Linux Wireless LAN Howto

Jean Tourrilhes

3 April 01

*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.

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. 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

This section describes the most common Wireless LAN products available on the market and their **compatibility** with Linux. I will make a short description of each product and will mainly focus on the drivers.

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 few people are using such hardware, so few of them are willing to develop, debug and maintain such a piece of code.

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.

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

Driver status : stable

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>
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
Mailing list :	http://lists.samba.org/pipermail/wireless/
Documentation :	man page, headers
Configuration :	Wireless Extensions & module parameters
Statistics :	Wireless Extensions
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...
Bugs :	May have some performance issues
License :	GPL
Vendor web page :	http://www.wavelan.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/

2.2.1 The device

Even if it uses the same name, the **Wavelan IEEE** product is completely different from the old Wavelan, 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 and mini PCI version of the hardware.

To confuse the issue, Lucent has recently renamed the Wavelan IEEE as **Orinoco** (Wavelan was better IMHO) and 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), 1stWave (**1stWave PC-Card**) and ARtem (**ARtem ComCard**). The Apple **Airport** is also derived from the Wavelan IEEE (see *section 2.5*).

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.

As with all IEEE 802.11 products, the Wavelan 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).

Nowadays, Lucent offers only the second generation of the Wavelan IEEE, 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").

2.2.2 The driver

Andreas Neuhaus was busy working to improve this driver. 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 2.3*), so this driver won't be maintained anymore...

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

2.3 Wavelan IEEE/Orinoco, PrismII and Symbol cards

Driver status :	fairly stable
Driver name :	orinoco_cs.o
Version :	v0.04
Where :	Linux kernel (2.4.3)
	http://www.ozlabs.org/people/dgibson/dldwd
Maintainers :	David Gibson <hermes@gibson.dropbear.id.au>
	Anton Blanchard <anton@samba.org>

Web page :	http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Orinoco.html
Mailing list :	http://lists.samba.org/pipermail/wireless/
Documentation :	man page, headers
Configuration :	Wireless Extensions only
Statistics :	Wireless Extensions
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...
Bugs :	Not fully functional on Prism2 and Symbol cards
License :	MPL and GPL
Vendor web pages :	[Too many to list here]

2.3.1 The devices

As explained in various sections, Lucent Wavelan-IEEE/Orinoco devices (see *section 2.2*), Intersil PrismII devices (see *section 2.10*) and Symbol High Rate devices (see *section 2.14*) 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), 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.

2.3.2 The driver

Anton Blanchard and *David Gibson* became official maintainers of the **wvlan_cs** driver (see *section 2.2*) 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 2.10*) and wrote a totally new driver combining the low level features of wlan-ng 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.

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.

The support of **PrismII cards** and clones is 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). However, the wlan-ng driver still has more features and is more tested...

The support of **Symbol cards** and OEM is limited. It is possible to make a card work in managed mode at 2 Mb/s without encryption, but not much else. More work is needed on this side, the code is there but doesn't work yet...

The driver does not support the USB and Mini-PCI version of the card. Only the Pcmcia cards are supported so far (including those in an ISA or PCI bridge). *Ben* is planning to add Airport support to this driver (see *section 2.5*), and *Anton* plan to backport this driver to the Pcmcia package.

2.4 Lucent Wavelan & Enterasys Roamabout (binary library driver)

Driver status :	stable
Driver name :	wavelan2_cs.o and roamabout_cs.o
Version :	v6.06
Where :	ftp://ftp.wavelan.com/pub/software/IEEE/PC_CARD/LINUX/
	http://www.enterasys.com/software/RoamAbout/
	ftp://projects.sourceforge.net/pub/pcmcia-cs/contrib/
Contact :	Lucent support <usasupport@wavelan.com>
	EnteraSys support <support@enterasys.com>
Maintainers :	Richard van Leeuwen <rleeuwen@lucent.com>
	Dean W. Gehnert <deang@tpi.com>
Documentation :	Extensive readme
Configuration :	Module parameters, Wireless Extensions
Statistics :	Wireless Extensions
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 :	WEP encryption, power management and microwave oven robustness
Non implemented :	Do not support all firmware releases
Bugs :	?
License :	Binary only for the core + OpenSource Linux wrapper
Vendor web page :	http://www.wavelan.com/
	http://www.enterasys.com/wireless/

2.4.1 The device

This is the same device as the previous entry (*section 2.2*).

