CHAPTER 17

WIRELESS TOOLS

Wireless networks offer the convenience of mobility and a reduced amount of network equipment. They also broadcast their presence, and possibly all of their data, to anyone who happens to be listening. The proliferation of wireless networks reintroduced many problems with clear-text protocols (communications in which sensitive data is not encrypted). They also permitted arbitrary users access to a corporation's internal network—absolutely bypassing the firewall or other security devices. The threats to wireless networks are not just limited to malicious users looking for open networks; anyone could sit in the parking lot and sniff the network's traffic.

Before we dive into two wireless tools, we should review a few wireless terms. Wired Equivalent Privacy (WEP) is an attempt to overcome the promiscuous nature of a wireless network. To sniff traffic on a wired network (one with CAT-5 cables, hubs, and switches), you first must physically connect to the network. For a wireless network, you merely need to be within proximity of an access point (AP). WEP is designed to provide encryption at the physical- and data-level layers of the network. In other words, it encrypts traffic regardless of the network protocol, such as TCP/IP or IPX. If a network is using WEP, traffic on it will be much harder to sniff; however, poor implementations of WEP have allowed a user to guess the encryption key and consequently view arbitrary traffic.

The other acronym that pops up quite a bit is the Service Set Identifier (SSID). The SSID is prepended to wireless packets. SSIDs provide a means for multiple access points to serve multiple networks while discriminating between packets. The SSID can be up to 32 characters long. Thus, one network might have an SSID of dev, and another network might have an SSID of DMZ. Even if the APs for these networks are close together, packets for the dev network will not enter the DMZ network by mistake. Thus, the SSID can be considered a sort of password to the AP, but one that is sent in clear text and is easy to discover if the SSID broadcast is enabled (or you wait long enough to catch a legitimate client connect to the AP). The SSID is a shared secret on the network, but it is similar to the SNMP community strings: they are all too often secrets that everyone knows. For example, here are some very common SSIDs:

- comcomcom
- Default SSID
- intel
- linksys
- Wireless
- WLAN

In addition to a computer and a wireless card, you can complement your wireless arsenal with a high-gain antenna and a Global Positioning System (GPS) unit. A high-gain antenna improves the range of your card, increasing the distance from which you can access a network. A GPS unit comes in handy when driving through areas on the prowl for network access points. Many tools incorporate the ability to record the access point's technical information (such as the SSID) as well as its location. Later, you could correlate the location on a map.

An external antenna is a good idea for improving your card's range from a few dozen meters to well past a kilometer. Several options are available, from $100 prebuilt antennas to high-gain antennas you can build yourself from cans and washers. A strong antenna not only lets you find distant networks, but it also lets you figure out how far away the data from your own wireless network is going.

Appropriate wireless drivers are necessary for many of the capabilities required by the tools covered in this section. Linux, FreeBSD, and Mac OSX (for Viha chipsets) have drivers that support the most common cards. The wireless cards of choice use Prism-based chipsets. Cisco and Orinoco (sometimes branded as Lucent) chipsets have adequate support as well. Currently, wireless cards that use a Broadcom chipset are to be avoided when using these wireless tools—the Broadcom drivers simply do not support the capabilities required. As a rule, you're pretty safe with any 802.11b card, but 802.11a and 802.11g cards tend to have inadequate drivers for Linux and FreeBSD. There are exceptions, but if you stick to Prism-based cards and check with some wireless-related newsgroups, you should do well.

NOTE The Linux ndiswrapper (*http://ndiswrapper.sf.net*) project enables Linux-based systems to take advantage of a Window's driver for a wireless device. So, even if a wireless card has no support for Linux, the ndiswrapper application enables Linux to use the card and access wireless networks. While this is perfect for associating to a network, this driver is designed to perform the basic functions necessary for networking. This driver won't let you use the advanced capabilities that a tool like Kismet provides. Check a card's chipset support before you buy it!

As a final note, it's important to realize that wireless networks have several implications for security. At its advent wireless (or "wi-fi") network security relied on WEP, which proved to be an insecure implementation of a cryptosystem. The encryption algorithms that it used weren't the problem; instead, it was the manner in which they were applied. As such, networks protected by WEP were in effect vulnerable to sniffing attacks that could reveal the encryption key used to protect all of the packets. The initial shortcomings of wireless security protocols were addressed by WPA and WPA2. These protocols improved the encryption scheme's implementation and also created per-user encryption. So, while a sniffing attack may still be possible, it is no longer as trivial to crack the encryption keys used to protect the wireless communications. Nevertheless, any wireless network must also consider the implications of having a network that is not physically bound by the walls of a building. The tools in this section focus on the discovery and inventory of wireless networks.

NETSTUMBLER

The NetStumbler tool, *http://www.netstumbler.com*, identifies wireless access points and peer networks. It does not sniff TCP/IP protocol data. Instead, it provides an easy method for enumerating wireless networks. You just launch the application, walk (or drive as in "wardriving") around an area, and watch as wireless devices pour into the list.

Implementation

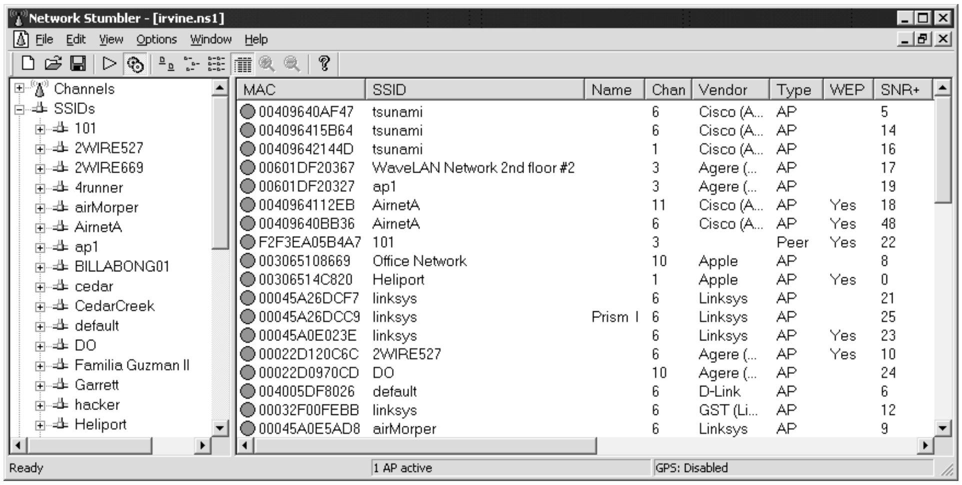
Even though NetStumbler appears to grab SSIDs from the ether, it works on a simple principle. It transmits connection requests to any listening access point with an SSID of ANY. Most APs respond to the request by sending their own SSID. Consequently, NetStumbler is not a passive sniffer. In other words, its traffic can be seen on the victim or target networks.

