# MadWifi UserDocs

[MadWifi UserDocs 1](#_Toc277951506)

[MadWifi 'First Time User' HOWTO 5](#_Toc277951507)

[Removing old modules 5](#_Toc277951508)

[Obtain and Install MadWifi Source 6](#_Toc277951509)

[Loading the MadWifi Module 7](#_Toc277951510)

[Creating an Interface 8](#_Toc277951511)

[Scanning for Access Points 9](#_Toc277951512)

[Connecting to an open AP 11](#_Toc277951513)

[Connecting to an AP with WEP 16](#_Toc277951514)

[Connecting to an AP with WPA 17](#_Toc277951515)

[Removing MadWifi 18](#_Toc277951516)

[More Documentation 18](#_Toc277951517)

[Using Ad-Hoc Mode 19](#_Toc277951518)

[New Code 19](#_Toc277951519)

[Old Code 19](#_Toc277951520)

[Common Instructions 19](#_Toc277951521)

[Setting a Client to Use WEP. 20](#_Toc277951522)

[Open vs Shared 21](#_Toc277951523)

[How do I get my card to use WEP 21](#_Toc277951524)

[Notes for TEW-443PI - Atheros chip revision A1.1R 22](#_Toc277951525)

[Setting up a Client Using WPA-PSK 22](#_Toc277951526)

[Prerequisites 23](#_Toc277951527)

[Instructions 23](#_Toc277951528)

[WPA on madwifi: AP & Station 25](#_Toc277951529)

[The Station (client) Side 26](#_Toc277951530)

[Now do the AP side 33](#_Toc277951531)

[Things That Did Not Work 39](#_Toc277951532)

[How do I get the card to use monitor mode 39](#_Toc277951533)

[MAC Address Filtering in AP Mode 39](#_Toc277951534)

[Introduction 39](#_Toc277951535)

[Steps 40](#_Toc277951536)

[Other Possibilities 40](#_Toc277951537)

[Security Issues 40](#_Toc277951538)

[How can I use my card as an access point? 41](#_Toc277951539)

[Beacon Interval 46](#_Toc277951540)

[DTIM period 47](#_Toc277951541)

[Using HostAP 47](#_Toc277951542)

[Introduction 47](#_Toc277951543)

[General note about bridging 48](#_Toc277951544)

[WPA-PSK setup 48](#_Toc277951545)

[WPA-EAP setup 63](#_Toc277951546)

[How do I see who's connected to my AP? 65](#_Toc277951547)

[How do I get my card to use turbo mode? 65](#_Toc277951548)

[Countrycode 68](#_Toc277951549)

[Which settings are determined by the regdomain/countrycode? 68](#_Toc277951550)

[What are regdomains and countrycodes? 68](#_Toc277951551)

[How can I change the regdomain or countrycode that Madwifi uses? 69](#_Toc277951552)

[Why doesn't MadWifi determine the regdomain and countrycode from beacon packets, like the NDIS wrapped Atheros windows driver does? 70](#_Toc277951553)

[Can an incorrect countrycode or regdomain lead to poor bit rates? 70](#_Toc277951554)

[Can I use two separate countrycodes on two cards in the same machine? 71](#_Toc277951555)

[This appears to be a list of Country Codes. 71](#_Toc277951556)

[List of allowed txpower and frequencies in different countries. 76](#_Toc277951557)

[Can I turn on my wireless card LEDs automatically? 114](#_Toc277951558)

[How can I change the MAC address of my card? 116](#_Toc277951559)

[How do I get the Cisco VPN client to work with madwifi? 117](#_Toc277951560)

[Bit-rate Selection Algorithms 118](#_Toc277951561)

[onoe 119](#_Toc277951562)

[amrr 119](#_Toc277951563)

[sample 119](#_Toc277951564)

[Bit-rate Selection Algorithm comparison 120](#_Toc277951565)

[Configuration 120](#_Toc277951566)

[Attachments 120](#_Toc277951567)

[Note about 802.11a channels 121](#_Toc277951568)

[Long Distance links with MadWifi 121](#_Toc277951569)

[Power and antenna gain 122](#_Toc277951570)

[ACK timeout and Slot time 122](#_Toc277951571)

[Performance expectations and measurements 123](#_Toc277951572)

[The athctrl tool 124](#_Toc277951573)

[The law 125](#_Toc277951574)

[Antenna Diversity 126](#_Toc277951575)

[Receiver diversity 126](#_Toc277951576)

[Transmitter Diversity 126](#_Toc277951577)

[Controlling diversity 127](#_Toc277951578)

[What doesn't work 129](#_Toc277951579)

[Configuring kismet with madwifi 129](#_Toc277951580)

[And with kismet-newcore?? 130](#_Toc277951581)

[Snort 130](#_Toc277951582)

[Disabling Background Scanning in Client Mode 131](#_Toc277951583)

[iwpriv extensions 132](#_Toc277951584)

[Locking to a Specific Mode 132](#_Toc277951585)

[Changing Authentication Mode. 132](#_Toc277951586)

[Manipulating the MAC white/black list 132](#_Toc277951587)

[Others 132](#_Toc277951588)

[ath\_info 133](#_Toc277951589)

[attachment 144](#_Toc277951590)

[Creating a Monitor Mode Interface 144](#_Toc277951591)

[FreeRADIUS/WinXP Authentication Setup 145](#_Toc277951592)

# MadWifi 'First Time User' HOWTO

Welcome to the MadWifi 'first time user' howto. This document is intended to be a complete set of instructions on how to get, install and use the latest MadWifi driver. No previous experience of wireless networking under Linux is assumed.

This howto describes the manual build process for MadWifi drivers. However your Linux distribution may already distribute pre-built (but old) MadWifi drivers. Distributions may also have their own way of building kernel modules for integration in the package management system. Have a look at [UserDocs/Distro](http://madwifi-project.org/wiki/UserDocs/Distro).

**Note:** This guide only shows you the steps to take for managed mode operation (aka. station). Refer to the [UserDocs](http://madwifi-project.org/wiki/UserDocs) for more information about other modes.

## Removing old modules

For this step you must be logged on as root.

First, set all your MadWifi devices down:

ifconfig ath0 down

ifconfig wifi0 down

#Repeat these 2 ifconfig lines for every MadWifi device you have (ath1, etc)

Assuming that you're inside the [MadWifi](http://madwifi-project.org/wiki/MadWifi) directory, execute the following scripts to remove the current modules from your system and its memory:

cd scripts

./madwifi-unload

./find-madwifi-modules.sh $(uname -r)

cd ..

You should then be asked if you are sure that you want to remove the old modules.

## Obtain and Install MadWifi Source

*For Debian (and Ubuntu?)*

# aptitude install module-assistant

# module-assistant auto-install madwifi-source

Go to "Loading the MadWifi Module" below

*Other Linux Distributions*

Requirements

Make sure you have Linux headers installed: sudo apt-get install linux-headers-$(uname -r)

(If you don't you will get an error: /lib/modules/2.6.24-19-server/build is missing, please set KERNELPATH. )

Please check [Requirements](http://madwifi-project.org/wiki/Requirements) before proceeding. This includes having an Atheros chipset physically installed.

Getting MadWifi Sources

The MADWiFi sources have been removed from sourceforge.net. This [UserDocs/GettingMadwifi](http://madwifi-project.org/wiki/UserDocs/GettingMadwifi) page refers to the best methods to obtain source.

Building MadWifi

Now that you have the MadWifi code, it's time to compile it into the actual driver. Thankfully, this is easy.

Assuming that you've met all of the requirements above, and you're inside the [MadWifi](http://madwifi-project.org/wiki/MadWifi) directory, you can just type:

make

Which will start the build process. Watch for any questions you might be prompted to answer - when it finishes, quickly scan through for any errors. If everything went according to plan, you can proceed to the next step. Make sure you have all the [Requirements](http://madwifi-project.org/wiki/Requirements) or the build process may fail.

Installing MadWifi

This step will take the built [MadWifi](http://madwifi-project.org/wiki/MadWifi), and install it on your system. Once again, make does all of the work for you.

This step needs to be done as *root*, so either type su and enter *root's* password, or if you have it set up (e.g. Ubuntu), prefix the following command with sudo.

To install the driver, type:

make install

This will copy all of the modules, tools and man pages to the correct directories on your system. You've now completed the basic install.

## Loading the MadWifi Module

This step will load the MadWifi driver module into your running system. This essentially lets all other software know how to talk to your MadWifi hardware.

This step needs to be done as *root*, so either type su and enter *root's* password, or if you have it set up (e.g. Ubuntu), prefix the following command with sudo.

To load the driver module, type:

modprobe ath\_pci

If after running this command ifconfig doesn't show the additional wireless interface you might need to reboot. If you have any problems with building the MadWifi driver, please refer to [UserDocs/BuildProblems](http://madwifi-project.org/wiki/UserDocs/BuildProblems).

## Creating an Interface

MadWiFi supports *virtual access points*, which means you can create more than one wireless device per wireless card. By default, a *sta* mode VAP is created, which is [MadWifi](http://madwifi-project.org/wiki/MadWifi) talk for a 'managed mode wireless interface'.

**If your svn snapshot is more recent than the 23rd January 2006, (**[**r1407**](http://madwifi-project.org/changeset/1407)**) than you can skip the following step:**

If not, then follow these instructions to make a normal *station* mode interface. Type (as root):

wlanconfig ath0 create wlandev wifi0 wlanmode sta

If [wlanconfig](http://madwifi-project.org/wiki/UserDocs/wlanconfig) doesn't work, you retry it after executing '[wlanconfig](http://madwifi-project.org/wiki/UserDocs/wlanconfig) ath0 destroy'.

Now, if you type iwconfig, you should see a list like the following:

eth0 no wireless extensions.

lo no wireless extensions.

wifi0 no wireless extensions.

ath0 IEEE 802.11g ESSID:""

Mode:Managed Frequency:2.457 GHz Access Point: 00:00:00:00:00:00

Bit Rate:0 kb/s Tx-Power:20 dBm Sensitivity=0/3

Retry:off RTS thr:off Fragment thr:off

Power Management:off

Link Quality=0/94 Signal level=-95 dBm Noise level=-95 dBm

Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0

Tx excessive retries:0 Invalid misc:0 Missed beacon:0

Then we need to bring up the wireless interface. This is done by typing (as root):

ifconfig ath0 up

There is more information on the creating of interfaces in [UserDocs](http://madwifi-project.org/wiki/UserDocs).

## Scanning for Access Points

If you know that there are some APs around, having a quick scan can be an excellent way of getting some instant gratification, and knowledge that everything's working OK.

The first step is to insert the scanning module. Type (as root):

modprobe wlan\_scan\_sta

Next, you can do the actual scan, which can be done in two ways.

The first way is specific to [MadWifi](http://madwifi-project.org/wiki/MadWifi). and gives you a nice, succinct results table.

This is done by issuing the command (again, as root):

wlanconfig ath0 list scan

This should give you a list that looks something like this:

SSID BSSID CHAN RATE S:N INT CAPS

eddie 00:06:25:e8:3a:05 6 54M 36:0 100 EPs

The second way is the more usual way of scanning (and works with other wireless cards), it also gives you somewhat more information, and is therefore a bit less easy to read. The command is:

iwlist ath0 scanning

Which gives an output which looks like this:

ath0 Scan completed :

Cell 01 - Address: 00:06:25:E8:3A:05

ESSID:"eddie"

Mode:Master

Frequency:2.437 GHz (Channel 6)

Quality=37/94 Signal level=-58 dBm Noise level=-95 dBm

Encryption key:on

Bit Rate:1 Mb/s

Bit Rate:2 Mb/s

Bit Rate:5.5 Mb/s

Bit Rate:11 Mb/s

Bit Rate:18 Mb/s

Bit Rate:24 Mb/s

Bit Rate:36 Mb/s

Bit Rate:54 Mb/s

Bit Rate:6 Mb/s

Bit Rate:9 Mb/s

Bit Rate:12 Mb/s

Bit Rate:48 Mb/s

Extra:bcn\_int=100

Especially useful is the line reading Encryption key:on, which indicates that this AP is running some kind of WEP.

If you get a message such as:

ath0 Failed to read scan data : Resource temporarily unavailable

instead of actual scan results, and you are in an environment that requires a shared encryption key, try running:

iwconfig ath0 key <yourkey>

iwpriv ath0 authmode 2

This will tell the card that it is operating in a restricted, shared-key environment, and thus it needs to use the key you supply with iwconfig. To use an open system key (which is often considered more secure) use [iwpriv](http://madwifi-project.org/wiki/UserDocs/iwpriv) authmode 1:

iwconfig ath0 key <yourkey>

iwpriv ath0 authmode 1

Once this is done, re-run the scan, and you may see proper results.

## Connecting to an open AP

If the scan you did above says "Encryption key:off", then you may not actually need to do anything to make the driver associate with the AP, since the driver will automatically connect to the one with the strongest signal. It's still a good idea to know how to tell it which AP/Network to connect to however.

Typing (as root):

iwconfig ath0 essid "eddie"

Will connect you to the AP with the ESSID (network name) eddie.

You can also specify which AP you want to connect to by using its MAC address (in topmost scan output, this is the field marked BSSID, and in the bottom one, it's the field called Address).

To specify the AP using its MAC/BSSID, type:

iwconfig ath0 ap 00:06:25:E8:3A:05

If you then decide you want to let the driver decide automatically which AP to associate with, you can type:

iwconfig ath0 ap any

Or:

iwconfig ath0 ap auto

Just being connected to an AP is like having an ethernet cable plugged into your machine - you're now 'on the network', However without getting an IP address you can't really do anything. For this reason the next step is to get an IP address, and again, this is fairly easy. The tools you use to get an IP with a wireless interface, are exactly the same as they are for a wired one.

First of all, you need to know if the network you're connecting to has a DHCP server (this is a server which gives you a network address automatically, and tells you how to do things like access the internet and perform DNS lookups).

There are several methods of finding out whether or not the network has DHCP:

* If you've used the wireless card in Windows (on this network), and you let it 'Obtain an IP address automatically', then the chances are that the network *does*.
* If you've used the wireless card in Windows (on this network), and you had to put in an IP address yourself, then it *doesn't*, and if you can, you should find the network details you used previously.
* If you've connected to some kind of combined wireless access point/router, then it almost certainly *will*.
* If you're in an internet Cafe, then it probably *does*.
* If there's someone around who knows about the network you're connected to, then ask them if it has DHCP.
* If you can't find out whether or not the network has DHCP, then try the method below anyway, it might work, and you'll have answered the question yourself.

*Connecting with DHCP* [*¶*](http://madwifi-project.org/wiki/UserDocs/FirstTimeHowTo#ConnectingwithDHCP)

There are various different DHCP clients available, and which one(s) are available is largely dependent on which GNU/Linux distribution you're using. The most common one is dhclient, which is what will be used here.

To get an IP address from a DHCP enabled network, type (as root):

dhclient ath0

You should then see something like:

Internet Software Consortium DHCP Client 2.0pl5

Copyright 1995, 1996, 1997, 1998, 1999 The Internet Software Consortium.

All rights reserved.

Please contribute if you find this software useful.

For info, please visit http://www.isc.org/dhcp-contrib.html

eth1: unknown hardware address type 24

eth1: unknown hardware address type 24

Listening on LPF/ath0/00:02:6f:20:14:81

Sending on LPF/ath0/00:02:6f:20:14:81

Sending on Socket/fallback/fallback-net

DHCPDISCOVER on ath0 to 255.255.255.255 port 67 interval 7

DHCPOFFER from 192.168.0.254

DHCPREQUEST on ath0 to 255.255.255.255 port 67

DHCPACK from 192.168.0.254

bound to 192.168.0.152 -- renewal in 7200 seconds.

The bottom line shows that the DHCP server allocated us the address 192.168.0.152. It will also have told the machine where to find a DNS server and gateway, if they're available.

Alternately, your system may have dhcpcd installed instead of dhclient:

dhcpcd ath0

If the network you've connected to is connected to the internet, then you should be able to type:

ping bbc.co.uk

To see if everything is working. If it is, you should get an output like this:

PING bbc.co.uk (212.58.224.131) 56(84) bytes of data.

64 bytes from rdirwww-vip.thdo.bbc.co.uk (212.58.224.131): icmp\_seq=1 ttl=119 time=15.4 ms

64 bytes from rdirwww-vip.thdo.bbc.co.uk (212.58.224.131): icmp\_seq=2 ttl=119 time=14.3 ms

64 bytes from rdirwww-vip.thdo.bbc.co.uk (212.58.224.131): icmp\_seq=3 ttl=119 time=15.1 ms

*Connecting without DHCP* [*¶*](http://madwifi-project.org/wiki/UserDocs/FirstTimeHowTo#ConnectingwithoutDHCP)

Connecting to a network without DHCP makes life a bit more complicated - you'll need to know a few details about the network, including:

* An IP address which you're allowed to use. This might be something like 192.168.0.10, but could be anything.
* The netmask of the network. This is usually 255.255.255.0, or 255.255.0.0, but again, it could be just about anything. The netmask describes the address range of the local network.
* The address of a local nameserver (if there is one). A nameserver is used to get the IP address from a hostname like **madwifi-project.org**, and vice versa.
* The address of the network's gateway (if there is one). A gateway allows you to connect to networks outside of the local subnet - usually the internet.
* The local domain name of the network (optional). This is the name which is prefixed to all hostnames in the local network. E.g.: suppose two machines on the local network are called ns1.localnet.com and ns2.localnet.com. In this case, the local domain name would be localnet.com.

Getting any one of the above pieces of information wrong could mean that your network connection doesn't work properly, but won't do any permanent damage.

To assign the IP address and netmask to the interface made above (ath0), type the following (as root):

ifconfig ath0 <IP address> netmask <netmask> up

To use the nameserver address, open /etc/resolv.conf in a text editor. Again, you need to do this as root:

gedit /etc/resolv.conf

If you prefer another editor, replace gedit with the command that runs it.

Once the file is opened, add a '#' to any lines which start with the word 'nameserver', and add your own lines like the following:

nameserver <nameserver IP>

If you have multiple nameserver addresses, you can add more than one 'nameserver' line. If you know the network's local domain, add a line like this:

search <local domain>

Then save the file, and close the editor.

The final step is to tell your machine to use the gateway (if there is one). To do this, you must use the route command, which tells the kernel how to accesses different subnets. As root, you should type:

route add default gw <gateway hostname>

Or:

route add default gw <gateway address>

You should then be able to 'see' out of the network and access the Internet. To check if everything worked, type:

ping bbc.co.uk

You should get an output like this:

PING bbc.co.uk (212.58.224.131) 56(84) bytes of data.

64 bytes from rdirwww-vip.thdo.bbc.co.uk (212.58.224.131): icmp\_seq=1 ttl=119 time=15.4 ms

64 bytes from rdirwww-vip.thdo.bbc.co.uk (212.58.224.131): icmp\_seq=2 ttl=119 time=14.3 ms

64 bytes from rdirwww-vip.thdo.bbc.co.uk (212.58.224.131): icmp\_seq=3 ttl=119 time=15.1 ms

## Connecting to an AP with WEP

Connecting to an AP using WEP encryption is very similar to connecting to an open AP. The main difference is that you need to specify the WEP key using **iwconfig**. You may also need to specify whether your WEP key is "open" or "shared" (this is a function of how the AP's WEP encryption is set up).

First, connect to the access point using "iwconfig" as described above. It is a good idea, when using WEP, to specify the AP you want to connect to (rather than just using "ap auto" or "ap any"). This is done by specifying either the AP's MAC address or its ESSID, as shown above.

Next, you may need to specify whether your WEP key is "open" or "shared." If you are using a shared WEP key, use [iwpriv](http://madwifi-project.org/wiki/UserDocs/iwpriv) authmode 2 by entering this command (as root):

iwpriv ath0 authmode 2

If you are using an open WEP key (which is often considered more secure) use [iwpriv](http://madwifi-project.org/wiki/UserDocs/iwpriv) authmode 1 by entering this command (as root):

iwpriv ath0 authmode 1

Once you have specified whether you are using a shared or open WEP key, enter the key by issuing this command (as root):

iwconfig ath0 key <wep key (in hex)>

Or, if you are using an ASCII key rather than a hexadecimal key, issue this command (as root):

iwconfig ath0 key <s:ASCII string of key>

This should establish a connection between your wireless card and your AP. The next step is to set up your internet connection by either using DHCP or by specifying the details of your internet connection, as explained above.

## Connecting to an AP with WPA

WPA currently offers the best security scheme currently available. This extra security takes a little bit more time and effort to setup, but it is well worth it. The following links are to pages that describe [MadWifi](http://madwifi-project.org/wiki/MadWifi)'s support for WPA, and how to take advantage of it:

\* [UserDocs/802.11i](http://madwifi-project.org/wiki/UserDocs/802.11i)

\* [UserDocs/WPA\_PSK\_on\_Both\_Ends](http://madwifi-project.org/wiki/UserDocs/WPA_PSK_on_Both_Ends#TheStationclientSide)

## Removing [MadWifi](http://madwifi-project.org/wiki/MadWifi)

Removing [MadWifi](http://madwifi-project.org/wiki/MadWifi) is easy, but needs to be done as root. First, change to the directory with the sources in (or, if you no longer have the sources, download them again). Then type:

make uninstall

This will run the scripts which find and remove any modules and tools which have been installed.

[Troubleshooting](http://madwifi-project.org/wiki/UserDocs/Troubleshooting)

If something goes wrong, then the first thing you should do is check back through what you've done, and make sure you did it right. It sounds simple, but when you aren't familiar with something, it's easy to make small mistakes.

