CHAPTER SIX

SSH

Hacks #66–71

Of all the types of hacks I encountered when preparing this book, it became obvious early on that *ssh* would require its own chapter. The *ssh* tool provides a very flexible and cryptographically secure method for connecting data streams together between machines over a network. Since the command line on a Linux system essentially consists of reading data from (and writing data to) files and pipelines, *ssh* makes it possible to sling data around a network more or less as if everything were taking place on a single machine. It does this in a fast, safe, and intuitive way, and makes for some very interesting (and powerful) hacks.

There are a couple of versions of *ssh* available for Linux. We'll assume that you're using OpenSSH v3.4p1 or later in the examples for this section. OpenSSH ships with all major Linux distributions, and is available at *http://www.openssh.com/*.

Quick Logins with ssh Client Keys

Using ssh keys instead of password authentication to speed up and automate logins

When you're an admin on more than a few machines, being able to navigate quickly to a shell on any given server is critical. Having to type "ssh my.server.com" (followed by a password) is not only tedious, but it breaks one's concentration. Suddenly having to shift from "where's the problem?" to "getting there" and back to "what's all this, then?" has led more than one admin to premature senility. It promotes the digital equivalent of "why did I come into this room, anyway?" (In addition, the problem is only made worse by /usr/games/fortune!)

At any rate, more effort spent logging into a machine means less effort spent solving problems. Recent versions of *ssh* offer a secure alternative to endlessly entering a password: public key exchange.