2.4.2 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 lack of Wireless Extensions support.

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 latest version adds support for the IBSS Ad-Hoc mode (see discussion above) Note that Enterasys/Cabletron is also distributing a slightly modified version of this driver.

But, as with any binary driver, you should check if your architecture and your version of the kernel and Pcmcia package are supported.

2.5 Apple Airport

Driver status :	stable
Driver name :	airport.o
Version :	0.9.3
Where :	http://ppclinux.apple.com/~benh/
Maintainer :	Benjamin Herrenschmidt <benh@kernel.crashing.org>
Documentation :	web page, headers
Configuration :	Wireless Extensions & module parameters
Statistics :	Wireless Extensions

Multi-devices :	No
Interoperability :	802.11-DS and 802.11-b, interoperate with Mac-OS ;-)
Other features :	MTU selection, multicast, promiscuous mode, power management, WEP hardware encryption
Non implemented :	Some optimisations...
Bugs :	SMP not fully tested
License :	GPL
Vendor web page :	http://www.apple.com/airport/

2.5.1 The device

The Apple **AirPort** is in fact the Lucent **Wavelan IEEE** repackaged, so has the same characteristic as the Wavelan (see *section 2.2*). 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 interoprable with other 802.11-b hardware.

2.5.2 The driver

Benjamin Herrenschmidt has ported the driver of *Andreas Neuhaus* (see *section 2.2*) 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 ;-)

2.6 Netwave AirSurfer & Xircom Netwave

Driver status :	fairly stable
Driver name :	netwave_cs.o
Version :	v 0.4.1
Where :	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
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/

2.6.1 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).

2.6.2 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).

2.7 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
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/

2.7.1 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.

2.7.2 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.

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

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/
Maintainers :	Mark S. Mathews <mark@linux-wlan.com>

	Peter Teuben <teuben@astro.umd.edu>
	Jason Hill <jhill@cs.berkeley.edu>
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
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

2.8.1 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)...

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 2.10*).

2.8.2 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).

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.

2.9 Z-Com LANEscape, ELSA MC2

Driver status :	?
Driver name :	wl24_cs.o
Version :	1.3 (stable), 2.03 (unstable) and 1.20 (Elsa full source)
Where :	http://www.boerde.de/~matthias/airnet/zcom/
	http://www.elsa.de
Maintainer :	?
Documentation :	README file
Configuration :	Specific tool
Statistics :	no

Multi-devices :	unknown
Interoperability :	802.11-FH (need updated firmware), interoperate with Windows
Other features :	-
Non implemented :	-
Bugs :	Must have the correct firmware revision.
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

2.9.1 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.

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 2.2*).

2.9.2 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, and doesn't work with the 802.11 compliant firmware.

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

Lately, *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...

2.10 Intersil PrismII based cards (Compaq, Samsung, D-Link, LinkSys and many others)

Driver status :	beta
Driver name :	prism2_cs.o

Version :	0.1.7 and 0.1.8-pre5
Where :	http://www.linux-wlan.com/linux-wlan
Maintainer :	Mark S. Mathews <mark@linux-wlan.com>
Mailing list :	http://www.linux-wlan.com/linux-wlan/
	http://www.lifix.fi/extarchive/lwlan/
Documentation :	Readme
Configuration :	Module parameters & configuration tool
Statistics :	Statistic tool
Multi-devices :	yes
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Quite exhaustive 802.11 support, PPC 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/DigitalHome/Wireless/
	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/Product.htm
	http://www.smc.com/
	http://www.ambicom.com/
	http://www.intersil.com/prism/