After that, if something still isn't working, you should have a look at [UserDocs/Troubleshooting](http://madwifi-project.org/wiki/UserDocs/Troubleshooting), a page of FAQs about Madwifi driver problems. There is also quite a lot of documentation on the Madwifi wiki, under [UserDocs](http://madwifi-project.org/wiki/UserDocs), having a look around the wiki might prove quite handy. If there's nothing on there that matches the problems you're having, then it might be a good idea to have a look at [Support](http://madwifi-project.org/wiki/Support). This page explains how to go about looking for, and getting help on Madwifi. In most cases, the solution to your problem will be available in either the mailing list archives, or by asking on the mailing list or IRC. Details of the both of these can be found on the [Support](http://madwifi-project.org/wiki/Support) page.

## More Documentation

If you want to view some of the documentation that comes with MADWiFi, you can find it all in the docs directory of the source tree. Once you are in the right directory, you can compile the file users-guide.tex to a PDF file by typing:

make

You will need a latex distribution installed for this to work.

You can also compile the users-guide as HTML, by installing [latex2html](http://www.latex2html.org/), and then using the command:

latex2html users-guide.tex

Which will create a directory called users-guide with its output inside.

*Attachments*

* [MadWifi First time user HOWTO.pdf](http://madwifi-project.org/attachment/wiki/UserDocs/FirstTimeHowTo/MadWifi%20First%20time%20user%20HOWTO.pdf) (44.3 kB) -Text in PDF format, added by arturo.montalvo@zmaw.de on 10/20/08 14:36:35.

# Using Ad-Hoc Mode

## New Code

Follow the instructions on [UserDocs/AdHocInterface](http://madwifi-project.org/wiki/UserDocs/AdHocInterface) before proceeding with the instructions below.

## Old Code

(optional) This [iwpriv](http://madwifi-project.org/wiki/UserDocs/iwpriv) command locks the NIC in the 802.11g-only mode

iwpriv ath0 mode 3

Set the type of network to ad-hoc

iwconfig ath0 mode ad-hoc

## Common Instructions

After you have your ad-hoc interface, set it up as normal:

Set the ESSID for the cell you want to connect to.

iwconfig ath0 essid yournetworkidhere

(optional) Set the network WEP key... Consult the iwconfig manpages to see other ways of setting a key. Here it is done using a string.

iwconfig ath0 key s:yourkeyhere

Set up your network using a static ip address.

ifconfig ath0 192.168.2.1

# Setting a Client to Use WEP.

First of all, you'll need to create a base device (does not apply to [MadWifi](http://madwifi-project.org/wiki/MadWifi)-Old). For information on how to proceed, see [UserDocs/StationInterface](http://madwifi-project.org/wiki/UserDocs/StationInterface).

The basic commands you need to execute are:

iwconfig ath0 key XXXXXXXXXX

iwconfig ath0 ap XX:XX:XX:XX:XX:XX

And then, if you feel the need:

iwconfig ath0 channel X

iwconfig ath0 essid foobar

iwconfig ath0 rate XXMB

You can also use a passphrase instead of a wep ‘key’ by issuing the command:

iwconfig ath0 key "s:passphrase"

When you want to disable WEP, you can try the command:

ifconfig ath0 down

iwconfig ath0 key open off

ifconfig ath0 up

# Open vs Shared

WEP can work in two different modes, open and shared. Some devices play better with one WEP variant or the other. You can switch like so:

iwpriv ath0 authmode 1 // open

iwpriv ath0 authmode 2 // shared

Somewhat counterintuitively, both variants support encryption and can be used in conjunction with iwconfig ath0 key HEX\_KEY\_HERE restricted. The primary difference between open and shared is that the latter has a challenge-response handshake during association. In a bizzare twist, the extra protocol makes shared *less secure* than open. (This is likely a side effect of the fact that WEP has been cracked for years.)

## How do I get my card to use WEP

If you have trouble with iwconfig when using WEP, specifically if you can’t set the essid or associate with an access point, then it may be the order in which you are issuing the iwconfig commands. Silas Bennett recommends the following order:

iwconfig ath0 key XXXXXXXXXX

iwconfig ath0 ap XX:XX:XX:XX:XX:XX

iwconfig ath0 channel X

iwconfig ath0 essid foobar

iwconfig ath0 rate XXMB

If you use debian, then the interfaces (in /etc/networks/interfaces used by ifup) commands would similarly be:

wireless\_key XXXXXXXXXX

wireless-ap XX:XX:XX:XX:XX:XX

wireless-channel X

wireless-essid foobar

wireless-rate XXMB

Where all of the Xs and ‘foobar’s are replaced by relevant values. All of the values can be used in ifupdown-roam’s ‘lifaces’. Silas also says that it is not normally necessary to put the lines after iwconfig ath0 ap XX:XX:XX:XX:XX:XX since the channel, essid and rate are set automatically on sucessfully associating with an access point. Also note that you can’t copy the second set of commands into [RedHat?](http://madwifi-project.org/wiki/RedHat)’s config files verbatim; they are debian specific.

More information can be found in the archives [here](http://sourceforge.net/mailarchive/forum.php?thread_id=3220174&forum_id=33958).

## Notes for TEW-443PI - Atheros chip revision A1.1R

Under PC Linux OPS which is a distro based on Mandriva you should setup under the harware options for your wireless card (Control Center -> Hardware -> Look at and configure harware) -> Set the current driver option: [autocreate](http://madwifi-project.org/wiki/UserDocs/autocreate): mode of your wireless card.

For my Trendnet TEW-443PI with Atheros revision A1.1R works after I added in the **Set current driver option** the same mode used for the network.

Ex.: I wanted to use my card in an adhoc network between two computers. After added "adhoc" (without quotes) as option for [autocreate](http://madwifi-project.org/wiki/UserDocs/autocreate), reboot. Then setup my network using wizard and the network works like a charm Before adding that option my card never want to work.

# Setting up a Client Using WPA-PSK

This is the information on how to manually install and configure WPA. However, most distributions provide wpasupplicant packages, obviating the need for this process. See [UserDocs/Distro](http://madwifi-project.org/wiki/UserDocs/Distro) for distribution specific information (recommended).

## Prerequisites

* A working madwifi installation, and wpa\_supplicant. Or, if you have 2.6.14 or greater, a directory with the built madwifi source. See [UserDocs/GettingMadwifi](http://madwifi-project.org/wiki/UserDocs/GettingMadwifi) for instructions.

## Instructions

Download the latest development pack or cvs tarball of [wpa\_supplicant](http://hostap.epitest.fi/wpa_supplicant/) and unpack it (or just check it out of CVS). Then execute:

cd wpa\_supplicant

and with your favorite editor edit (you may have to create it first) the .config file so that it contains the following:

If the above criteria is satisfied, then the following .config file can be used to compile wpa\_supplicant:

CONFIG\_DRIVER\_WEXT=y

CONFIG\_CTRL\_IFACE=y

Use the following for Linux versions less than 2.6.14:

CONFIG\_DRIVER\_MADWIFI=y

CFLAGS += -I/path/to/madwifi-ng

CONFIG\_CTRL\_IFACE=y

Be sure that /path/to/madwifi-ng points to your [MadWifi](http://madwifi-project.org/wiki/MadWifi) source directory, where you built it. E.g. if you have [MadWifi](http://madwifi-project.org/wiki/MadWifi) folder in /usr/src/madwifi, then the correct flag is:

CFLAGS += -I/usr/src/madwifi

Compile the source with make and copy the files wpa\_cli, wpa\_passphrase and wpa\_supplicant into an appropriate directory e.g. /usr/local/sbin. Then execute:

wpa\_passphrase YOURSSID yourpassphrase

with the SSID of your AP and the passphrase you’ve entered in its WPA-PSK configuration. You’ll receive an output, which looks like this:

network={

ssid="YOURSSID"

#psk="yourpassphrase"

psk=edda86468aa67c3f71c0bbaf7828aedccd320f9011d63e699f5381a5b77e0c2a

}

Copy this output in a new file called /etc/wpa\_supplicant.conf and change the permissions after you’ve finished editing:

chmod 640 /etc/wpa\_supplicant.conf

To get your network device up and running, you could execute this command:

wpa\_supplicant -BW -Dwext -iath0 -c/etc/wpa\_supplicant.conf

wpa\_supplicant -B -Dwext -iath0 -c/etc/wpa\_supplicant.conf

Or, if you are using a Linux version smaller than 2.6.14, substitute the madwifi driver:

-Dmadwifi

(I'm not certain but two lines above the -BW might be a mistake. AFAIAC I changed it in -B and it worked, though it didn't work with -BW) If you omit -B from the above commands, then the terminal where you’ve started the wpa\_supplicant should now show the communication between your wlan card and the AP. -B tells the wpa\_supplicant process to run in the background.

For more information, take a look at README and at wpa\_supplicant.conf files supplied with the lastest wpa\_supplicant package.

# WPA on madwifi: AP & Station

**02/21/06 News** To make WPA2 (WPA2=RSN) or Hostapd-0.5.1 using the instructions below, You Must Use the development branch of hostapd posted after 02/21/06 04:00:00 (earlier releases still need to be patched). Please see:

* Ticket [~~#241~~](http://madwifi-project.org/ticket/241)
* Download madwifi from the svn release 1453 or greater - see [r1453](http://madwifi-project.org/changeset/1453) for details on the RSN patches.
* Use hostapd-0.5.1 (posted after 02/21/06 04:00:00) or later.

This page shows how to set up Pre-Shared Key WPA security on both the Access Point (AP) and the Station for **madwifi**, code. This does not cover setting up a Radius server, nor how to become a supplicant/client station to a Radius server. Pre-Shared keys are adequate for home and small networks, compared to the alternatives (WEP, Open) and much easier than a Radius server. NB: the example hostapd.conf setup has the standard bridge instruction commented out . As set, hostapd expects that bridging/routing/firewalling is managed by netfilters and iptables (ie: by hand or with Shorewall or the like). See the bottom of this doc for the standard bridging setup.

You will need to download, configure, and build programs for both the Station and the Access Point. You will need to construct runtime configuration files for the programs once you've built them. The examples below show wpa\_supplicant and hostapd version 0.4.7, **you must substitute your own version numbers** (ie: 0.5.0, etc) for those shown in the example scripts.

Further, you **must** already have your kernel source and build, and the madwifi source and drivers built and available. See [UserDocs/KernelConfig](http://madwifi-project.org/wiki/UserDocs/KernelConfig) and the Wiki [UserDocs](http://madwifi-project.org/wiki/UserDocs) for your distribution to find instructions. These examples all presume you've chosen **/usr/src/** as the source code directory-if you have made other choices, substitute your chosen source directory for **/usr/src/** in the scripts and config files below. The programs need to be run on the AP and on the Stations before you can authenticate the wireless connection and begin to associate and network. Once you've gotten everything running, you can automate your startups.

**Additional Documentation**

* See [UserDocs/802.11i](http://madwifi-project.org/wiki/UserDocs/802.11i) for the madwifi Station WPA Howto.
* See [UserDocs/HostAP](http://madwifi-project.org/wiki/UserDocs/HostAP) for more examples of both the host and supplicant setups.
* See the README file in your wpa\_supplicant source directory for details about the workstation implementation.
* See the madwifi.conf file in your hostapd source directory. This proved to be best example of hostapd.conf for madwifi that I found.
* See the README file in your hostapd source Directory for details on the AP implementation.
* See <http://www.emma.cam.ac.uk/teaching/computing/Instructions/802.1x/Linux.cfm> for an overview of WPA usage.
* For general introductions to networking see the HOWTOs,<http://www.tldp.org/> and <http://linux-net.osdl.org/index.php/Main_Page>

## The Station (client) Side

### Get wpa\_supplicant

On All workstations that want to subscribe to the WPA enabled AP, do the following:

Go here: <http://hostap.epitest.fi/wpa_supplicant/> and get: **Latest stable release:** (as of this writing: wpa\_supplicant-0.4.7.tar.gz). Unpack it to **/usr/src/wpa\_supplicant-0.4.7**

### Copy this to /usr/src/wpa\_supplicant-0.4.7/.config

CONFIG\_DRIVER\_MADWIFI=y

CFLAGS += -I/usr/src/madwifi-ng

CONFIG\_CTRL\_IFACE=y

### Now build wpa\_supplicant on the station

cd /usr/src/wpa\_supplicant-0.4.7

make clean

make

make install

### Run **/usr/src/wpa\_supplicant-0.4.7/wpa\_passphrase** to make your PSK

From **/usr/src/wpa\_supplicant-0.4.7/** execute:

wpa\_passphrase My\_WPA\_Protected\_AP\_ESSID "Some\_Decent\_PassPhrase\_of\_up\_64\_Characters" >> /etc/wpa\_supplicant.conf

with the SSID of your AP **My\_WPA\_Protected\_AP\_ESSID** and the passphrase of your choice **Some\_Decent\_PassPhrase\_of\_up\_64\_Characters** . This writes

network={

ssid="My\_WPA\_Protected\_AP\_ESSID"

#psk="Some\_Decent\_PassPhrase\_of\_up\_64\_Characters"

psk=701459761a3d17c5ddead0deafbeeffeedbadf00dc659db31e2e3d36f00a12b1

}

to **/etc/wpa\_supplicant.conf**

Edit **/etc/wpa\_supplicant.conf** and add the lines into the network section created above:

key\_mgmt=WPA-PSK

# proto=WPA

proto=RSN

Add the following line to **/etc/wpa\_supplicant.conf** prior to the network section:

ap\_scan=0

Uncomment the proto=WPA line (and comment out the RSN line) to enable WPA. Leave as is for WPA2 (stronger encryption).

If you want to make a user control GUI interface to wpa\_supplicant, see [UserDocs/WPA\_PSK\_on\_Both\_Ends/wpa\_gui](http://madwifi-project.org/wiki/UserDocs/WPA_PSK_on_Both_Ends/wpa_gui) for information about using wpa\_gui with wpa\_supplicant.

Change the permissions of **/etc/wpa\_supplicant.conf** with: **chmod 640 /etc/wpa\_supplicant.conf**

In Madwifi releases after [r1407](http://madwifi-project.org/changeset/1407), modprobe ath\_pci automatically creates ath0 (see [UserDocs/autocreate](http://madwifi-project.org/wiki/UserDocs/autocreate). If you are using Madwifi releases prior to [r1408](http://madwifi-project.org/changeset/1408), you must use [wlanconfig](http://madwifi-project.org/wiki/UserDocs/wlanconfig) to create the station device ath0. The examples below reflect the [autocreate](http://madwifi-project.org/wiki/UserDocs/autocreate)/post [r1407](http://madwifi-project.org/changeset/1407) behavior.

Once you have completed the AP side as described below, you should see something like this when you run wpa\_supplicant with debugging(-dd) on (from a terminal as root).

Note: If you encounter "Argument list too long" errors in your wpa\_supplicant debug output, use -Dwext rather than -Dmadwifi in the example below (assuming you're running a kernel that's version 1.6.15 or later).

For example:

* modprobe ath\_pci *(if running pre-*[*r1408*](http://madwifi-project.org/changeset/1408) *Madwifi code, then also:* [*wlanconfig*](http://madwifi-project.org/wiki/UserDocs/wlanconfig) *ath0 create wlandev wifi0 wlanmode sta)*
* iwconfig ath0 essid "My\_WPA\_Protected\_AP\_ESSID"
* ifconfig ath0 192.168.0.100 up
* /usr/local/bin/wpa\_supplicant -dd -Dmadwifi -iath0 -c/etc/wpa\_supplicant.conf

yields:

State: GROUP\_HANDSHAKE -> COMPLETED

CTRL-EVENT-CONNECTED - Connection to 00:0f:b4:a1:3f:47 completed (auth)

==================================================================

and **iwconfig ath0** shows:

ath0 IEEE 802.11g ESSID:"My\_WPA\_Protected\_AP\_ESSID" Nickname:"YOUR\_HOSTNAME"

Mode:Managed Frequency:2.422GHz Access Point: 00:0D:B3:1A:E2:67

Bit Rate:54Mb/s Tx-Power:18 dBm Sensitivity=0/3

Retry:off RTS thr:off Fragment thr:off

Encryption key:59B8-0286-FEED-DEAF-BEEF-F00D-192B Security mode:restricted

Power Management:off

Link Quality:43/94 Signal level:-52 dBm Noise level:-95 dBm

Rx invalid nwid:3 Rx invalid crypt:0 Rx invalid frag:0

Tx excessive retries:0 Invalid misc:0 Missed beacon:0

You should be able to use the network now, and your wireless data is encrypted and authenticated with WPA. If you have the interface up, but things like ping fail with huge error rates, check and assure that your firewall is not blocking ath0, and that you have a default route set to ath0.

You can terminate the running wpa\_supplicant with **CTRL-C** from the terminal you've launched it from, or with **pkill wpa\_supplicant**.

### To automate your Station device, execute this script

#!/bin/sh

modprobe ath\_pci (If running pre-r1408 Madwifi code, then also: wlanconfig ath0 create wlandev wifi0 wlanmode sta)

iwconfig ath0 essid "My\_WPA\_Protected\_AP\_ESSID"

ifconfig ath0 192.168.0.100 up

wpa\_supplicant -Bw -Dmadwifi -iath0 -c/etc/wpa\_supplicant.conf

The above is only an example. You may have a better way to invoke the modprobe ath\_pci (ie: /etc/modprobe.conf, /etc/modules.conf, or the like). Your distribution may offer a means to launch this script when you insert a pccard, or start the network. See [UserDocs/Distro](http://madwifi-project.org/wiki/UserDocs/Distro) for your distribution for more information.

### An Advanced Example of wpa\_supplicant

This is example of setting up wpa\_supplicant to run on a [WRAP](http://madwifi-project.org/wiki/UserDocs/WRAP). Do all the setups as above, but for the make wpa\_supplicant- also you'll need to first build your kernel with the AES option as shown below.

**Platform**

*OpenWrt, BCM4710A0 CPU, Asus WL-500G AP: atheros 5212 wireless miniPCI card, madwifi-ng 2005-12-15 snapshot, wpa\_supplicant-0.4.7*

**Compile the kernel with**

*CONFIG\_CRYPTO\_AES=m*

After the regular modprobe **ath\_pci**, also modprobe **aes**

Make wpa\_supplicant as described above, but with this **/usr/src/wpa\_supplicant-0.4.7/.config**

(You *NEED* to change is the path in this line: CFLAGS += -I/root/drivers/madwifi-ng)

CONFIG\_DRIVER\_ATMEL=n

CONFIG\_DRIVER\_HOSTAP=n

CONFIG\_DRIVER\_IPW=n

CONFIG\_DRIVER\_MADWIFI=y

# Point this to your madwifi(-ng) sources

CFLAGS += -I/root/drivers/madwifi-ng

CONFIG\_DRIVER\_NDISWRAPPER=n

CONFIG\_DRIVER\_PRISM54=n

CONFIG\_DRIVER\_WEXT=y

CONFIG\_DRIVER\_WIRED=y

CONFIG\_WIRELESS\_EXTENSION=y

#CONFIG\_DRIVER\_NDIS=y

#CONFIG\_DRIVER\_HERMES=y

#CONFIG\_DRIVER\_BROADCOM=y

CONFIG\_IEEE8021X\_EAPOL=y

CONFIG\_EAP\_MD5=y

CONFIG\_EAP\_MSCHAPV2=y

CONFIG\_EAP\_FAST=y

CONFIG\_EAP\_TLS=y

CONFIG\_EAP\_PEAP=y

CONFIG\_EAP\_TTLS=y

CONFIG\_EAP\_GTC=y

CONFIG\_EAP\_OTP=y

CONFIG\_EAP\_SIM=y

CONFIG\_EAP\_PSK=y

CONFIG\_EAP\_PAX=y

CONFIG\_EAP\_LEAP=y

CONFIG\_EAP\_AKA=y

CONFIG\_PKCS12=y

CONFIG\_SMARTCARD=y

CONFIG\_PCSC=y

CONFIG\_CTRL\_IFACE=y

CONFIG\_READLINE=y

If wpa\_supplicant -D madwifi -i ath0 -c /etc/wpa\_supplicant.conf -dd -t fails to associate and you see something like:

Setting scan request: 0 sec 100000 usec

BSSID 00:0f:66:c8:8b:14 blacklist count incremented to 2

State: GROUP\_HANDSHAKE -> DISCONNECTED

EAPOL: External notification - portEnabled=0

EAPOL: SUPP\_PAE entering state DISCONNECTED

...then :

"ioctl[IEEE80211\_IOCTL\_SETKEY]: No such device or address". | ioctl[unknown???]: No such device or address"), and what should read "RX EAPOL from 00:12:17:b8:1c:db" when wpa\_supplicant is launched with the debug log -dd option ubstead reads "ioctl[unknown???]: No such device or address"

You've probably not gotten the kernel config right or the **aes** module loaded.