When you launch NetStumbler and start a capture file, it begins to search for access points. Figure 17-1 shows some examples of access points. The right pane displays the MAC address of the AP and its corresponding information such as its WEP status, SSID, signal strength, and coordinates if a GPS unit is attached to the computer.

The left pane contains three tree views: Channels, SSIDs, and Filters. The Channels and SSIDs views break down the results into obvious fields. The Filters view also shows APs, but only if they meet certain criteria. Table 17-1 describes each of the default filters.

The most difficult part of using NetStumbler is locating wireless networks. NetStumbler's web site enables users to upload their own capture files, complete with SSID and GPS information. Then anyone can query the web site's database to view the geographic location of access points.

NOTE Many access points support the ability not to broadcast the SSID. In this case, NetStumbler will not discover the AP.



MAC	SSID	Name	Chan	Vendor	Type	WEP	SNR+
00409640AF47	tsunami		6	Cisco (A...	AP		5
004096415B64	tsunami		6	Cisco (A...	AP		14
00409642144D	tsunami		1	Cisco (A...	AP		16
00601DF20367	WaveLAN Network 2nd floor #2		3	Agere (...	AP		17
00601DF20327	ap1		3	Agere (...	AP		19
0040964112EB	AirnetA		11	Cisco (A...	AP	Yes	18
00409640BB36	AirnetA		6	Cisco (A...	AP	Yes	48
F2F3EA05B4A7	101		3		Peer	Yes	22
003065108669	Office Network		10	Apple	AP		8
00306514C820	Heliport		1	Apple	AP	Yes	0
00045A26DCF7	linksys		6	Linksys	AP		21
00045A26DCC9	linksys	Prism I	6	Linksys	AP		25
00045A0E023E	linksys		6	Linksys	AP	Yes	23
00022D120C6C	2WIRE527		6	Agere (...	AP	Yes	10
00022D0970CD	DO		10	Agere (...	AP		24
004005DF8026	default		6	D-Link	AP		6
00032F00FEBB	linksys		6	GST (Li...	AP		12
00045A0E5AD8	airMorper		6	Linksys	AP		9

Ready | 1 AP active | GPS: Disabled

Figure 17-1. Detecting wireless networks

Filter Name	Description
Encryption Off	Lists all devices that do not have WEP enabled. This implies that you would be able to sniff the network's traffic.
Encryption On	Lists all devices that have WEP enabled. Early WEP implementations were insecure, and their traffic could be decrypted.
ESS (AP)	The Extended Service Set ID (ESSID) is an alphanumeric code shared by all APs and wireless clients that participate on the same wireless network. It enables multiple APs to serve the same network, which is important for physically and logically large networks. Thus, two APs could use the same channel and even have overlapping coverage but serve two unique wireless networks. The default ESSID is well known for a few APs: Cisco (tsunami), 3COM (101), and Agere (WaveLAN network).
IBSS (Peer)	This filter represents another wireless card in a peer-to-peer or ad hoc mode. The concept is similar to a crossover cable on wired networks. This allows two (or more) wireless cards to communicate with each other without the presence of an AP.
CF Pollable	These APs respond to specific beacon packets to determine periods in which to broadcast. An AP that supports contention-free (CF) transmission is used to reduce collisions and improve bandwidth.
Short Preamble	An alternate method for specifying data in the 802.11b physical layer. The abbreviated preamble is used for time-sensitive applications such as voice-over IP or streaming media.

Table 17-1. NetStumbler Filters

AIROPEEK

AiroPeek, from *http://www.wildpackets.com/products/airopeek*, actually lets you peek into the data transmitted across a wireless network. It goes beyond the capability of NetStumbler by displaying, for example, web traffic. This aspect of AiroPeek places it into the category of a packet capture tool such as tcpdump.

Implementation

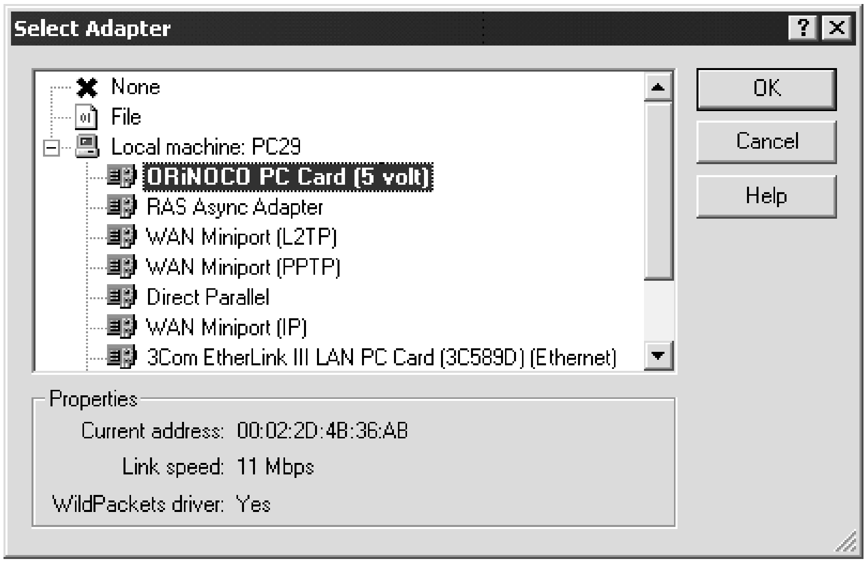
The most important prerequisite for AiroPeek is obtaining a wireless card with the correct firmware that permits promiscuous mode. AiroPeek supports Cisco Systems 340 Series, Cisco Systems 350, Symbol Spectrum24 11 Mbps DS, Nortel Networks e-mobility 802.11 WLAN, Intel PRO/Wireless 2011 LAN, 3Com AirConnect 11 Mbps WLAN, and Lucent ORiNOCO PC (Silver/Gold) cards. For cards that require a specific firmware, the drivers are available from the WildPackets web site.