2.10.1 The device

The PrismII chipset is the successor of the PrismI chipset, described in the previous section (see *section 2.8*), 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** (rumored to be selling a rebadged Samsung card), with a Pcmcia card, a PCI card and an Access Point. Aironet and Symbol also uses the PrismII chipset, but with their own MAC controllers (see *section 2.17* and *section 2.14*). Other Prism vendors like **ZoomAir**, **Nokia** and **GemTek** are slowly releasing their own version of the PrismII card. Some big networking vendors like **D-Link**, **LinkSys** and **SMC** were

also quick to jump on this new opportunity for them, as well as many smaller vendors like **AddTron** and **Ambicom**...

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).

A few words about **Ad-Hoc modes** : people have reported to me that the default Ad-Hoc mode of those cards interoperate with the Wavelan Ad-Hoc demo mode and is therefore not 802.11-b compliant. It seems that some of the newer firmwares and drivers for some of these cards have the option to select the IBSS Ad-Hoc mode which is 802.11-b compliant.

2.10.2 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.

Mark told me that he was planning to add Wireless Extension to this already very complete driver, so stay tuned !

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

2.11 Samsung MagicLAN (binary library driver)

Driver status :	beta
Driver name :	swld11_cs.o
Version :	1.20
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
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/

2.11.1 The device

The Samsung **MagicLAN** is one of the various products based on the Intersil PrismII chipset (see *section 2.10* 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...

2.11.2 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 2.10*) 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 exhaustive support for Wireless Extensions. Quite a few people are reporting that this driver works better for them...

2.12 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
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/

2.12.1 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.

2.12.2 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...

The current driver doesn't support the RangeLan802 line, but you may contact *Dave* if you would like to see a driver for RangeLan802.

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...

2.13 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 <lkeyser@digitalsquare.com>
Mailing list :	http://sourceforge.net/mail/?group_id=11099
Documentation :	Readme file
Configuration :	module parameters, partial support of Wireless Extensions
Statistics :	None
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

2.13.1 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.

2.13.2 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...).

2.14 Symbol Spectrum24 High Rate, 3Com AirConnect and Intel PRO/ Wireless

Driver status :	Beta (Pcmcia only)
Driver name :	spectrum24t_cs.o
Version :	1.01
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 <lkeyser@digitalsquare.com>

Mailing list :	http://sourceforge.net/mail/?group_id=11099
Documentation :	Readme file
Configuration :	module parameters
Statistics :	None
Multi-devices :	-
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Multicast, power management and WEP encryption
Non implemented :	-
Bugs :	-
License :	GPL or BSD
Vendor web page :	http://www.symbol.com/products/wireless/wireless.html
	http://www.intel.com/network/products/wireless.htm
	http://www.3Com.com/mobile/wireless/solution.html

2.14.1 The device

Despite beeing a long time proponent of Frequency Hopping, Symbol couldn't ignore the success of 802.11-b. After a strategic agrement 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 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. But, unfortunately for us, Symbol doesn't sell yet any of their famous "all-in-one" products with 802.11-b.

The Symbol product is composed of the Intersil PrismII chipset (see *section 2.10*) 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.

2.14.2 The driver

The driver was initially written by TriplePoint, and *Lee* has taken over the maintainance. 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 writen, 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 latest version, 1.01, fixes some bugs related to higher bit rate (11 Mb/s) and encryption.

2.15 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
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/

2.15.1 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 5*), but they realised pretty quickly the BlueTooth would not fulfill the need of the Wireless LAN market. Ericsson is also pushing hard **HiperLAN II** (see *section 5*), 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 2.25*). 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 2.14*). As such, this product has all the usual 802.11-b features...

2.15.2 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 2.25* - 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...

2.16 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 :	?
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

2.16.1 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 below). 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.

2.16.2 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.