## Now do the AP side

### Get hostapd

Go here: <http://hostap.epitest.fi/hostapd/> and get: **Latest stable release** (as of this writing: <http://hostap.epitest.fi/releases/hostapd-0.4.7.tar.gz>) unpack it to /usr/src/hostapd-0.4.7

### Copy the below to /usr/src/hostapd-0.4.7/.config

# Driver interface for madwifi driver

CONFIG\_DRIVER\_MADWIFI=y

CFLAGS += -I/usr/src/madwifi-ng # change to reflect local setup; directory for madwifi src

# IEEE 802.11F/IAPP

CONFIG\_IAPP=y

# WPA2/IEEE 802.11i RSN pre-authentication

CONFIG\_RSN\_PREAUTH=y

# Integrated EAP server

CONFIG\_EAP=y

# EAP-MD5 for the integrated EAP server

CONFIG\_EAP\_MD5=y

# EAP-TLS for the integrated EAP server

CONFIG\_EAP\_TLS=y

# EAP-MSCHAPv2 for the integrated EAP server

CONFIG\_EAP\_MSCHAPV2=y

# EAP-PEAP for the integrated EAP server

CONFIG\_EAP\_PEAP=y

# EAP-GTC for the integrated EAP server

CONFIG\_EAP\_GTC=y

# EAP-TTLS for the integrated EAP server

CONFIG\_EAP\_TTLS=y

# EAP-PSK for the integrated EAP server

#CONFIG\_EAP\_PSK=y

# PKCS#12 (PFX) support (reads private keys or certificates from .p12 or .pfx files)

CONFIG\_PKCS12=y

# RADIUS authentication server. Access the integrated EAP server from external hosts using RADIUS.

CONFIG\_RADIUS\_SERVER=y

Change to the /usr/src/hostapd-0.4.7 directory and:

* make clean
* make
* make install

**Caution**, hostapd's **make clean** doesn't get the installed /usr/local/bin copies, and **make install** won't overwrite them. You must hand erase them if you have to run it 2x+

Add the ssid and psk generated above to your /etc/hostapd.conf on the AP as shown below

### Copy the below to /etc/hostapd.conf (uncomment the bridge line for standard bridging)

# An additional configuration parameter, bridge,

# must be used to notify hostapd if the interface is included in a bridge.

#bridge=br0 # Enable this for standard bridging, leave disabled for netfilter firewalls

interface=ath0

driver=madwifi

logger\_syslog=-1

logger\_syslog\_level=2

logger\_stdout=--1

logger\_stdout\_level=2

debug=0

ctrl\_interface\_group=0

macaddr\_acl=0

deny\_mac\_file=/etc/hostapd.deny

auth\_algs=3

eapol\_key\_index\_workaround=0

eap\_server=0

dump\_file=/tmp/hostapd.dump

ssid="My\_WPA\_Protected\_AP\_ESSID"

wpa=3

wpa\_psk=701459761a3d17c5ddead0deafbeeffeedbadf00dc659db31e2e3d36f00a12b1

wpa\_key\_mgmt=WPA-PSK

wpa\_pairwise=TKIP CCMP

### Successful hostapd Run Log With Debugging Output

[root@cablin]# **hostapd -dd /etc/hostapd.conf**

Configuration file: /etc/hostapd.conf

madwifi\_set\_iface\_flags: dev\_up=0

Using interface ath0 with hwaddr 00:0a:b5:89:ea:b7 and ssid 'alex'

madwifi\_set\_ieee8021x: enabled=1

madwifi\_configure\_wpa: group key cipher=1

madwifi\_configure\_wpa: pairwise key ciphers=0xa

madwifi\_configure\_wpa: key management algorithms=0x2

madwifi\_configure\_wpa: rsn capabilities=0x0

madwifi\_configure\_wpa: enable WPA= 0x1

madwifi\_set\_iface\_flags: dev\_up=1

madwifi\_set\_privacy: enabled=1

WPA: group state machine entering state GTK\_INIT

WPA: group state machine entering state SETKEYSDONE

madwifi\_set\_key: alg=TKIP addr=00:00:00:00:00:00 key\_idx=1

Flushing old station entries

madwifi\_sta\_deauth: addr=ff:ff:ff:ff:ff:ff reason\_code=3

Deauthenticate all stations

l2\_packet\_receive - recvfrom: Network is down

When a station succesfully connects you will see things (amoung many) like:

ath0: STA 00:0f:b5:62:b1:71 WPA: received EAPOL-Key frame (2/2 Group)

WPA: 00:0f:b5:62:b1:71 WPA\_PTK\_GROUP entering state REKEYESTABLISHED

ath0: STA 00:0f:b5:62:b1:71 WPA: group key handshake completed (WPA)

WPA: group state machine entering state SETKEYSDONE

madwifi\_set\_key: alg=TKIP addr=00:00:00:00:00:00 key\_idx=1

WPA: 00:0f:b5:62:b1:71 WPA\_PTK\_GROUP entering state IDLE

Checking STA 00:0f:b5:62:b1:71 inactivity:

Station has been active

If you see something successful like the above you can kill hostapd and turn down debugging by setting the debug values in /etc/hostapd to 0 or 1 (your choice), then relaunch hostapd without the -dd option.

If the AP has netfilters/iptables and or a firewall manager like Shorewall (my favorite) running on the AP, expect to manage bridging/routing/firewalling without **brctl** or **br0**. See your firewall docs for instructions.

To Launch hostapd in the background (mode G only, ip 192.168.0.2 in this example), at every startup from a script**.**

#!/bin/sh

/sbin/modprobe ath\_pci autocreate=ap

iwconfig ath0 essid "My\_WPA\_Protected\_AP\_ESSID"

iwpriv ath0 mode 3

ifconfig ath0 192.168.0.2 up

hostapd -B /etc/hostapd.conf

NB: The above is just an example. Be certain to include the '[autocreate](http://madwifi-project.org/wiki/UserDocs/autocreate)=ap' parameter to modprobe ath\_pci (as shown above), or you will have to '[wlanconfig](http://madwifi-project.org/wiki/UserDocs/wlanconfig) ath0 destroy; [wlanconfig](http://madwifi-project.org/wiki/UserDocs/wlanconfig) ath0 create wlandev wifi0 wlanmode ap' to start the AP VAP. You will also need to do this if you are using the pre-[r1408](http://madwifi-project.org/changeset/1408) Madwifi code.

### Standard Bridging

Alternatively, to build an AP with standard bridging ath0 to eth0 (mode G only, ip 192.168.0.2 in this example), uncomment the #br0 line in hostapd.conf (see above) and:

/sbin/modprobe ath\_pci autocreate=ap

iwconfig ath0 essid "My\_WPA\_Protected\_AP\_ESSID"

iwpriv ath0 mode 3

brctl addbr br0

brctl addif br0 eth0

brctl addif br0 ath0

brctl setfd br0 1

ifconfig ath0 up

ifconfig eth0 up

ifconfig br0 192.168.0.2 up

hostapd -dd /etc/hostapd.conf

See **man brctl**, and <http://linux-net.osdl.org/index.php/Bridge> for more about bridging.

## Things That Did Not Work

* The hostapd.conf from users-guide appendix
* Don't begin with a copy /usr/src/hostapd/defconfig file as base, use MADWIFI.CONF instead.
* Don't follow the README regarding editing the makefile to set the madwifi directory, do that in the .config

# How do I get the card to use monitor mode

To create a monitor mode VAP, see: [UserDocs/MonitorModeInterface](http://madwifi-project.org/wiki/UserDocs/MonitorModeInterface). After that, it won't be necessary to use the command iwconfig ath0 mode monitor.

You can just add another VAP in monitor mode and it will act just like the rawdev used to in past days of madwifi-old - you can send packets to that device, and they will go out like they would on the old rawdevice.

# MAC Address Filtering in AP Mode

## Introduction

This short introduction explains how to set up an access control list (ACL) or whitelist in AP mode. It is assumed that you already know how to set up a madwifi AP, but if you don’t, there is a brief howto at [UserDocs/SimpleAccessPoint](http://madwifi-project.org/wiki/UserDocs/SimpleAccessPoint).

## Steps

1. First, make sure your card is not set to any particular mode or essid.
2. Run:
   * To flush the list of MAC addresses:
   * iwpriv ath0 maccmd 3
   * To make the list a whitelist:
   * iwpriv ath0 maccmd 1
3. Put the card in master mode:
4. iwconfig ath0 mode master essid test
5. ifconfig ath0 up
6. At this point, nothing will be able to connect to the AP, since the whitelist is empty. To rectify this, you need to add some MACs to the list:
7. iwpriv ath0 addmac 00:01:02:03:04:05

Obviously you need to change this, and repeat as needed for other addresses you want to allow.

1. Hey presto! Your AP whitelist should be working perfectly.

## Other Possibilities

The most obvious change you could make to this is to make it into a blacklist. Fortunately, this is dead easy. Just change the line reading [iwpriv](http://madwifi-project.org/wiki/UserDocs/iwpriv) ath0 maccmd 1 to [iwpriv](http://madwifi-project.org/wiki/UserDocs/iwpriv) ath0 maccmd 2.

## Security Issues

Do not rely solely on a MAC whitelist for your APs security; it is far too easy to spoof.

# How can I use my card as an access point?

NOTE: because a station VAP is created by default, [wlanconfig](http://madwifi-project.org/wiki/UserDocs/wlanconfig) will fail to create an AP VAP (for [MadWifi](http://madwifi-project.org/wiki/MadWifi) releases >1407) unless the parameter '[autocreate](http://madwifi-project.org/wiki/UserDocs/autocreate)=none' is passed when loading the ath\_pci module. See [UserDocs/autocreate](http://madwifi-project.org/wiki/UserDocs/autocreate) for details.

To create an interface (called *ath0*) as an access point, issue the command:

wlanconfig ath0 create wlandev wifi0 wlanmode ap

This will allow clients to see your network and associate with it. To do anything useful beyond having network access to just the machine with the wireless NIC, you will need to set up some sort of routing, most likely install a DHCP server on the router, etc.

If you want the box to behave like a commercial AP, bridging the wireless and wired networks:

Under Debian (or Ubuntu, etc.), this can be accomplished almost entirely within /etc/network/interfaces, which gives Debian systems a standardized way to setup network interfaces. Ensure that bridging is available in the kernel (either modules or builtin), then use something like the following in your /etc/network/interfaces:

# Bring up ath0 with the correct wifi settings

# "manual" causes it to bring up the interface without TCP/IP (which will be setup on the bridge interface br0)

auto ath0

iface ath0 inet manual

# ensure ath0 is down (never fails because of "true")

pre-up wlanconfig ath0 destroy || true

# set up the ath0 device in AP mode before bringing up the interface (unless you're using AutoCreate)

pre-up wlanconfig ath0 create wlandev wifi0 wlanmode ap

# remove the ath0 device when bringing the interface down

post-down wlanconfig ath0 destroy

# set master mode, channel, and the essid

wireless-mode master

wireless-channel 11

wireless-essid YourESSID

# IF you use WEP, put the key here:

#wireless-key 1234-1234-1234-1234

# Bring up the bridge device, br0

# Using DHCP to assign an IP, etc. may work as well,

# but this illustrates setting up a static IP for the interface.

# The IP will be accessible from both the wired and wireless sides of the network.

auto br0

iface br0 inet static

# Assign your IP address, subnet and mask, broadcast address, and default gateway

address 192.168.0.[x]

network 192.168.0.0

netmask 255.255.255.0

broadcast 192.168.0.255

gateway 192.168.0.1

# Bridge eth0 and ath0 with br0.

bridge\_ports eth0 ath0

# You do \*not\* need a separate entry for eth0 in this file, because you are not configuring it

# or bringing it up as a separate interface at all. Mentioning it in the bridge\_ports directive

# is enough.

I am successfully using this setup to bridge my wired and wireless networks, along with [hostapd](http://hostap.epitest.fi/hostapd/) for WPA authentication.

Without bridging: Here's another method where you just add the info to /etc/networks/interfaces. I'm using this on ubuntu. I haven't figured out where to stick the 802.1g switch, but this gets the access point up, and brings it up at boot time:

iface ath0 inet static

address 10.42.2.1

netmask 255.255.255.0

broadcast 10.42.2.255

wireless-essid your\_essid

wireless-mode master

wireless-key 1223-1234-1234-1234 # well, not really, use yours.

My configuration with ubuntu 8.04. What I have, a modem/router managing internet, with my PC connected to it, and with that PC with a PCI atheros card installed and configured as an AP, where I can connect from my PC-WIFI, navigating without problems.

Note: Some times (too many), ubuntu hangs when ath0 is up, just at the start. Be careful with configuring an automatic IFUP because if linux hangs you will need to edit this file through a live-CD.

The keys, when the AP and the PC interfaces (eth0 and ath0) share the same netmask (this is what I believe), you do not need to bridge the interfaces.

What I need, the etc/network/interfaces file

auto lo

iface lo inet loopback

auto eth0

iface eth0 inet static

address 192.168.1.8

netmask 255.255.255.0

gateway 192.168.1.1

auto ath0

iface ath0 inet static

pre-up wlanconfig ath0 create wlandev wifi0 wlanmode ap

post-down wlanconfig ath0 destroy

wireless-mode master

wireless-essid "DEXTER"

wireless-channel 1

address 10.0.0.1

netmask 255.255.255.0

With this, the PC-WIFI can connect to the DEXTER network. Now you need to allow the modem/router to manage the ath0 connections, which will be obtained through IPTABLES.

My iptables script, just to allow the navigation (you have to build your firewall rules)

# activate ip forwading

/bin/echo "1" > /proc/sys/net/ipv4/ip\_forward

# Flush any rules

iptables --table filter --flush FORWARD

iptables --table filter --flush INPUT

iptables --table filter --flush OUTPUT

# Drop default policies

iptables --table filter --policy INPUT ACCEPT

iptables --table filter --policy OUTPUT ACCEPT

iptables --table filter --policy FORWARD ACCEPT

# Free loopback connexions

iptables --table filter --append INPUT --in-interface lo --jump ACCEPT

iptables --table filter --append OUTPUT --out-interface lo --jump ACCEPT

# -------------------------#

# ROUTING

# -------------------------#

iptables --table nat \

--append POSTROUTING \

--out-interface eth0 \

--source 10.0.0.0/255.255.255.0 \

--jump SNAT \

--to-source 192.168.1.8

And that's all. This works, but should be used just as a first step in the building of a WIFI secure spot. As a last note, the PC-WIFI should have an IP like 10.0.0.8 pointing to the 10.0.0.1 as the gateway, with the same 255.255.255.0 netmask, and the DNS properly introduced as well.

# Beacon Interval

Set beacon interval value:

iwpriv ath0 bintval x

Replace x by the desired beacon interval in ms

Get beacon interval value:

iwpriv ath0 get\_bintval

You'll get something like that:

ath0 get\_bintval:100

That means the beacon interval value is 100ms. Beacon intervals close to the upper limit of 500 ms should be avoided when using wpa\_supplicant. There appears to be a problem with repeated association and disassociation of the client with the AP.

# DTIM period

DTIM stands for Delivery Traffic Indication Message.

A DTIM is a countdown mechanism for WLAN Access Points, informing clients of the next window for listening to broadcast and multicast messages. When the Access Point has buffered broadcast or multicast messages for associated clients, it sends the next DTIM with a DTIM Interval value. Access Point clients hear the beacons and awaken to receive the broadcast and multicast messages.

Set the beacon DTIM period:

iwpriv ath0 dtim\_period x

Replace 'x' with the desired DTIM period in ms   
  
  
Get the current beacon DTIM in ms:

iwpriv ath0 get\_dtim\_period

You'll get something like that:

ath0 get\_dtim\_period:1

# Using [HostAP](http://madwifi-project.org/wiki/UserDocs/HostAP)

## Introduction

From the [HostAP](http://hostap.epitest.fi/hostapd) website:

*hostapd is a user space daemon for access point and authentication servers. It implements IEEE 802.11 access point management, IEEE 802.1X/WPA/WPA2/EAP authenticators, RADIUS client, EAP server, and RADIUS authentication server. The current version supports Linux (*[*HostAP*](http://madwifi-project.org/wiki/UserDocs/HostAP)*, madwifi, Prism54 drivers) and FreeBSD (net80211).*

So why would you want to run hostapd? Because you want to run your wifi client in WPA or WPA2 mode for increased security.

WPA is available in two flavors: WPA-EAP (also referred to as WPA-Enterprise) and WPA-PSK. [WPA-PSK](http://madwifi-project.org/wiki/UserDocs/HostAP#WPAPSKsetup) uses pre-shared keys, while [WPA-EAP](http://madwifi-project.org/wiki/UserDocs/HostAP#WPAEAPsetup) makes use of an external authentication server.

## General note about bridging

If you're bridging between the wireless and wired (ie using brctl), you **must** add a line such as:

bridge=br0

or whatever the bridge name you are using to the config file, or the WPA key exchange packets will get eaten by the bridge.

Symptom of this problem is that the host AP will retransmit the WPA key exchange packets (3 times) then deassociate the authenticating STA. The associating STA will receive the initial EAP-KEY packets, and respond, but the transmissions from the STA never reach the host AP daemon.

To avoid any further problems when connecting to the AP the first time it is best to set the forwarding delay of the bridge to 0. If not you can experience similar problems when first attempting to connect to a WPA-secured AP.

brctl setfd br0 0

## WPA-PSK setup

This is an example hostapd.conf file from a Debian box running hostapd in WPA-PSK mode. It is setup to use allow/deny lists and to only operate in WPA-PSK mode.

