Sobald der Daemon läuft, läßt sich über das Dienstprogramm *wpa_cli* sein Betriebszustand abfragen, vorausgesetzt, man hat Zugriff auf die von ihm angelegte Kontrolldatei. Nach dem Start läßt sich so mit dem Befehl *status* eine Übersicht über die gegenwärtige Verschlüsselungskonfiguration anzeigen, und auch ein erneuter Scan der erreichbaren Netze kann mit dem Kommando *scan* erzwungen werden.

Eine Übersicht über alle verfügbaren Befehle, mit denen sich das Verhalten des Daemons in vielerlei Hinsicht beeinflussen läßt, zeigt der naheliegende Befehl *help* an.

4.5.6 Wardriving

Seit die WLAN-Technik Einzug in Privathaushalte und Firmen gefunden hat, bilden die kabellosen Netze ein breites Ziel für Angreifer aller Art. Kaum einem Anwender oder Administrator wäre je in den Sinn gekommen, eine frei nutzbare Netzwerkdose außerhalb des Firmengeländes zu installieren, dennoch sind viele Funknetze vollkommen ungesichert und ermöglichen damit völlig fremden Personen Zugriff auf das lokale Netz.

Der Begriff des »Wardriving« geistert dabei regelmäßig durch die Boulevardmagazine, die gerade bei Computerthemen nicht unbedingt durch fundierte Berichterstattung glänzen. Dabei malen die Redakteure ein sehr eindimensionales Bild über böse Hacker, die in den Rechnern unbescholtenen Bürgern randalieren möchten. Nicht nur, daß hier – wie leider inzwischen allgemein üblich – der Terminus des Hackers mit dem dem Crackers vermischt wird. Die Beweggründe der Wardriver sind vielschichtig und oftmals nicht einmal bösartig. Der martialische Begriff selbst läßt sich als »Wireless Access Revolution Driving« aufschlüsseln, teilweise ist auch das Synonym »WiLDering« für »Wireless LAN Driving« in Gebrauch.

Im allgemeinen bedienen sich Wardriver eines Notebooks, geeigneter WLAN-Hardware (PCMCIA-Karte, USB, integrierter Empfänger) und oft auch eines GPS-Empfängers. Je nach technischer Versiertheit und Aufwand finden auch externe WLAN-Antennen Verwendung, die außerhalb des Autos – dessen Fahrgastzelle wie ein abschirmender Faradayscher Käfig wirkt – angebracht wird. Mit dieser Installation machen sich die Wardriver auf dem Weg und fahren ihre Strecke ab. Spezielle Software auf dem Notebook horcht nun über die WLAN-Antenne (passiv) nach erreichbaren Funknetzen, während der GPS-Empfänger laufend die Position des Fahrzeugs feststellt. Gefundene Netze werden mit ihren Daten und ihrer Position verzeichnet, so daß sie auf einer Karte dargestellt werden können.

Doch was ist der Antrieb hinter den Lauschtouren? Der Personenkreis, der solchen Tätigkeiten nachgeht, läßt sich grob in drei Gruppen einteilen:

1. Statistiker und Aufklärer:
   Die erste Gruppe ist lediglich an der Verbreitung der Funknetze interessiert, und in der Tat ist es interessant, die wechselnde Dichte an Netzen in verschiedenen Straßen und Gegenden einer Stadt zu beobachten. Ebenso lassen sich die Anzahl der gesicherten und ungesicherten Netze in ein Verhältnis setzen, das Rückschlüsse auf das Gefährdungs- und Sicherheitsbewußtsein der Bevölkerung zuläßt. Oft möchten Mitglieder dieser Gruppe auch auf Sicherheitsprobleme hinweisen, die gerade – aber nicht nur – bei privaten Installationen wenig Beachtung finden.