When you first launch AiroPeek, you will be prompted for an adapter to use for capturing data. Simply highlight the correct card and click OK. Figure 17-2 shows an example of this window.

AiroPeek is now ready to capture packets. Select Capture from the main menu. A screen similar to the one shown in Figure 17-3 greets you. Now most wireless traffic that passes within range of your wireless card can be captured.

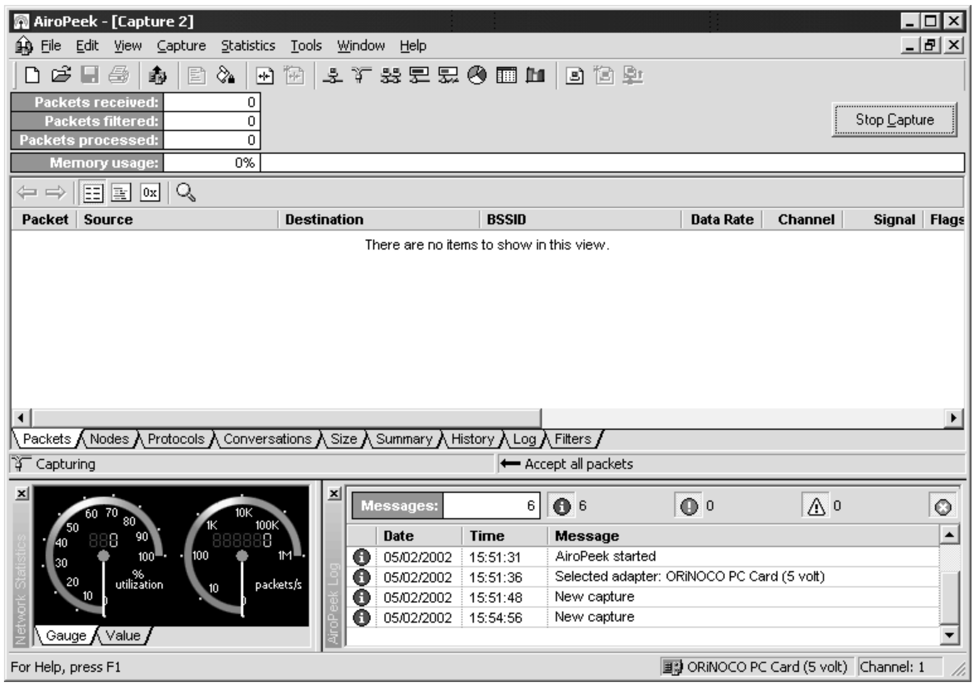
If multiple wireless networks are in the area or a large amount of traffic is occurring, you can use triggers to narrow down the amount of data collected.

TIP You can decrypt WEP-protected traffic if you know the correct WEP key. Set the key by choosing Tools | Options | 802.11 | WEP Key Set | Edit Key Sets.



Current address: 00:02:2D:4B:36:AB
Link speed: 11 Mbps
WildPackets driver: Yes

Figure 17-2. Select a wireless adapter.


AiroPeek - [Capture 2]
File Edit View Capture Statistics Tools Window Help
Stop Capture
Packets Nodes Protocols Conversations Size Summary History Log Filters
Gauge Value
Network Statistics
AiroPeek Log
% utilization
packets/s

Packets received: 0
Packets filtered: 0
Packets processed: 0
Memory usage: 0%

Packet	Source	Destination	BSSID	Data Rate	Channel	Signal	Flags

There are no items to show in this view.

Capturing   Accept all packets

Messages: 6   6   0   0

Date	Time	Message
05/02/2002	15:51:31	AiroPeek started
05/02/2002	15:51:36	Selected adapter: ORiNOCO PC Card (5 volt)
05/02/2002	15:51:48	New capture
05/02/2002	15:54:56	New capture

For Help, press F1   ORiNOCO PC Card (5 volt)   Channel: 1

Figure 17-3. Capture wireless traffic.

From this point on, AiroPeek is just another network sniffer. Use it to validate that traffic is being encrypted or to determine how much network information from the wired network leaks to the wireless network. Here are some typical scenarios:

- *Verify that WEP is enabled*. Without the proper WEP key, AiroPeek will not be able to view any of the data.
- *Verify that MAC-based access is working.* MAC-based access permits wireless cards with only a specific hardware MAC address to access the wireless network. Other network cards may see the traffic but will not be able to access the network.
- *Identify at-risk protocols on the wireless network.* Use AiroPeek to determine what type of traffic goes across the wireless portion of the network. Is domain authentication passed? Are NT LAN Manager hashes being passed between file shares? Are any clear-text protocols in use? Even if WEP is enabled on the network, a malicious insider with knowledge of the WEP key could still watch traffic.