##### hostapd configuration file ##############################################

# Empty lines and lines starting with # are ignored

# AP netdevice name (without 'ap' prefix, i.e., wlan0 uses wlan0ap for

# management frames)

interface=ath0

# Driver interface type (hostap/wired/madwifi/prism54; default: hostap)

driver=madwifi

# hostapd event logger configuration

#

# Two output method: syslog and stdout (only usable if not forking to

# background).

#

# Module bitfield (ORed bitfield of modules that will be logged; -1 = all

# modules):

# bit 0 (1) = IEEE 802.11

# bit 1 (2) = IEEE 802.1X

# bit 2 (4) = RADIUS

# bit 3 (8) = WPA

# bit 4 (16) = driver interface

# bit 5 (32) = IAPP

#

# Levels (minimum value for logged events):

# 0 = verbose debugging

# 1 = debugging

# 2 = informational messages

# 3 = notification

# 4 = warning

#

logger\_syslog=-1

logger\_syslog\_level=2

logger\_stdout=-1

logger\_stdout\_level=1

# Debugging: 0 = no, 1 = minimal, 2 = verbose, 3 = msg dumps, 4 = excessive

debug=0

# Dump file for state information (on SIGUSR1)

dump\_file=/tmp/hostapd.dump

# Interface for separate control program. If this is specified, hostapd

# will create this directory and a UNIX domain socket for listening to requests

# from external programs (CLI/GUI, etc.) for status information and

# configuration. The socket file will be named based on the interface name, so

# multiple hostapd processes/interfaces can be run at the same time if more

# than one interface is used.

# /var/run/hostapd is the recommended directory for sockets and by default,

# hostapd\_cli will use it when trying to connect with hostapd.

ctrl\_interface=/var/run/hostapd

# Access control for the control interface can be configured by setting the

# directory to allow only members of a group to use sockets. This way, it is

# possible to run hostapd as root (since it needs to change network

# configuration and open raw sockets) and still allow GUI/CLI components to be

# run as non-root users. However, since the control interface can be used to

# change the network configuration, this access needs to be protected in many

# cases. By default, hostapd is configured to use gid 0 (root). If you

# want to allow non-root users to use the contron interface, add a new group

# and change this value to match with that group. Add users that should have

# control interface access to this group.

#

# This variable can be a group name or gid.

#ctrl\_interface\_group=wheel

ctrl\_interface\_group=0

##### IEEE 802.11 related configuration #######################################

# SSID to be used in IEEE 802.11 management frames

ssid=YOURSSID

# Station MAC address -based authentication

# 0 = accept unless in deny list

# 1 = deny unless in accept list

# 2 = use external RADIUS server (accept/deny lists are searched first)

macaddr\_acl=0

# Accept/deny lists are read from separate files (containing list of

# MAC addresses, one per line). Use absolute path name to make sure that the

# files can be read on SIGHUP configuration reloads.

accept\_mac\_file=/etc/hostapd/hostapd.accept

deny\_mac\_file=/etc/hostapd/hostapd.deny

# IEEE 802.11 specifies two authentication algorithms. hostapd can be

# configured to allow both of these or only one. Open system authentication

# should be used with IEEE 802.1X.

# Bit fields of allowed authentication algorithms:

# bit 0 = Open System Authentication

# bit 1 = Shared Key Authentication (requires WEP)

auth\_algs=1

# Associate as a station to another AP while still acting as an AP on the same

# channel.

#assoc\_ap\_addr=00:12:34:56:78:9a

##### IEEE 802.1X (and IEEE 802.1aa/D4) related configuration #################

# Require IEEE 802.1X authorization

#ieee8021x=0

# Use integrated EAP authenticator instead of external RADIUS authentication

# server

#eap\_authenticator=0

# Path for EAP authenticator user database

#eap\_user\_file=/etc/hostapd/eap\_user

# CA certificate (PEM or DER file) for EAP-TLS/PEAP/TTLS

#ca\_cert=/etc/hostapd/ca.pem

# Server certificate (PEM or DER file) for EAP-TLS/PEAP/TTLS

#server\_cert=/etc/hostapd/server.pem

# Private key matching with the server certificate for EAP-TLS/PEAP/TTLS

# This may point to the same file as server\_cert if both certificate and key

# are included in a single file. PKCS#12 (PFX) file (.p12/.pfx) can also be

# used by commenting out server\_cert and specifying the PFX file as the

# private\_key.

#private\_key=/etc/hostapd/server.prv

# Passphrase for private key

#private\_key\_passwd=secret passphrase

# Configuration data for EAP-SIM database/authentication gateway interface.

# This is a text string in implementation specific format. The example

# implementation in eap\_sim\_db.c uses this as the file name for the GSM

# authentication triplets.

#eap\_sim\_db=/etc/hostapd/sim\_db

# Optional displayable message sent with EAP Request-Identity

#eap\_message=hello

# WEP rekeying (disabled if key lengths are not set or are set to 0)

# Key lengths for default/broadcast and individual/unicast keys:

# 5 = 40-bit WEP (also known as 64-bit WEP with 40 secret bits)

# 13 = 104-bit WEP (also known as 128-bit WEP with 104 secret bits)

#wep\_key\_len\_broadcast=5

#wep\_key\_len\_unicast=5

# Rekeying period in seconds. 0 = do not rekey (i.e., set keys only once)

#wep\_rekey\_period=300

# EAPOL-Key index workaround (set bit7) for WinXP Supplicant (needed only if

# only broadcast keys are used)

#eapol\_key\_index\_workaround=0

# EAP reauthentication period in seconds (default: 3600 seconds; 0 = disable

# reauthentication).

#eap\_reauth\_period=3600

##### IEEE 802.11f - Inter-Access Point Protocol (IAPP) #######################

# Interface to be used for IAPP broadcast packets

#iapp\_interface=eth0

##### RADIUS configuration ####################################################

# for IEEE 802.1X with external Authentication Server, IEEE 802.11

# authentication with external ACL for MAC addresses, and accounting

# The own IP address of the access point (used as NAS-IP-Address)

#own\_ip\_addr=127.0.0.1

# Optional NAS-Identifier string for RADIUS messages. When used, this should be

# a unique to the NAS within the scope of the RADIUS server. For example, a

# fully qualified domain name can be used here.

#nas\_identifier=ap.example.com

# RADIUS authentication server

#auth\_server\_addr=127.0.0.1

#auth\_server\_port=1812

#auth\_server\_shared\_secret=secret

# RADIUS accounting server

#acct\_server\_addr=127.0.0.1

#acct\_server\_port=1813

#acct\_server\_shared\_secret=secret

# Secondary RADIUS servers; to be used if primary one does not reply to

# RADIUS packets. These are optional and there can be more than one secondary

# server listed.

#auth\_server\_addr=127.0.0.2

#auth\_server\_port=1812

#auth\_server\_shared\_secret=secret2

#

#acct\_server\_addr=127.0.0.2

#acct\_server\_port=1813

#acct\_server\_shared\_secret=secret2

# Retry interval for trying to return to the primary RADIUS server (in

# seconds). RADIUS client code will automatically try to use the next server

# when the current server is not replying to requests. If this interval is set,

# primary server will be retried after configured amount of time even if the

# currently used secondary server is still working.

#radius\_retry\_primary\_interval=600

# Interim accounting update interval

# If this is set (larger than 0) and acct\_server is configured, hostapd will

# send interim accounting updates every N seconds. Note: if set, this overrides

# possible Acct-Interim-Interval attribute in Access-Accept message. Thus, this

# value should not be configured in hostapd.conf, if RADIUS server is used to

# control the interim interval.

# This value should not be less 600 (10 minutes) and must not be less than

# 60 (1 minute).

#radius\_acct\_interim\_interval=600

# hostapd can be used as a RADIUS authentication server for other hosts. This

# requires that the integrated EAP authenticator is also enabled and both

# authentication services are sharing the same configuration.

# File name of the RADIUS clients configuration for the RADIUS server. If this

# commented out, RADIUS server is disabled.

#radius\_server\_clients=/etc/hostapd/radius\_clients

# The UDP port number for the RADIUS authentication server

#radius\_server\_auth\_port=1812

##### WPA/IEEE 802.11i configuration ##########################################

# Enable WPA. Setting this variable configures the AP to require WPA (either

# WPA-PSK or WPA-RADIUS/EAP based on other configuration). For WPA-PSK, either

# wpa\_psk or wpa\_passphrase must be set and wpa\_key\_mgmt must include WPA-PSK.

# For WPA-RADIUS/EAP, ieee8021x must be set (but without dynamic WEP keys),

# RADIUS authentication server must be configured, and WPA-EAP must be included

# in wpa\_key\_mgmt.

# This field is a bit field that can be used to enable WPA (IEEE 802.11i/D3.0)

# and/or WPA2 (full IEEE 802.11i/RSN):

# bit0 = WPA

# bit1 = IEEE 802.11i/RSN (WPA2) (dot11RSNAEnabled)

wpa=1

# WPA pre-shared keys for WPA-PSK. This can be either entered as a 256-bit

# secret in hex format (64 hex digits), wpa\_psk, or as an ASCII passphrase

# (8..63 characters) that will be converted to PSK. This conversion uses SSID

# so the PSK changes when ASCII passphrase is used and the SSID is changed.

# wpa\_psk (dot11RSNAConfigPSKValue)

# wpa\_passphrase (dot11RSNAConfigPSKPassPhrase)

#wpa\_psk=0123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef

wpa\_passphrase=yoursecretpassphrase

# Optionally, WPA PSKs can be read from a separate text file (containing list

# of (PSK,MAC address) pairs. This allows more than one PSK to be configured.

# Use absolute path name to make sure that the files can be read on SIGHUP

# configuration reloads.

#wpa\_psk\_file=/etc/hostapd/wpa\_psk

# Set of accepted key management algorithms (WPA-PSK, WPA-EAP, or both). The

# entries are separated with a space.

# (dot11RSNAConfigAuthenticationSuitesTable)

#wpa\_key\_mgmt=WPA-PSK WPA-EAP

wpa\_key\_mgmt=WPA-PSK

# Set of accepted cipher suites (encryption algorithms) for pairwise keys

# (unicast packets). This is a space separated list of algorithms:

# CCMP = AES in Counter mode with CBC-MAC [RFC 3610, IEEE 802.11i/D7.0]

# TKIP = Temporal Key Integrity Protocol [IEEE 802.11i/D7.0]

# Group cipher suite (encryption algorithm for broadcast and multicast frames)

# is automatically selected based on this configuration. If only CCMP is

# allowed as the pairwise cipher, group cipher will also be CCMP. Otherwise,

# TKIP will be used as the group cipher.

# (dot11RSNAConfigPairwiseCiphersTable)

wpa\_pairwise=TKIP CCMP

# Time interval for rekeying GTK (broadcast/multicast encryption keys) in

# seconds. (dot11RSNAConfigGroupRekeyTime)

#wpa\_group\_rekey=600

# Rekey GTK when any STA that possesses the current GTK is leaving the BSS.

# (dot11RSNAConfigGroupRekeyStrict)

#wpa\_strict\_rekey=1

# Time interval for rekeying GMK (master key used internally to generate GTKs

# (in seconds).

#wpa\_gmk\_rekey=86400

# Enable IEEE 802.11i/RSN/WPA2 pre-authentication. This is used to speed up

# roaming be pre-authenticating IEEE 802.1X/EAP part of the full RSN

# authentication and key handshake before actually associating with a new AP.

# (dot11RSNAPreauthenticationEnabled)

#rsn\_preauth=1

#

# Space separated list of interfaces from which pre-authentication frames are

# accepted (e.g., 'eth0' or 'eth0 wlan0wds0'. This list should include all

# interface that are used for connections to other APs. This could include

# wired interfaces and WDS links. The normal wireless data interface towards

# associated stations (e.g., wlan0) should not be added, since

# pre-authentication is only used with APs other than the currently associated

# one.

#rsn\_preauth\_interfaces=eth0

Another (very basic) example for a WPA-PSK setup can be found [here](http://www.rikanise.net/files/hostapd.conf/).

## WPA-EAP setup

This example covers setting up hostapd to use WPA and an external RADIUS server with EAP authentication utilizing PEAP and MSCHAPv2 This is the working hostapd.conf file used to authenticate against [FreeRADIUS](http://www.freeradius.org/) v1.0.5.

##### hostapd configuration file ##############################################

bridge=br0 # bridge interface usually br0

interface=ath0 #atheros interface

driver=madwifi #driver type

ssid=wpa-test #set essid

#macaddr\_acl=2 #optional macaddress authentication instead of user/password pair -- macaddr\_acl=2 tells hostapd to use radius

#accept\_mac\_file=/etc/hostapd.accept

#deny\_mac\_file=/etc/hostapd.deny

ieee8021x=1

auth\_algs=1

eap\_server=0

eapol\_key\_index\_workaround=1

###Radius Setup

own\_ip\_addr=10.0.0.1

nas\_identifier=test.5gwireless.com

auth\_server\_addr=10.0.0.200

auth\_server\_port=1812

auth\_server\_shared\_secret=testing123

acct\_server\_addr=10.0.0.200

acct\_server\_port=1813

acct\_server\_shared\_secret=testing123

###WPA

wpa=1

wpa\_key\_mgmt=WPA-EAP

wpa\_pairwise=TKIP

wpa\_group\_rekey=300

wpa\_gmk\_rekey=640

FreeRADIUS and Windows XP supplicant setup are explained [here](http://text.dslreports.com/forum/remark,9286052~mode=flat/).

# How do I see who's connected to my AP?

Issue the command:

wlanconfig ath0 list sta

Or just:

wlanconfig ath0 list

This should give you a nicely formatted list. See man wlanconfig 8 for more information.

**If you are using Madwifi-Old**

This is fairly simple, you just need to issue the command:

iwlist ath0 peers

Obviously modifying the ath0 where needed.

Since this command is regarded obsolete and will be removed from new wireless tools versions, you can also get the information (even more detailed) from following command:

cat /proc/sys/net/wlan/ath0/associated\_sta

# How do I get my card to use turbo mode?

See [here](http://madwifi-project.org/wiki/ChipsetFeatures/SuperAG#TurboMode) for a short explanation of what turbo mode is.

You can set your card to **static** turbo mode in master, managed and ad-hoc mode. Unfortunately, the *driver does not support dynamic turbo mode* yet. This means that non-turbo cards will not be able to associate with your AP, and also [MadWifi](http://madwifi-project.org/wiki/MadWifi) cards not in turbo modes ( or, apparently, other makes of cards) won’t be able to see it.

Turbo mode is just a double channel (2\*20MHz). If the driver is set to 54 mbit rate (static, no auto) and one enables turbo, that will give you 108 mbit link speed. **There is no reported rate called 108 mbit**, in turbo mode the real rate is the reported rate\*2.

One may need to set the countrycode ( [UserDocs/CountryCode](http://madwifi-project.org/wiki/UserDocs/CountryCode) ) to 0 to get turbo mode working.

Issue the following commands:

iwpriv ath0 mode 3 # this forces it to G-only mode, but you can also use A mode (1))

iwpriv ath0 turbo 1 # enable turbo

iwpriv ath0 mode 5 # required if the interface is in 11a managed (STA) mode

This might not enable turbo ;) Depends on you countrycode ! For example 840 (USA)

# wlanconfig ath0 list chann

Channel 1 : 2412 Mhz 11g Channel 48 : 5240 Mhz 11a Dynamic

Channel 2 : 2417 Mhz 11g Channel 50 : 5250 Mhz 11a Static

Channel 3 : 2422 Mhz 11g Channel 52 : 5260 Mhz 11a

Channel 4 : 2427 Mhz 11g Channel 56 : 5280 Mhz 11a Dynamic

Channel 5 : 2432 Mhz 11g Channel 58 : 5290 Mhz 11a Static

Channel 6 : 2437 Mhz 11g Dynamic Channel 60 : 5300 Mhz 11a

Channel 7 : 2442 Mhz 11g Channel 64 : 5320 Mhz 11a

Channel 8 : 2447 Mhz 11g Channel 149 : 5745 Mhz 11a

Channel 9 : 2452 Mhz 11g Channel 152 : 5760 Mhz 11a Static

Channel 10 : 2457 Mhz 11g Channel 153 : 5765 Mhz 11a Dynamic

Channel 11 : 2462 Mhz 11g Channel 157 : 5785 Mhz 11a

Channel 36 : 5180 Mhz 11a Channel 160 : 5800 Mhz 11a Static

Channel 40 : 5200 Mhz 11a Dynamic Channel 161 : 5805 Mhz 11a Dynamic

Channel 42 : 5210 Mhz 11a Static Channel 165 : 5825 Mhz 11a

Channel 44 : 5220 Mhz 11a

Then to enable/force turbo in master (AP) or ad-hoc mode :

# iwconfig ath0 channel 0

# iwpriv ath0 mode 1 # 11a mode

# iwconfig ath0 channel 42 # channel 42 : 5210 Mhz 11a Static turbo

# iwconfig ath0 ... #

ath0 IEEE 802.11Ta ... # 11a + Turbo enabled (static turbo).

In 11a STA mode the following is also required:

# iwpriv ath0 mode 5

otherwise the STA will never see the AP.

In STA mode, it appears that attempts to set the channel are ignored so

# iwconfig ath0 channel x

is futile - the radio will just scan and lock.

Question:

# [iwpriv](http://madwifi-project.org/wiki/UserDocs/iwpriv) ath0 turbo 0 # disables turbo ? driver doesn't care ? don't know # [iwpriv](http://madwifi-project.org/wiki/UserDocs/iwpriv) ath0 turbo 1 # enables turbo ? :) Driver doesn't care ? or it is adjusted depending on countrycode automatically ?

}}}

# Countrycode

This page is the FAQ about Regdomain and Countrycode.

## Which settings are determined by the regdomain/countrycode?

The countrycode and regdomain settings control the channels the card can operate on, as well as maximum transmission power, whether active scanning can be used, where and how ad-hoc mode can be used and also whether DFS and TPC (802.11h) support is required.

## What are regdomains and countrycodes?

The regdomain outlines regions of the world which share similar regulatory conditions. Countrycode allows a finer selection of one of the countries inside the given regdomain. Both settings are stored inside the EEPROM of your card, which will be used by default.

In addition to the “real” regdomains, there are so called “world wide roaming” pseudo regdomain codes that seem to cover the union of [ETSI](http://www.etsi.org/) and [FCC](http://www.fcc.gov/) requirements. This might bring some restrictions for you: ETSI requires stuff like DFS and TPC to be enabled, and disallows other features like active scanning in certain frequency ranges, which is not required by FCC. You’ll find them programmed into most IBM-made cards.

Use

sysctl -a | grep country

to determine your current Country Code setting.

Use

sysctl -a | grep regdomain

to determine your current Regdomain setting.

## How can I change the regdomain or countrycode that Madwifi uses?

Whilst it isn’t possible to change the regdomain manually without messing with the EEPROM of your card (kids, don’t do this at home!), you can manually override the default countrycode (as long as it belongs to the regdomain which is used by your card) without causing any harm. This can be done with the module parameter countrycode:

modprobe ath\_pci countrycode=276 # country code for germany

Country codes follow the numbering scheme defined in ISO 3166, which can be found [here](http://www.davros.org/misc/iso3166.html) for example.  
**Important**: strip leading zeros from country code values mentioned in the standard. Example: in the standard *056* is given as code for Belgium, so you use *countrycode=56* when loading the driver.

The values for regdomain don't follow any ISO scheme, but are manufacturing codes introduced by Atheros. There is no publically available list of valid regdomain codes and their meaning, but since it isn't possible to easily change/override the regdomain EEPROM settings this doesn't matter anyway.

This is a preliminary list of regdomains for people who want to play with changing their regdomain using the ar5k utility

0x00 (wildcarded)

0x10 (FCC)

0x20 (DOC)

0x30 (ETSI)

0x31 (Spain)

0x32 (France)

0x37 (South Africa)

0x40 (MKK-Japan)

0xFF (debug)

Most "universal/worldwide" cards use zero as the value, some 0xFF

This is by no means a valid or complete list and should be revised and added to.

## Why doesn't [MadWifi](http://madwifi-project.org/wiki/MadWifi) determine the regdomain and countrycode from beacon packets, like the NDIS wrapped Atheros windows driver does?

Madwifi supports more modes than just station and ad-hoc. If a Madwifi driven card is being used in master mode (i.e. as an AP) then looking at beacon packets is clearly inappropriate, since they will have been produced by the madwifi driver itself.

## Can an incorrect countrycode or regdomain lead to poor bit rates?

Not in principle.

## Can I use two separate countrycodes on two cards in the same machine?

No, you can't.

## This appears to be a list of Country Codes.

<http://www.unicode.org/onlinedat/countries.html>

**FAQs - FCC Part 15 Power Limits(1)**

*How much power can I transmit on a 2.4 GHz 10 dBi omni and still be legal?*

The FCC regulations for PtMP allows 36 dBm (4 watts) EIRP when omni antennas are used. This is 30 dBm (1 watt) into a 6 dBi antenna. If you use a 10 dBi omni antenna, you must limit your transmitter (or amplifier) to 26 dBm (10 + 26 = 36 dBm). For a PtP panel antenna, more power is allowed (see separate FAQ on this). Power is measured at the antenna connector, so subtract any cable loss between the amplifier and the antenna. Refer to the following table:

Power at antenna (dBm/Watts) Antenna Gain (dBi) EIRP (dBm) EIRP (watts)

30 dBm (1 W) 6 36 4

27 dBm (500 mW) 9 36 4

24 dBm (250 mW) 12 36 4

21 dBm (125 mW) 15 36 4

18 dBm (62 mW) 18 36 4

15 dBm (31 mW) 21 36 4

12 dBm (15 mW) 24 36 4

*Can I use any antenna of my choice for my AP or CPE antenna?*

Yes, up to the highest gain antenna specified in the FCC certification information or the product literature that accompanies the device. (See FCC 04-165 adopted July 8, 2004, 15.204(c)) Those people in countries other than the US will need to consult their own regulations. If you are not sure if the antenna that you plan to use is certified or authorized with the radio system, ask the radio or antenna manufacturer/vendor. If you have the radio FCC ID, you can check on the FCC web site for certification information.

*How much power can I transmit with in my 2.4 GHz Point-to-Point system?*

According to FCC regulations, 2.4 GHz Part 15.247 point-to-point transmitters may use a 30 dBm transmitter with a 6 dBi antenna. For a 3 dB increase in antenna gain, the transmitter power output must be reduced by 1 dB. Power is measured at the antenna connector, so subtract any cable loss between the amplifier and the antenna. Refer to the following table. (also see Canadian Rules)

Power at antenna (dBm/watts) Max Antenna Gain (dBi) EIRP (dBm) EIRP (watts)

30 dBm (1 W) 6 36 4

29 9 38 6.3

28 12 40 10

27 dBm (500 mW) 15 42 16

26 18 44 25

25 21 46 39.8

24 dBm (250 mW) 24 48 63

23 27 50 100

22 30 52 158

*Is the Customer or Client (CPE) system considered PtMP or PtP?*

If the CPE system (or Subscriber Unit - SU) only talks with the POP/AP and is at a fixed location, then it is considered to be PtP and can use power and antenna gain associated with PtP systems, as shown below. (This has been verified by FCC Certified systems using a 26 dBm radio and a 17 dBi antenna) If a CPE system is part of a mesh network, then it is considered PtMP.

*Should I use 2.4 GHz or 5.8 GHz for my WLAN or WISP system?*

Currently, most systems use either IEEE 802.11 or 802.11b operating between 2.4 and 2.4835 GHz. As these frequencies become more congested, the U-NII Band 3 at 5.725 - 5.825 GHz (IEEE 802.11a) will be used more. 5.8 GHz also offers data transmission rates greater than 11 MB/s. However, more antenna gain will be necessary at 5.8 GHz for the same distance on 2.4 GHz. 5.8 GHz will have a smaller Fresnel zone, so there may be certain advantages when shooting a signal through a tight space between trees or buildings. The WCS and MMDS frequencies between 2.1 and 2.7 GHz are also available to FCC-licensed users. (See IEEE 802.16a)

*What frequencies are available to WLAN outside the US?*

The 2400-2500 MHz band is used worldwide. There are certain channels within this band that are allocated to certain regions, however. The 5725-5825 MHz band is used in the US & Canada with 4 watts EIRP (PtMP). Europe uses the HiperLAN frequencies of 5470-5725 MHz outdoors with 1 watt EIRP. The indoor band at 5 GHz is 5150-5250 MHz in US/Japan and 5150-5350 in Europe. There are also frequencies between 3.4 and 4.0 GHz which are available in Canada, Asia and Africa and the Far East. (See IEEE 802.16a) *How much power can I transmit on a 5.3 GHz 10 dBi omni and still be legal?*

The FCC regulations for PtMP and PtP allows only 30 dBm (1 watt) EIRP in the UNII-2 band. This is 24 dBm (250 mW) into a 6 dBi antenna. If you use a 10 dBi antenna, you must limit your transmitter (or amplifier) to 20 dBm (10 + 20 = 30 dBm). For a 15 dBi panel antenna, this allows a 15 dBm transmitter (or amplifier). Power is measured at the antenna connector, so subtract any cable loss between the amplifier and the antenna. Refer to the following table:

Power at antenna (dBm/Watts) Antenna Gain (dBi) EIRP (dBm) EIRP (watts)

24 dBm (250 mW) 6 30 1

21 dBm (125 mW) 9 30 1

18 dBm (62 mW) 12 30 1

15 dBm (31 mW) 15 30 1

12 dBm (15 mW) 18 30 1

9 dBm (7 mW) 21 30 1

6 dBm (4 mW) 24 30 1

*How much power can I transmit on a 7 dBi omni on 5.8 GHz and still be legal?*

The FCC regulations for PtMP allows 36 dBm (4 watts) EIRP in the UNII-3 band. This is 30 dBm (1 watt) into a 6 dBi antenna. If you use a 7 dBi antenna, you must limit your transmitter (or amplifier) to 29 dBm (7 + 29 = 36 dBm). For a 15 dBi sector antenna, this allows a 21 dBm transmitter (or amplifier). Power is measured at the antenna connector, so subtract any cable loss between the amplifier and the antenna. Refer to the following table:

Power at antenna (dBm/Watts) Antenna Gain (dBi) EIRP (dBm) EIRP (watts)

30 dBm (1 W) 6 36 4

27 dBm (500 mW) 9 36 4

24 dBm (250 mW) 12 36 4

21 dBm (125 mW) 15 36 4

18 dBm (62 mW) 18 36 4

15 dBm (31 mW) 21 36 4

12 dBm (15 mW) 24 36 4

*How much power can I legally transmit on a 23 dBi panel at 5.8 GHz?*

The FCC regulations for UNII-3 wideband digital fixed PtP transmitters allows a maximum 30 dBm (or 17 dBm + 10logB) output with directional antennas up to 23 dBi gain without any corresponding reduction in transmitter power. Maximum EIRP is 53 dBm (200 watts). Power is measured at the antenna connector, so subtract any cable loss between the amplifier and the antenna. Refer to the following table:

Power at antenna (dBm/Watts) Antenna Gain (dBi) EIRP (dBm) EIRP (watts)

30 dBm (1 W) 6 36 4

30 dBm (1 W) 9 39 8

30 dBm (1 W) 12 42 16

30 dBm (1 W) 15 45 31

30 dBm (1 W) 18 48 62

30 dBm (1 W) 21 51 125

30 dBm (1 W) 23 53 200

(1) Adapted with permission from a document that appears on a number of websites including qorvus, michwave, adaletwireless, wifinerd, and globalspec.