2. Huckepacksurfer
   Die zweite Gruppe sucht einen funktionierenden Internetzugang, den sie jedoch nicht »mißbrauchen«, sondern nur kurz ausborgen möchte, um beispielsweise ihr E-Mail-Postfach abzufragen oder eine vergessene Wegbeschreibung abzurufen. Diese ungebetenen Gäste bemühen sich in der Regel, keinen Schaden durch übermäßige Nutzung zu verursachen und lassen auch die Daten des Besitzers in Frieden – schließlich wollen sie nicht an diese gelangen, sondern an ihre eigenen. Wie jemand, der den Hauseigentümer höflich um die Nutzung der Toilette bittet, wollen sie ihr digitales Örtchen sauber und natürlich auch ohne gestohlenes Rasierwasser hinterlassen.
3. Böswillige Angreifer
   Natürlich darf auch nicht verschwiegen werden, daß längst nicht alle Wardriver solche Vorsicht walten lassen; Im Übergangsbereich zur Cracker-Szene finden sich auch weitaus unangenehmere Zeitgenossen, die gezielt Netze angreifen, um entweder an die dort gespeicherten Daten zu gelangen oder die infiltrierte Struktur als Sprungbrett für Attacken auf weitere Systeme zu nutzen. Schnell unterhöhlt ein ungesicherter Access-Point die penibel und restriktiv gepflegte Firewall, die das Unternehmen zum Internet abschirmt. Benutzt der Cracker das Unternehmensnetz als Sprungbrett, ist er schon über alle Berge, noch bevor die Logdateien des Opfers ausgewertet sind und auf den Besitzer des infiltrierten WLANs verweisen.

Ein Schutz vor reinen Wardrivern ist kaum möglich. Durch das rein passive Auffangen der ausgesandten Signale können diese bereits einen Access-Point feststellen. Auch die oftmals als »Stealth-Modus« betitelte Funktion zum Ausblenden der ESSID aus den Beacon-Paketen macht das WLAN nicht unsichtbar. Sie entfernt lediglich den klar lesbaren Netzwerknamen aus den Signalpaketen des Zugangsknotens, so daß es Beobachtern erschwert wird, sich mit dem Netz zu assoziieren. Der »Schutz« wird jedoch ausgehebelt, sobald sich ein berechtigter Teilnehmer in das Netz einklinkt. Er überträgt den vorher noch versteckten Namen im Klartext, das Netz ist enttarnt. Das Verstecken von WLANs fällt daher in den Bereich der »Security through Obscurity«, echte Sicherheit bringt nur eine Verschlüsselung des Datenverkehrs über WLAN-eigene Maßnahmen wie WEP oder bevorzugt WPA, oder auch IPSec und VPN-Techniken.

Neben dem klassischen Wardriving, also dem Aufspüren von Funknetzen aus einem Auto heraus, gibt es noch weitere Schwesterdisziplinen. So horchen Hacker beim Warflying oder Warstorming ihre Umgebung aus einem Flugzeug oder Hubschrauber nach Netzwerken ab, etwas weniger spektakulär geht es beim Warwalking zu. Hier findet die gesamte Tätigkeit zu Fuß statt, meist mit einem Notebook im Rucksack, oft werden auch entsprechend ausgerüstete PDAs genutzt. Eine weitere Subkultur ist das sogenannte WarChalking. Angelehnt an Gaunerzinken kennzeichnen Teilnehmer dieser Gruppe hier gefundene Netze über spezielle Kreidesymbole an Hauswänden, Bürgersteigen oder anderen Objekten in direkter Nachbarschaft des entdeckten Access-Points. Dabei kennzeichnet ein geschlossener Kreis ein gesichertes WLAN (beispielsweise WEP, WPA), zwei entgegengesetzte Kreishälften dagegen zeigen ein offenes WLAN an. Manchmal werden auch ESSID und WEP-Kennwort daneben festgehalten. In manchen Städten ist es auch üblich, ein verfügbares offenes WLAN mit dem Wort »bellum« (lateinisch für Krieg) zu kennzeichnen. Wird das Netzwerk schließlich abgesichert, streichen Warchalking-Kundige das Wort aus und ersetzen es durch Frieden (»pax«).

Der Sniffer Kismet

Wer sich selbst einmal ein Bild über diese diffus dargestellte Tätigkeit machen möchte, der benötigt nicht viel. Ein normales Notebook samt installiertem Linux und eine WLAN-Karte, die den Monitor-Modus unterstützt, reichen aus, um sich auf den Streifzug nach offenen und verschlossenen Netzen zu machen.