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

Driver status :	stable
Driver name :	ISA, PCI : airo.o
	Pcmcia : airo_cs.o
Version :	1.8 - 2001/03/27
Where :	Pcmcia package (3.1.25)
Maintainers :	Benjamin Reed <breed@almaden.ibm.com>
	Javier Achirica <achirica@ttd.net>
Web pages :	http://sourceforge.net/projects/airo-linux/
	http://www.cse.ucsc.edu/~breed/airo.html
Mailing list :	http://csl.cse.ucsc.edu/pipermail/aironet/
Documentation :	README file
Configuration :	/proc interface and Wireless Extensions
Statistics :	/proc interface and Wireless Extensions
Multi-devices :	N/A
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Support ad-hoc and managed mode, and WEP (encryption).
Non implemented :	-
Bugs :	Doesn't work with SMP, no promiscuous support.
License :	MPL
Vendor web page :	http://www.aironet.com/products/

2.17.1 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 aquire Telxon. After a short independant life, Aironet was aquired by Cisco.

The previous section was dealing with Aironet old pre-802.11 products (see *section 2.15*), 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 2.8*), 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 2.10*).

After the aquisition 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).

2.17.2 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. SMP is not yet supported. Promiscuous doesn't work because the Aironet firmware doesn't support this feature.

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 the most complete Wireless Extension support of all.

Then, *Javier Achirica* added to the driver the Cisco proprietary API, which allow communication with Cisco utilities (see *section 2.19*) 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.

2.18 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 and above
Maintainer :	Elmer Joandi <Elmer.Joandi@ut.ee>
Documentation :	Configure.help file
Configuration :	/proc interface
Statistics :	/proc interface
Multi-devices :	-
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Support ad-hoc and managed mode, and WEP (encryption).
Non implemented :	Pcmcia interface
Bugs :	Buggy SMP support.
License :	GPL
Vendor web page :	http://www.aironet.com/products/

2.18.1 The device

This is the same device as the previous entry (*section 2.17*).

2.18.2 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.

2.19 Cisco/Aironet 802.11 (Cisco driver)

Driver status :	stable (rock solid)
Driver name :	ISA, PCI : airo.o
	Pcmcia : airo_cs.o

Version :	1.000.2
Where :	http://www.cisco.com/public/sw-center/sw-wireless.shtml
Maintainer :	Cisco
Documentation :	Text files
Configuration :	Cisco utilities
Statistics :	Cisco utilities
Multi-devices :	-
Interoperability :	802.11-DS and 802.11-b, interoperate with Windows
Other features :	Support ad-hoc and managed mode, and WEP (encryption).
Non implemented :	-
Bugs :	Doesn't work with kernel 2.4.X
License :	Cisco open source license
Vendor web page :	http://www.aironet.com/products/

2.19.1 The device

This is the same device as the two previous entry (*section 2.17*).

2.19.2 The driver

Recently, Cisco decided to get more involved with supporting their Wireless LAN cards under Linux. Rather than developping an entirely new driver, they decided to base their work on *Ben*'s driver (*section 2.17*), 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, since Cisco grabed a snapshot of *Ben*'s driver (*section 2.17*), it improved as well. 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 Wireless Extensions (which is not in this driver). However, I beleive 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.

2.20 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 <corey@world.std.com>
Web page :	http://world.std.com/~corey/raylink.html
Documentation :	README file + headers
Configuration :	modules parameters and Wireless Extension (read only)
Statistics :	Wireless Extensions
Multi-devices :	yes (except for module parameters setting)
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

2.20.1 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.

2.20.2 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.

2.21 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
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/

2.21.1 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 2.8*), 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 aquired No Wires Needed, Intersil can offer 2 different 802.11 MAC controller !

2.21.2 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.

2.22 No Wires Needed 1148

Driver status :	stable
Driver name :	poldhu_cs.o
Version :	0.1.1 (for kernel 2.4.0)
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
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/

2.22.1 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 2.21*) : 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 2.10*).

2.22.2 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 2.21*).

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).