## List of allowed txpower and frequencies in different countries.

From www.nuclearcat.com/athmap.txt (now appearently offline)

DEBUG (DB, 0x1ff, 511) NO\_ENUMRD (0x0, 0)

2312G 5.0 2317G 5.0 2322G 5.0 2327G 5.0 2332G 5.0 2337G 5.0

2342G 5.0 2347G 5.0 2352G 5.0 2357G 5.0 2362G 5.0 2367G 5.0

2372G 5.0 2412G 5.0 2417G 5.0 2422G 5.0 2427G 5.0 2432G 5.0

2437G 5.0 2442G 5.0 2447G 5.0 2452G 5.0 2457G 5.0 2462G 5.0

2467G 5.0 2472G 5.0 2512G 5.0 2517G 5.0 2522G 5.0 2527G 5.0

2532G 5.0 2537G 5.0 2542G 5.0 2547G 5.0 2552G 5.0 2557G 5.0

2562G 5.0 2567G 5.0 2572G 5.0 2577G 5.0 2582G 5.0 2587G 5.0

2592G 5.0 2597G 5.0 2602G 5.0 2607G 5.0 2612G 5.0 2617G 5.0

2622G 5.0 2627G 5.0 2632G 5.0 2637G 5.0 2642G 5.0 2647G 5.0

2652G 5.0 2657G 5.0 2662G 5.0 2667G 5.0 2672G 5.0 2677G 5.0

2682G 5.0 2687G 5.0 2692G 5.0 2697G 5.0 2484B 5.0 5120A 5.0

5140A 5.0 5160A 5.0 5180A 5.0 5200A 5.0 5220A 5.0 5240A 5.0

5260A 5.0 5280A 5.0 5300A 5.0 5320A 5.0 5340A 5.0 5360A 5.0

5380A 5.0 5400A 5.0 5420A 5.0 5440A 5.0 5460A 5.0 5480A 5.0

5500A 5.0 5520A 5.0 5540A 5.0 5560A 5.0 5580A 5.0 5600A 5.0

5620A 5.0 5640A 5.0 5660A 5.0 5680A 5.0 5700A 5.0 5745A 5.0

5765A 5.0 5785A 5.0 5805A 5.0 5825A 5.0 5130T 5.0 5150T 5.0

5170T 5.0 5190T 5.0 5210T 5.0 5230T 5.0 5250T 5.0 5270T 5.0

5290T 5.0 5310T 5.0 5330T 5.0 5350T 5.0 5370T 5.0 5390T 5.0

5410T 5.0 5430T 5.0 5450T 5.0 5470T 5.0 5490T 5.0 5510T 5.0

5530T 5.0 5550T 5.0 5570T 5.0 5590T 5.0 5610T 5.0 5630T 5.0

5650T 5.0

FCC1\_FCCA (0x10, 16)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 5180A 14.0

5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0 5280A 16.0 5300A 16.0

5320A 13.5 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0 5825A 18.0

5210T 14.0 5250T 14.0 5290T 15.0 5760T 16.0 5800T 16.0

ALBANIA (AL, 0x8, 8) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

ALGERIA (DZ, 0xc, 12) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

ARGENTINA (AR, 0x20, 32) APL3\_WORLD (0x47, 71)

2412B 18.0 2417B 18.0 2422B 18.0 2427B 18.0 2432B 18.0 2437B 18.0

2442B 18.0 2447B 18.0 2452B 18.0 2457B 18.0 2462B 18.0 2467B 18.0

2472B 18.0 5280A 16.0 5300A 16.0 5320A 13.5 5745A 18.0 5765A 18.0

5785A 18.0 5805A 18.0

ARMENIA (AM, 0x33, 51) ETSI4\_WORLD (0x30, 48)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0

AUSTRALIA (AU, 0x24, 36) FCC2\_WORLD (0x21, 33)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 14.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 13.5 5745A 18.0 5765A 18.0 5785A 18.0

5805A 18.0 5825A 18.0

AUSTRIA (AT, 0x28, 40) ETSI5\_WORLD (0x39, 57)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 15.0 5200A 15.0 5220A 15.0 5240A 15.0

AZERBAIJAN (AZ, 0x1f, 31) ETSI4\_WORLD (0x30, 48)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0

BAHRAIN (BH, 0x30, 48) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

BELARUS (BY, 0x70, 112) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

BELGIUM (BE, 0x38, 56) ETSI4\_WORLD (0x30, 48)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0

BELIZE (BZ, 0x54, 84) APL1\_ETSIC (0x55, 85)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0 5825A 18.0

BOLVIA (BO, 0x44, 68) APL1\_ETSIC (0x55, 85)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0 5825A 18.0

BRAZIL (BR, 0x4c, 76) NULL1\_ETSIC (0x8, 8)

2412B 18.0 2417B 18.0 2422B 18.0 2427B 18.0 2432B 18.0 2437B 18.0

2442B 18.0 2447B 18.0 2452B 18.0 2457B 18.0 2462B 18.0 2467B 18.0

2472B 18.0

BRUNEI DARUSSALAM (BN, 0x60, 96) APL1\_WORLD (0x52, 82)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0 5825A 18.0

BULGARIA (BG, 0x64, 100) ETSI6\_WORLD (0x34, 52)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5500A 17.0 5520A 17.0 5540A 17.0 5560A 17.0 5580A 17.0

5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0 5680A 17.0 5700A 17.0

CANADA (CA, 0x7c, 124) FCC2\_FCCA (0x20, 32)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 5180A 14.0

5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0 5280A 16.0 5300A 16.0

5320A 13.5 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0 5825A 18.0

CHILE (CL, 0x98, 152) APL5\_WORLD (0x58, 88)

2412B 18.0 2417B 18.0 2422B 18.0 2427B 18.0 2432B 18.0 2437B 18.0

2442B 18.0 2447B 18.0 2452B 18.0 2457B 18.0 2462B 18.0 2467B 18.0

2472B 18.0 5745A 17.0 5765A 17.0 5785A 17.0 5805A 17.0 5825A 17.0

CHINA (CN, 0x9c, 156) APL1\_WORLD (0x52, 82)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0 5825A 18.0

COLOMBIA (CO, 0xaa, 170) FCC1\_FCCA (0x10, 16)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 5180A 14.0

5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0 5280A 16.0 5300A 16.0

5320A 13.5 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0 5825A 18.0

COSTA RICA (CR, 0xbc, 188) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

CROATIA (HR, 0xbf, 191) ETSI3\_WORLD (0x36, 54)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0

CYPRUS (CY, 0xc4, 196) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

CZECH REPUBLIC (CZ, 0xcb, 203) ETSI3\_WORLD (0x36, 54)

2412B 18.0 2417B 18.0 2422B 18.0 2427B 18.0 2432B 18.0 2437B 18.0

2442B 18.0 2447B 18.0 2452B 18.0 2457B 18.0 2462B 18.0 2467B 18.0

2472B 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0

DENMARK (DK, 0xd0, 208) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

DOMINICAN REPUBLIC (DO, 0xd6, 214) FCC1\_FCCA (0x10, 16)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 5180A 14.0

5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0 5280A 16.0 5300A 16.0

5320A 13.5 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0 5825A 18.0

5210T 14.0 5250T 14.0 5290T 15.0 5760T 16.0 5800T 16.0

ECUADOR (EC, 0xda, 218) NULL1\_WORLD (0x3, 3)

2412B 18.0 2417B 18.0 2422B 18.0 2427B 18.0 2432B 18.0 2437B 18.0

2442B 18.0 2447B 18.0 2452B 18.0 2457B 18.0 2462B 18.0 2467B 18.0

2472B 18.0

EGYPT (EG, 0x332, 818) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

EL SALVADOR (SV, 0xde, 222) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

ESTONIA (EE, 0xe9, 233) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

FINLAND (FI, 0xf6, 246) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

FRANCE (FR, 0xfa, 250) ETSI3\_WORLD (0x36, 54)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0

GEORGIA (GE, 0x10c, 268) ETSI4\_WORLD (0x30, 48)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0

GERMANY (DE, 0x114, 276) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

GREECE (GR, 0x12c, 300) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

GUATEMALA (GT, 0x140, 320) FCC1\_FCCA (0x10, 16)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 5180A 14.0

5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0 5280A 16.0 5300A 16.0

5320A 13.5 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0 5825A 18.0

5210T 14.0 5250T 14.0 5290T 15.0 5760T 16.0 5800T 16.0

HONDURAS (HN, 0x154, 340) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

HONG KONG (HK, 0x158, 344) FCC2\_WORLD (0x21, 33)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 14.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 13.5 5745A 18.0 5765A 18.0 5785A 18.0

5805A 18.0 5825A 18.0

HUNGARY (HU, 0x15c, 348) ETSI2\_WORLD (0x35, 53)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0

ICELAND (IS, 0x160, 352) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

INDIA (IN, 0x164, 356) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

INDONESIA (ID, 0x168, 360) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

IRAN (IR, 0x16c, 364) APL1\_WORLD (0x52, 82)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0 5825A 18.0

IRELAND (IE, 0x174, 372) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

ISRAEL (IL, 0x178, 376) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

ITALY (IT, 0x17c, 380) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

JAPAN (JP, 0x188, 392) MKK1\_MKKA (0x40, 64)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 2484B 18.0 5170A 16.0 5190A 16.0 5210A 16.0 5230A 16.0

JAPAN1 (J1, 0x189, 393) MKK1\_MKKB (0x41, 65)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 2484B 18.0 5170A 16.0 5190A 16.0 5210A 16.0 5230A 16.0

JAPAN2 (J2, 0x18a, 394) MKK1\_FCCA (0x48, 72)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 5170A 16.0

5190A 16.0 5210A 16.0 5230A 16.0

JAPAN3 (J3, 0x18b, 395) MKK2\_MKKA (0x43, 67)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 2484B 18.0 4920A 16.0 4940A 16.0 4960A 16.0 4980A 16.0

5040A 16.0 5060A 16.0 5080A 16.0 5170A 16.0 5190A 16.0 5210A 16.0

5230A 16.0

JAPAN4 (J4, 0x18c, 396) MKK1\_MKKA1 (0x4a, 74)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 2484B 18.0 5170A 16.0 5190A 16.0 5210A 16.0 5230A 16.0

JAPAN5 (J5, 0x18d, 397) MKK1\_MKKA2 (0x4b, 75)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 2484B 18.0 5170A 16.0 5190A 16.0 5210A 16.0 5230A 16.0

JORDAN (JO, 0x190, 400) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

KAZAKHSTAN (KZ, 0x18e, 398) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

NORTH KOREA (KP, 0x198, 408) APL2\_WORLD (0x45, 69)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0

KOREA REPUBLIC (KR, 0x19a, 410) APL2\_WORLD (0x45, 69)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0

KOREA REPUBLIC2 (K2, 0x19b, 411) APL2\_APLD (0x49, 73)

2312G 18.0 2317G 18.0 2322G 18.0 2327G 18.0 2332G 18.0 2337G 18.0

2342G 18.0 2347G 18.0 2352G 18.0 2357G 18.0 2362G 18.0 2367G 18.0

2372G 18.0 2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0

2437G 18.0 2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0

2467G 18.0 2472G 18.0 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0

KUWAIT (KW, 0x19e, 414) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

LATVIA (LV, 0x1ac, 428) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

LEBANON (LB, 0x1a6, 422) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

LIECHTENSTEIN (LI, 0x1b6, 438) ETSI2\_WORLD (0x35, 53)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0

LITHUANIA (LT, 0x1b8, 440) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

LUXEMBOURG (LU, 0x1ba, 442) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

MACAU (MO, 0x1be, 446) FCC2\_WORLD (0x21, 33)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 14.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 13.5 5745A 18.0 5765A 18.0 5785A 18.0

5805A 18.0 5825A 18.0

MACEDONIA (MK, 0x327, 807) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

MALAYSIA (MY, 0x1ca, 458) NULL1\_WORLD (0x3, 3)

2412B 18.0 2417B 18.0 2422B 18.0 2427B 18.0 2432B 18.0 2437B 18.0

2442B 18.0 2447B 18.0 2452B 18.0 2457B 18.0 2462B 18.0 2467B 18.0

2472B 18.0

MEXICO (MX, 0x1e4, 484) FCC1\_FCCA (0x10, 16)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 5180A 14.0

5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0 5280A 16.0 5300A 16.0

5320A 13.5 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0 5825A 18.0

5210T 14.0 5250T 14.0 5290T 15.0 5760T 16.0 5800T 16.0

MONACO (MC, 0x1ec, 492) ETSI4\_WORLD (0x30, 48)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0

MOROCCO (MA, 0x1f8, 504) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

NETHERLANDS (NL, 0x210, 528) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

NEW ZEALAND (NZ, 0x22a, 554) FCC2\_ETSIC (0x22, 34)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 14.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 13.5 5745A 18.0 5765A 18.0 5785A 18.0

5805A 18.0 5825A 18.0

NORWAY (NO, 0x242, 578) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

OMAN (OM, 0x200, 512) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

PAKISTAN (PK, 0x24a, 586) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

PANAMA (PA, 0x24f, 591) FCC1\_FCCA (0x10, 16)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 5180A 14.0

5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0 5280A 16.0 5300A 16.0

5320A 13.5 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0 5825A 18.0

5210T 14.0 5250T 14.0 5290T 15.0 5760T 16.0 5800T 16.0

PERU (PE, 0x25c, 604) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

PHILIPPINES (PH, 0x260, 608) FCC1\_WORLD (0x11, 17)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 14.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 13.5 5745A 18.0 5765A 18.0 5785A 18.0

5805A 18.0 5825A 18.0 5210T 14.0 5250T 14.0 5290T 15.0 5760T 16.0

5800T 16.0

POLAND (PL, 0x268, 616) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

PORTUGAL (PT, 0x26c, 620) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

PUERTO RICO (PR, 0x276, 630) FCC1\_FCCA (0x10, 16)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 5180A 14.0

5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0 5280A 16.0 5300A 16.0

5320A 13.5 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0 5825A 18.0

5210T 14.0 5250T 14.0 5290T 15.0 5760T 16.0 5800T 16.0

QATAR (QA, 0x27a, 634) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

ROMANIA (RO, 0x282, 642) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

RUSSIA (RU, 0x283, 643) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

SAUDI ARABIA (SA, 0x2aa, 682) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

SINGAPORE (SG, 0x2be, 702) APL4\_WORLD (0x42, 66)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 14.0 5200A 16.0 5220A 16.0 5240A 16.0 5745A 18.0

5765A 18.0 5785A 18.0 5805A 18.0 5825A 18.0

SLOVAK REPUBLIC (SK, 0x2bf, 703) ETSI3\_WORLD (0x36, 54)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0

SLOVENIA (SI, 0x2c1, 705) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

SOUTH AFRICA (ZA, 0x2c6, 710) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

SPAIN (ES, 0x2d4, 724) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

SWEDEN (SE, 0x2f0, 752) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

SWITZERLAND (CH, 0x2f4, 756) ETSI2\_WORLD (0x35, 53)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0

SYRIA (SY, 0x2f8, 760) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

TAIWAN (TW, 0x9e, 158) APL3\_WORLD (0x47, 71)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5280A 16.0 5300A 16.0 5320A 13.5 5745A 18.0 5765A 18.0

5785A 18.0 5805A 18.0

THAILAND (TH, 0x2fc, 764) APL2\_WORLD (0x45, 69)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0

TRINIDAD & TOBAGO (TT, 0x30c, 780) ETSI4\_WORLD (0x30, 48)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0

TUNISIA (TN, 0x314, 788) ETSI3\_WORLD (0x36, 54)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0

TURKEY (TR, 0x318, 792) ETSI3\_WORLD (0x36, 54)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0

UKRAINE (UA, 0x324, 804) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

UNITED ARAB EMIRATES (AE, 0x310, 784) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

UNITED KINGDOM (GB, 0x33a, 826) ETSI1\_WORLD (0x37, 55)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5180A 16.0 5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0

5280A 16.0 5300A 16.0 5320A 16.0 5500A 17.0 5520A 17.0 5540A 17.0

5560A 17.0 5580A 17.0 5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0

5680A 17.0 5700A 17.0

UNITED STATES (US, 0x348, 840) FCC1\_FCCA (0x10, 16)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 5180A 14.0

5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0 5280A 16.0 5300A 16.0

5320A 13.5 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0 5825A 18.0

5210T 14.0 5250T 14.0 5290T 15.0 5760T 16.0 5800T 16.0

URUGUAY (UY, 0x35a, 858) APL2\_WORLD (0x45, 69)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0

UZBEKISTAN (UZ, 0x35c, 860) FCC3\_FCCA (0x3a, 58)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 5180A 14.0

5200A 16.0 5220A 16.0 5240A 16.0 5260A 16.0 5280A 16.0 5300A 16.0

5320A 13.5 5500A 17.0 5520A 17.0 5540A 17.0 5560A 17.0 5580A 17.0

5600A 17.0 5620A 17.0 5640A 17.0 5660A 17.0 5680A 17.0 5700A 17.0

5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0 5825A 18.0 5210T 14.0

5250T 14.0 5290T 15.0 5760T 16.0 5800T 16.0

VENEZUELA (VE, 0x35e, 862) APL2\_ETSIC (0x56, 86)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0 5745A 18.0 5765A 18.0 5785A 18.0 5805A 18.0

VIET NAM (VN, 0x2c0, 704) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

YEMEN (YE, 0x377, 887) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

ZIMBABWE (ZW, 0x2cc, 716) NULL1\_WORLD (0x3, 3)

2412G 18.0 2417G 18.0 2422G 18.0 2427G 18.0 2432G 18.0 2437G 18.0

2442G 18.0 2447G 18.0 2452G 18.0 2457G 18.0 2462G 18.0 2467G 18.0

2472G 18.0

My computer has extra LEDs for wireless information, how can I make them work?

If the LED is connected to the chipsets GPIO pins, then the following commands should make it blink slowly on power up, and then switch to a 3 second blink period on association (similar to the LED behaviour on PC Cards).

Use:

sysctl -w dev.wifi0.ledpin=1

sysctl -w dev.wifi0.softled=1

For AR5007EG the value of dev.wifi0.ledpin must be 3

*For* [*MadWifi*](http://madwifi-project.org/wiki/MadWifi)*-Old use:*

sysctl -w dev.ath0.ledpin=1

sysctl -w dev.ath0.softled=1

Note that sysctl parameters are not saved in non-volatile memory, so you’ll want to put these commands in a local boot script or just add the following lines to /etc/sysctl.conf:

dev.wifi0.ledpin=1

dev.wifi0.softled=1

*For* [*MadWifi*](http://madwifi-project.org/wiki/MadWifi)*-Old use:*

dev.ath0.ledpin=1

dev.ath0.softled=1

For the owners of Gigabyte GN-WIAG02 minipci card: I found out that the led is on pin 2, so in this case dev.wifi0.ledpin=2 is the proper choice. Note that I am using it with Dell D610 laptop, this could also be important.

For the owners of IBM Thinkpads (at least R51 and R51e series): the following settings will light up your led:

sysctl -w dev.wifi0.ledpin=0

sysctl -w dev.wifi0.softled=1

For the Thinkpad R60e and X40 set both values to 1.

## Can I turn on my wireless card LEDs automatically?

If you were able to turn on your LEDs as described above, then most likely yes.

### Using udev

[udev](http://www.kernel.org/pub/linux/utils/kernel/hotplug/udev.html) *"works entirely in userspace, using hotplug events the kernel sends whenever a device is added or removed from the kernel."* udev can also *"run a particular program when a device is connected or disconnected."*

To make udev enable the LEDs, create a short script that calls the correct sysctl commands and then add a [udev rule](http://www.reactivated.net/writing_udev_rules.html): KERNEL=="ath0", RUN+="/path/to/led/script"

To see if udev manages your wireless card, try **$ udevinfo -a -p /sys/class/net/ath0** and if you see descriptions about your card fill the screen, then udev manages your wireless card.

### Working example

The PCMCIA wireless card [SMC WCBT-G?](http://madwifi-project.org/wiki/BigOldCompatibilityList#SMCWCBTG/SMCWCBTGEU) has two LEDs: one for Link (which lights when the card is up) and the other for Activity (which blinks with traffic). Both LEDs can be turned on using the following script:

#!/bin/bash

# Set the Link LED to behave like link.

sysctl -w dev.wifi0.ledpin=1

sysctl -w dev.wifi0.softled=1

sysctl -w dev.wifi0.softled=0

# Set the Activity LED to behave like activity.

sysctl -w dev.wifi0.ledpin=0

sysctl -w dev.wifi0.softled=1

* Save this script as "leds"
* Make it executable with **$ chmod +x leds**
* Place it in your system path with something like **# mv leds /usr/local/bin**
* Add a udev rule to **/etc/udev/udev.d/10-local.rules** KERNEL=="ath0", RUN+="/usr/local/bin/leds"
* Reload udev's rules with **# udevcontrol reload\_rules**
* Test your new rule with **$ udevtest /class/net/ath0**

$ udevtest /sys/class/net/ath0

This program is for debugging only, it does not create any node,

or run any program specified by a RUN key. It may show incorrect results,

if rules match against subsystem specfic kernel event variables.

main: looking at device '/class/net/ath0' from subsystem 'net'

wait\_for\_sysfs: file '/sys/class/net/ath0/address' appeared after 0 loops

udev\_rules\_get\_name: rule applied, 'ath0' becomes 'ath0'

main: run: 'ifplugd.agent'

main: run: '/usr/local/bin/leds'

main: run: 'socket:/org/kernel/udev/monitor'

main: run: 'net.agent'

main: run: 'socket:/org/freedesktop/hal/udev\_event'

The next time udev dectects the wireless card being connected to the system, it will call /usr/local/bin/leds and the LEDs will turn on. Anytime you want to turn on the LEDs, you can also issue **# leds**.