- *Debug the wireless network.* As a system administrator, you've likely been asked "Why is the network slow?" at least a dozen times. A tool such as AiroPeek can help you debug the network to determine whether communications problems exist between servers, unresponsive hosts, or interfering traffic.
- *Determine the network's range.* Perform a simple test to determine how far your network propagates. For example, ride the elevator up and down a few floors (if you're in such a building) to determine who else can see your network. Walk outside the building until you lose the signal. This test is useful only if you're also using a high-gain antenna. Highly directional antennas on the order of 20 dB gain are available. These antennas can receive very weak signals, but they have a narrow angle in which they work most efficiently. This means that someone who wishes to eavesdrop on your network from a distance must be patient and use a tripod (or other stationary device) to capture signals. In the end, you'll want to know how far your network reaches, so don't rely on a laptop's antenna.

WELLENREITER

Wellenreiter is a user-friendly tool that offers simple, straightforward wireless packet capture. It comes in two flavors: a Perl-based script that supports Linux and a C++ version that supports the wider Unix population as well as some handheld devices. Consequently, Wellenreiter is good for quickly putting together simple, unsophisticated wireless auditing tools.

Implementation

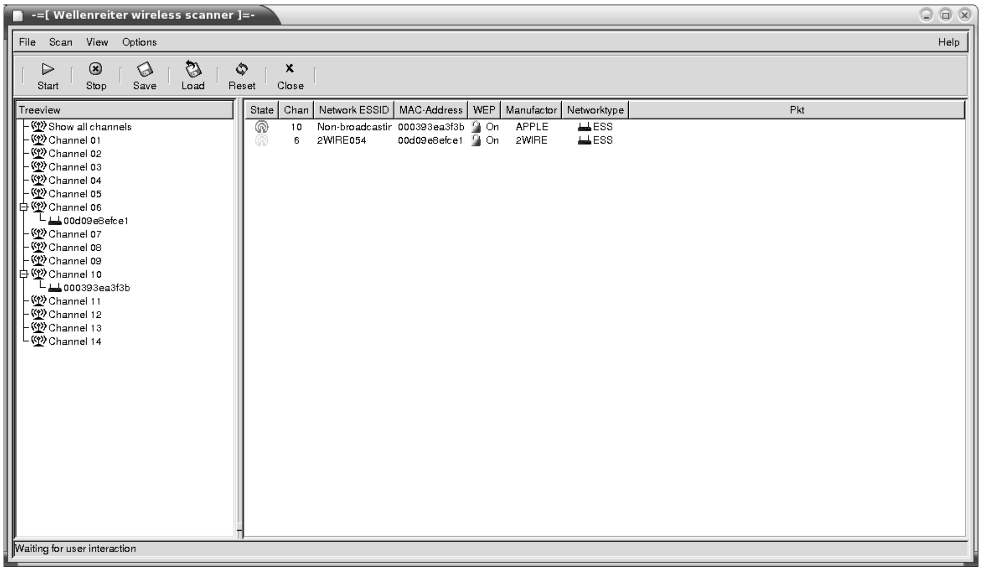
If you have used a Perl script, you've probably come across the problem of installing certain modules required by the script. The Wellenreiter Perl script is no exception. It requires the Net::Pcap module, which is readily available from *http://www.cpan.org*. The interface requires the GIMP Toolkit (GTK) module, but that is most likely already present on most systems.

TIP If you have trouble installing the Net::Pcap module on a Linux distribution, verify that you have the perl-devel-*.rpm installed. The module requires certain headers to compile correctly.

Once you have installed the necessary Perl modules, you are ready to use Wellenreiter. It will automatically handle the configuration and monitoring mode for most Cisco-, Lucent-, and Prism2-based cards. Therefore, all you need to do is execute the script with root privileges.

```
# perl Wellenreiter.pl
```

The interface is simple (see Figure 17-4). The left pane lists channels monitored by your card and the right pane displays the SSIDs discovered.


-=[Wellenreiter wireless scanner]=-
File
Scan
View
Options
Help
Start
Stop
Save
Load
Reset
Close
Treeview
Waiting for user interaction

```
Show all channels
Channel 01
Channel 02
Channel 03
Channel 04
Channel 05
Channel 06
  00d09e8efce1
Channel 07
Channel 08
Channel 09
Channel 10
  000393ea3f3b
Channel 11
Channel 12
Channel 13
Channel 14
```

State	Chan	Network ESSID	MAC-Address	WEP	Manufactor	Networktype	Pkt
	10	Non-broadcastir	000393ea3f3b	On	APPLE	ESS	
	6	2WIRE054	00d09e8efce1	On	2WIRE	ESS	

Figure 17-4. Wellenreiter in action

By default, Wellenreiter saves a binary packet capture to the user's home directory. Look for *.dump files with a timestamp in the name. These are in pcap format (remember Net::Pcap?) and can be viewed with tcpdump or Ethereal. You can also read data from a GPS device, but if you're interested in mobile wireless discovery, kismet might be better suited to your needs.

KISMET

Kismet's capabilities and usefulness have grown significantly since its first release. It is one of the most robust open-source wireless tools available. You will also find that the web site, *http://www.kismetwireless.net*, also provides some excellent forums for selecting the appropriate equipment to accomplish your task. Once you delve into the wireless arena, you will realize that a good antenna (or set of antennas) and a GPS device add more quality to the data collected.

TIP If Kismet turns out to be more than you need on a BSD system, you can try the bsd-airtools in the security directory of the BSD Ports collection. It has an interface similar to Kismet, but it does not have all of the capabilities and data handling that Kismet offers.

Implementation

Kismet compiles on most Unix systems (including Mac OS X) and Cygwin for Windows. You can download the latest stable release from *http://www.kismetwireless.net*, or you can play with development releases via CVS access.