2.23 Nokia C110/C111

Driver status :	???
Driver name :	nokia_c110.o
Version :	1.00
Where :	http://forum.nokia.com/main/1,13682,5_63_10,00.html
Maintainer :	Nokia
Documentation :	Readme
Configuration :	Specific Pcmcia scripts
Statistics :	/proc file
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/

2.23.1 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 2.8*), and Nokia continued selling those cards, renamed Nokia **C020** and **C021**. While beeing busy working on BlueTooth and HiperLanII, Nokia didn't forgot 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 2.10*). 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).

2.23.2 The driver

Nokia quietly made this driver available in the registered only part of their web site and don't seem to mention it anywhere, fortunately I have my spies to inform me ;-) This driver is only for the C110/C111, on the other hand the C020 is supposed to work with the linux-wlan package (see *section 2.8*).

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.

2.24 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
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/

2.24.1 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.

2.24.2 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...

2.25 BreezeCom BreezeNet PRO Pcmcia

Driver status :	stable
Driver name :	brzcom_cs.o
Version :	1.0
Where :	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
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/

2.25.1 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 2.26*). The first two Pcmcia cards were limited in term of bit-rate (only 1 Mb/s), and have lower transmit power.

2.25.2 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 **newave_cs** driver (in all good Pcmcia packages). For the first pro.11 Pcmcia card, you may use the **ray_cs** driver (in recent Pcmcia packages - and therefore get 802.11 compliance). For the DS.11 card (802.11-b compliant), you may use the **swallow_cs** driver.

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 maintainance 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 ;-)

2.26 BreezeCom BreezeNet (not Pcmcia)

Driver status :	not needed (for Pcmcia, see above)
Driver name :	-
Version :	-
Where :	-
Maintainer :	none
Documentation :	none
Configuration :	none
Statistics :	none
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/

2.26.1 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.

2.26.2 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.

2.27 Not supported

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...

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

WebGear Aviator 900 MHz : connect to the parallel port and offer cable replacement solution. No functional Linux driver yet.

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.

2.28 A note on driver licenses

Donald Becker's web page alerted me on the **license and copyright** issues for networking drivers (see http://cesdis.gsfc.nasa.gov/linux/misc/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 tough, 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.

2.29 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).

To have a good picture of all the devices available and their characteristics, you should redirect your favourite browser to :

http://hydra.carleton.ca/info/wlan.html

2.30 Other Wireless technologies

2.30.1 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.

2.30.2 Radio Amateur and AX25 (HAM)

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

2.30.3 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://www.cs.uit.no/linux-irda/

http://mobilix.org/howtos.html

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

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

2.30.4 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 5* 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**.

More information on BlueTooth for Linux is available at :

http://delbert.matlock.com/linux-bluetooth.htm

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

http://developer.axis.com/software/bluetooth/

http://freshmeat.net/projects/bluedrekar/

2.30.5 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 generalitie.

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 2.30.3*) to connect your mobile phone to your laptop. That's done with PPP over RfComm or gnokii.

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

3 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*.

3.1 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.

3.2 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 *Pcmcia* form factor, some vendors offer as well *ISA* and *PCI* versions of their cards, and most other vendors offer some ISA-to-Pcmcia bridge (separately or included in the box) to allow to plug a Pcmcia card in a desktop. I've seen some cards in *PC104* form factor (for embedded applications), and I expect sooner or later to see cards in *Compact Flash* form factor or with a *USB* interface.

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...

3.3 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 are **802.11-FH** and **802.11-DS**. 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. For **802.11-b**, it is just a simple extension of 802.11-DS (adding 5.5 and 11 Mb/s speeds), and products seem to already interoperate at higher speed.

Other standards (such as HiperLan, SWAP and 802.11@5GHz) 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.

3.4 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 have to **compile** the driver source code (some Linux distribution may offer precompiled modules). 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 the package. Otherwise, see the installation instructions coming with the driver.

