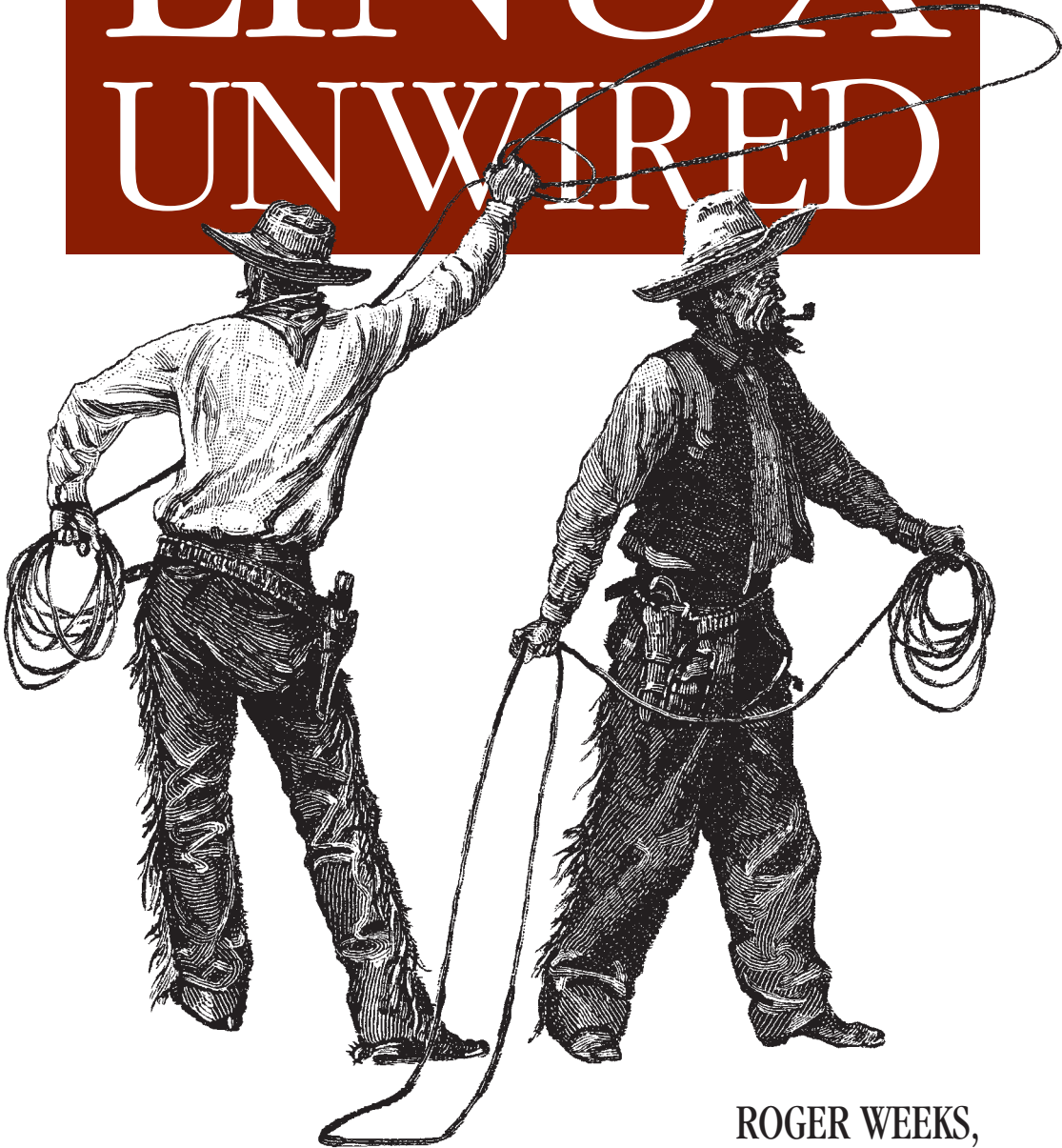


GOING WIRELESS AND LIVING THE UNTETHERED LIFESTYLE

# LINUX UNWIRED



O'REILLY®

ROGER WEEKS,  
EDD DUMBILL & BRIAN JEPSON

# Getting On the Network



Assuming that you didn't encounter any problems in Chapter 2, you should now have a functional wireless network adapter, and the knowledge to configure and use it under Linux. If you have a wireless network set up at home or at work, chances are you will use this network most of the time.