NOTE Do not be misled by the comment that kismet will compile on Windows in the Cygwin environment. It will compile easily with the `./configure --disable-pcap` option, but it will not function as a server. In other words, you will not be able to use kismet to sniff with a wireless card in a Windows system. On the other hand, you will be able to use the client functionality of a Windows-based kismet.

```
$ cd /usr/local/src
$ svn co http://svn.kismetwireless.net/code/trunk kismet-devel
```

Once you have downloaded the source code, follow the usual `./configure; make; make install` routine. For the most part, kismet will be able to auto-detect the options and drivers available on your system. The only exception is when you wish to compile with Ethereal wiretaps, which enables compatibility with several binary packet capture and storage formats. The Ethereal source code must have already been compiled, but it is not necessary to install Ethereal. To do this, pass the following option to ./configure:

```
$ ./configure --with-ethereal=/path/to/ethereal/source
```

NOTE Ethereal support is currently disabled. This does not affect capture files, which can still be stored in pcap (tcpdump) format. If you're not sure why you would need this support, then don't worry about it. If you really need the support, try uncommenting the appropriate lines in the configure file (or hard-code the etherealdir value).

As a final convenience, you can install kismet with SUID root privileges using the `suidinstall` target to `make`. This means that any user can launch kismet. It also implies that any user can take advantage of kismet's sniffing capabilities or compromise the system through a security hole in the kismet binary. This is really more a matter of choice and policy than security.

Configuring the Server and Client

Kismet has two main pieces: a server for collecting wireless data and a client for presenting the data to the user. You can also create drones, which are secondary servers and useful for large distributed wireless sniffing networks. This chapter will just focus on the server and client; once you understand those, a drone can be quickly configured. Once the binaries have been compiled, a few steps need to be taken before kismet is ready to sniff. The kismet.conf and kismet_ui.conf files need to be configured.

NOTE Kismet can run in a third "drone" mode, which also has a kismet_drone.conf file. This mode is more useful for distributed systems in which administrators wish to monitor particular areas. Drones are merely collection points for data, much like servers.

By default, the configuration files are located in the /usr/local/etc directory. Table 17-2 shows some important lines of the kismet.conf file (the file used by the server).

Name	Value	Description
`Version`	`2005.01.R1a`	Tracks the version for which the configuration file was initially built. Make sure that this is close, if not an exact, match to the version of the binary that you use. If you notice that the configuration files have not changed even though you have installed a newer version of kismet, use the `forceinstall` target when issuing the `make` command. Beware that this will overwrite previous configuration files.
`servername`	`Kismet`	A mnemonic for keeping track of servers. Change this to whatever you want.
`suiduser`	`user name`	This must be a nonroot user defined on your system. Kismet starts with root privileges to set the wireless cards in monitor mode but then drops to nonroot status while it collects data. Depending on your personal paranoia, this is either a must-have security defense or a convenient option.
`source`	`cisco,eth0,ciscosource`	This defines the driver, card name, and descriptive name for the wireless card. The driver must match a supported card (there are many!), and the card name must match the interface defined on your system (such as eth0 or wlan0). It is possible to define multiple sources, all of which can be used by the server.
`channelvelocity`	`n`	When configured to scan multiple channels, Kismet will scan *n* channels per second. Minimum 1 Maximum 10

Table 17-2. /user/local/etc/kismet.conf Options That Should Be Modified

If you have changed these options, you can immediately start sniffing with kismet. If you chose the route of a suidinstall, type **kismet** and everything will start correctly. Otherwise, you will have to perform two steps.

```
# kismet_server
# su - user
$ kismet_client
```

It is best if you switch `user` (`su` command) from root to the `suiduser` defined in the kismet.conf file.

Tweaking the Server and Client

As long as you specify the `suiduser` and `source` options in the kismet.conf file you can launch kismet as a single, host-based wireless sniffer. If you want to delve into more advanced capabilities, you'll need to modify other values and install additional software, as shown in Table 17-3.

Name	Value	Description
`tcpport`	`2501`	The socket on which the server listens for client connections.
`allowedhosts`	`127.0.0.1`	IP addresses or networks in CIDR notation (e.g. 10.0.1.0/24) that are permitted to connect to the server. This has no effect on what hosts are sniffed from the wireless network.
`maxclients`	`5`	The maximum number of remote clients that may connect to the server.
`gps`	`true`	Enable or disable if a GPS device is present.
`gpshost`	`localhost:2947`	Most GPS devices connect to the computer via a serial or USB cable. The software used to read data from the device opens a socket so that other applications can access the data. The most popular package, gpsd, listens on this port by default. Normally, it doesn't make sense to specify a host other than localhost.
`gpsmodelock`	`false`	Set to true only if you are having trouble with GPS data capture.
`enablesources`	`prismsource`	If you create multiple `source` definitions, you should explicitly enable them with this option.

Table 17-3. More kismet.conf Settings

Name	Value	Description
`sourcechannels`	`prismsource:1,6`	This option presents a unique method of distributing scan duties among multiple cards. For example, it is possible to have multiple USB and PCMCIA wireless cards attached to a single laptop. If you had two cards, you could set one to monitor all channels and the other to monitor only channel 6 (possibly the most-used channel): `sourcechannels=prism1:1,11,2,7,3,8,4,9,5,10` `sourcechannels=prism2:6`
`alert`	`NETSTUMBLER, 5/min,2`	Kismet generates an alert for 10 predefined types of traffic. These relate to specific probes and packets sent by tools such as Wellenreiter, NetStumbler, and Airjack. Typically, they represent malicious activity on the wireless network.
`logtemplate`	`%n-%d-%i.%l`	The `logtemplate` can specify paths and filenames. You may find it useful to create a directory hierarchy based on date (`%d`) or log type (`%l`). The only drawback to this method is that the directory must exist before kismet starts; otherwise, it will not save the file.

Table 17-3. More kismet.conf Settings *(continued)*

It is possible to have the server play sounds or use speech when networks are detected, but it makes more sense to set these options in the client configuration file, kismet_ui.conf. The configuration directives in Table 17-4 are not essential to kismet's operation, but they provide battery information and enable various audio cues when networks are discovered.