Once you've got the driver compiled, you must tell your system about it. For pcmcia drivers, the package has its own **configuration scripts** (see pcmcia documentation). For other drivers, you will have to edit the system configuration scripts. You will likely to have to add a *ifconfig* line in some script (/etc/init.d/network for the *Debian*). There is many other networks configuration files in /etc. For modules, you need as well to change some stuff in the module configuration files. See the list of readings below for more information.

3.5 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)

3.6 Driver parameters

A lot of users are confused when it comes to set **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).

As it is explained nowhere correctly, I disgress a bit and give you a few hints...

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**, 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 done in the pcmcia scripts.

3.7 More parameters to configure

The most obvious difference with Ethernet is that there is **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* or *domain*, *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 95 *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...

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.

3.8 Where to get information about your Wireless LAN

- The official documentation that come with your product.
- Manufacturer web page and support.
- Linux driver source code, documentation (headers, man pages), maintainer.

3.9 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 know who is there, anybody and anything can use the radio band.

To try to separate everyone out there, most products define some *network identifier* (Network ID, Network Name or Domain, in 802.11 it is also called 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 don't interfere with 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**.

3.10 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.

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 on a wireless LAN (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 explicitely 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 bridge with most products out there, and forget about turning it into an Access Point.

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...

3.11 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 the access point, and they are 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.

3.12 Performance (speed)

Most people want to know how fast it goes, and complain that they can't get the speed written on the box, and that the number seems low, even if in real use they are far away from saturating the network. Even when converting the byte per seconds to bit per seconds, there is no denying that the **TCP throughput** is lower than the **signalling rate**. This is because the Wireless LANs are slower to start with and on top of that uses less efficiently the available bandwidth.

Most Wireless LANs have a **signalling rate** around 1 or 2 Mb/s. The signalling rate is the speed the bits are send over the air (Ethernet is 10 Mb/s), but doesn't account of all the overhead of the protocols.

The Wireless LAN protocols have usually a higher overhead than their wired counterpart (like Ethernet) because of some technological limitations and to improve the reliability and the coverage of the Wireless LAN (optimisation trade-offs). On the other hand, Wireless LANs protocols are also usually less sensitive to high load (the throughput doesn't drop when you overload the network - which could happen more often).

Some protocols also adapt the signalling rate depending on the quality of the link (for example a 2 level modulation 2FSK/4FSK). When the link is clear and reception is strong, it will use the fastest rate, but when there start to be noise or the device is further away, it goes down to the more robust rate. The throughput that you will get will depend on that as well (for example the high speed might be only usable in line of sight).

3.13 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).

3.14 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.

3.15 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** :-)

3.16 Security and Privacy

Because they use radio waves, wireless LANs are usually perceived as a security problem (in fact, it's more likely that you will get hacked from the Internet, and it's also possible to read your screen and your Ethernet cable from across the street, with the correct equipment).

For most users, the **network identifier** will be enough protection against casual users : other people can't accidentally join your network by mistake, unless they guess the correct network identifier. Trying to discover the network identifier is usually not as easy as it seems, unless you use social engineering...

Some people are more concerned about those issues or may want to increase the security of their system. Some Wireless LANs offer **MAC level encryption** (sometimes only as an option), which is designed for those concerned users and target security equivalent to a having a shared Ethernet cable, and a few systems offer stronger encryption (but still considered weak by some experts because of the distributed single key and the lack of per-packet authentication).

With these schemes, each packet transmitted over the network is individually encrypted, and the card refuse unencrypted data. This encryption is totally

transparent to the higher layer and the user just need to set the same encryption key in the access point and all nodes of the network.

If you are totally paranoid about security or your life depend on the data beeing transmitted, use **IPsec** or **SSL**, or don't use a computer at all.

3.17 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

3.18 Tuning

More and more, 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 performance.

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, 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 paper). Let's hope the vendors will work on that...

4 Wireless Extensions for Linux

See the document about Wireless Extensions.

5 Wireless LAN technology overview

See the document about the technology overview.