for some clues.

6.6 Wireless hardware compatibility

- Linux Wireless Howto - Second section, organised by chipset
- Linux hardware surveys - Not always complete
- Community wireless web sites - Usually most up-to-date, by vendor
- Linux wireless tools web sites - List cards compatible with the specific tool
- Mailing list archives - Use a search engine

6.7 Driver installation

The reader should be familiar with some of the documents listed in the *Useful readings chapter* below, because the information here mainly acts as a complement to them. A good knowledge of your Wireless LAN is also a prerequisite before switching to Linux.

Most Wireless LAN vendors have tried to make things easy and offer product with an interface as similar as possible as **Ethernet**, and which work mostly the same way. So, a bit of background on Ethernet and the general Linux networking is welcomed (see below).

The operating system need a piece of software to interface to the hardware. That is the role of the **driver**. Basically, when Linux gives to the driver a packet to send, the driver have to copy the packet to the hardware and toggle the correct bits in the correct register on the card to send it. It is the same when the card generates an interrupt, the driver go and read the packet and give it to Linux. Of course, the driver needs to know about the specific hardware details and the specific operating system ways.

In conclusion, you must check first if the driver for your Wireless LAN exists (see *previous section*), because in many case it proves to be quite useful...

With Linux, you normally have to **compile** the driver source code, however most Linux distribution offer precompiled modules for the most popular drivers. There is usually two compilation options : drivers compiled staticaly in the **kernel** and as a **module**. If the driver is already in the kernel sources, the compilation is quite simple (you have to enable it in the kernel configuration, static or module). If it is in the **Pcmcia package**, you just need to install this package.

Otherwise, see the installation instructions coming with the driver which should detail the procedure to install it.

Once you've got the driver compiled and installed, you must tell your system about it. This involve getting the driver to load (see *section 6.9*), wireless configuration (see *section 6.10*) and network configuration (see *section 6.12*).

6.8 Useful Linux related readings

- Ethernet HowTo - How to install and configure most of the network drivers
- Net2 HowTo - The network stack story
- Module HowTo - To compile you driver as a module
- Pcmcia HowTo - An excellent medicine for pcmcia drivers
- AX25 HowTo - AX25 and Radio Amateur users should enjoy this one
- The Linux Network Administration guide - A lot of background and tips about networking and Linux
- Your distribution documentation or FAQ (if any)

6.9 Driver loading and driver parameters

If your card is removable (Pcmcia, USB), you usually want the system to **automatically load** the driver when the card is plugged or activated. The Pcmcia subsystem is usually very good at doing this, and has it's own scripts or uses the Hotplug scripts. The USB subsystem uses HotPlug scripts.

For non-removable cards, things are simpler. If the driver is compiled staticaly in the kernel, it will be loaded at boot time. If it's a module, you can use **Hotplug** or **udev** to automatically load the driver, or you can load it as needed using /etc/modules.conf (see below).

Some older drivers (typically ISA) may load only with some required **driver parameters**. Those parameters usually allow to specify the base address and interrupt of the hardware to avoid scanning, but also might be used for multi-device configuration or wireless specific parameters (see below). Note that for all modern Plug-and-Play cards (PCI), those extra parameters are no longer necessary, as the system figure things by itself, so you can omit them (or set them to 0).

A lot of users are confused when it comes to set driver loading and parameters. As it is explained nowhere correctly, I disgress a bit and give you a few hints on how to load drivers with and without parameters...