Name	Value	Description
`apm`	`true`	Displays remaining battery charge.
`sound`	`true`	Plays a sound for specific events: new network, traffic, junk traffic, GPS locked, GPS lost, and alert.
`soundplay`	`/usr/bin/play`	The path to the application that plays .wav files.
`speech`	`false`	Uses the Festival text-to-speech engine to report discovered networks.
`festival`	`/usr/bin/festival`	You will need to download and compile the Festival speech engine. This option defines the location of the binary. See *http://www.cstr.ed.ac.uk/projects/festival/*.

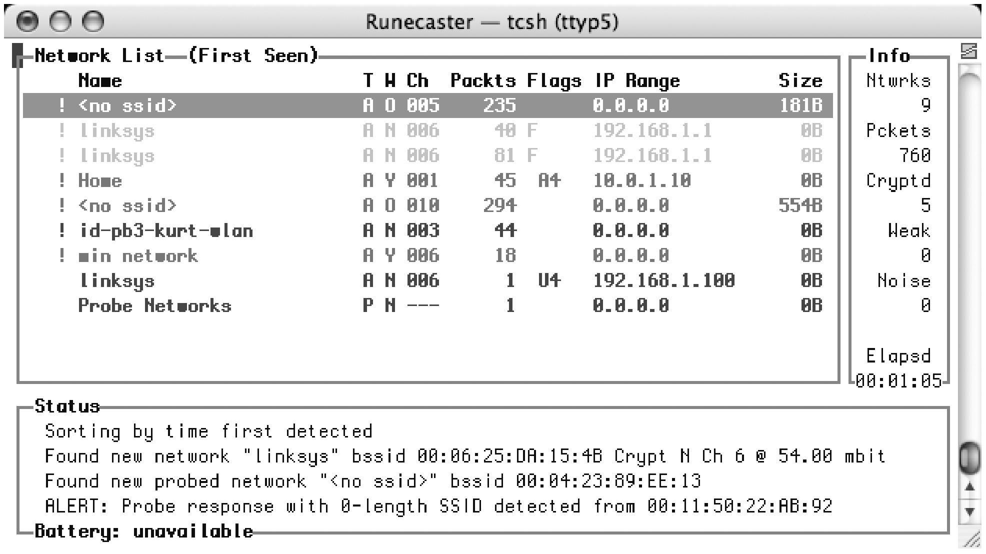
Table 17-4. Important kismet_ui.conf Settings

Kismet Commands

Kismet provides helpful instructions from the client. Press `h` to access the Help menu. Press `x` to close any pop-up window, including the Help menu. To use any of the commands to view information about an SSID or list its associated clients, you must take kismet out of auto-sort mode. Do this with the `s` command followed by the type of sort to use, such as `f` for first seen. When you want to view more details about a network, use the arrow keys to highlight the desired SSID, and press `i` for information. An example of captured wireless traffic is shown in Figure 17-5. Notice that this will tell you whether the SSID is cloaked (not being broadcast by the access point), the MAC address of the device, traffic rate, use of WEP, channel, and GPS information if available.

Press `c` to view the clients of a particular SSID. This shows another auto-sorted list of MAC addresses and related information. Take this out of auto-sort mode (press `s` and then `f`), and then highlight a client and press `i` to display more information. Figure 17-6 shows an example of an OS X kismet client. You might notice, however, that the information does not appear reliable. The MAC address is incomplete and no IP address is associated with the client. This tends to happen for weaker signals. Wireless networks do not have the fidelity of wired networks, so expect to capture noise and bad data.

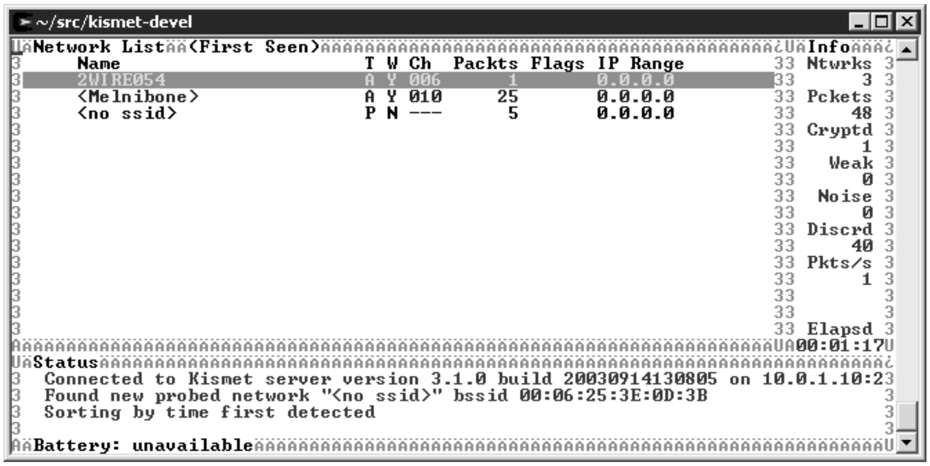
To view detailed information about a wireless network or a particular wireless client, highlight the SSID or MAC address and press `i`. Figure 17-7 shows an example of the information available for wireless networks. The information regarding the wireless network represents the configuration settings of the network's base station. Figure 17-8



```
Network List—(First Seen)                                              Info
    Name                          T W Ch  Packts Flags IP Range      Size  Ntwrks
  ! <no ssid>                     A O 005    235       0.0.0.0       181B      9
  ! linksys                       A N 006     40 F     192.168.1.1     0B  Pckets
  ! linksys                       A N 006     81 F     192.168.1.1     0B    760
  ! Home                          A Y 001     45  A4   10.0.1.10       0B  Cryptd
  ! <no ssid>                     A O 010    294       0.0.0.0       554B      5
  ! id-pb3-kurt-wlan              A N 003     44       0.0.0.0         0B    Weak
  ! win network                   A Y 006     18       0.0.0.0         0B      0
    linksys                       A N 006      1  U4   192.168.1.100   0B   Noise
    Probe Networks                P N ---      1       0.0.0.0         0B      0

                                                                           Elapsd
                                                                         00:01:05
Status
 Sorting by time first detected
 Found new network "linksys" bssid 00:06:25:DA:15:4B Crypt N Ch 6 @ 54.00 mbit
 Found new probed network "<no ssid>" bssid 00:04:23:89:EE:13
 ALERT: Probe response with 0-length SSID detected from 00:11:50:22:AB:92
Battery: unavailable
```