Unter Linux hat sich die Software *Kismet* etabliert, die auf Netzschicht 2 als Sniffer und Detektor für WLANs dient. Die Palette der unterstützten Karten ist dabei weit gefaßt. Sofern die Karte in den RFMon-Modus versetzt werden kann, ist ein Einsatz mit dem Programm möglich. Dabei leitet die Karte alle Pakete weiter, die sie aus dem Äther fischen kann, unabhängig von der eigenen Netzwerkzugehörigkeit. Oftmals hängt dies auch vom verwendeten Treiber ab. Die bekannten Prism2-Karten unterstützen dieses Verfahren nicht mit dem kerneleigenen Treiber, im Zusammenspiel mit dem HostAP- oder einem modifizierten Modul lassen jedoch auch sie sich zum stöbern einsetzen. Ihre Nachfolgerhardware mit Prism54-Treiber ist von Haus aus dazu fähig, seit kurzem auch die in Centrino-Notebooks integrierten Intel-WLAN-Chipsätze.

Kismet läßt sich von der Projekthomepage *http://www.kismetwireless.net/* im GPL-lizenzierten Quelltext herunterladen, in Paketform liegt das Programm jedoch allen bekannten Distributionen bei. Kismet verfolgt eine flexibel gegliederte Client/Server-Architektur. Der Kismet-Server verwaltet die Hardware, über die das System sein Ohr im Äther hält und bereitet die empfangenen Pakete auf. Über einen Netzwerk-Socket können nun Client-Programme Kontakt mit ihm aufnehmen, um die aufgefangenen Informationen darzustellen; Diese Clients können wiederum die Informationen mehrerer Server in ihrer Präsentation vereinen.

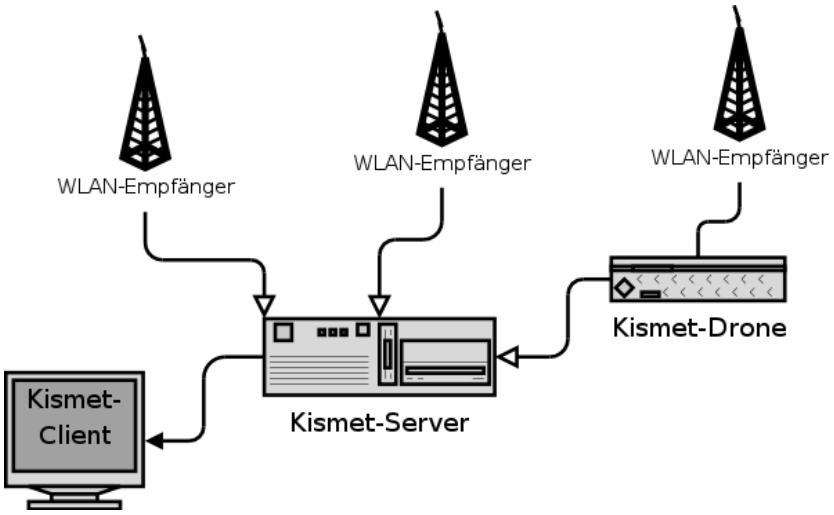

Bild 4.52: Kismet besteht aus mehreren Komponenten, die in einer Client/Server-Architektur ineinandergreifen

Die Konfiguration des Servers erfolgt in der Datei */etc/kismet/kismet.conf*. Sie definiert die WLAN-Geräte, über die das Programm verfügen soll, und legt auch das allgemeine Verhalten des Servers fest. Aufgrund seiner Interaktion mit der WLAN-Hardware muß Kismet mit root-Rechten gestartet werden, die es jedoch aus Sicherheitsgründen alsbald ablegen möchte; es wechselt dazu seine Benutzeridentität hin zu einer gewöhnlichen Zugangskennung, die über die Einstellung *suiduser* festgelegt wird. In der Regel ist dies der normale Benutzeraccount des Administrators, zu dem das Programm automatisch nach dem Start wechselt.

