Chapter 1

Why Use Linux?

Welcome to the wide, wide world of Linux! The Linux operating system has gotten a tremendous amount of press over the past few years, and this book explains why you might want to see what all the fuss is about. In slightly over 10 years, Linux has blossomed from its roots in two free software projects to its adoption as the operating system of choice for most enterprise computing centers. It is also the default operating system used on inexpensive PCs sold at many consumer electronics stores. You can hardly open a computer magazine today without seeing articles about Linux, ads for companies selling Linux distributions, and ads for companies offering computers running different versions of Linux.

Linux is different and unique enough that anyone who is trying to learn about it has plenty of questions. Linux is free, so why are people selling it? It involves something called open source, but what the heck is that? What's this Unix thing I keep hearing about — is that the same thing? What do acronyms like GNU, GNOME, KDE, and X mean? How can there be so many different versions of one operating system? Can I actually do anything useful with Linux, or is it only for computer science students, geeks, nerds, and hackers?

This book answers those questions and more. This chapter provides all background information that you'll need to understand what Linux is, where it came from, and why it has become so popular. History lessons aren't usually a part of learning about a computer operating system, but Linux has strong social and philosophical components that make it more interesting than an off-the-shelf operating system that is just the product of some company. As you can see in this book, Linux is a powerful, dynamic operating system that is being worked on simultaneously by tens of thousands of developers all over the world. If you're just looking for a solution to your computing problems and aren't particularly interested in changing the world, that's fine — you don't have to wave the Linux flag just because you decide to use it. Plenty of other people are willing and eager to do that. For many people, Linux is just a powerful, inexpensive operating system that comes with tons of free, entertaining, and easy-to-use software that lets them do what they need to do.

Two points are worth clearing up right off the bat. First, what is Linux? Conversationally, Linux is the collective name for an operating system kernel and its associated applications, just like any other computer operating system. A *kernel* is the common name for the single program that is the actual operating system and runs directly on the hardware, managing your memory, disks, and peripherals, starting and stopping other programs, handling requests for resources from applications, and so on. As discussed in more detail later, *Linux* is technically just the name of the kernel — most of the applications that anyone uses with Linux come from other free software projects. However, a *Linux distribution* is the common term for a Linux kernel, a set of applications that can run on top of it (regardless of where they come from), and a tool to install everything and configure your system.

Secondly, how can Linux be free and sold at the same time? That's confusing, but easy enough to answer. The source code for Linux is indeed freely available from thousands of sites on the Internet. Anyone who wants it can get it, but building a complete system out of it that you can easily install on a computer is another thing entirely. When people sell Linux, they are selling a prepackaged and installable collection of free things — they're basically just charging you for the media that it comes on and the time and effort they invested in putting it all together; they're also "charging in advance" for any customer support that you might need if you encounter installation or initial configuration problems.

Now that those are out of the way, let's explore a bit of the history of Linux to give you an idea of where it comes from, why it is so powerful and popular, and how Linux distributions make it easy for you to install and get started with this exciting, powerful operating system.

The Evolution of Linux

The inspiration for Linux is an older operating system known as Unix, which was developed at Bell Labs in Murray Hill, New Jersey, beginning in 1969. The name Unix is a pun of sorts on Multics, which was an operating system research and development project being done at Bell Labs (then part of AT&T) in conjunction with the U.S. government, General Electric, MIT, and others. As the Multics program disintegrated from too much complexity and cost and time overruns, a number of ex-Multics developers and their associates continued their operating systems research, among them Ken Thompson, Dennis Ritchie, Doug McIlroy, Rudd Canaday, and J. F. Ossanna. They began developing a smaller, lighter-weight operating system on a much smaller system, a DEC PDP-7. Multics stood for *Multiplexed Information and Computing Service*, so UNICS was introduced as a humorous name that supposedly stood for *Uniplexed Information and Computing Service*, referring to the fact that Unix was developed and ran on much smaller systems than the mainframe on which Multics was to have run. Unix was eventually adopted as a sound-alike for UNICS, and the name that we now know was born.