For driver compiled **staticaly in the kernel**, the parameters are passed on the kernel command line. The syntax is "ether=*irq,base,name*" where *base* is the base address, *irq* the interrupt and *name* the device logical name (ex : *eth0*). The kernel command line is passed by *lilo* (or *loadlin*) itself, so in fact it means that you add in /etc/lilo.conf a line which look like this :

```
append="reboot=warm ether=0,0,eth1 ether=10,0x3E0,eth2 ether=11,0x390,eth3"
```

For drivers compiled as **modules** (but which are not for removable devices), the parameter interface is much more flexible and each driver may be different, so you must look in the documentation. Basically, the driver define a set of parameters by their name and you may set for each keyword an array (one value for each instance of the hardware). The module configuration is usually done in /etc/conf.modules like this :

```
alias eth1 hp100
alias eth2 wavelan
options wavelan io=0x3E0,0x390 name=eth2,eth3 irq=10,11
```

For **Pcmcia modules**, the configuration is usually done in the pcmcia scripts in the directory /etc/pcmcia/, and you should check the Pcmcia Howto for details. Note that most modern distributions may use the HotPlug scripts. Usually, you don't need extra driver parameters, as Pcmcia is Plug-and-Play, and all driver part of the Pcmcia package are already preconfigured for proper auto-loading.

However, you need to make sure the Pcmcia subsystem load the driver you desire, if there are multiple drivers bound to the same device you may end up with an unexpected driver. In this case, you need to edit the various Hotplug config files if using Hotplug, or the Pcmcia config files (in /etc/pcmcia/ - *grep* is your friend).

For **USB modules**, you may use the HotPlug scripts. USB usually don't require any driver parameters, but again, you need to make sure the proper driver is loaded.

Before following up with the wireless configuration, you may want to make sure the driver is properly loaded, recognise the hardware and can initialise it. This can be done by checking the message logs (with *dmesg*).

6.10 Wireless configuration (more parameters to configure)

The most obvious difference with Ethernet cards is that there are **more parameters** to configure. In order to communicate, all nodes of the network must have those parameters configured the same. Some examples are : *frequency* or *hopping pattern*, *network id*, *domain* or *essid*, *encryption key* (for security)...

Under *Windows,* the installation program usually opens a nice window and asks the user to enter these parameters, or sets them to a default value. Some drivers set those parameters in a permanent storage in the device (*EEprom*), so the Linux driver is able to reuse them. But, the current tendency is to scrap the EEprom and to use the Windows *registry* to save those parameters instead. Of course, the Linux driver can't retrieve the parameters in those conditions.

The **Wireless Extensions** (see *next section*) has been designed to simplify the process of setting those parameters under Linux by providing an unified interface across drivers, but not all drivers support (yet) the Wireless Extensions...

For the majority of drivers that support it, Wireless Extensions allow to change parameters at run time (using various tools - see my web pages) or be use at boot time or card insertion time through various initialisation scripts (this is usually distribution specific). Most Linux distributions nowadays fully integrate wireless configuration in their network configuration. The **Wireless Tools** package contains additional documentation on all this.

http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html

The Wireless Tools also allow you to **diagnose** various basic problems (configuration not applied, not connected to the Access Point, invalid encryption key) and monitor the link. You need to make sure that the wireless link is established if you want your network setup to be successful.

Advanced security configuration, either 802.1x or WPA configuration, is done through **wpa_supplicant** or **xsupplicant**, which are daemons/applications handling those protocols. For those that prefer GUI, **NetworkManager** is a tools that allow to configure both the Wireless Extensions and wpa_supplicant graphically.

Drivers that don't use Wireless Extensions can use a variety of methods for configuration. Some have their own dedicated tools, some use a /proc interface and some use modules parameters, or a combination of these. The *previous sections* of the Wireless Howto should give you some hints on the subject.

In conclusion, you must read your documentation to know what parameters need to be set, what they are used for, and look the Linux driver documentation to know how to set them under Linux. See below for a suggested list of information sources.

It is usually quite a good idea to install the Wireless Lan first under some mainstream operating system with the official vendors driver and tools, to have a feeling of how the beast does work. You might also compare the performance before and after :-)

Once you've got all those new parameters set, your Wireless LAN should be up and running.

