Preface

I wrote this book because I believe you should be able to learn what your computer does. You should be able to make your software do what you want it to do (within the reasonable limits of its capabilities, of course). The key to attaining this power lies in understanding the fundamentals of what the software does and how it works, and that's what this book is all about. You should never have to fight with a computer.

Linux is a great platform for learning because it doesn't try to hide anything from you. In particular, most system configuration can be found in plaintext files that are easy enough to read. The only tricky part is figuring out which parts are responsible for what and how it all fits together.

Who Should Read This Book?

Your interest in learning how Linux works may have come from any number of sources. In the professional realm, operations and DevOps folks need to know nearly everything that you'll find in this book. Linux software architects and developers should also know this material in order to make the best use of the operating system. Researchers and students, often left to run their own Linux systems, will also find that this book provides useful explanations for *why* things are set up the way they are.

Then there are the tinkerers—people who just love to play around with their computers for fun, profit, or both. Want to know why certain things work while others don't? Want to know what happens if you move something around? You're probably a tinkerer.

Prerequisites

Although Linux is beloved by programmers, you do not need to be a programmer to read this book; you need only basic computer-user knowledge. That is, you should be able to bumble around a GUI (especially the installer and settings interface for a Linux distribution) and know what files and directories (folders) are. You should also be prepared to check additional documentation on your system and on the Web. As mentioned earlier, the most important thing you need is to be ready and willing to play around with your computer.

How to Read This Book

Building the requisite knowledge is a challenge in tackling any technical subject. When explaining how software systems work, things can get *really* complicated. Too much detail bogs down the reader and makes the important stuff difficult to grasp (the human brain just can't process so many new concepts at once), but too little detail leaves the reader in the dark and unprepared for later material.

I've designed most chapters to tackle the most important material first: the basic information that you'll need in order to progress. In places, I've simplified things in order to keep focus. As a chapter progresses, you'll see much more detail, especially in the last few sections. Do you need to know those bits right away? In most cases, no, as I often note. If your eyes start to glaze over when faced with a lot of extra details about stuff that you only just learned, don't hesitate to skip ahead to the next chapter or just take a break. The nitty-gritty will still be there waiting for you.

A Hands-On Approach

However you choose to proceed through this book, you should have a Linux machine in front of you, preferably one that you're confident abusing with experiments. You might prefer to play around with a virtual installation—I used VirtualBox to test much of the material in this book. You should have superuser (root)

access, but you should use a regular user account most of the time. You'll mostly work at the command line, in a terminal window or a remote session. If you haven't worked much in this environment, no problem; Chapter 2 will bring you up to speed.

Commands in this book will typically look like this:

```
$ ls /
[some output]
```

Enter the text in bold; the non-bolded text that follows is what the machine spits back. The `$` is the prompt for your regular user account. If you see a `#` as a prompt, you should be superuser. (More on that in Chapter 2.)

How This Book is Organized

I've grouped the book's chapters into three basic parts. The first is introductory, giving you a bird's-eye view of the system and then offering hands-on experience with some tools you'll need for as long as you run Linux. Next, you'll explore each part of the system in more detail, from device management to network configuration, following the general order in which the system starts. Finally, you'll get a tour of some pieces of a running system, learn some essential skills, and get some insight into the tools that programmers use.

With the exception of Chapter 2, most of the early chapters heavily involve the Linux kernel, but you'll work your way into user space as the book progresses. (If you don't know what I'm talking about here, don't worry; I'll explain in Chapter 1.)

The material here is meant to be as distribution-agnostic as possible. Having said this, it can be tedious to cover all variations in systems software, so I've tried to cover the two major distribution families: Debian (including Ubuntu) and RHEL/Fedora/CentOS. It's also focused on desktop and server installations. There is a significant amount of carryover into embedded systems, such as Android and OpenWRT, but it's up to you to discover the differences on those platforms.

What's New in the Second Edition?

The first edition of this book dealt primarily with the user-centric side of a Linux system. It focused on understanding how the parts worked and how to get them humming. At that time, many parts of Linux were difficult to install and configure properly.

This is happily no longer the case thanks to the hard work of the people who write software and create Linux distributions. With this in mind, I have omitted some older and perhaps less relevant material (such as a detailed explanation of printing) in favor of an expanded discussion of the Linux kernel's role in every Linux distribution. You probably interact with the kernel more than you realize, and I've taken special care to note where.

Of course, so much of the original subject matter in this book has changed over the years, and I've taken pains to sort through the material in the first edition in search of updates. Of particular interest is how Linux boots and how it manages devices. I've also taken care to rearrange material to match the interests and needs of current readers.

One thing that hasn't changed is the size of this book. I want to give you the stuff that you need to get on the fast track, and that includes explaining certain details along the way that can be hard to grasp, but I don't want you to have to become a weightlifter in order to pick up this book. When you're on top of the important subjects here, you should have no trouble seeking out and understanding more details.

I've also omitted some of the historical information that was in the first edition, primarily to keep you focused. If you're interested in Linux and how it relates to the history of Unix, pick up Peter H. Salus's *The Daemon,*

the Gnu, and the Penguin (Reed Media Services, 2008)—it does a great job of explaining how the software we use has evolved over time.

A Note on Terminology

There's a fair amount of debate over the names of certain elements of operating systems. Even "Linux" itself is game for this—should it be "Linux," or should it be "GNU/Linux" to reflect that the operating system also contains pieces from the GNU Project? Throughout this book, I've tried to use the most common, least awkward names possible.