The original motivation for developing the Unix operating system and file system on the PDP-7 was to facilitate running a game that Thompson had originally written for Multics, called Space Travel. As the capabilities of Unix grew, the operating system and associated utilities became much more interesting than the game, and the rest is history. The Unix team quickly outgrew the PDP-7, so the group eventually got a newer, more powerful machine, a DEC PDP-11. The purchase of the new machine was justified by using Unix as the foundation for a text processing system for the Bell Labs patent office, which was the first commercial, real-world use of Unix.

The C language

The process of moving any operating system and its applications to new hardware, known as *porting*, is typically extremely time-consuming. Unix was originally written in assembly language, which differs from machine to machine. To simplify porting, Thompson developed a language known as B, based on an older language from Xerox PARC known as BCPL, and attempted to write much of Unix in the higher-level B language, but his rewrite was hampered by shortcomings in the language.

Ritchie added concepts such as types and data structures to B, eventually ending up with a new language, which he called C. In 1973, Thompson rewrote as much of Unix as possible in C.

Minimizing the amount of machine-specific code required by Unix made porting it to new computer hardware much easier because only the machine-specific routines had to be rewritten for the new hardware. This portability was one of the big reasons for the initial popularity of Unix. The core portion of Unix remained written in machine-specific assembly language, but this was reduced to a few hundred lines. The portions of Unix written in C (which was most of it, at this point) could simply be recompiled for the new hardware once the compiler itself was ported to support the new machine's hardware instruction set.

Unix goes public

At the time that Unix development began, AT&T was operating under a government ruling that broke up its monopoly on the telephone system. To prevent AT&T from emerging as a monopoly in other areas, this ruling also prohibited the company from branching into other lines of business. Thus, it couldn't market its new operating system. For this reason, it was decided to freely license Unix to various academic and research institutions.

Once Unix hit the academic and research communities, its future was assured. Researchers and graduate students quickly adopted Unix as their platform of choice, using it as a daily platform as well as the reference platforms for many research efforts. Much of the reason for this quick acceptance was that the source code for Unix was freely available to licensees, which made it easy for developers to see how the operating system worked, fix problems that they discovered, and extend its capabilities. Later in this chapter, I come back to the notion of how having the source code for an operating system can help popularize it and facilitate its adoption — this same notion of freely available source code is one of the driving factors for the adoption and popularization of Linux.

Aside from the source-code issue, Unix was interesting to researchers and developers for several other reasons. I won't put you to sleep with all of the reasons (there are plenty of other books that can do that for you), but some of the most significant reasons are interesting. First, Unix was a true multiuser, multitasking operating system that ran on relatively small, relatively inexpensive machines. This meant that entire research groups could share access to and development time on the same machine. Secondly, Unix introduced a very interesting way of accessing and using devices. All devices attached to a Unix system, regardless of whether they are disk drives, printers, or network devices, can be accessed as streams of characters. This provides a single, consistent way of reading from or writing to any device, unlike the complex, specialized commands that other systems at the time required for accessing each different kind of device. (Some devices, such as disk drives, also offer block-oriented interfaces for performance reasons, but you always have access to the raw, character-oriented device.) Finally, Unix was the operating system chosen for development of the initial TCP/IP implementation, which means that Unix and the Internet (and before that, the ARPANET) have always gone hand in hand. The original TCP/IP implementation was done at the University of California at Berkeley (UCB), on Unix Version 7. UCB's collection of Unix enhancements eventually got to be large enough that they turned into the popular Berkeley Standard Distribution (BSD) of Unix, whose source code was freely available to any Unix licensee. (Conceptual derivatives of BSD are freely available today as FreeBSD, NetBSD, and OpenBSD.) All Unix systems since 1980 or so have supported TCP/IP networking.

Unix is traditionally associated with workstations, which are relatively high-powered desktop systems with sophisticated graphics capabilities. Today's workstation vendors, primarily Hewlett-Packard, IBM, Silicon Graphics, and Sun Microsystems, all purchased licenses for Unix from AT&T;

they have continued enhancing it and developing software that enables their version of Unix to take full advantage of the capabilities of their hardware.

Unix development is still taking place today, and almost all computer vendors now offer some version of Unix. However, Unix is rapidly being eclipsed by Linux for a variety of reasons, most notably lower cost, higher power, vendor-independence, and books like this one.