If, however, you have Linux installed on a notebook PC, chances are you're often in transit, and you probably want to find and use wireless networks in cities, airports, hotels, and conferences.

This chapter discusses tools and techniques that allow you to find available wireless networks, whether they are fee-based or free.

## Hotspots

It would be pretty much impossible for any notebook user not to have heard the term *hotspot*. Wireless hotspots are popping up in many locations; coffee shops, airports, hotels, conferences, restaurants, city parks, and libraries are just a few places where you might find a hotspot.

You can easily build your own hotspot, and we cover this in detail in Chapter 6. A hotspot requires at least one access point, a good antenna that covers the needed area, a broadband Internet connection, and some form of access control (if you want to restrict access).

Most hotspots are built around these four basic pieces. Some use DSL as their broadband Internet connection, while many of the commercial hotspots use a T1 line or other dedicated circuit. However, many hotspots are simply in a house or apartment, particularly in dense urban areas, and these connections are DSL, cable, or even simply dial-up.

Before you leave for a trip, research online to find hotspots along the way to your destination. To find both fee-based and free hotspots, consult the following web sites:

*WiFinder*

<http://www.wifinder.com/search.php>

*HotSpotList*

<http://www.hotspotlist.com>

*T-Mobile Hotspots*

<http://www.t-mobile.com/hotspot>

*Wi-Fi Zone Finder*

<http://www.wi-fizone.org/zoneLocator.asp>

*JiWire*

<http://www.jiwire.com>

## Wireless Hotspot Providers

There are an increasing number of commercial hotspot providers, ranging from large companies, such as T-Mobile and WayPort, to small operations in local coffee shops, and wireless aggregators that allow you to access multiple networks from different hotspot providers.

Nearly all of these providers restrict access to their hotspots through a *captive portal*. This form of access control intercepts all TCP/IP traffic. To gain access through a captive portal, simply open a web browser and attempt to navigate to any web page, such as <http://www.oreilly.com>. Your browser traffic is intercepted and redirected to the login screen of the hotspot's portal software. Figure 3-1 shows a typical hotspot login screen.

With commercial hotspot providers, you have a number of payment choices for access. The large operators all offer monthly subscriptions in addition to pay-as-you-go pricing. This is convenient if you don't want to sign up with a specific provider or if you don't travel enough to justify the \$20–40 per month that most monthly subscriptions cost.

If you travel frequently, you may want to sign up with one of the wireless hotspot providers. Deciding which one to use is tricky. It really depends on where you think you may spend the most time. T-Mobile provides access in nearly all Starbucks coffee shops, as well as Borders bookstores, Kinko's copy centers, and many airports. Surf and Sip has neatly taken up many of the non-Starbucks coffee shops in major cities. WayPort is a good choice if you need hotel or airport access.