6.11 Where to get information about your Wireless LAN configuration

- The official documentation that come with your product.
- Manufacturer web page and support.
- Linux driver source code, documentation (headers, man pages), maintainer.
- Documentation and man pages in the Wireless Tools package.
- Network configuration documentation for your specific Linux distribution

6.12 Network configuration

After having setup the wireless parameters, you need to configure the regular network parameters. At the minimum, you need to tell the system to use DHCP or

not. Very often, you may have to configure an IP address, Gateway and a DNS server. More complex setup (such as a Wireless Router) will require some route and firewall configuration.

All this is exactly similar to standard Ethernet cards, and often specific to each distribution, so please check the relevant documentation. Note however that bridging very often won't work (see *section 6.14*).

6.13 Wireless LAN deployment

From the network administrator point of view, the main problem with Wireless LANs is that the **medium is shared**. If on a cable you can physically know who is connected, however anybody and anything can use the radio band.

To try to separate everyone out there, most products define some *network identifier* (802.11 calls it ESSID). This is a number or character string which is used to identify all the people wanting to be one the same **logical network**. Networks using different *network identifiers* still share the bandwidth, but are logically separate and ignore each other.

This situation is not totally ideal, so that's why usually you have some **distinct channels** (or *frequencies*, or *hopping patterns*). People on distinct channels use different part of the bandwidth, so don't interfere at all. If you want to install multiple independent networks in the same area, this is the way to go.

The Wireless LAN has only a limited range, so you may reach only device within that range. This is usually why you should define some cells where everybody is in range. If you want those cells to communicate or a node to move across cells, you should install an **access point** in each of those and configure those with the same *network identifier* (and add an Ethernet segment between the access points).

On the other hand, some time you just want to quickly set up a network between a group of nodes and don't want to build an infrastructure. Most Wireless LANs offer **ad-hoc networking**, allowing you to just do that (apart from TCP configuration).

Some network administrators are also a bit scared by security problem over the medium. The only solution is to use **encryption**.

6.14 Access Points, Home Gateways and Ethernet bridging

Most **Access Points** act as a MAC level bridge, allowing the Wireless LAN to be a natural extension of a wired network. They are deployed in a cellular fashion, and provide extended security, management and roaming.

On the other hand, the **Home Gateways** allow a single cell to be connected to a WAN, like a modem, a cable modem or a DSL access. The set of features is quite different, and they offer NAT/masquerading and PPP configuration. Most Home Gateways also have an internal Ethernet switch and act as a MAC level bridge between their wireless and internal wired side.

The conventional **Ethernet bridging** method (promiscuous sniffing) doesn't work with most wireless LAN standard, because of the header encapsulation and the interactions with link layer retransmissions. In other word, most often, when

you use a software bridge software on a wireless LAN card (such as the Linux bridge on a 802.11 card), it doesn't work (moreover, quite often promiscuous is broken as well).

The driver could work around this restriction by creating its own MAC headers (802.11 headers instead of 802.3, and putting the right bits in the right place), but in fact most vendors don't provide the specification on how to this with their hardware (when they don't explicitly prevent it in hardware, to force you to buy their Access Points).

In other words, don't expect to use your Linux PC as a wireless bridge or wireless Access Points with most products out there, and forget about turning it into an Access Point. Of course, there are some exceptions that are listed in the driver descriptions (*section 2*).

The workaround is to set the wireless LAN in **ad-hoc mode** and to use other methods, such as routing, masquerading, IP bridging, ARP proxying...

6.15 Point to point links (connecting different LANs by wireless)

Most Wireless LANs are designed to be used as a local area network, where all the nodes can see each other or can see an access point, and each node is attached to other networks through a single access point (or not at all in ad-hoc mode).

Some people have asked me question on how to use Wireless LANs to connect different LANs together using wireless technology, usually those LANs are in **distant places** (across the street). Most of the time, you can't use a Wireless LAN because you don't have a fully connected topology (some node can't see each other, it's more a set of point to point links) and you may need to use directional antennas to overcome the distance.

I've never personally tried this, but I see 2 ways to achieve this.

The first solution is to use **Wireless Bridges**. Each Wireless Bridge is connected to one of the LAN section and redirect the traffic over the air to the correct destination. There is many products on the market, they are a bit expensive but very flexible, transparent and optimised for the task.