Ein einzelner Kismet-Server kann mehrere WLAN-Geräte gleichzeitig abfragen. Die Datenquellen werden mit dem Schlüsselwort *source* definiert; durch Kommata getrennt erwartet die Konfigurationsdatei hier drei Werte, die den Treiber, den Gerätenamen und eine eindeutige Identifikation der Quelle erwarten.

```
source=ipw2100,wlan0,centrino
source=prism54g,wlan1,prism
```

Diese beiden Zeilen definieren gleich zwei Datenquellen: Zum einen kann Kismet auf die integrierte WLAN-Funktionalität des Centrino-Notebooks zugreifen, zum anderen liefert auch die zusätzlich eingesteckte WLAN-Karte mit Prism54-Chip ihre Pakete an das Spürprogramm. Welche Treiber durch Kismet unterstützt werden, listet das Programm ausführlich in der beiliegenden README-Datei auf.

Die Verwendung mehrerer Paketquellen hat Vorteile: Da 802.11b/g auf mehreren Kanälen funkt, muß Kismet durch die Frequenzen wechseln, um alle denkbaren Sender abzudecken. Dieses Channelhopping wird mit der Option *channelhop=true* aktiviert und mit der Einstellung *channelvelocity=5* reguliert, die Zahl gibt dabei die Anzahl an Kanälen an, die in einer Sekunde durchlaufen werden. Durch *channelsplit=true* verteilt Kismet die Kanäle auf die Karten, so daß die Gefahr sinkt, ein aktives WLAN zu verpassen.

Die Funkkanäle, die Kismet überprüft, werden über die *defaultchannel*-Direktiven festgelegt:

```
# 802.11b
defaultchannels=IEEE80211b:1,7,13,2,8,3,14,9,4,10,5,11,6,12
# 802.11g
defaultchannels=IEEE80211g:1,7,13,2,8,3,14,9,4,10,5,11,6,12
```

Dabei läßt sich das Spektrum für jedes Mitglied der WLAN-Familie separat festlegen. Die Reihenfolge resultiert aus der Überlappung der Kanäle. Wer der automatischen Verteilung der Kanäle nicht traut, kann sie auch manuell auf die einzelnen WLAN-Endgeräte verteilen:

```
# Die Prism-Karte überwacht laufend Kanal 6
sourcechannels=prism:6
```

Karten, deren Verhalten hier nicht explizit festgelegt wird, wechseln weiterhin durch die Standardkanäle.

Neben WLAN-Paketen verarbeitet Kismet auch GPS-Signale. Über die Satelliten des weltweiten Positionssystems kann ein Empfänger so weltweit seine eigene – und damit auch die der gefundenen Funknetze – bestimmen. GPS-Empfänger zur Nutzung mit einem Notebook oder PDA sind inzwischen nicht mehr teuer und verfügen meist über eine serielle, USB- oder Bluetooth-Anbindung. In nahezu jedem Fall entsprechen sie jedoch dem NMEA-Standard, der das Protokoll genau festlegt. Auf diese Weise läßt sich jeder Empfänger auch unter Linux nutzen, da die Geräte in der Regel eine serielle Schnittstelle nachahmen, entweder über einem im USB-Gerät integrierten USB-nach-Seriell-Konverter oder über die entsprechenden Bluetooth-Standards. Unter Linux verwaltet der GPS-Daemon *gpsd* die kleinen Geräte und erlaubt so mehreren Programmen gleichzeitig, über einen Netzwerkport die Positionsdaten abzufragen.

Verfügt das Notebook über GPS-Empfang, fragt Kismet den laufenden GPS-Daemon ab, vorausgesetzt, diese Funktion ist in der Konfigurationsdatei aktiviert:

```
# Do we have a GPS?
gps=true
# Host:port that GPSD is running on.  This can be localhost OR remote!
gpshost=localhost:2947
```

Der Kismet-Server führt sowohl Positions- als auch WLAN-Informationen zusammen, und stellt sie den angebundenen Kismet-Clients zur Verfügung. Wer sich dieser Daten bedienen darf, legt eine eigene Sektion in der Konfigurationsdatei fest:

```
# Port to serve GUI data
tcpport=2501
# People allowed to connect, comma seperated IP addresses or network/mask
# blocks.  Netmasks can be expressed as dotted quad (/255.255.255.0) or as
# numbers (/24)
allowedhosts=127.0.0.1
# Maximum number of concurrent GUI's
maxclients=5
```

Der Client selbst verfügt über die eigene Konfigurationsdatei */etc/kismet/kismet_ui.conf*. Die wichtigste Einstellung hier ist wohl *host=localhost:2501*, mit der der zu kontaktierende Server angegeben wird.