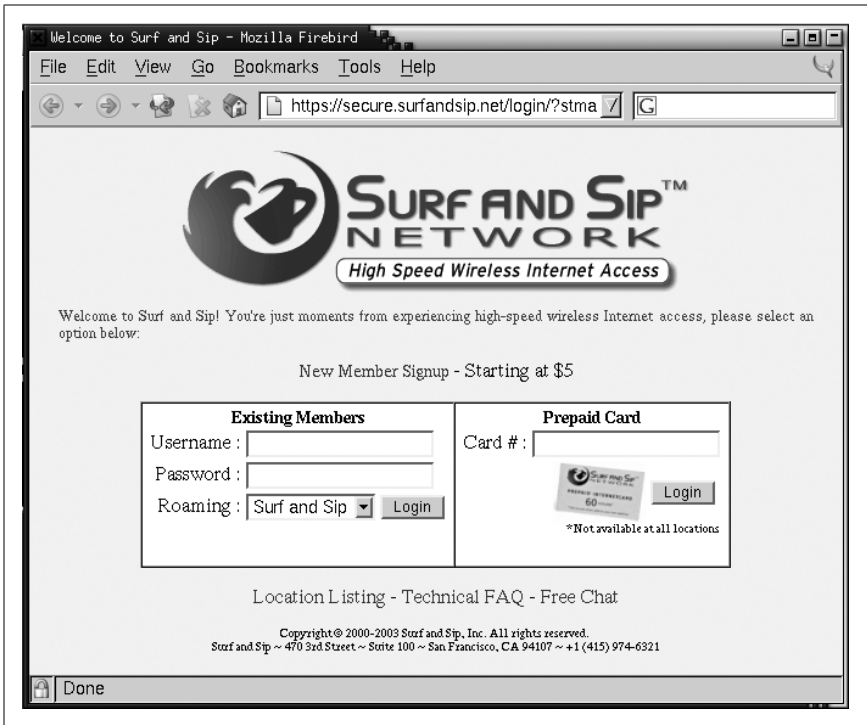


Figure 3-1. Typical hotspot login to a captive portal

Associating with a hotspot provider is easy. They all have easy-to-identify SSIDs. You can also locate their hotspots ahead of time using their web pages. Table 3-1 lists some major hotspot providers, their SSIDs, and their web pages for locating their hotspots.

Table 3-1. Hotspot providers, SSIDs, and location finders

Hotspot provider	SSID	Web location finder
Cometa	Cometa-Hotspot	<a href="http://www.cometa-hotspot.com/locations/">http://www.cometa-hotspot.com/locations/</a>
STSN	STSN	<a href="http://www.stsn.com/hotel_locator.php">http://www.stsn.com/hotel_locator.php</a>
Surf and Sip	SurfandSip	<a href="http://www.surfandsip.com">http://www.surfandsip.com</a>
T-Mobile	tmobile	<a href="http://locations.hotspot.t-mobile.com/">http://locations.hotspot.t-mobile.com/</a>
Verizon Wireless	Verizon	<a href="http://www.verizonwireless.com/wifi/hot_spot/">http://www.verizonwireless.com/wifi/hot_spot/</a>
WayPort	wayport	<a href="http://www.wayport.com/locations">http://www.wayport.com/locations</a>

## Wireless Aggregators

With the rise in availability of commercial hotspot providers comes a conundrum: which hotspot provider do you sign up with? As you've seen, there are many providers, and each of them has different coverage. If you're a real road warrior, using several different hotspots could cost quite a bit.

Wireless aggregators have come into the market to address this problem. You sign up for an account with the aggregator, and through its revenue-sharing agreements with different hotspot providers, you are able to use many different hotspots while maintaining a single account with one company.

That's the theory. In practice, roaming is still very difficult, especially for non-Windows users. Boingo (<http://www.boingo.com>), the largest aggregator, requires the use of proprietary software on your notebook, and as of this writing, that software is Windows-only. There are reports of adventurous people running the Boingo software using a Windows emulator like Wine, but we're not going to attempt to cover that here. Unless the web-based captive portal offers a roaming option, roaming with Boingo and Linux isn't possible at this time.

Two other aggregators fall into the same category: Trustive (<http://www.trustive.com/>) provides only a Windows client software package, and iPass (<http://www.ipass.com>), while providing clients for Windows, Windows CE/Pocket PC, Mac OS X, and Mac OS, does not provide a Linux software client.

Fortunately, there is at least one roaming company that has gotten it right: FatPort. FatPort's roaming customers don't need any special software. Its locations and partner locations all use captive portal software that requires only a web browser.