### Working example for a Philips SNN6500 PCMCIA card on Gentoo

See <http://dev.gentoo.org/~wschlich/misc/scripts/udev/atheros-wlan-card-led-activation/ath_led.sh>

# How can I change the MAC address of my card?

With madwifi-ng, you change the MAC of the master device and VAP mac addresses are all based upon that. This must be done **before creating** any VAPs for the corresponding master device.

Ideally, load madwifi with [autocreate](http://madwifi-project.org/wiki/UserDocs/autocreate)=none as in this example:

modprobe ath\_pci autocreate=none

Alternatively, you can try destroying any VAPs using [wlanconfig](http://madwifi-project.org/wiki/UserDocs/wlanconfig) destroy, as in this example:

wlanconfig ath0 destroy

You cannot use ifconfig to change the mac at this time. It should work with newer netlink API (iproute or macchanger). The suggested approach is to use macchanger to alter the address. Here's an example of changing the master device's MAC with macchanger:

ip link set dev wifi0 down

macchanger --mac=xx:xx:xx:xx:xx:xx wifi0

ip link set dev wifi0 up

NOTE: macchanger can also make random addresses (--random) or have another MAC for the same vendor (--another). It's more specialized tool for the job than iproute2. You can get macchanger at <http://www.alobbs.com/macchanger/>

You should also be able to use iproute2 (it uses netlink) to do the same thing with:

ip link set dev wifi0 down

ip link set addr xx:xx:xx:xx:xx:xx dev wifi0

ip link set dev wifi0 up

Now create your VAPs

wlanconfig ath0 create wlandev wifi0 wlanmode sta

# How do I get the Cisco VPN client to work with madwifi?

Apparently if you use version 4.0.4A or later, it might work right out of the box, but for previous versions or if you are having problems, read on…

According to Tom Marshall, vpnclient needs fixing. This is what Tom has to say:

vpnclient has (bad, evil) code to look for acceptable interfaces based on their name. You need to add “ath” to that list. In interceptor.c you will find the supported\_device() function. Add these lines at the appropriate place:

else **if**(!strncmp(dev->name,"ath",3) && (dev->name[3]>='0' &&

+dev->name[3]<='9'))

{

rc=1;

}

Then recompile it and you should find that it works. Alternatively, you could rename the interface.

For version 4.0.4A or later, you might find that you can connect to your VPN server, but you may not get any other traffic through. If "/sbin/ifconfig wifi0" (or whatever your wifi device is) shows receive frame errors, then the issue is related to the latest supported\_device() code in interceptor.c.

According to Mitch Sukalski the code intercepts incoming traffic for both the wifiX and athX devices. It should only intercept the athX device. Modify the supported\_device() code in interceptor.c to something like this:

**int** rc=0;

**if** (strncmp(dev->name,"wifi",4) == 0)

{

rc=0;

}

**else** **if**(dev->type == ARPHRD\_ETHER)

{

rc=1;

}

**else** **if**(dev->type == ARPHRD\_PPP)

{

rc=1;

}

**return** rc;

# Bit-rate Selection Algorithms

802.11a, b, and g, have multiple bitrates (like 1Mbps, 6Mbps, 54Mbps, etc) from which transmitters can choose when sending data. Higher bit-rates allow high quality links to transmit more data, but suffer from loss on many links. Lower bit-rates, in general, have a lower loss probability and can transmit data over more links. Each link may have multiple bit-rates that deliver packets, and the throughput a particular bit-rate achieves is a function of the bit-rate and delivery probability. A few different algorithms that choose the bit-rate for each packet are included in madwifi; In the current driver, there are three available:

## onoe

A simple, robust algorithm that decreases the bit-rate when packets, on average, need at least 1 retry. Increases the bit-rate when less than 10% of packets require a retry. It performs very well in indoor 802.11b environments.

## amrr

Uses binary exponential backoff to avoid attempting to increase the bit-rate too often. It was conceived for high latency systems.

For more info, read <http://www-sop.inria.fr/rapports/sophia/RR-5208.html> ([local copy](http://madwifi-project.org/attachment/wiki/UserDocs/RateControl/RR-5208.pdf))

## sample

SampleRate chooses the bit-rate it predicts will provide the most throughput based on estimates of the expected per-packet transmission time each bit-rate. SampleRate periodically sends packets at bit-rates other than the current one to estimate when another bit-rate will provide better performance.

You can find some statistics on the operation of the sample rate control in /proc/net/madwifi/ath?/ratestats\_{250,1600,3000}. The ratestats\_\* files contain the stats used to make the decisions. The algorithm is estimating the amount of medium required to transmit a packet at each of the bit-rates. Since the loss rate is also a function of packet size, it gathers statistics for 250, 1600, and 3000 byte packets. So ratestats\_1600 for example represents the loss rates for packets of sizes between 251 and 1600 bytes. So if you are sending mostly 1500 byte Ethernet packets you would be interested mostly in the ratestats\_1600 proc entry.

Failed/pkts means the number of packets that failed out of the total number of packets sent (at that bit-rate). So if you transmit a packet and receive an ack it would be 0/1. If you transmit a packet 3 times, and then only get an ack on the forth retransmission it would be 3/4.

Average Tries means how many times on average you need to send the packet at the bit-rate before receiving an ack. If the success rate is 100% then average tries = 1. If the success rate is 50% then average tries is about 2.

For more info, read <http://www.pdos.lcs.mit.edu/papers/jbicket-ms.pdf> ([local copy](http://madwifi-project.org/attachment/wiki/UserDocs/RateControl/jbicket-ms.pdf))

## Bit-rate Selection Algorithm comparison

A comparison of different algorithms can be found in [John Bicket's paper](http://www.pdos.lcs.mit.edu/papers/jbicket-ms.pdf) ([local copy](http://madwifi-project.org/attachment/wiki/UserDocs/RateControl/jbicket-ms.pdf)) as well.

## Configuration

Currently the default algorithm is ath\_rate\_sample; most users say it provides the highest throughput and it performs well even on lossy links where other algorithms may have trouble selecting the bit-rate with the highest throughput. To change it, edit the following line in [Makefile](http://madwifi-project.org/wiki/UserDocs/Makefile).inc:

ATH\_RATE=ath\_rate/sample

Or call make like this:

make ATH\_RATE=ath\_rate/sample

You can, of course, substitute sample with amrr or onoe as appropriate for your desired configuration.

## Attachments

* [RR-5208.pdf](http://madwifi-project.org/attachment/wiki/UserDocs/RateControl/RR-5208.pdf) (323.2 kB) -Local copy of paper on AMRR, added by mrenzmann on 11/02/05 10:46:04.
* [jbicket-ms.pdf](http://madwifi-project.org/attachment/wiki/UserDocs/RateControl/jbicket-ms.pdf) (0.7 MB) -Local copy of paper on SAMPLE, added by mrenzmann on 11/02/05 10:47:04.
* [pam2006s2.pdf](http://madwifi-project.org/attachment/wiki/UserDocs/RateControl/pam2006s2.pdf) (240.5 kB) -Comparison of the different RCAs; it got published in PAM 2006., added by anonymous on 03/27/06 09:23:26.

# Note about 802.11a channels

As you probably already know, the list of channels available via [MadWifi](http://madwifi-project.org/wiki/MadWifi) on 11a is (in the US reg domain):

* Channel 36 : 5.18 GHz
* Channel 40 : 5.2 GHz
* Channel 42 : 5.21 GHz (T)
* Channel 44 : 5.22 GHz
* Channel 48 : 5.24 GHz
* Channel 50 : 5.25 GHz (T)
* Channel 52 : 5.26 GHz
* Channel 56 : 5.28 GHz
* Channel 58 : 5.29 GHz (T)
* Channel 60 : 5.3 GHz
* Channel 64 : 5.32 GHz
* Channel 149 : 5.745 GHz
* Channel 152 : 5.76 GHz (T)
* Channel 153 : 5.765 GHz
* Channel 157 : 5.785 GHz
* Channel 160 : 5.8 GHz (T)
* Channel 161 : 5.805 GHz
* Channel 165 : 5.825 GHz

Note however that channels 42, 50, 58, 152 and 160 are not available for non-turbo mode operation. This is no bug, but related to the way turbo mode works. In the 11a frequency ranges there are five channels (42, 50, 58, 152, 160) available for use with turbo mode, and these "turbo channels" use non-standard center frequencies.

<http://www.super-g.com/collateral/atheros_superg_whitepaper.pdf> has a small graphics explaining the differences in channel allocation between non-turbo and turbo. See section "Super G Key Features / Dynamic Turbo" (page 12).

# Long Distance links with [MadWifi](http://madwifi-project.org/wiki/MadWifi)

In order to run wireless links over long distances you need to keep a number of things in mind. Foremost you need to think about power and gain: do your radio and antenna combinations result in a sufficient signal-to-noise margin at the receiving ends? After that you need to tune a couple of parameters in the 802.11 protocol to enable proper communication over long distances. Finally, you may want to understand how the links are performing.

For long distance station connections to an AP, please also see [eXtendedRangeXR](http://madwifi-project.org/wiki/ChipsetFeatures/SuperAG#eXtendedRangeXR) regarding the Atheros XR features.

## Power and antenna gain

In order to communicate over longer distances the stock antennas in your laptop and on your access point are unlikely to be sufficient! Some of the options are to purchase higher power radios, to add external bidirection amplifiers, to use high gain antennas, and to do all the above! Note that there is no magic with antennas: in order to get gain you have to get directionality as well. This is pretty obvious if you're looking at dish antennas but it's also the case when looking at high gain omnidirectional antennas! For example, if you buy a 9dBi omni (for operation at 2.4Ghz) you are getting an antenna that is uniform (omni) 360 degrees around, but that has a vertical beamwidth of only about 14 degrees. What this means is that if you mount your antenna vertically, you get 3dB less gain (i.e. only 6dB gain) if you are 7 degrees up or down from horizontal with respect to the antenna. So if you are mounting the antenna in your attic and expect miracles 2 floors down, or if you are dreaming of putting the antenna on that mountain peak next door, think again... (There are onmis with an "electrical downtilt" specifically made for high locations.) For some antenna inspirations take a look at <http://www.pacwireless.com/> and <http://www.hyperlinktech.com/>.

One very handy tool as you are pondering over power and gain is a link analysis calculator. A free one is available at <http://www.hol4g.com/rl/TS/hwdt.aspx> and will give you a first estimate on how much signal to noise margin you can expect. If you arrive at less than 10dB you shouldn't expect much in terms of bandwidth and count yourself lucky if you can associate at all. You probably need more than 16dB to get into the higher bit rates (say 12Mps and above).

Take a look at [UserDocs/AntennaDiversity](http://madwifi-project.org/wiki/UserDocs/AntennaDiversity). You'll want to disable transmit diversity if you only have one antenna connected.

## ACK timeout and Slot time

In 802.11a/b/g all data transmissions are acknowledged by the receiving radio and the transmitter makes a number of retransmission attempts if such an ack is not received. (Note that there are ways to send unacknowledged packets using multicast or multimedia features.) The acknowledgments affect long distance links in that the transmitter waits for a limited amount of time before retrying. If the ACK timeout is set too short, the transmitter will start retransmitting before an ACK could have possibly been received and this retransmission may well actually interfere with an ACK that is "on it's way".(It is important to note that this retransmission will occur after a random backoff) The end result is that actual throughput is very low and the number of retransmissions is excessively high. If, conversely, the ACK timeout is set too long, the transmitter waits unncesessarily long before retransmitting in the case no ACK is received. This represents lost time and thus reduces the throughput of the link.

In addition to the ACK timeout, there are a number of other time constants that need to be adjusted for long distance links. These time constants have to do with the collision sensing and avoidance parts of the protocol.

The bottom line is that you need to determine the distance between the radios (or the maximum distance in the case of a mobile installation), calculate the time of flight of the packets in microseconds, and then set the ACK timeout to a little more than a round-trip time as the CTS timeout as well as the Slot time to the one-way time. These settings are available in /proc/sys/dev/wifiX as slottime, ctstimeout, and acktimeout. The easiest way to change these settings is using the athctrl utility provided with the driver. For example, athctrl -d 15000 sets these parameters appropriately for stations located 15000 meters apart (approx 9.4 miles). Note that it is important that all stations that are communicating with each other use the same value. So if you have an access point in a point-to-multipoint set-up where one client is 10000 meters away and the other is 15000 meters away then you should run athctrl -d 15000 on all three nodes.

One little problem with the slot time is that 802.11g requires it to be switched between 9us and 20us depending on whether a 802.11b client is associated or not (or something like that). You will thus see the slot time suddenly be reset to one of these values if any association operation takes place on your access point. There is a patch in the works to lock the slot time to what you set it. This description will be updated when that goes into the code base...

## Performance expectations and measurements

The first measurement you are likely to do is to look at the SNR (signal to noise ratio) or quality values displayed by iwconfig, iwlist, and athstats. What the values mean depends on the driver and differs from chipset to chipset. For [MadWifi](http://madwifi-project.org/wiki/MadWifi) there is only a single value that means anything and that's the signal-to-noise or quality value. This value is in dB above the noise floor. That means the SNR or quality or rssi values reported by the various applications are all derived from so-called rssi (received signal strength indication) values placed into the tx/rx descriptors by the chipset and retrieved by the driver.

How about the signal strength and noise floor values you may see? Well, the hardware only returns rssi/SNR measurements and the driver simply sets the noise floor value returned to various tools to a constant -95dB. From that some tools calculate signal strength to be noise floor + signal to noise. So, in other words, if you are using [MadWifi](http://madwifi-project.org/wiki/MadWifi) you may as well only look at SNR and ignore all other values as they don't contain any additional information. Comments in the driver further explain: "If you assume that the noise floor is -95, which is an excellent assumption 99.5 % of the time, then you can derive the absolute signal level (i.e. -95 + rssi). There are some other slight factors to take into account depending on whether the rssi measurement is from 11b, 11g, or 11a. These differences are at most 2db and can be documented."

## The athctrl tool

As mentioned in the [ACK timeout and Slottime](http://madwifi-project.org/wiki/UserDocs/LongDistance#ACKtimeoutandSlottime) section, athctrl tool is the primary method for tweaking the link distance related settings on the Atheros radios. It will adjust the CTS and ACK timeout values based upon the distance between your links.

The following example sets first Atheros radio to a distance of 22531 meters or 14 miles. Please notice the differences in usage for athctrl between [MadWifi](http://madwifi-project.org/wiki/MadWifi) and [MadWifi](http://madwifi-project.org/wiki/MadWifi)-Old due to the old interface naming scheme.

athctrl -i wifi0 -d 22531

*For* [*MadWifi*](http://madwifi-project.org/wiki/MadWifi)*-Old:*

athctrl -i ath0 -d 22531

**Note**: The formula used for the calculation of the tuned parameters is not fully correct. It might be that you have to give a slightly different (larger or shorter) distance in order to get optimal results.

MORE

## The law

### FCC power limits

See [www.fcc.gov](http://www.fcc.gov)

MORE

### Australia

Power in 2.4GHz band is limited to 4W EIRP. There are 13 channels; see <http://www.melbpc.org.au/pcupdate/2404/2404article6.htm>

### Europe

### Belgium

Power is limited to 100mw p.i.r.e with the antenna given with your hardware. Your network is limited to 300m.   
You have to buy a licence to [IBPT](http://www.ibpt.be) if you want more: 850 Euro for a private network (only once) You have to declare to the IBPT what hardware you use. You also have to notice them when you change or add hardware.

### Italy

<http://www.comunicazioni.it/it/index.php?IdPag=699> (in Italian)   
<http://www.comunicazioni.it/it/index.php?IdPag=1039> (in Italian)

### France

<http://www.arcep.fr/index.php?id=8126> on 2.4Ghz band (in french)   
<http://www.arcep.fr/index.php?id=8125> on 5Ghz band (in french)

### Switzerland

<http://www.bakom.admin.ch/themen/technologie/01223/> (2.4 and 5 GHz bands; in multiple languages)

### México

<http://www.cofetel.gob.mx> (In Spanish)   
  
MORE

# Antenna Diversity

Antenna diversity is the term used when the receiver and transmitter use multiple antennas in order to improve the quality of reception/transmission. Many Atheros radios have two antenna connectors which allows you to connect two antennas (NB: [MadWifi](http://madwifi-project.org/wiki/MadWifi) does not yet support MIMO, the 3+ antenna array devices). Two antennas can be beneficial in a number of scenarios. One typical scenario is called polarization diversity. For example, most laptops have two small dipole antennas oriented differently because as you move around with the laptop one or the other antenna may "line up" better with the access point antenna polarization. There is also spatial diversity which exploits the fact that sometimes two antennas spaced more than a few (~10) wavelengths apart have different reception conditions due to reflections or fading, or other effects.

## Receiver diversity

The way receive antenna diversity works is that the radio hardware listens to the begining of the incoming transmission (the preamble stuff) and compares the signal strength on both antennas. It then selects the stronger antenna and receives the body of the packet on that antenna. All this is performed in hardware, the driver can only enable/disable diversity, and if it's disabled it can select which antenna to receive on.

## Transmitter Diversity

Transmitter diversity works differently in that the driver needs to tell the radio which antenna to use. In madwifi-old the driver logic was to start transmitting to a station on the default antenna (lowest numbered) and then keep track of the receiving antenna for packets received from that station. If three consecutive packet receptions from a station occur on "the other" antenna then the driver changes the transmit antenna to match the receive antenna. The operation of madwifi-ng is probably the same, but this has not been confirmed.

Apparently the switch that selects which antenna transmits tends to have about a 10dB-15dB attenuation on the non-transmitting antenna. So there is some power that goes to that non-transmitting antenna. Also, if you see fluctuations in output power levels of about that amount on one antenna, then it may be diversity kicking in.

You will definitely want to disable transmitter diversity **if you only have one antenna connected**. Otherwise 50% of your broadcast and multicast packets (ARP, OSPF) will go out on the wrong antenna. (See [r1430](http://madwifi-project.org/changeset/1430), [~~#326~~](http://madwifi-project.org/ticket/326), [~~#168~~](http://madwifi-project.org/ticket/168).)

## Controlling diversity

Antenna diversity can be controlled using sysctl (/proc/sys/dev/wifiX/{diversity,txantenna,rxantenna}):

diversity:

0: no

1: yes

txantenna and rxantenna:

0: auto

1: antenna 1

2: antenna 2

Which connector corresponds to antenna 1 and which to antenna 2 depends on the particular radio hardware. For the popular Wistron CM-9 [MiniPCI](http://madwifi-project.org/wiki/UserDocs/MiniPCI) the connector close to the corner of the card is antenna 1, but contrarywise another popular Wistron CM-10 (DCMA-81)[MiniPCI](http://madwifi-project.org/wiki/UserDocs/MiniPCI) (half-size) has connector close to the inside of the card is antena 1 and close to the corner of the card is antenna 2. Do not mix up these CM´s cards!

For example, to disable diversity and select antenna 1 for receive and transmit you'd do the following:

sysctl -w dev.wifi0.diversity=0

sysctl -w dev.wifi0.txantenna=1

sysctl -w dev.wifi0.rxantenna=1

To save these settings, add them to /etc/sysctl.conf.

If your distribution doesn't include a sysctl.conf file, here's a sh snippet to run from /etc/rc.local:

#!/bin/sh

# sysc\_atha.sh

# Usage: sh sysc\_atha.sh [tx rx div]

#

t=`cat /proc/sys/dev/wifi0/txantenna`

r=`cat /proc/sys/dev/wifi0/rxantenna`

d=`cat /proc/sys/dev/wifi0/diversity`

t=${1:-$t}

r=${2:-$r}

d=${3:-$d}

(sysctl -w dev.wifi0.txantenna=$t

sysctl -w dev.wifi0.rxantenna=$r

sysctl -w dev.wifi0.diversity=$d

)|tr '\n' ' '|sed s/dev.wifi0.//g

echo

e.g.:

sh sysc\_atha.sh 2 2 0

txantenna = 2 rxantenna = 2 diversity = 0

## What doesn't work

Note that connecting antennas with different coverage areas doesn't really work. When reading about multiple antennas some people think they can connect one high-gain long range antenna for a long distance link and one omnidirectional antenna to serve for local connectivity to a single accesspoint radio. Or one high gain pointing in one direction and a second pointing in a different direction. While this can be made to work in principle, there are a number of problems. The first is that on an accesspoint beacons are only sent out one antenna, so stations on the other antenna are out of luck. The second problem is that when the AP transmits the stations on the other antenna don't hear the transmission and therefore don't do the collision avoidance properly. If you use RTS/CTS then also some stations don't get them at all. The bottom line is that diversity works well when one antenna is a few dB better than the other, which allows higher data rates and fewer packet losses. It does not work well if half the stations only hear one antenna and not at all the other.

## Configuring [kismet](http://madwifi-project.org/wiki/UserDocs/kismet) with madwifi

This page only addresses the [**kismet**](http://madwifi-project.org/wiki/UserDocs/kismet)**.conf** configurations for the *source* parameter, see the [kismet](http://madwifi-project.org/wiki/UserDocs/kismet) docs for general instructions.

You **MUST** have the latest development code, stable [kismet](http://madwifi-project.org/wiki/UserDocs/kismet) only supports [MadWifi](http://madwifi-project.org/wiki/MadWifi)-Old.