The second is to use normal Wireless LAN cards, and to plug them in a **router** (for example a Linux PC). I recommend to use a Wireless LAN supporting *RTS/CTS* if you have more than one link, and to set them in ad-hoc mode (no access point). Each LAN segment must have a different *IP subnet*, and the wireless link must have it's own subnet (it can be a private subnet if you use masquerading). After much configuration of the routing tables of your network, you should be able to get it working.

Some people using the Aironet Arlan cards for this kind of application have made a very nice **Arlan Wireless Routing Howto**, and I believe it can apply to most other Wireless LANs as well :

http://www.rage.net/wireless/wireless-howto.html

Note that it is not always possible to use a **bridging software** on top of a Wireless LAN card, and that is why I do recommend routing (or proxy-ARP + IP

forwarding). Setting the card in promiscuous mode won't give the behavior expected by the bridge, because of the interaction with MAC level retransmissions. Some drivers are clever enough, and by playing directly with the 802.11 headers (if the hardware allows it), they can allow bridging to work, but most drivers are not.

6.16 Performance (speed)

Most people want to know how fast it goes, their first surprise after benchmarking is that they can't get the speed written on the box, and that the number seems low. Even when converting the byte per seconds to bit per seconds, it is obvious that the **TCP throughput** is much lower than the **signalling rate**. This is because the Wireless LANs are slower to start with , may not use their fastest speed and on top of that have higher overhead.

Most Wireless LANs products only specify their highest **signalling rate** (also called bitrate). The signalling rate is the speed the bits are send over the air (802.11b is up to 11 Mb/s, 802.11g is up to 54 Mb/s) and it doesn't account of all the overhead of the various protocols.

Most protocols have **multiple rates** and adapt the signalling rate depending on the quality of the link (for example 802.11b can run at 1, 2, 5.5 and 11 Mb/s). When the link is clear and reception is strong, it uses the fastest rate, but when there is noise/interference or the devices are further away, it downgrades to a more robust rate, which is slower. The throughput that you will get will depend on which rate you are using, for example the highest speed might be only usable in line of sight. The environment conditions constantly change, so most products adjust their rate continuously, and the bandwith available can be unpredictable.

The Wireless LAN protocols also have usually a **higher overhead** than their wired counterpart (such as Ethernet) because of some technological limitations and to improve the reliability and the coverage of the Wireless LAN (optimisation trade-offs). Wireless protocols have a lot more protocol overhead, such as contention, large headers, encryption, management frames... Retransmissions are needed to overcome various radio conditions. Part of the protocol overhead is fixed and does not scale with the bit rate, which means that at the highest rate this overhead is becoming proportionally larger. At 11 Mb/s the TCP throughput can be almost reduced to 50%.

6.17 Reliability

Most Wireless LANs protocols include mechanisms to improve the **reliability** of the packet transmissions to be at the same level or even better than Ethernet (*MAC level retransmissions* for example). Anyway, if you are using a protocol such as TCP (the default under Linux), you will be fully protected again any loss or corruption of data over the air. In other word, when you copy a file across the radio, it can't be corrupted (but it might fail completely).

6.18 Coverage

As said earlier, people get excited about speed, and they often don't realise that the main measure of performance of a wireless LAN is the **coverage**, and by

a wide margin. This includes maximum distance between nodes, resistance to interferences and ability to keep connectivity in a wide range of conditions.

The **propagation** of radio transmissions is influenced by many factors. Walls and floors tend to decrease and reflect the signal, and background noise make it more difficult to extract. The channel quality vary quite a lot over the time (*fading*).

Depending on the quality of reception, the error rate will change (forcing packet retransmissions), or the system may switch to a more robust (and slower) mode (fragmentation or modulation), so the actual throughput will vary from good to nothing.

Because of the way radio transmission are affected by the environment, it is quite difficult to predict the comportment of the system and to define a range. You will have some good, fair and bad area/period, the closer you are the more likely you are to be in a good one.