Although FatPort is based primarily in Canada, it has a wide range of partner agreements with Surf and Sip, Boingo, and iPass. While not a complete coverage of all roaming sites, this is an excellent option for the Linux user who is constantly on the road. FatPort accounts range from hourly rates to yearly subscriptions. Check out <http://www.fatport.com> for more details.

## Open Hotspots

Just as the software world is split into proprietary and open source, the hotspot world is populated with commercial hotspots (which we've covered) and open hotspots. These open wireless networks span a wide range of locations and philosophies:

- Businesses providing free wireless access as an incentive to customers. Hotels, coffee shops, restaurants, bookstores, and other businesses are all using free wireless access as a way to bring in customers and entice them to stay.
- Public places serving up hotspots as a public service. Libraries, city parks, town squares, city halls, and other publicly owned spaces view free wireless access as a way of promoting their city, county, or other locations, and attracting visitors.
- Community wireless groups working with businesses, governments, and private citizens placing hotspots in all sorts of locations, including apartment buildings, parks, downtown areas, and any place that would benefit from free wireless access. Many community groups view this as a way to better their neighborhoods.

Open hotspots are a mixed bag. You may simply be associating with a wireless router in someone's apartment, connected to his DSL line. On the other hand, it may be a custom-built Linux-based access point in a New York City park, installed by NYCWireless (<http://www.nycwireless.net>), with a T1 or DSL backhaul.

Access control is also going to vary. If you connect to someone's home network with an SSID of "default" or "linksys," chances are you won't find a captive portal or any other form of access control in place. Many community and business that open hotspots have some sort of access control in place, such as a web page that asks you to agree to a Terms of Service (ToS) agreement before you are allowed to use the network.

A good place to locate open hotspots is the Personal Telco Project in Portland, Oregon. Visit its Wireless Communities site at <http://www.personaltelco.net/index.cgi/WirelessCommunities>. A second place to look for hotspots is WiFi-Maps at <http://www.wifimaps.com>. This site, while still in development, shows you hotspots all over the world.

## Associating with Hotspots

To associate your Linux notebook with an open or commercial hotspot, you have a couple of options. If you know the SSID of the hotspot, simply set the SSID using `iwconfig`:

```
$ iwconfig eth1 ESSID SurfandSip
```

Once you've done this, fire up your favorite web browser, attempt to navigate to any web page, and you will be redirected to the hotspot captive portal login, as shown in Figure 3-1.

If you've settled in a coffee shop that has an unknown hotspot provider, the first thing you can try is:

```
$ iwconfig eth1 ESSID any
```

If there is a hotspot in range, your card should find and associate with it. This can be tricky, especially if you're in a densely populated urban area. For example, sitting in a coffee shop in San Francisco, we were able to associate with four different SSIDs. The signal strength from the coffee shop hotspot was not as strong as a neighboring open hotspot located in someone's apartment.

In these cases, you want to identify all of the access points in your immediate area before you decide which one to associate with. There are several methods of finding access points with Linux, and we cover each one in turn.

## Wireless Network Discovery

If your network card supports it, the easiest method of locating available wireless networks is included with the Wireless Tools, which you installed in Chapter 2. The `iwlist` command supports a scanning parameter that lists any access points in range. It's worth noting, however, that some wireless card drivers do not support this feature. Chief among them is the `orinoco_cs` driver. If you're using this driver, you must use one of the alternative discovery methods next.

To determine if your card and driver support scanning, execute the `iwlist` command with no other parameters. If you see "scanning" listed in the output, you should be able to scan for available access points. Note that you must have root access to use this command.

```
# iwlist
Usage: iwlist [interface] frequency
          [interface] channel
          [interface] ap
          [interface] accesspoints
          [interface] bitrate
          [interface] rate
          [interface] encryption
          [interface] key
          [interface] power
          [interface] txpower
          [interface] retry
          [interface] scanning
```

Once you've determined that you can use the scanning parameter, execute the command. You must specify the network adapter that corresponds to your wireless card (`eth1` in the following example). Again, you must have root access.