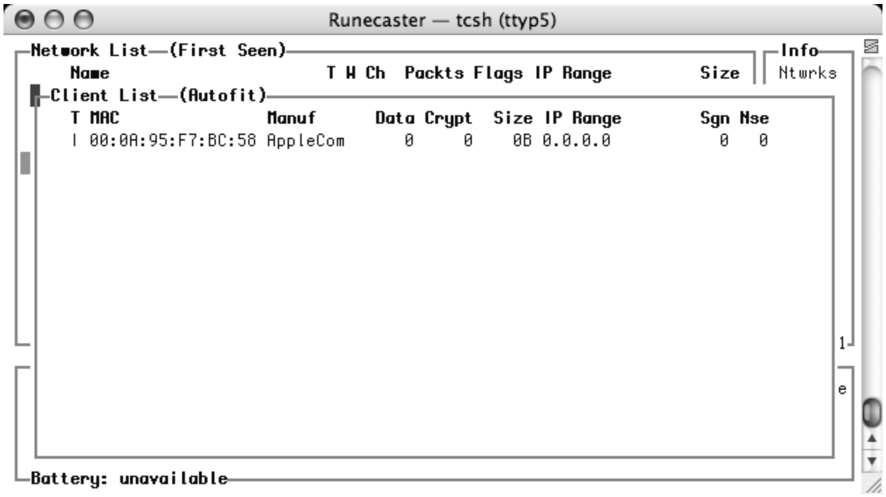
Figure 17-5. Linux kismet_client



```
~/src/kismet-devel
ÚÄNetwork ListÄÄ(First Seen)ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ¿ÚÄInfoÄÄÄ¿
3      Name                          T W Ch  Packts Flags IP Range        33 Ntwrks 3
3      2WIRE054                      A Y 006      1       0.0.0.0         33      3 3
3      <Melnibone>                   A Y 010     25       0.0.0.0         33 Pckets 3
3      <no ssid>                     P N ---      5       0.0.0.0         33     48 3
3                                                                         33 Cryptd 3
3                                                                         33      1 3
3                                                                         33   Weak 3
3                                                                         33      0 3
3                                                                         33  Noise 3
3                                                                         33      0 3
3                                                                         33 Discrd 3
3                                                                         33     40 3
3                                                                         33 Pkts/s 3
3                                                                         33      1 3
3                                                                         33        3
3                                                                         33        3
3                                                                         33 Elapsd 3
ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÙÄ00:01:17Ù
ÚÄStatusÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ¿
3  Connected to Kismet server version 3.1.0 build 20030914130805 on 10.0.1.10:23
3  Found new probed network "<no ssid>" bssid 00:06:25:3E:0D:3B                3
3  Sorting by time first detected                                               3
3                                                                               3
ÀÄBattery: unavailableÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÙ
```

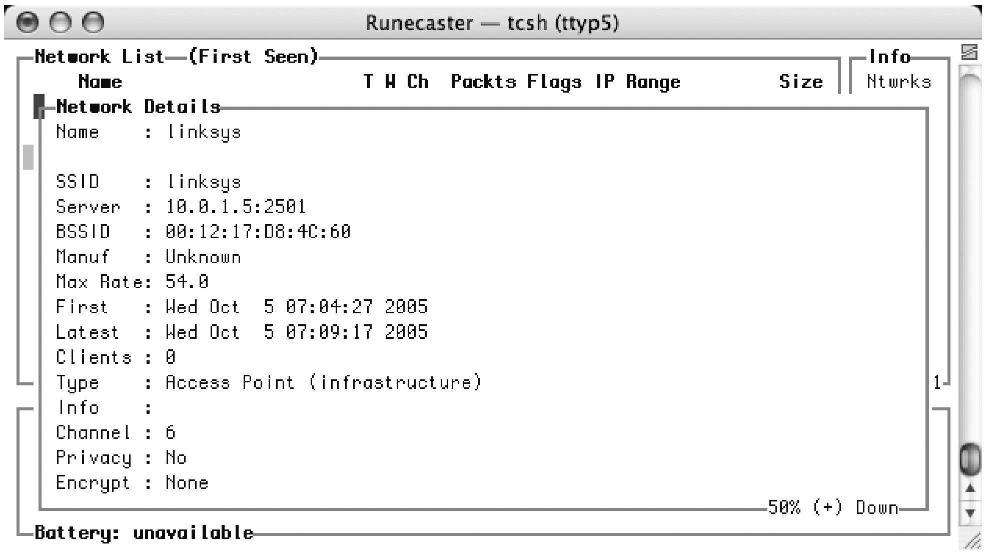
Figure 17-6. OS X kismet client

shows an example of the information available for a client that is using the wireless network. A wireless client is any device whose network traffic uses the wireless network. Thus, kismet will observe "wired" clients (clients that do not have a wireless network card) if they communicate with wireless clients or the base station.



```
Runecaster — tcsh (ttyp5)
Network List—(First Seen)                                            Info
     Name                    T W Ch  Packts Flags IP Range     Size  Ntwrks
  Client List—(Autofit)
     T MAC               Manuf       Data Crypt  Size IP Range       Sgn Nse
     | 00:0A:95:F7:BC:58 AppleCom       0     0    0B 0.0.0.0          0   0
                                                                          1
                                                                          e
Battery: unavailable
```

Figure 17-7. Press `i` on a highlighted SSID to view network information.



```
Network List—(First Seen)                                              Info
    Name                        T W Ch  Packts Flags IP Range        Size   Ntwrks
 Network Details
 Name    : linksys

 SSID    : linksys
 Server  : 10.0.1.5:2501
 BSSID   : 00:12:17:D8:4C:60
 Manuf   : Unknown
 Max Rate: 54.0
 First   : Wed Oct  5 07:04:27 2005
 Latest  : Wed Oct  5 07:09:17 2005
 Clients : 0
 Type    : Access Point (infrastructure)                                     1
 Info    :
 Channel : 6
 Privacy : No
 Encrypt : None
                                                              50% (+) Down
Battery: unavailable
```

Figure 17-8. Press `i` on a highlighted network to view client information.