6.19 Mobility

One of the main advantage of Wireless LANs is that they offer **mobility**. It mean that even when moving around, you retain your connection to the network.

Of course, this mobility is limited by the range of the Wireless LAN. To extend the range, you must cover the area with **access points**, which very often include **roaming** : you switch transparently to the closer access point which provide you a connection to the rest of the world and nodes out of range.

Note that most cheap Access Point don't include roaming (to force you to buy the more expensive Access Point version), and that Access Points of different vendors usually don't fully interoperate (the Access Points don't talk to each other).

If you want to move across IP subnets, this is time to try **Mobile IP** :-)

6.20 Security and Privacy

Because they use radio waves, wireless LANs are usually perceived as a security problem. In fact, it's much more likely that you will get hacked from the Internet or that somebody will tap your phone line at the back of your house. It is also possible to read your screen and your Ethernet cable from across the street, with the correct equipment, so nothing is bullet proof. But it's important to assess the security threat and the possible remedies.

First of all, any **attack** on a Wireless LAN will be necessary **local** (physical proximity), because of the limited range of radio. Most "attacks" will be your neighbor accidentally connecting to your network due to improper configuration. Directional antennas will allow the attacker to be further away than regular nodes, but only with line of sight. But this limit greatly the scope of the risk.

Wireless LAN, because they use digital transmissions, can *not* be listen to with a regular radio scanner. The only practical way to attack a Wireless LAN is to use another Wireless card compatible with it. The attacker may mostly try to do two things, snoop on your communications (for example to read the e-mail you are sending) or access your resources (for example your access to the Internet).

For most users, the **network identifier (ESSID)** will be enough protection against casual users : other people can't accidentally join your network by mistake, unless they guess the correct network identifier or really try to attack you. Obviously, you should change the default network identifier, and most products allow you to hide it, which you should.

Some people are more concerned about those issues or may want to increase the security of their system. Most Wireless LANs offer **MAC level encryption**. There is different levels of encryption, and not all security protocols are equal. A simple encryption like WEP was designed to offer security equivalent to a having a shared Ethernet cable (i.e. not much). Complex encryption like WPA are much more secure, and impossible to break-in in practice, but they are usually more complex to set up and manage. Finally, for high security, various techniques are available at the higher layers of the protocol stack.

The basis of all MAC level encryption scheme is that each packet transmitted over the network is individually encrypted with a secret key, and the card refuses unencrypted data or data encrypted with a different key. This encryption is totally transparent to the higher layer, everything is handled at the MAC layer, as opposed to VPN solutions that create encrypted tunnels at higher layers.

Simple encryption schemes, such as WEP, are widely available and supported, and are usually **easy to set-up**, the same encryption key is set up in the access point and all nodes of the network, no central authority is needed. However, these schemes are considered weak because they use a single key distributed to all nodes, don't have key management facility and lacks per-packet authentication. On top of that, the encryption algorithm, RC4, is not considered very strong. Over the years, various attacks have been found on **WEP**, the MAC level encryption scheme of 802.11, which illustrate that WEP is not more secure than what it was designed for (and probably less). In any case, enabling WEP is better than nothing, it can be seen as putting a "do not enter" sign on your network.

Various encryption schemes have been released for 802.11 to address the shortcomings of WEP. The initial version of **802.1x** adds elaborated key management to WEP, and changes the WEP key frequently. Today, this is not considered very secure, because it is still based on WEP. **WPA** and **WPA2** improve on 802.1x by having better key management and using AES instead of RC4. Those are considered state of the art and are beginning to be well supported and interoperable, but they are more complex to deploy and require a central authority (usually implemented in an Access Point or a Wireless Controller).

For higher level of security, I would advise to use a **security solution independent of the wireless link** (like IPsec or a regular VPN, see below). This separation of wireless and security has other advantages, for example the security solution can be updated independently of the hardware as needed, and hardware from different vendors can be mixed.

