

WIRELESS HACKS™

2nd
Edition

*Tips & Tools for Building, Extending,
and Securing Your Network*



O'REILLY®

Rob Flickenger & Roger Weeks

Wireless Hacks™, Second Edition

by Rob Flickenger and Roger Weeks

Copyright © 2006 O'Reilly Media, Inc. All rights reserved.
Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly Media, Inc. books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*safari.oreilly.com*). For more information, contact our corporate/institutional sales department: (800) 998-9938 or *corporate@oreilly.com*.

Editor:	Brian Sawyer	Production Editor:	Philip Dangler
Series Editor:	Rael Dornfest	Cover Designer:	Ellie Volckhausen
Executive Editor:	Dale Dougherty	Interior Designer:	David Futato

Printing History:

September 2003: First Edition.

November 2005: Second Edition.

Nutshell Handbook, the Nutshell Handbook logo, and the O'Reilly logo are registered trademarks of O'Reilly Media, Inc. "Hacks," "The Hacks Series," and related trade dress are trademarks of O'Reilly Media, Inc. The association between the image of a wire cutter and the topic of wireless technology is a trademark of O'Reilly Media, Inc.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and O'Reilly Media, Inc. was aware of a trademark claim, the designations have been printed in caps or initial caps. All other trademarks are property of their respective owners.

While every precaution has been taken in the preparation of this book, the publisher and authors assume no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein.

Small print: The technologies discussed in this publication, the limitations on these technologies that technology and content owners seek to impose, and the laws actually limiting the use of these technologies are constantly changing. Thus, some of the hacks described in this publication may not work, may cause unintended harm to systems on which they are used, or may not be consistent with applicable user agreements. Your use of these hacks is at your own risk, and O'Reilly Media, Inc. disclaims responsibility for any damage or expense resulting from their use. In any event, you should take care that your use of these hacks does not violate any applicable laws, including copyright laws.



This book uses RepKover™, a durable and flexible lay-flat binding.

ISBN: 0-596-10144-9

[M]



HACK

#1

Set Up Bluetooth on Linux

Linux kernels from 2.6 onward have easy-to-use tools for Bluetooth.

Prior to the release of the 2.6 Linux kernel, getting Bluetooth support involved compiling your own kernel as well as the necessary utilities. There were also multiple Bluetooth stacks available, each with their own features, adapter support, and quirks. In 2.6, the BlueZ stack was crowned as the officially supported way to use Bluetooth in Linux, and that's the focus of this hack.

First, make sure you have a supported Bluetooth adapter. You used to be able to find a reasonably current list of BlueZ-supported hardware at <http://www.holtmann.org/linux/bluetooth/devices.html>. However, as of March 2005, this information has been removed because of threatened legal action from the Bluetooth SIG. What this basically means is that the association of companies who maintain the Bluetooth standard don't want anyone to advertise that their devices are *compliant* with Linux unless you pay the SIG a lot of money and fill out a bunch of paperwork. So, you're on your own here. Probably the best place to get advice is in the BlueZ Users mailing list, which can be found at <http://www.bluez.org/lists.html>.

Next, you'll need to make sure that your kernel has Bluetooth support enabled. All distributions shipping the 2.6 kernel have Bluetooth support. 2.4 kernels shipped with both the Red Hat 9.0 and Debian Sarge distributions already include Bluetooth support. You can test your kernel for Bluetooth support by running `modprobe rfcomm` as root. If the `modprobe` fails, you'll need to install the packages that support Bluetooth.

Red Hat and Fedora users should install these packages using *yum* or *rpm*. This assumes you're using GNOME as your window manager:

```
yum install bluez-utils gnome-bluetooth
```

Likewise, Debian and Ubuntu users should install using *apt*:

```
apt-get install bluez-utils gnome-bluetooth
```

This next bit is for UART-based (that is, non-USB) devices only, so if you're using a USB Bluetooth adapter, you can skip ahead. Serial-style USB devices, which include serial dongles and PCMCIA cards, need to be explicitly *attached* to the Bluetooth host controller interface, using the *hciattach* utility. When you connect the device, the appropriate kernel driver might be loaded automatically, leaving a log entry in `/var/log/messages`.

If you're using a UART-based device, you may see a reference to a `/dev/ttySn` serial device, where *n* is some integer. In any event, you can try attaching the device to the Bluetooth host controller device by running `/sbin/hciattach /`

`dev/ttySn` any from the command line. Like any good Unix utility, you know that `hciattach` worked if it returns without printing anything. If it doesn't work, make sure you have the right device and check the manpage for other options.

Assuming that the `hciattach` command did work, you will want to add a reference to this device to your `/etc/bluetooth/uart` file, so that the device can be appropriately attached to the Bluetooth host controller interface at boot time. If this file doesn't exist, create it. Add a single line to this file that reads `/dev/ttySn` any, replacing `n` with the appropriate serial device number.

Now that you have everything installed, plug in your Bluetooth adapter and try running `/etc/init.d/bluetooth start` as root. In Debian and Ubuntu, start Bluetooth with `/etc/init.d/bluez-utils start`. You should see some appropriate status messages in your `/var/log/messages`. Assuming everything works, you might want to add the Bluetooth script to the appropriate `rc.d` directory for your default run level with the `chkconfig` utility or via a manual symlink. Chances are good your package install has already added this for you, but it's a good idea to check.

Now run `hciconfig` from the command line. You should see something like this:

```
hci0:  Type: USB
      BD Address: 00:11:22:33:44:55 ACL MTU: 192:8  SCO MTU: 64:8
      UP RUNNING PSCAN ISCAN
      RX bytes:99 acl:0 sco:0 events:13 errors:0
      TX bytes:296 acl:0 sco:0 commands:12 errors:0
```

If you don't see anything like this, make sure that `hcid` is running and that there aren't any error messages in `/var/log/messages`. The `BD Address` shown is the unique Bluetooth identifier for your adapter, much like an Ethernet MAC address.

Now, bring another Bluetooth device within range of your computer, and make sure that the device is visible to Bluetooth scans. Then, run `hcitool scan` from the command line. It might take up to 15 or 20 seconds to complete its scan, and then it should display something like this:

```
$ hcitool scan
Scanning ...
    00:99:88:77:66:55      Nokia3650
```

You can now test the device to see which services it supports, using `sdptool browse 00:99:88:77:66:55`. You should see a lengthy list of supported services, providing information that can be used to configure access to those services.