If you don't have it, you can grab latest development code with:

svn co http://svn.kismetwireless.net/code/trunk kismet-devel

Kismet-devel will create its own VAP. You may want to remove all prior VAPs before launching [kismet](http://madwifi-project.org/wiki/UserDocs/kismet).   
Edit [kismet](http://madwifi-project.org/wiki/UserDocs/kismet).conf and search for source:

source=madwifi\_g,wifi0,AtherosG

See <http://www.kismetwireless.net/documentation.shtml#readme>, or the README from your svn download for the most current information.

Kismet supports monitoring A, B, or G channels using one of the following sources:

* madwifi\_a
* madwifi\_ab
* madwifi\_ag
* madwifi\_g
* madwifi\_b

Be sure your capture interface is **wifiX** (the base device). NB: The [MadWifi](http://madwifi-project.org/wiki/MadWifi)-Old interface is athX.

Latest SVN builds of Kismet-devel, as of 31 March 2006, autodetects whether madwifi-old, or madwifi-ng is used. Both old style and ng style capturesource statements in the [kismet](http://madwifi-project.org/wiki/UserDocs/kismet).conf file are supported.

## And with [kismet](http://madwifi-project.org/wiki/UserDocs/kismet)-newcore??

Somebody wrote how to use [kismet](http://madwifi-project.org/wiki/UserDocs/kismet)-newcore, lorcon and madwifi together. A little bit complicated, but then again ...

<http://www.757.org/~joat/wiki/index.php/Kismet_NewCore>

# Snort

[Snort](http://madwifi-project.org/wiki/UserDocs/Snort) with [MadWifi](http://madwifi-project.org/wiki/MadWifi) driver on a Atheros CM9 miniPCI wireless interface

See snort-wireless <http://www.snort-wireless.org/> for more information.

ifconfig ath0 down

wlanconfig ath0 destroy

wlanconfig ath0 create wlandev wifi0 wlanmode monitor

ifconfig ath0 up

snort -devw -i ath0

Be sure you see beacon from your ap, then configure your **snort.conf** configuration file (configure --enable-wireless) and invoke

*snort -c /your/path/to/snort.conf*

NB: [Snort](http://madwifi-project.org/wiki/UserDocs/Snort)-2.4.3 may have some problems.

# Disabling Background Scanning in Client Mode

In a dedicated link topology where you never want your client to roam to a different access point, you will want to disable client background scanning. In a bridged backhaul topology it was found that this significantly increased total throughput and possibly stability. Basically, by default madwifi clients will stop passing traffic temporarily to do a background scan of available access points. So this is unnecessary and undesirable in a dedicated link topology.

The results of disabling this functionality were extremely positive in allowing better bi-directional and single direction throughputs.

To disable it:

iwpriv ath0 bgscan 0

Compile time

If the home connection used is opened when the computer is turned on and closed when the computer is shuted down, it should be better to keep always the driver in the kernel by compilling the drivers directly in it: see /madwifi-0.9.4/patches/README. Install (install.sh) the drivers and recompile the kernel with just the Default transmission rate: Sample (kernel: Device Drivers - Network device support - Wireless LAN - Atheros PCI/Cardbus cards). And leave the others options unchecked.

# [iwpriv](http://madwifi-project.org/wiki/UserDocs/iwpriv) extensions

There are many private ioctls available in MADWifi, here are a few useful ones.

## Locking to a Specific Mode

To lock the card to a specific mode, use:

* iwpriv ath0 mode 11a To lock to 11a only.
* iwpriv ath0 mode 11b To lock to 11b only.
* iwpriv ath0 mode 11g To lock to 11g only.
* iwpriv ath0 mode 0 (default) autoselect mode.

## Changing Authentication Mode.

Use:

* iwpriv ath0 authmode 1 To use open authentication.
* iwpriv ath0 authmode 2 To use shared key authentication.
* iwpriv ath0 authmode 3 To use 802.1x authentication.

## Manipulating the MAC white/black list

Use:

* iwpriv ath0 maccmd 3 To clear the MAC list.
* iwpriv ath0 maccmd 1 To make the list a whitelist.
* iwpriv ath0 maccmd 2 To make the list a blacklist.
* iwpriv ath0 addmac 00:11:22:33:44:55 To add a mac address to the list.
* iwpriv ath0 delmac 00:11:22:33:44:55 To delete a mac from the list.
* iwpriv ath0 kickmac 00:11:22:33:44:55 To send a disassociation frame to an associated station.

## Others

For some hints on other value controllable with [iwpriv](http://madwifi-project.org/wiki/UserDocs/iwpriv), see the source files [net80211/ieee80211\_ioctl.h](http://madwifi-project.org/browser/madwifi/trunk/net80211/ieee80211_ioctl.h) around line 550 and [net80211/ieee80211\_wireless.c](http://madwifi-project.org/browser/madwifi/trunk/net80211/ieee80211_wireless.c) around line 5000. There are short descriptions in the comments. Perhaps someone can transfer this cleanly to the wiki.

[Using iwpriv with a Ubiquity SuperRange Cardbus](http://madwifi-project.org/wiki/UbuquitySRCBiwpriv)

# ath\_info

ath\_info is a tool to get detailled information about the hardware. Since the output of lspci and the dmesg information provided by the HAL is not always sufficient to correctly identify chipset versions we appreciate ath\_info output to be attached to bug reports, for both [ath5k](http://madwifi-project.org/wiki/About/ath5k) and madwifi - especially the first few lines which identify the chipsets are important:

Here is an example:

# ath\_info 0xa0010000

-==Device Information==-

MAC Version: 5212 (0x50)

MAC Revision: 5213 (0x56)

Device type: 3

5Ghz PHY Revision: 5111 (0x17)

2Ghz PHY Revision: 2111 (0x23)

/============== EEPROM Information =============\

| EEPROM Version: 3.4 | EEPROM Size: 16 kbit |

| EEMAP: 0 | Reg. Domain: 0x00 |

|================= Capabilities ================|

| 802.11a Support: yes | Turboa disabled: no |

| 802.11b Support: yes | Turbog disabled: no |

| 802.11g Support: yes | 2GHzXR disabled: no |

| RFKill Support: no | 5GHzXR disabled: no |

| 32KHz Crystal: no | |

\===============================================/

There is more output which helps us in the development of [ath5k](http://madwifi-project.org/wiki/About/ath5k):

/=========================================================\

| Calibration data common for all modes |

|=========================================================|

| CCK/OFDM gain delta: 00 |

| CCK/OFDM power delta: 15 |

| Scaled CCK delta: 00 |

| 2Ghz Antenna gain: 00 |

| 5Ghz Antenna gain: 00 |

| Turbo 2W maximum dbm: 32 |

| Target power start: 000 |

| EAR Start: 000 |

\=========================================================/

/=========================================================\

| Calibration data for 802.11a operation |

|=========================================================|

| I power: 0x00 | Q power: 0x00 |

| Use fixed bias: 0x00 | Max turbo power: 0x20 |

| Max XR power: 0x00 | Switch Settling Time: 0x2d |

| Tx/Rx attenuation: 0x07 | TX end to XLNA On: 0x02 |

| TX end to XPA Off: 0x00 | TX end to XPA On: 0x0e |

| 62db Threshold: 0x0f | XLNA gain: 0x0d |

| XPD: 0x01 | XPD gain: 0x0d |

| I gain: 0x0a | Tx/Rx margin: 0x00 |

| False detect backoff: 0x07 | Noise Floor Threshold: -54 |

| ADC desired size: -32 | PGA desired size: -80 |

|=========================================================|

| Antenna control 0: 0x00 | Antenna control 1: 0x01 |

| Antenna control 2: 0x22 | Antenna control 3: 0x02 |

| Antenna control 4: 0x22 | Antenna control 5: 0x02 |

| Antenna control 6: 0x02 | Antenna control 7: 0x21 |

| Antenna control 8: 0x01 | Antenna control 9: 0x21 |

| Antenna control 10: 0x01 | Antenna control 11: 0x00 |

|=========================================================|

| Octave Band 0: 1 | db 0: 2 |

| Octave Band 1: 1 | db 1: 2 |

| Octave Band 2: 1 | db 2: 2 |

| Octave Band 3: 1 | db 3: 2 |

\=========================================================/

/=================== Per channel power calibration ====================\

| Freq | pwr\_0 | pwr\_1 | pwr\_2 | pwr\_3 |pwrx3\_0|pwrx3\_1|pwrx3\_2|max\_pwr|

| | pcdac | pcdac | pcdac | pcdac | pcdac | pcdac | pcdac | |

|======|=======|=======|=======|=======|=======|=======|=======|=======|

| 5230 | 0.00 | 63.01 | 20.01 | 51.00 | 18.01 | 34.02 | 41.00 | 51.00 |

| | [167] | [198] | [222] | [227] | [20] | [35] | [63] | |

|======|=======|=======|=======|=======|=======|=======|=======|=======|

| 5170 | 0.01 | 63.01 | 21.01 | 35.00 | 22.01 | 34.02 | 42.00 | 35.00 |

| | [168] | [168] | [195] | [204] | [20] | [35] | [63] | |

|======|=======|=======|=======|=======|=======|=======|=======|=======|

| 5400 | 0.02 | 63.01 | 22.01 | 19.02 | 22.01 | 35.02 | 42.00 | 19.02 |

| | [233] | [233] | [260] | [269] | [20] | [35] | [63] | |

|======|=======|=======|=======|=======|=======|=======|=======|=======|

| 5300 | 0.02 | 63.01 | 21.01 | 19.01 | 18.01 | 35.02 | 41.00 | 19.01 |

| | [168] | [169] | [198] | [211] | [20] | [35] | [63] | |

|======|=======|=======|=======|=======|=======|=======|=======|=======|

| 5650 | 8.02 | 63.01 | 23.01 | 36.00 | 18.01 | 34.02 | 40.00 | 36.00 |

| | [103] | [134] | [160] | [169] | [20] | [35] | [63] | |

|======|=======|=======|=======|=======|=======|=======|=======|=======|

| 5500 | 8.03 | 63.01 | 23.01 | 4.00 | 14.01 | 33.02 | 39.00 | 4.00 |

| | [38] | [69] | [97] | [110] | [20] | [35] | [63] | |

|======|=======|=======|=======|=======|=======|=======|=======|=======|

| 5825 | 8.02 | 63.01 | 22.01 | 19.03 | 10.01 | 33.02 | 39.00 | 19.03 |

| | [37] | [67] | [93] | [98] | [20] | [35] | [63] | |

|======|=======|=======|=======|=======|=======|=======|=======|=======|

| 5725 | 0.00 | 63.01 | 20.01 | 50.03 | 10.01 | 32.02 | 40.00 | 50.03 |

| | [38] | [69] | [93] | [98] | [20] | [35] | [63] | |

|======|=======|=======|=======|=======|=======|=======|=======|=======|

| 5825 | 0.00 | 63.01 | 20.01 | 50.03 | 10.01 | 32.02 | 40.00 | 50.03 |

| | [38] | [69] | [93] | [98] | [20] | [35] | [63] | |

|======|=======|=======|=======|=======|=======|=======|=======|=======|

| 5825 | 0.00 | 63.01 | 20.01 | 50.03 | 10.01 | 32.02 | 40.00 | 50.03 |

| | [38] | [69] | [93] | [98] | [20] | [35] | [63] | |

\======================================================================/

/=========================================================\

| Calibration data for 802.11b operation |

|=========================================================|

| I power: 0x00 | Q power: 0x00 |

| Use fixed bias: 0x00 | Max turbo power: 0x00 |

| Max XR power: 0x00 | Switch Settling Time: 0x23 |

| Tx/Rx attenuation: 0x0f | TX end to XLNA On: 0x02 |

| TX end to XPA Off: 0x00 | TX end to XPA On: 0x07 |

| 62db Threshold: 0x1c | XLNA gain: 0x0d |

| XPD: 0x01 | XPD gain: 0x0b |

| I gain: 0x0a | Tx/Rx margin: 0x00 |

| False detect backoff: 0x00 | Noise Floor Threshold: -1 |

| ADC desired size: -38 | PGA desired size: -80 |

|=========================================================|

| Antenna control 0: 0x00 | Antenna control 1: 0x08 |

| Antenna control 2: 0x14 | Antenna control 3: 0x04 |

| Antenna control 4: 0x10 | Antenna control 5: 0x00 |

| Antenna control 6: 0x04 | Antenna control 7: 0x18 |

| Antenna control 8: 0x08 | Antenna control 9: 0x10 |

| Antenna control 10: 0x00 | Antenna control 11: 0x00 |

|=========================================================|

| Octave Band 0: 4 | db 0: 4 |

| Octave Band 1: 2 | db 1: 2 |

| Octave Band 2: 0 | db 2: 0 |

| Octave Band 3: 0 | db 3: 0 |

\=========================================================/

/=================== Per channel power calibration ====================\

| Freq | pwr\_0 | pwr\_1 | pwr\_2 | pwr\_3 |pwrx3\_0|pwrx3\_1|pwrx3\_2|max\_pwr|

| | pcdac | pcdac | pcdac | pcdac | pcdac | pcdac | pcdac | |

\======================================================================/

/=========================================================\

| Calibration data for 802.11g operation |

|=========================================================|

| I power: 0x00 | Q power: 0x00 |

| Use fixed bias: 0x00 | Max turbo power: 0x00 |

| Max XR power: 0x00 | Switch Settling Time: 0x2d |

| Tx/Rx attenuation: 0x0f | TX end to XLNA On: 0x02 |

| TX end to XPA Off: 0x00 | TX end to XPA On: 0x0e |

| 62db Threshold: 0x1c | XLNA gain: 0x0d |

| XPD: 0x01 | XPD gain: 0x0b |

| I gain: 0x0a | Tx/Rx margin: 0x00 |

| False detect backoff: 0x00 | Noise Floor Threshold: -1 |

| ADC desired size: -32 | PGA desired size: -80 |

|=========================================================|

| Antenna control 0: 0x00 | Antenna control 1: 0x08 |

| Antenna control 2: 0x14 | Antenna control 3: 0x04 |

| Antenna control 4: 0x10 | Antenna control 5: 0x00 |

| Antenna control 6: 0x04 | Antenna control 7: 0x18 |

| Antenna control 8: 0x08 | Antenna control 9: 0x10 |

| Antenna control 10: 0x00 | Antenna control 11: 0x01 |

|=========================================================|

| Octave Band 0: 3 | db 0: 3 |

| Octave Band 1: 2 | db 1: 2 |

| Octave Band 2: 0 | db 2: 0 |

| Octave Band 3: 0 | db 3: 0 |

\=========================================================/

/=================== Per channel power calibration ====================\

| Freq | pwr\_0 | pwr\_1 | pwr\_2 | pwr\_3 |pwrx3\_0|pwrx3\_1|pwrx3\_2|max\_pwr|

| | pcdac | pcdac | pcdac | pcdac | pcdac | pcdac | pcdac | |

\======================================================================/

GPIO registers: CR 0x00000000, DO 0x00000000, DI 0x00000005

STA\_ID0: fe:f8:36:68:c9:35

STA\_ID1: 0x000035c9, AP: 0, IBSS: 0, KeyCache Disable: 0

TIMER0: 0x00000000, TBTT: 0, TU: 0x00000000

TIMER1: 0x00000000, DMAb: 0, TU: 0x00000000 (0)

TIMER2: 0x00000000, SWBA: 0, TU: 0x00000000 (0)

TIMER3: 0x00000000, ATIM: 0, TU: 0x00000000 (0)

TSF: 0x0000000000000000, TSFTU: 0, TU: 0x00000000

BEACON: 0x003ed7ff

LAST\_TSTP: 0xf6f5df66

Building ath\_info

Now ath\_info is maintained independently from madwifi:

svn co http://madwifi-project.org/svn/ath\_info/trunk ath\_info

cd ath\_info/

make

Using ath\_info

You have to tell ath\_info the memory address it should operate on. This is the memory address of the Atheros card shown by lspci -v. Or just use the following script fragment:

athmem=`lspci -vd 168c: |sed -n 's/.\*Memory at \([^ ]\*\).\*/0x\1/p'`

ath\_info $athmem

Advanced Usage

ath\_info can do more than that. check with ath\_info -h:

ath\_info [-w [-g N:M]] [-v] [-f] [-d] [-R addr] [-W addr val] <base\_address> [<name1> <val1> [<name2> <val2> ...]]

-w write values into EEPROM

-g N:M set GPIO N to level M (only used with -w)

-v verbose output

-f force; suppress question before writing

-d dump eeprom (file 'ath-eeprom-dump.bin' and screen)

-R <addr> read register at <addr> (hex)

-W <addr> <val> write <val> (hex) into register at <addr> (hex)

<base\_address> device base address (see lspci output)

- read info:

ath\_info <base\_address>

- set regdomain to N:

ath\_info -w <base\_address> regdomain N

- set a PCI id field to value N:

ath\_info -w <base\_address> <field> N

where <field> is on of:

pci\_dev\_id pci\_vendor\_id pci\_class pci\_rev\_id pci\_subsys\_dev\_id pci\_subsys\_vendor\_id regdomain

You may need to set a GPIO to a certain value in order to enable

writing to the EEPROM with newer chipsets, e.g. set GPIO 4 to low:

ath\_info -g 4:0 -w <base\_address> regdomain N

DISCLAIMER: The authors are not responsible for any damages caused by

this program. Writing improper values may damage the card or cause

unlawful radio transmissions!

# attachment

## Creating a Monitor Mode Interface

This document does not apply to "old" versions of [MadWifi](http://madwifi-project.org/wiki/MadWifi) (Prior to November 2005).

NOTE: because a station VAP is created by default, [wlanconfig](http://madwifi-project.org/wiki/UserDocs/wlanconfig) will fail to create an AP VAP (for [MadWifi](http://madwifi-project.org/wiki/MadWifi) releases >1407) unless the parameter '[autocreate](http://madwifi-project.org/wiki/UserDocs/autocreate)=none' is passed when loading the ath\_pci module. See [UserDocs/autocreate](http://madwifi-project.org/wiki/UserDocs/autocreate) for details.

First

wlanconfig ath0 destroy

And then to create an interface (called *ath0*) in monitor mode, issue the command:

wlanconfig ath0 create wlandev wifi0 wlanmode monitor

Or,

wlanconfig ath create wlandev wifi0 wlanmode monitor

The only difference is the lack of '0' in the second command. The kernel will give the interface the lowest free integer as its suffix. More information and examples can be found in *man* [*wlanconfig*](http://madwifi-project.org/wiki/UserDocs/wlanconfig) *8*, and [UserDocs/MonitorMode](http://madwifi-project.org/wiki/UserDocs/MonitorMode).

To finish bring the interface up

ifconfig ath0 up

By default, monitor mode receives packets with prism2 headers prepended on them. To change this, you must run one of the following:

* Only 802.11 headers: echo '801' > /proc/sys/net/ath0/dev\_type
* Prism2 headers: echo '802' > /proc/sys/net/ath0/dev\_type
* Radiotap headers: echo '803' > /proc/sys/net/ath0/dev\_type
* Atheros Descriptors: echo '804' > /proc/sys/net/ath0/dev\_type

NOTE: tcpdump and related programs may be unable to use filters such as "wlan src" on packets with extra headers. There is a program "prism-strip" which will strip Prism2 headers from a capture file, enabling filters to be used on an existing file if the capture cannot be repeated with only 802.11 headers

FreeRADIUS/WinXP Authentication Setup  
  
This post describes how to build a FreeRADIUS server for TLS and PEAP authentication, and how to configure the Windows XP clients (supplicants). The server is configured for a home (or test) network.  
  
Three papers have been written about TLS authentication with a FreeRADIUS server:  
  
1) www.missl.cs.umd.edu/wireless/eaptls  
2) www.freeradius.org/doc/EAPTLS.pdf  
3) www.denobula.com  
  
These papers provide an excellent background, but are somewhat out of date. Where appropriate, I will simply refer to these documents rather than repeating the information. I recommend that you follow the steps I give below rather than the steps in these documents.  
  
In the steps below, I give examples from the FreeRADIUS server that I installed yesterday in my Red Hat 9 computer. If you follow this example, please make the needed changes to the names of the files. I installed the FreeRADIUS and OpenSSL files in special local directories. This ensures that there is no interaction between the base Linux files and the new files. It also allows you to easily remove all of the newly installed files.  
  
One word of caution: Be prepared for some frustration. The FreeRADIUS and OpenSSL snapshots used in constructing the server are beta software. Don't be surprised if you encounter some problems.  
  
1. Download and Install OpenSSL and FreeRADIUS   
  
The first step is to download and install the latest snapshot versions of OpenSSL and FreeRADIUS.  
  
a. OpenSSL -- Download the latest OpenSSL-0.9.7-stable snapshot. I downloaded the OpenSSL snapshot to my home directory. The snapshots are located at:  
  
»ftp://[ftp.openssl.org/snapshot/](ftp://ftp.openssl.org/snapshot/)  
  
Then I used the following nine steps:  
  
mkdir -p /usr/src/802/openssl   
cd /usr/src/802/openssl  
cp /home/jbibe/openssl-0.9.7-stable-SNAP-20040202.tar.gz \  
openssl-0.9.7-stable-SNAP-20040202.tar.gz  
  
gunzip openssl-0.9.7-stable-SNAP-20040202.tar.gz  
tar xvf openssl-0.9.7-stable-SNAP-20040202.tar  
cd openssl-0.9.7-stable-SNAP-20040202  
  
./config shared --prefix=/usr/local/openssl  
make  
make install  
  
That completes the work with OpenSSL, except for building the required certificates.  
  