I would still argue that most home users don't really need security beyond preventing people accidentally joining their network. Most web site that you will access over wireless and that handle critical information already use **SSL** to encrypt those transactions, which is proven to be very secure, and there exist

secure versions of POP or IMAP (still using SSL). Users that access remote Intranets most often already uses VPN software. So all critical transactions are often already protected, or easy to protect.

For those who need it, like if you are paranoid, your life depends on it, or if the wireless link is already behind a firewall, you need to set up a **encrypted tunnel** above the wireless link with a VPN protocol. Various VPN protocols can be used with Linux, such as **IPsec**, **PPTP** and **SSH** (IPsec considered is the ultimate security solution). You will need to setup a VPN gateway on the other side of the Access Points and the VPN software on every wireless device. That's usually complicated to setup, but that's the price for security.

6.21 Benchmarks

For all the reasons described above, I think it is quite tricky to **benchmark** Wireless LANs, and measuring coverage or throughput in isolation is not fair. Remember that **coverage** matters much more than raw performance in real life. This is why I don't give any performance numbers. Some computer magazine publish from time to time some extensive review of all those products and try to do some performance comparison more or less real life.

If you want to test the throughput of your device, you should use a tool called *Netperf*. You might want to submit your results in the database...

http://www.netperf.org/netperf/NetperfPage.html

6.22 Tuning

Quite often, vendors are delivering products in configuration that look good in benchmarks and doesn't perform as well in real life. This is where you need to **tune** a few parameters to get better usability and performance, in those cases where the hardware does not do that for you automatically.

Most 802.11g and 802.11n Access Points have two modes, **mixed mode** and exclusive mode. Mixed mode makes sure that the Access Point interoperate well with legacy and older devices, but is slower than the exclusive mode. Obviously, you want all your devices to get the best performance, not a selected few, so you need to set the mode appropriate for your device mix.

Many vendors have **proprietary extensions**, that enable them to claim very high speeds. These extensions, especially channel bonding (wider channels), may confuse other devices, so in some cases you may want to turn them off.

Some vendors ship products with **RTS/CTS disabled**. This is the best setting if you have only two nodes, but when you have a fair number of nodes active at the same time, RTS/CTS can increase the performance. And of course, if you have *hidden nodes*, you can not get away without it...

Some vendors also tend to set the **number of MAC retransmissions** a bit low for my taste. If you use TCP, this might improve your performance slightly under good conditions, because TCP do its own retransmissions. However, all applications not using TCP (ping, RTP, NFS...) might suffer from the packet losses.

On a related note, you can play with the **bit-rate** setting of the card. Most cards nowadays include a rate-adaptation scheme, which adapt the bit-rate to the

range : basically the card try transmitting at the highest rate and decrease the rate in case of packet losses. However, in configuration with large number of active nodes, packet losses also come from the contention process, so disabling this rate adaptation scheme (forcing the highest rate) can increase performance in some cases.

If you have **interferers** in the band, you might need to enable *fragmentation* (send smaller packet to fit between interferences) and to *raise the sensibility* (tell the card to ignore the noise). Of course, the best thing is to eliminate the interferer, if possible.

If you have different Access Points and have enabled roaming, you should also set carefully the **roaming threshold**, which is the point (in signal strength) at which the card search for a new Access Point. If you set it too low, the card will spend to much time with a non optimal AP (getting a poorer throughput), and if you set it too high the card will waste time searching for a new AP too often.

To finish, all this fine tuning and optimisation could/should be done in the card itself, not by the end user, the card itself has most of the information it needs to optimise those settings and the algorithms are not that complex (I describe some in my latest papers). This ensure that users automatically get the best performance in any condition. Most modern wireless hardware is now getting much better at doing this **automatic tuning**, and nowadays you need less and less to do any manual tuning (and actually tuning may be counter-productive in some cases).

7 Wireless Extensions for Linux

See the document about Wireless Extensions.

8 Wireless LAN technology overview

See the document about the technology overview.