## # iwlist eth1 scanning

```
eth1 Scan completed :
  Cell 01 - Address: 00:02:6F:01:76:31
    ESSID:"NoCat "
    Mode:Master
    Frequency: 2.462GHz
    Quality:0/92 Signal level:-50 dBm Noise level:-100 dBm
    Encryption key:off
    Bit Rate:1Mb/s
    Bit Rate:2Mb/s
    Bit Rate:5.5Mb/s
    Bit Rate:11Mb/s

  Cell 02 - Address: 00:30:65:03:E7:0A
    Essid:"SurfandSip "
    Mode:Master
    Frequency:2.422GHz
    Quality:0/92 Signal level:-66 dBm Noise level:-96 dBm
    Encryption key:off
    Bit Rate:1Mb/s
    Bit Rate:2Mb/s
    Bit Rate:5.5Mb/s
    Bit Rate:11Mb/s
```

Now that you've obtained a list of available networks, see what providers are in your area, and make a decision on the hotspot to use. scanning shows you relative signal strengths, so pay attention. You don't necessarily want to associate with the weakest hotspot in the area.

Note also that the scanning output gives you the frequency of each hotspot as well as whether encryption (WEP) is enabled.

## Kismet

In contrast to the small bit of information you can glean by using iwlist scanning, Kismet is a seriously advanced wireless diagnostic tool. It is a passive network scanner, similar to commercial tools such as Network Associates' Sniffer Wireless and AiropEEK. It is designed from the ground up specifically for scanning wireless networks, so it detects all 802.11 traffic from both access points and wireless clients. It can find "closed" networks (some access points allow you to disable the broadcast of the SSID) by monitoring traffic sent from clients, and it logs all raw 802.11 frames in standard *pcap(3)* format for later use with other specialized tools such as Ethereal, an open source network protocol analyzer.

To take advantage of Kismet's advanced features, you need a wireless card and driver capable of entering RF Monitor or promiscuous mode. Cards in this category include the Prism-based cards using the `host_ap` driver and the Cisco Aironet cards using the `airo` driver. Kismet also works well with Atheros-based 802.11a/g cards using the `madwifi` driver. However, if you need monitor mode in the `madwifi` driver, download the latest CVS driver code. Finally, you'll need a patched `orinoco_cs` driver or the latest CVS version of the `orinoco_cs` code to support monitor mode with Orinoco cards. We covered this in detail in Chapter 2.

Kismet is available as a package with most distributions. Debian users can install Kismet using `apt-get`:

```
apt-get install kismet
```

Red Hat and Fedora users can obtain RPM packages from <http://www.rpmfind.net>. Mandrake users can install Kismet using `urpmi`:

```
urpmi kismet
```

If you want to read Kismet's dump files in Ethereal, you must download the source code for Kismet from <http://www.kismetwireless.net>. Also, Ethereal must be installed from source, and the Ethereal source code tree must be available. Change into the Kismet source directory, and configure Kismet as follows:

```
# ./configure --with-ethereal=/your/ethereal/source/path/here
```

Once that is done, build Kismet with standard compile commands:

```
make  
make dep  
make install
```

Once Kismet is compiled or installed from source, you must edit `/usr/local/etc/kismet.conf` to suit your system. If you've installed from package, the file is probably located in `/etc/kismet.conf`. At a minimum, you must edit the `source=` line to match your hardware. The format for this line is `driver,device,description`. For example, with a Prism card, edit the line to read:

```
source=hostap_cs,wlan0,Prism
```

See the comments in the `kismet.conf` file for more information on supported drivers.

If you want Kismet to play sound effects when it finds new SSIDs, it will. By default, it expects `/usr/bin/play` to be installed, which is part of the Sox sound utilities, but any command-line audio player works. All of the audio and other display parameters are configured in `/usr/local/etc/kismet_ui.conf`.