When you perform the config, make, and make-install here and in the FreeRADIUS install described below, I recommend that you log the information. For example, instead of using the simple "make" command, use:  
  
make > mymake.log 2>&1  
  
If you encounter problems, you can review mymake.log (or myconfig.log, or myinstall.log) for errors.  
  
b. FreeRadius -- Download the latest FreeRADIUS snapshot. Again, I downloaded the file to my home directory. The snapshot is located at:  
  
»ftp://[ftp.freeradius.org/pub/radius/CVS-snapshots/](ftp://ftp.freeradius.org/pub/radius/CVS-snapshots/)  
  
Then I used the following nine steps:  
  
mkdir -p /usr/src/802/radius  
cd /usr/src/802/radius  
cp /home/jbibe/freeradius-snapshot-20040203.tar.gz \  
freeradius-snapshot-20040203.tar.gz  
  
gunzip freeradius-snapshot-20040203.tar.gz  
tar xvf freeradius-snapshot-20040203.tar  
cd freeradius-snapshot-20040203  
  
./configure --with-openssl-includes=/usr/local/openssl/include \  
--with-openssl-libraries=/usr/local/openssl/lib \  
--prefix=/usr/local/radius  
make  
make install  
  
That completes the work with FreeRADIUS, except for building certificates, making the changes to the FreeRADIUS configuration files, moving the server certificates to their final location, and building a wrapper for radiusd.  
  
2. Produce Certificates  
  
Server and client certificates are needed for TLS and PEAP. To produce the required certificates, I recommend that you use CA.all that is included with FreeRADIUS. CA.all uses the configuration information in openssl.cnf.  
  
a. openssl.cnf -- Update openssl.cnf for your configuration. The configuration file is located at:  
  
/usr/local/openssl/ssl  
  
A portion of the information from my openssl.cnf is given below. (The company information is does not describe an actual company located in Brentwood, TN.) Note that the configuration information includes the password "whatever". It is the certificate password.  
  
When CA.all executes, it uses this information three times. The first pass through this information produces the root certificates. If you set up your configuration as shown below, you will be able to accept all of the settings in the first pass. The second pass through this information produces the client certificates. You only need to change the commonName to the client name. In my case, I changed the commonName to jbibe. The third pass through this information produces the server certificates. You only need to change the commonName to the server name. In my case, I changed the commonName to micron.  
  
----- Example -------------------------------------------

...  
# req\_extensions = v3\_req  
   
# The extensions to add to a certificate request  
   
[ req\_distinguished\_name ]  
   
countryName = Country Name (2 letter code)  
countryName\_default = US  
countryName\_min = 2  
countryName\_max = 2  
   
stateOrProvinceName = State or Province Name (full name)  
stateOrProvinceName\_default = Tennessee  
   
localityName = Locality Name (eg, city)  
localityName\_default = Brentwood  
   
0.organizationName = Organization Name (eg, company)  
0.organizationName\_default = Helava  
   
organizationalUnitName = Organizational Unit Name  
organizationalUnitName\_default = Engineering  
   
commonName = Common Name (eg, YOUR name)  
commonName\_max = 64  
commonName\_default = HAI  
   
emailAddress = Email Address  
emailAddress\_max = 40  
emailAddress\_default = ohb@cmcast.net  
   
# SET-ex3 = SET extension number 3  
   
[ req\_attributes ]  
   
challengePassword = A challenge password  
challengePassword\_min = 4  
challengePassword\_max = 20  
challengePassword\_default = whatever  
   
unstructuredName = An optional company name

---------------------------------------------------------   
  
b. CA.all -- Update the CA.all script for your requirements. The file is located at:  
  
/usr/src/802/radius/freeradius-snapshot-20040203/scripts  
  
If you use the default password "whatever", you only need to verify that the path in the script points to the installed openssl information. No changes should be necessary, but there is one gotcha. At about line 30, the path will probably be in error. Look for the following line and update the path as needed.   
  
echo "newreq.pem" | /usr/local/openssl/ssl/misc/CA.pl -newca  
  
When CA.all executes, it produces nine certificates:  
  
root.pem, root.p12, root.der  
cert-clt.pem, cert-clt.p12, cert-clt.der  
cert-srv.pem, cert-srv.p12, cert-srv.der  
  
For TLS and PEAP, the server needs root.pem and cert-srv.pem. For TLS, the Windows XP client needs root.der and cert-clt.p12. For PEAP, the Windows XP client needs root.der.  
  
In the event that you want to use TLS authentication with multiple clients, Document 3 provides the needed script. Look for the CA.clt script in Section 6.  
  
3. Configure Server for TLS  
  
There are only a few changes and additions needed for TLS authentication. The clients.conf, users, and radiusd.conf are located at:  
  
/usr/local/radius/etc/raddb  
  
a. clients.conf -- This file contains the basic configuration for the Access Point. Look for the following line then uncomment and modify as appropriate:  
  
#client 192.168.0.0/24 {

client 192.168.1.0/24 {  
 secret = AP\_Shared\_Secret  
 shortname = WLAN  
}

b. users -- This file contains the basic user information. Look for the following line and then add the user name:  
  
#"John Doe" Auth-Type := Local, User-Password == "hello"  
#  
  
jbibe  
  
Note that for TLS, you should not include an Auth-Type or a password. The server is able to determine the correct Auth-Type, and a password is not needed because the client uses a client certificate for authentication.  
  
c. radiusd.conf -- This file contains the server configuration information. Look for the following lines and then change the default\_eap\_type from md5 to tls:

eap {  
 default\_eap\_type = md5

Change md5 to tls.  
  
Move down to the following line, and then uncomment and modify the information, as shown below. Note that I placed the server certificates, dh file and random file in a new directory 1x on my system. Modify the path as needed for your server:  
  
#tls {

tls {  
 private\_key\_password = whatever  
 private\_key\_file = /usr/local/radius/etc/1x/cert-srv.pem  
 certificate\_file = /usr/local/radius/etc/1x/cert-srv.pem  
 CA\_file = /usr/local/radius/etc/1x/root.pem  
 dh\_file = /usr/local/radius/etc/1x/dh  
 random\_file = /usr/local/radius/etc/1x/random  
 fragment\_size = 1024  
 include\_length = yes  
}

No other changes are needed in radiusd.conf for TLS.  
  
d. Server Certificates, DH File, and Random File -- I added a new directory 1x in the radius etc directory, and then copied the server certificates (root.pem and cert-srv.pem) into the directory. Finally, I used the following trick to produce dh and random:  
  
date > dh  
date > random  
  
If you prefer, use your keyboard to enter some random characters in these files. Or even better, use the OpenSSL tools to produce the random information for these files.  
  
e. Run-Radius -- The only server addition remaining is wrapper for radiusd. I added a new file run-radius in the /usr/local/radius/sbin directory. The script is from Document 3:  
  
----- Wrapper Script ------------------------------------  
#!/bin/sh -x  
  
LD\_LIBRARY\_PATH=/usr/local/openssl/lib   
LD\_PRELOAD=/usr/local/openssl/lib/libcrypto.so  
  
export LD\_LIBRARY\_PATH LD\_PRELOAD  
  
/usr/local/radius/sbin/radiusd $@  
---------------------------------------------------------  
  
After entering and saving the script, make run-radius executable:  
  
chmod u=rwx run-radius  
  
The server is complete.  
  
4. Install Windows XP Certificates and Setup Client for TLS  
  
The Windows XP certificates need to be installed, and client needs to be configured. I recommend that you follow Raymond McKay's example in Document 3, Section 10, XP Client (Supplicant) Setup. When this step is complete, the client is ready.  
  
5. AP Setup  
  
The AP configuration needs to be modified. This is the setup I used with my ZyXEL B-1000v2. (I assume that the B-1000 has been configured previously to use WEP keys and MAC addresses.)  
  
At the wireless 802.1x tab:  
  
Wireless Port Control = Authentication Required  
ReAuthentication Timer = 1800 seconds  
Idle Timeout = 3600 seconds  
Authentication Database = RADIUS only  
Dynamic WEP Key Exchange = 128-bit WEP  
  
At the RADIUS tab for authentication:  
  
Active = Yes  
Server IP = 192.168.1.10  
Port Number = 1812  
Shared Secret = AP\_Shared\_Secret  
  
6. Test TLS  
  
The final step is to test the server. With Windows XP computer off, start the server in the debug mode by entering:  
  
/usr/local/radius/sbin/run-radius -X -A  
  
The server should start, displaying various debug information before it displays:  
  
----- Example --------------------------------------------  
  
Listening on IP address \*, ports 1812/udp and 1813/udp, with proxy on 1814/udp.  
Ready to process requests  
  
----------------------------------------------------------  
  
If you don't see the message, look through the debug information for errors and missing information. If you see this message, start the Windows XP computer.  
  
When the Windows XP starts, you will see various messages and certificates exchanged between the client and the server. If all is well, you should see the client authenticated and the user logged on. The following partial example is from Document 3. It shows the last few lines of a successful authentication:  
  
----- Example ---------------------------------------------  
...  
MS-MPPE-Recv-Key = 0xe032765ca06c052e5fe7c2a7534a4252daec44a08505bdb459d4  
fa81e70390f2221d2b06071eb0625e0ba67452a890909662  
MS-MPPE-Send-Key = 0xe03131ce085bc266127528e749bd4753d3e1702df2d4d8c080351  
380f52eae2c24a9fa78015c24e0d140bcd01b23d6c0cacc  
EAP-Message = "\003\_\000\004"  
Message-Authenticator = 0x00000000000000000000000000000000  
Finished request 5  
Going to the next request  
-----------------------------------------------------------  
  
If you see MS-MPPE-Recv-Key and MS-MPPE-Send-Key, the server authenticated the client. You should be able to surf.  
  
7. Change Server Configuration for PEAP  
  
To change the server for PEAP authentication, only a few changes need to be made.  
  
a. users -- Return to the users file and add the user password:  
  
jbibe User-Password == "My-XP-Password"  
  
b. Radiusd.conf -- Return to the radiusd.conf file and make the following changes:  
  
Change the default\_eap\_type from tls to peap:

eap {  
 default\_eap\_type = peap

Move to the PEAP section below the TLS section and uncomment the following lines:

peap {  
 default\_eap\_type = mschapv2  
}

The server is now ready for PEAP authentication.  
  
8. Change Windows XP for PEAP  
  
On the Wireless Network tab, select the network and click Configure to open the network properties. Then  
  
Select the Authentication tab  
Select Protected EAP on the drop-down list  
Click Properties  
Enable "Validate server certificate"  
In Trusted Root Certification Authorities list, enable the root.der certificate.  
In Select Authentication Method, select "Secured password (EAP-MSCHAPv2)"  
Click Configure  
If desired, enable "Automatically use my Windows logon name and password".  
  
I did not enable "Automatically use my Windows ..." In my HP laptop, the software adds HP\\ before the user name; e.g., HP\\jbibe. If you don't enable this option, windows will ask for your user name and password the first time the laptop tries to connect to the network. The computer will then use the user name and password exactly as entered.  
  
On the original Authentication screen, I disabled the "Authenticate as computer when computer information is available"  
  
Windows XP is now ready for testing.  
  
9. Test PEAP  
  
The final step is to test the server. With Windows XP computer off, start the server in the debug mode by entering:  
  
/usr/local/radius/sbin/run-radius -X -A  
  
The server should start, displaying various debug information. If it displays "Ready to process requests", the server is running. This message is identical to the TLS start message. If you review the debug information, you will see additional messages as peap and mschapv2 start.  
  
If you see the Ready message, start the Windows XP computer. As the client and server communicate, you will see various messages exchanged. If all is well, you should see the client authenticated and the user logged on. Again you will see the MS-MPPE-Recv-Key and the MS-MPPE-Send-Key.  
  
If you review the debug messages, you will see the TLS tunnel being built. Once it is built, you will see verification that messages are passing through the tunnel. Finally, you will see the user authenticated.  
[reply](http://text.dslreports.com/speak/remark,9286052)

[bbarrera](http://text.dslreports.com/profile/226051) @ 5th Feb 12:38AM:  
**Re: FreeRADIUS/WinXP Authentication Setup**  
  
Any particular reason you used CVS snapshot of FreeRadius instead of version 0.9.3 released in November?  
[reply](http://text.dslreports.com/speak/remark,9292093)

[No\_Strings](http://text.dslreports.com/profile/520919) @ 5th Feb 12:48AM:  
**Re: FreeRADIUS/WinXP Authentication Setup**  
  
PEAP is not available in the released version.  
  
Rumblings on the freeradius mailing list indicate that a 1.0 release is fairly close.  
--  
I will not take other people's medicine.  
[reply](http://text.dslreports.com/speak/remark,9292163)

[bbarrera](http://text.dslreports.com/profile/226051) @ 5th Feb 01:10AM:  
**Re: FreeRADIUS/WinXP Authentication Setup**  
  
Thanks. Assume FreeRadius 0.9.3 is OK for TLS. Wasn't planning on using PEAP so I'll give the latest release a try.  
[reply](http://text.dslreports.com/speak/remark,9292310)

[No\_Strings](http://text.dslreports.com/profile/520919) @ 5th Feb 01:13AM:  
**Re: FreeRADIUS/WinXP Authentication Setup**  
  
TLS is indeed included in 0.9.3. Running here.  
--  
I will not take other people's medicine.  
[reply](http://text.dslreports.com/speak/remark,9292325)

[jbibe](http://text.dslreports.com/profile/322587) @ 5th Feb 05:33AM:  
**Re: FreeRADIUS/WinXP Authentication Setup**  
  
If you only need TLS, you can use FreeRadius-0.9.3 and OpenSSL-0.9.7c, both stable releases. Since FreeRadius-0.9.3 does not support "--with-openssl/include=" and "--with-openssl-libraries=", you will need to use:  
  
./configure --prefix=/usr/local/radius  
  
After configure executes, edit the rlm\_eap\_tls Makefile as shown in the Document 3. Then perform  
  
make  
make install  
  
In the users file, you will need to enter the Auth-Type, as follows:  
  
jbibe Auth-Type := EAP  
  
Also, you will need to verify that the information in the Authorization section of radiusd.conf file matches the information in Document 3.  
  
If you follow the How-To advice in Document 3, you do not need to download and install two OpenSSL versions. OpenSSL-0.9.7c will support the creation of the required certificates and the operation of FreeRadius-0.9.3.  
[reply](http://text.dslreports.com/speak/remark,9293037)

[bbarrera](http://text.dslreports.com/profile/226051) @ 5th Feb 09:51AM:  
**Re: FreeRADIUS/WinXP Authentication Setup**  
  
Thanks! :)  
[reply](http://text.dslreports.com/speak/remark,9294102)

[DrTCP](http://text.dslreports.com/profile/117326) @ 5th Feb 07:22PM:  
**Re: FreeRADIUS/WinXP Authentication Setup**  
  
Regarding shared libraries:  
  
If you install openssl under /usr/local/openssl as instructed above and try to run:  
  
/usr/local/openssl/bin/openssl it might fail because of shared library not found.   
  
To solve this I've added:  
  
/usr/local/openssl/lib   
  
to the end of /etc/ld.so.conf and as **root** user (su - ) regenerated ld.so.cache file by executing:  
  
ldconfig -v  
  
It might also be possible to solve the library problems by creating symlinks from /usr/lib/openssl/lib/\* to /lib but I've not tested it.  
[reply](http://text.dslreports.com/speak/remark,9299643)

[No\_Strings](http://text.dslreports.com/profile/520919) @ 5th Feb 07:41PM:  
**Re: FreeRADIUS/WinXP Authentication Setup**  
  
A great tip!  
  
Can you help me find the hair I pulled out trying to work around that?  
--  
Metaphors mixed while you wait.  
[reply](http://text.dslreports.com/speak/remark,9299881)

[jbibe](http://text.dslreports.com/profile/322587) @ 5th Feb 09:14PM:  
**Re: FreeRADIUS/WinXP Authentication Setup**  
  
All of the FreeRADIUS documents recommend installing the OpenSSL files as shown in the initial post. I have always followed their advice, but it might not be the best choice. I have been reviewing the OpenSSL installation recommendations today. According to their information, the OpenSSL files are installed in /usr/local/ssl, unless the user specifies another location. In this latter case, they recommend that the user run config as follows:  
  
./config --prefix=/usr/local --openssldir=/usr/local/openssl  
  
This installs the bin files at /usr/local/bin, the lib files at /usr/local/lib, and the include files at /usr/local/include/openssl. And it installs the OpenSSL configuration files at /usr/local/openssl.  
  
This install recommendation might prevent the problem that you have identified.  
[reply](http://text.dslreports.com/speak/remark,9300785)

[DrTCP](http://text.dslreports.com/profile/117326) @ 5th Feb 11:22PM:  
**Re: FreeRADIUS/WinXP Authentication Setup**  
  
Thanks for passing the recommended way.   
  
I also appreciate all the valueable information you have collected, verified and documented here.

said by [jbibe](http://text.dslreports.com/profile/322587):

This install recommendation might prevent the problem that you have identified.

Yes, it does prevent the problem that way because /lib and /bin/lib and /usr/local/lib are in the default library search path.  
  
However, it might interact with the existing openssl installed on my system. When openssl is installed under /usr/local/openssl the uninstallation is darn easy :) You do not have to hunt individual files among others.  
  
Well yet another way to solve this problem is by defining two environment variables (your 3rd reference).  
  
»[www.impossiblereflex.com/8021x/e···TO.htm#8](http://www.impossiblereflex.com/8021x/eap-tls-HOWTO.htm#8)  
[reply](http://text.dslreports.com/speak/remark,9302080)

[jbibe](http://text.dslreports.com/profile/322587) @ 6th Feb 04:48AM:  
**Re: FreeRADIUS/WinXP Authentication Setup**

said by [DrTCP](http://text.dslreports.com/profile/117326):

However, it might interact with the existing openssl installed on my system.

Alan DeKok, one of the lead designers of FreeRADIUS, is always commenting in the FreeRADIUS mail list about the Linker linking to the wrong version when you have two versions of OpenSSL on your computer. When users encounter a segmentation fault and they have two versions of OpenSSL installed, his usual recommendation is to include the --disable-shared option when building the server. I have never experimented with this option, so I can't comment on its effectiveness.

quote:

When openssl is installed under /usr/local/openssl the uninstallation is darn easy :) You do not have to hunt individual files among others.

I love the ability to remove all the installed information with a single command.  
  
**Edit:** Alan DeKok's suggestion to add the --disable-shared option works. I installed a second FreeRADIUS server, using openssl-SNAP-20040202 and freeradius-snapshot-20040205. I selected these versions because they would probably fail to operate properly. This time I used the OpenSSL recommended config commands as follows:  
  
./config shared --prefix=/usr/local --openssldir=/usr/local/opensnap  
make  
make install  
  
Then I installed FreeRADIUS using these commands:  
  
./configure --with-openssl-includes=/usr/local/includes/openssl \  
--with-openssl-libraries=/usr/local/lib \  
--prefix=/usr/local/radsnap  
make  
make install  
  
After configuring the server, I started the new server in the debug mode. No trouble was encountered until the WinXP client tried to connect. During the authentication, the server crashed with a segmentation error.  
  
I went back, added the --disable-shared option to the ./configure, and ran ./configure, make, and make install again. Once this change was made, the new server operated properly. It authenticates TLS and PEAP with WinXP clients.  
  
I now have two working FreeRADIUS servers on my RH computer. :)  
[reply](http://text.dslreports.com/speak/remark,9303449)

[jbibe](http://text.dslreports.com/profile/322587) @ 6th Feb 05:21AM:  
**Re: FreeRADIUS/WinXP Authentication Setup**  
  
In my post, I used  
  
date > dh  
date > random  
  
to build the required random files. It is better to use the following methods:  
  
openssl gendh >> dh  
dd if=/dev/urandom of=random count=2  
  
These methods were suggested recently in the FreeRADIUS mailing list.  
  
I usually build the server with the simple date method when I am experimenting. Once the server is operational, I use the built-in tools to produce the final dh and random files.  
[reply](http://text.dslreports.com/speak/remark,9303502)

[anon](http://text.dslreports.com/profile/0) @ 6th Feb 09:57AM:  
**msg deleted**  
  
deleted by a moderator  
[reply](http://text.dslreports.com/speak/remark,9304597)

[jamesv](http://text.dslreports.com/profile/782487) @ 21st Feb 12:08AM:  
**Re: FreeRADIUS/WinXP Authentication Setup**  
  
What should the three Common Names be? I thought Windows cared about such things.  
[reply](http://text.dslreports.com/speak/remark,9457635)

[jbibe](http://text.dslreports.com/profile/322587) @ 21st Feb 06:18AM:  
**Re: FreeRADIUS/WinXP Authentication Setup**  
  
You select what is convenient for your application. For example, a homeowner might select the family last name for the root certificates, and the first names of the family members for the cert-clt certificates.  
  
If you are using TLS authentication, WinXP compares the common name in the client certificate (cert-clt.p12) with the login name. If the names don't match, WinXP will ask the user to identify the correct credentials.  
[reply](http://text.dslreports.com/speak/remark,9458960)

[jamesv](http://text.dslreports.com/profile/782487) @ 21st Feb 05:08PM:  
**Re: FreeRADIUS/WinXP Authentication Setup**

said by [jbibe](http://text.dslreports.com/profile/322587):

If you are using TLS authentication, WinXP compares the common name in the client certificate (cert-clt.p12) with the login name. If the names don't match, WinXP will ask the user to identify the correct credentials.

OK. Larry, Moe & Curly would be fine, but for TLS is would be more convenient for "Moe" to be the Windows logon name.  
  
I guess this means 802.11 WPA/Radius won't work unless someone is logged in, at least in the Windows world?  
  
Thanks. I'm working on this right now (FreeBSD).  
[reply](http://text.dslreports.com/speak/remark,9462720)

[jbibe](http://text.dslreports.com/profile/322587) @ 21st Feb 06:12PM:  
**Re: FreeRADIUS/WinXP Authentication Setup**  
  
It is not necessary for the WinXP user to log in. WinXP will ask for the user to identify the correct credentials for TLS (or PEAP) authentication the first time the user tries to connect to the network. Once the user identifies the credentials, anyone with access to that computer will have complete access to the LAN. For a home user, this might be satisfactory.  
[reply](http://text.dslreports.com/speak/remark,9463241)