Als Besonderheit bietet Kismet die Möglichkeit, Ereignisse über Tonausgaben mitzuteilen. Diese nützliche Fähigkeit (gerade für Wardriver, die sich so auf die Straße konzentrieren können) baut auf zwei Standbeinen auf. Zum einen spielt Kismet vordefinierte Sounddateien als Signal ab, zum anderen bedient sich das Programm des Sprachsynthesizers *Festival*, um flexibel Auskunft über neue Netze zu geben.

```
# Does the GUI use sound?
# NOT to be confused with "sound" option later, which is for the SERVER to make
# noise on whatever host it's running on.
sound=true
# Path to sound player
soundplay=/usr/bin/aplay
# Optional parameters to pass to the player
```

```
# soundopts=--volume=.3
# New network found
sound_new=/usr/share/kismet/wav/new_network.wav
# Wepped new network
# sound_new_wep=${prefix}/com/kismet/wav/new_wep_network.wav
# Network traffic sound
sound_traffic=/usr/share/kismet/wav/traffic.wav
# Network junk traffic found
sound_junktraffic=/usr/share/kismet/wav/junk_traffic.wav
# GPS lock aquired sound
# sound_gpslock=/usr/share/kismet/wav/foo.wav
# GPS lock lost sound
# sound_gpslost=/usr/share/kismet/wav/bar.wav
# Alert sound
sound_alert=/usr/share/kismet/wav/alert.wav
```

Sowohl Kismet-Server als auch der Client können diese Sounds ausgeben; in der Regel wird der Anwender aber die Ausgabe aus der Clientsoftware bevorzugen. Ausnahmen sind jedoch embedded Systeme, die autonom und ohne viel Ballast arbeiten sollen. Über die Wahl der Option *soundplay* läßt sich ein Ausgabeprogramm wählen, das optimal mit der Anwendungsumgebung und damit mit vorhandenen Sound-Daemons wie *artsd* oder *esd* zusammenarbeitet.

Über Festival generiert Kismet auch dynamische Mitteilungen. Dabei bedient sich das Programm der beiden Vorlagen *speech_encrypted* und *speech_unencrypted*, sobald sich ein neues verschlüsseltes oder offenes WLAN in Reichweite befindet. Der dort abgelegte Satz wird dem Sprachsynthese-Programm übergeben, nachdem vorher die Platzhalter durch die ermittelten Werte des neuen Netzes ersetzt wurden:

```
# Soll künstliche Sprache verwendet werden?
speech=true
# Pfad zum Syntheseprogramm "Festival"
festival=/usr/bin/festival
# Buchstabiertypus (speech/nato/spell)
speech_type=nato
# %b wird durch die BSSID (MAC) des Netzes ersetzt
# %s wird durch die SSID - den Netzwerknamen - ersetzt
# %c entspricht dem Funkkanal
# %r ist die maximale Datenrate
speech_encrypted=New network detected, s.s.i.d. %s, channel %c, network
encrypted.
speech_unencrypted=New network detected, s.s.i.d. %s, channel %c, network open.
```