When Kismet is running, your wireless card will be in RF monitoring mode. Note that once in this mode, your card can no longer associate with wireless networks, so you may not have a network connection.

Now execute the `kismet` command using your normal user ID. You don't have to run the Kismet user interface as root. You should see the Kismet screen as shown in Figure 3-2.



Figure 3-2. The main Kismet screen

Kismet incorporates a hopping algorithm to switch between radio channels in order to find all the networks in your locations. This makes your card hop between radio channels. The hop pattern is configurable to your needs. See the `kismet_hopper` manpage for details. Note that newer versions of Kismet call `kismet_hopper` automatically.

By default, Kismet initially scans the network list based on the last time it saw traffic from each network. This list constantly changes, making it difficult, if not impossible, to select any one network for more detailed information.

To keep the list from constantly changing, manage the scanning sort order by hitting `s` at any time, followed by the desired sort order. For example, to sort by SSID, hit `ss`. Now use the arrow keys to select a network for further details. Press `h` at any time to see keystroke help and `q` to close any pop-up windows.

To get more information on a specific network, select it using the arrow keys and press **i**. You will see a more detailed screen as shown in Figure 3-3.

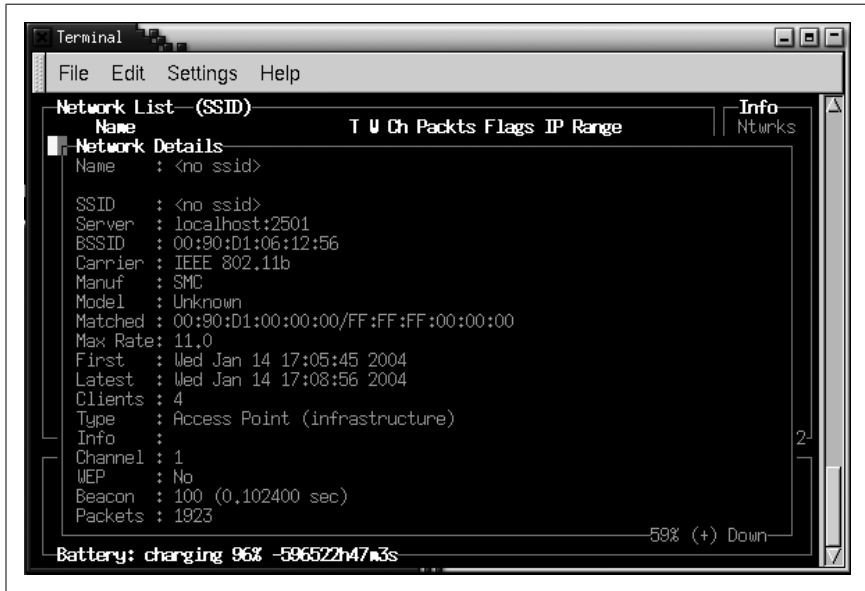


Figure 3-3. Kismet's detailed network information

Kismet finds closed networks (networks that do not broadcast their SSID). If there is no network traffic coming from a client of that network, Kismet lists the SSID with a name of `<no_ssid>`. Once Kismet sees a frame of traffic from a client, the SSID updates.

Note that your card is now out of monitor mode, but the original network settings are not returned. You can physically eject the card and reinsert or execute:

```
# cardctl reset
```

## AP Radar

The previous methods are perfectly usable and provide you with all sorts of information regarding the available wireless networks near you. These are manual methods that don't approach the level of ease in wireless detection and configuration that is offered with other operating systems.

AP Radar is an attempt to make detection of and connecting to wireless networks easier and more manageable. It is both a graphical network discovery tool and a wireless profile manager. Using the Wireless Extensions, it has the ability to watch for wireless networks while staying associated to your

existing network. It focuses on automating tasks, so that when you come in range of your home network, you are automatically connected.