Kismet distinguishes a network's wireless protocol (802.11b, 802.11g, 802.11a) and the presence of encryption, such as WEP or WPA.

Expanding Kismet's Capabilities

Mobile auditing counts as one of kismet's best points. It has two capabilities that support users who want to do some walking, biking, or driving to discover a wireless presence. The first capability falls under the category of "user-friendly"—sound and speech. The second capability, GPS data collection, has more utility for auditing networks.

Speech and sound are useful for discreet auditing because you can place a laptop or handheld device in a backpack, purse, or jacket pocket and monitor activity through an earpiece. Audio clues also aid war drivers; they provide feedback when it is more important to drive than to read the laptop screen. Thus, a speech engine makes the effort truly hands-free. The Festival engine is available from *http://www.cstr.ed.ac.uk/projects/festival/*. The readme and install files provide all of the information necessary to get things started.

Using GPS with kismet lets you add spatial information to the wireless network. This is especially important when you want to identify how far your wireless network propagates outside a building or even between floors of a building. Kismet does not include software to collect data from a GPS device; a different tool handles this job: GpsDrive. The comprehensive GpsDrive application, available at *http://gpsdrive.kraftvoll.at/index.shtml*, contains more than you will need for just using kismet. At its core, GpsDrive uses a daemon named *gpsd* to collect data from a serial port connection to a GPS device. Refer to Table 17-5 for some command-line options to get *gpsd* up and running.

Option	Description
`-D<n>`	Debug level. Higher values for *n* correspond to more verbose output.
`-S`	Port number. This is the TCP port number on which gpsd opens a listener. Do not confuse this with a listening port for the actual GPS device. Default is 2947.

Table 17-5. gpsd Command-line Options

TIP Compiling GpsDrive requires the usual triumvirate of `./configure; make; make install`. The only drawback is that a complete GTK development environment must be available. This may necessitate the installation of a half-dozen or so RedHat Package Managers (RPMs). Nevertheless, the process is simple once the environment is set up correctly.

Recording GPS information also enables you to take advantage of Kismet's mapping utility, gpsmap. The gpsmap program combines GPS coordinates and sniffed network information with maps of the target location.

Case Study: Networks Without Borders

Wardriving grew out of the same culture that spawned war dialing (see Chapter 18). Instead of looking for computers by randomly dialing phone numbers, wardriving looks for computers by wandering an area. The amount of information that becomes available ranges from solely the SSID to IP addresses, usernames, and passwords. In some cases, a network will even offer a DHCP address to the wandering wireless card. Obviously, the security implications are severe. The NetStumbler web site contains a map of North America that contains access points discovered by casual observers. Although this tool doesn't grab every username or password on the wireless network (check out AiroPeek or kismet for that), it provides a clear illustration of the pervasiveness of wireless networks and the necessity for strong protocols to support the security of these networks.

Simply being able to view the SSID does not mean that the wireless network is insecure. Network administrators can encrypt access with strong WEP implementations or layered cryptography implementations and lock down access based on a card's MAC address. A wireless network's security increases greatly when combined with a VPN implemented on a protocol such as IPsec. In fact, many network

Networks Without Borders *(continued)*

administrators place wireless access points in an untrusted segment of the network. Then they require legitimate users to authenticate to a centralized authentication server (such as LDAP or an Active Directory) or establish a VPN into the network. The ease in which unauthorized users can obtain a wireless signal outside of the physical boundaries of a building necessitates such steps. As we will see in the upcoming tools, there are more threats to the network than merely finding out its name, or SSID.

Case Study: WEP Insecurities

Wireless networks are not relegated to business offices and corporate networks. They can also pop up in residential areas, airports, and large retail establishments. Finding the presence of a wireless network does not necessarily have a security implication, but being able to view data does. In May 2002, an anonymous hacker reported finding wireless networks in several large department stores such as Best Buy, Wal-Mart, and Home Depot. Although it isn't clear whether credit card information was being transmitted unencrypted, this case does drive home the point that someone sitting in the parking lot could collect quite a few credit card numbers in a single day.

Even if the traffic is encrypted, WEP implementations are vulnerable to active and passive attacks that enable a third party to identity the WEP key by analyzing packets. Thus, it is not sufficient to rely only on WEP for data security. Vendors may claim that their WEP security is based on 40- or 64-bit encryption, but the truth here is slightly muddled. The secret key in both of these cases is a 40-bit value. The next 24 bits (which make up the 64-bit key) are part of the initialization vector (IV) that changes for each packet. Researchers from AT&T Labs and Rice University (*http:// www.cs.jhu.edu/~astubble/600.412/s-c-papers/wireless2.pdf*) discovered a method for breaking the IV generation scheme and discerning the WEP key based on passive monitoring of 5–6 million packets. At first, this number may appear large, but a partially loaded network easily generates this many packets in a few hours. University of Maryland researchers (*http://www.cs.umd.edu/~waa/wireless.pdf*) identified a similar weakness in WEP and vendor implementations.

Flooding the AP with de auth commands will greatly reduce the amount of time needed to do this.

WPA, now available on Windows XP systems, and the 802.11i protocol provide significant improvements over WEP. Additionally, many vendors have upgraded their firmware to silently squash any of the "weak" initialization vectors from being used by the card. So, while WEP should be a concern when installing a wireless network, its weaknesses should not be a detractor for most implementations. Combining wireless network access with a VPN or other encryption layer addresses most security concerns.