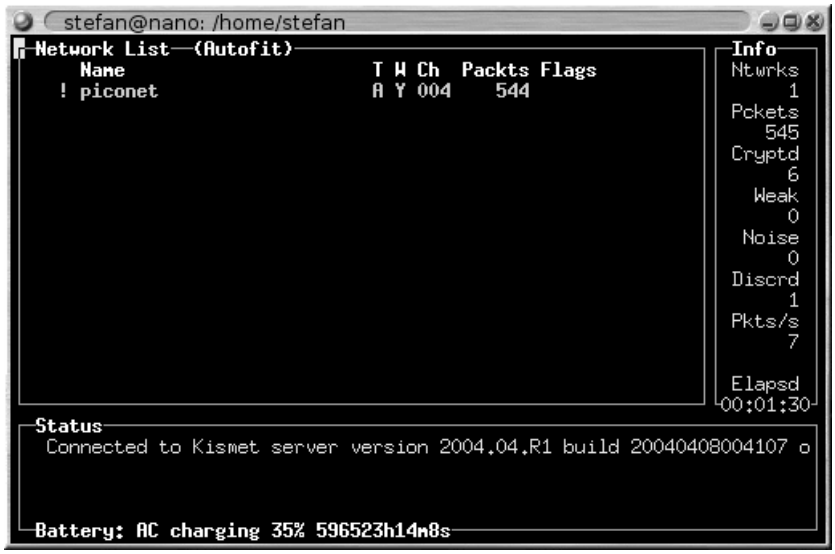

Bild 4.53: Der »kismet_client« listet die gefundenen Funknetze übersichtlich auf

Gestartet wird das Spürprogramm als Root mit dem Befehl *kismet*. Dieses Skript startet zuerst den Server (*kismet_server*) und dockt sogleich das Kismet-Benutzerinterface (*kismet_client*) an diesen an.

Die Benutzeroberfläche gliedert sich in drei Hauptsegmente. Im großen Hauptfenster listet das Programm alle gefundenen WLAN-Netze auf, während am rechten Rand Statusinformationen über die Anzahl der gefundenen Netze, aufgefangene Pakete, sowie über das Channel-Hopping der überwachten WLAN-Schnittstellen ihren Platz finden. Der untere Bereich des Schirms dient zur Anzeige diverser Statusmitteilungen, wie dem Auftreten anormaler WLAN-Aktivitäten anderer Teilnehmer.

Um weitere Informationen über die entdeckten Netze anzuzeigen, muß zunächst der Sortiermodus gewechselt werden. Nach einem Druck auf die Taste [s] läßt sich ein neuer Sortiermodus wählen. Danach kann mit den Pfeiltasten durch die entdeckten Netze navigiert werden, die Eingabetaste ruft weitere Informationen zu einem bestimmten Netz ab.

Die Typen der gefundenen WLAN-Stationen werden über ihre Farbe und ihren Kennbuchstaben und der T-Spalte angezeigt. Grüne Einträgen zeigen WEP-gesicherte Netze an, während gelbe Netze kein WEP benutzen; bei rotgefärbten Einträgen hat Kismet vermutlich sogar festgestellt, daß der Access-Point in seiner Standardkonfiguration betrieben wird.

Soll ein Netz näher untersucht werden, kann das Channel-Hopping unterbunden werden, indem der Server mit *L* (Lock) auf ein bestimmtes Netz eingerastet wird. Die Erfassungsrate der Pakete wird dabei sofort ansteigen, da keine Pakete mehr durch das Wechseln der Kanäle verlorengehen. Die Taste [d] zeigt die Zeichenketten an, die Kismet aus dem Datenstrom erkennen kann; genauere Analysen sind mit Ethereal möglich, das auch die Netzdumps des Kismet-Sniffers lesen kann.