AP Radar is the work of Don Park, and you can obtain it from the project's SourceForge development site at <http://apradar.sourceforge.net>. Currently, it is available as an RPM package or as a source file. In order to get the package running, you must have GNOME Version 2. You'll also need a 2.4.20 kernel or higher, or any 2.6 kernel.

To compile AP Radar from source, you must have the GTK+ header files and libraries, as well as the GTKmm header files and libraries. Users of Mandrake, RedHat, and other distributions that use RPM should see the AP Radar README file for a list of required RPMs.

Debian users should be able to install the same packages via apt-get; however, you must set up apt to obtain packages from the testing or unstable trees. See the `sources.list` manpage for details.

To build AP Radar from source, uncompress the source code file and change into the newly created directory. The commands to compile are standard, although the filename and top-level directory name will differ if you are using a newer version than we did:

```
$ tar xzvf apradar-0.50.tar.gz
$ cd apradar-0.50
$ ./configure
$ make
$ su -c "make install"
```

AP Radar works with a number of wireless cards and drivers. To determine whether AP Radar will run with your card and driver, execute `iwlist` scanning:

```
# iwlist wlan0 scanning
```

You should see some output like the following:

```
eth1 Scan completed :
Cell 01 - Address: 00:02:6F:01:76:31
ESSID:"NoCat "
Mode:Master
Frequency: 2.462GHz
Quality:0/92 Signal level:-50 dBm Noise level:-100 dBm
Encryption key:off
Bit Rate:1Mb/s
Bit Rate:2Mb/s
Bit Rate:5.5Mb/s
Bit Rate:11Mb/s
```

If you see anything else, chances are AP Radar will not function with your card. Some reasons for this include the use of the following drivers:

### *Orinoco\_cs driver, wlan, wavelan, and wavelan2 drivers*

None of these drivers currently support wireless scanning. Patches are available for the `orinoco_cs` driver to enable scanning, and the CVS code for `orinoco_cs` also supports scanning. See Chapter 2 for more details.

### *host\_ap driver*

If you are using the newest `host_ap` code, Version 0.1.3 (as of this writing), you must execute the following command as root for AP Radar to function properly:

```
# iwpriv wlan0 host_roaming 1
```

Once you install AP Radar and determine that it will function with your wireless card/driver, simply start it as root:

```
# apradar
```

If you experience problems starting AP Radar, it may be due to oddities in your wireless card driver and how it writes status to `/proc/net/wireless`. In order to avoid this problem, start AP Radar by specifying the interface name (`ath0` in the following example):

```
# apradar -i ath0
```

The AP Radar main screen appears, as shown in Figure 3-4.

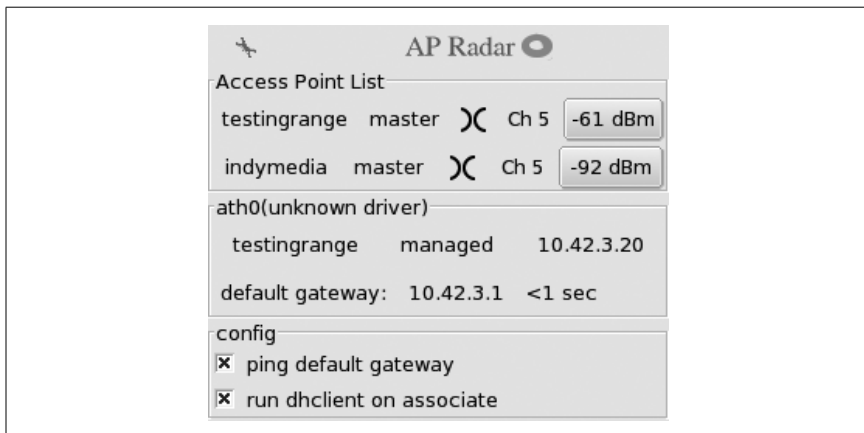


Figure 3-4. AP Radar main screen

AP Radar shows all access points that are in range. Almost every field on the screen is either clickable or provides you with information when you hover the mouse over it.

To associate with any of the access points shown under Access Point List, simply click on the name of the access point. By default, AP Radar not only associates your wireless card with the selected AP, but it runs `dhclient` to obtain an IP address via DHCP.

This and one other option can be set by clicking on the red symbol at the top of the AP Radar screen. You can set two options:

#### *Ping default gateway*

This monitors the gateway that you receive from DHCP. When it does not receive a response from a ping after more than a second, AP Radar assumes that the gateway is out of range.

#### *Run dhclient on associate*

This allows you to specify whether you want AP Radar to obtain a DHCP address for your PC after it associates with an access point. Turn this off if you need to use static addressing.

In addition to displaying the SSID, mode, and channel and signal strength for each access point, AP Radar also displays whether WEP is enabled by displaying the warchalking symbol for the network. See the later section “Warchalking.”

## Wardriving

Back in the good ol’ days of hacking, *wardialing* was (and still is) the act of having a computer use a modem to dial phone numbers from a list or mathematically step through all possible numbers in a telephone exchange. Malicious hackers noted each line that had an answering modem and went back to those numbers to find systems that could be compromised.

With the proliferation of notebook computers, handheld computers, and wireless network cards, the term *wardriving* has been coined. When you wardrive, usually a two-man team takes off: one driving and the other handling the wireless scanning. In dense urban areas, a wardrive can locate hundreds if not thousands of active SSIDs.

With some added equipment such as external antennas and a GPS receiver, wardrivers can log each wireless network and place them on a physical map. <http://www.wifimaps.com> is just one example of a collaborative effort to place wardriving maps from all over the world in an online database. Kismet (discussed previously) makes an excellent tool for wardriving, and it interfaces with GPS systems. See Chapter 10 and the Kismet documentation for details.

People wardrive for different reasons. While many people do it simply for enjoyment or for the technical knowledge gained, there are also those who have more illicit purposes in mind. Some wardrivers are specifically out there looking for insecure networks that can be compromised for various purposes.

Wardriving may not be legal in your area. While it does not appear to be illegal in the United States, there are many countries where it is considered a crime.

## Warflying

In the same vein, *warflying* is conducted by those lucky people who can afford to rent a plane for a few hours or who actually have their own plane. Warflyers generally need external antennas to pick up wireless networks below the plane.

If you think this practice sounds too far-fetched to be true, Google for the phrase “warflying”. You’ll be surprised at how many people do this.

## Warchalking

During the Great Depression, many people in the United States were homeless because of economic conditions. Tramps and hobos traveled the country looking for work and food. Due to scarcity of work, hobos were not welcome in many places. Over time, hobos devised a set of logos that could be written in chalk or stone, or carved in trees near various houses, restaurants, and other places. These logos could communicate everything from “free food” to “you will be beaten.”

You can visit the following web sites for more symbols used by the hobos:

- <http://www.slackaction.com/signroll.htm>
- <http://sedaliakatydepot.com/hobo.htm>

Matt Jones, an Internet product designer, operates a web site (<http://blackbeltjones.com>) that serves primarily as the Londoner’s online resume and portfolio. In 2002, Jones combined the practice of using a sniffer tool to detect a wireless network with the hobos’ set of logos to come up with the symbols for wireless networks (see Figure 3-5).




KEY	SYMBOL
OPEN NODE	ssid  bandwidth
CLOSED NODE	ssid 
WEP NODE	ssid      access contact  bandwidth

Figure 3-5. Warchalking symbols

Using these symbols, wireless users can discover if there is an available wireless network for their use. He was inspired by architecture students “chalking up the pavement” on his way to lunch. During a lunch, Jones and a friend, who had recently been discussing hobo signs, called their idea warchalking. You can learn more at <http://www.warchalking.org>.