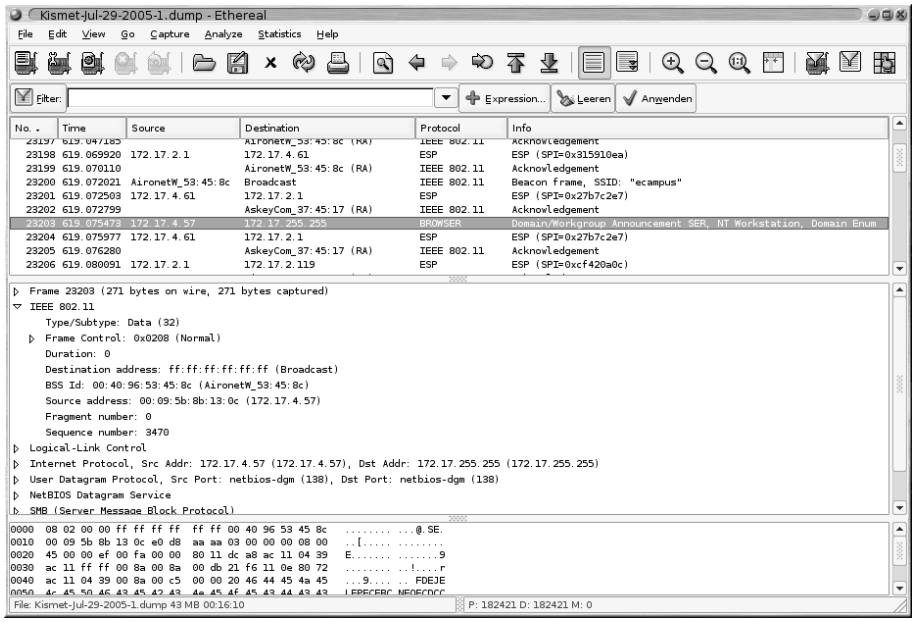

Bild 4.54: Ethereal analysiert die durch Kismet aufgefangenen Pakete

Natürlich sind die Pakete für Ethereal nur lesbar, wenn das WLAN unverschlüsselt ist. Ist der WEP-Schlüssel des Netzes bekannt, kann Kismet die erfaßten Pakete vor dem Speichern dekodieren. Dazu wird der Schlüssel in die *kismet.conf* eingetragen:

```
wepkey=00:DE:AD:C0:DE:00,FEEDFACEDEADBEEF0102030405060708090O
```

Empfängt Kismet nun Pakete von der angegebenen BSSID (MAC-Adresse), wendet es den angegebenen Schlüssel darauf an. Aus Sicherheitsgründen übermittelt es die Schlüssel nicht an die Clients; ist dies trotzdem gewünscht, kann das mit *allowkeytransmit=true* aktiviert werden.

WLANs kartographieren

Um die gefundenen WLANs grafisch darzustellen, findet oft das Programm *GPSDrive* (http://www.gpsdrive.cc/) Verwendung. Findet es bei seinem Start einen laufenden Kismet-Server, meldet es sich automatisch an diesem als Client an. Gefundene WLANs werden automatisch in die MySQL-Datenbank eingetragen und auf der Karte plaziert, so daß ein spannender Überblick über die WLAN-Versorgung bestimmter Regionen entsteht.

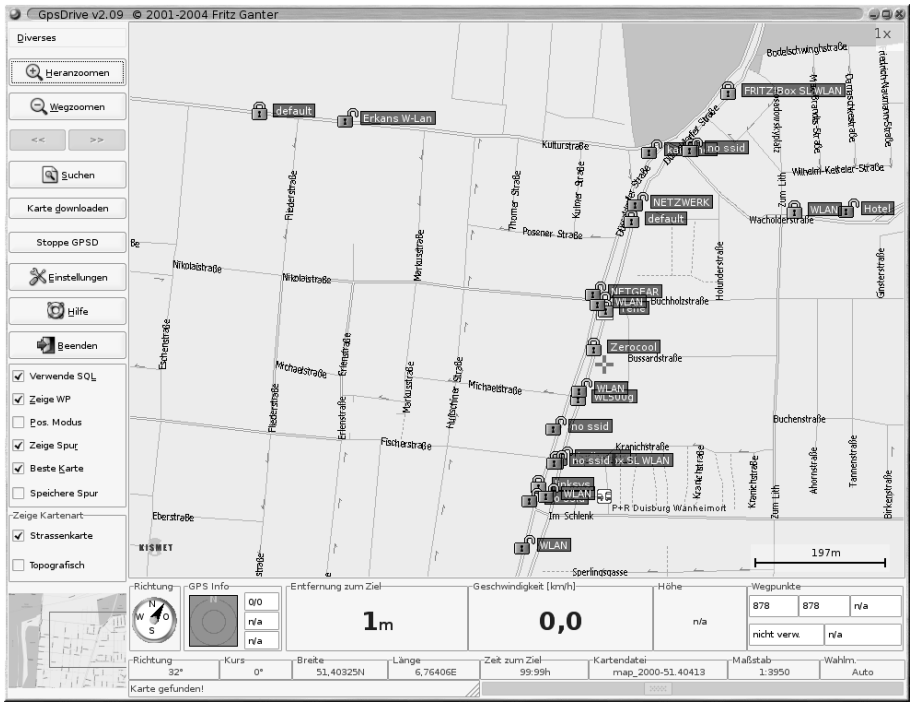

Bild 4.55: GPSDrive agiert als Kismet-Client und stellt gefundene WLANs auf einer Straßenkarte dar