The GNU project

While the source code for Unix has always been available to licensees, the popularity of Unix has ensured that many other people wanted access to it. Originally owned by AT&T, some versions of the source code escaped into the wild thanks to its popularity in less controlled academic and research environments; but there were always legions of lawyers standing around, waiting to enforce AT&T's ownership and copyrights. AT&T's possessive attitude irritated some of the more progressive members of the software community, most notably Richard Stallman, who, in 1983, announced the GNU project, a project dedicated to developing software with source code that would always be freely available. Stallman is also well known as the founder of the Free Software Foundation (FSF, `http://www.fsf.org`), which was formed in 1985 as an institution to help raise funds for the GNU project, and to generally evangelize and popularize the concept of free software.

The GNU project (`http://www.gnu.org`) consists of a completely free operating system kernel and a huge collection of associated utilities. GNU is a recursive acronym that stands for *GNU's Not Unix*. Because the source code for every part of the GNU project is freely available, it is known as an *open source* project—the source code is always freely available from the FSF site and from anyone who uses it. Interestingly enough, though the source code is freely available, it is not unlicensed—the FSF owns the copyrights to all of the GNU source code and distributes it under the terms of open source licenses such as the GNU General Public License (GPL) and the Lesser GNU Public License (LGPL). These licenses largely stipulate that anyone who redistributes any GPL or LGPL software must also redistribute the source code for their version of that software, though the LGPL provides some exclusions for programs that simply link to libraries provided with GNU software.

The GNU project includes an operating system kernel known as Hurd and a better-known collection of enhanced versions of all of the standard Unix utilities, including an extremely popular collection of free compilers known as GNU Compiler Collection (commonly called GCC). Hurd is an acronym that stands for *Hird of Unix-Replacing Daemons*, whereas Hird stands for *Hurd of Interfaces Representing Depth*—those wacky computer scientists and their acronyms! GCC provides free and powerful compilers for the Ada, C, C++, Objective-C, Fortran, and Java languages. The GCC compilers are the most popular and widely used collection of compilers that have ever existed in the world of computer science.

Note

The C compiler in GCC is known as `gcc`, which is a frequent source of recursive confusion.

The development of GNU utilities occurred quickly, because they could easily be written, compiled, and tested on a variety of existing platforms, especially those on which the GCC compilers were already available. However, developing a new kernel is a somewhat more complex task. As of this writing, the GNU Hurd kernel is functional and usable, but is still not quite ready for prime time. Why? Largely because of a Finnish college student's project to develop a free, powerful kernel—and his friendly, open announcement on the Internet in 1991 asking for feedback.

Enter Linus

In 1991, Linus Torvalds was a college student at the University of Helsinki in Finland. Interested in computer operating systems and somewhat frustrated by the fact that the source code for the academic operating system that he was working with was not freely available and redistributable, Linus decided to write his own. His initial announcement was the following:

```
From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)
Newsgroups: comp.os.minix
Subject: What would you like to see most in minix?
Summary: small poll for my new operating system
Message-ID: <1991Aug25.205708.9541@klaava.Helsinki.FI>
Date: 25 Aug 91 20:57:08 GMT
Organization: University of Helsinki

Hello everybody out there using minix -
I'm doing a (free) operating system (just a hobby, won't be big and
professional like gnu) for 386(486) AT clones. This has been brewing
since april, and is starting to get ready. I'd like any feedback on
things people like/dislike in minix, as my OS resembles it somewhat
(same physical layout of the file-system (due to practical reasons)
among other things).
I've currently ported bash(1.08) and gcc(1.40), and things seem to work.
This implies that I'll get something practical within a few months, and
I'd like to know what features most people would want. Any suggestions
are welcome, but I won't promise I'll implement them :-)
Linus (torvalds@kruuna.helsinki.fi)
PS. Yes - it's free of any minix code, and it has a multi-threaded fs.
It is NOT protable (uses 386 task switching etc), and it probably never
will support anything other than AT-harddisks, as that's all I have :-(.
```

Linus's request for feedback sparked a few e-mail messages asking for copies and offering suggestions. The true communal development of Linux began a few months later, when he offered the source code to anyone who was interested and asked for people to contribute fixes and enhancements back to him. As you can see from the announcement, Linus was using the GNU utilities both for his kernel development and as the basic collection of things that ran on top of his new kernel. It's fair to say that Linux never would have happened without the GNU project, and for that reason many purists still refer to Linux as GNU/Linux to acknowledge the contributions of the GNU project to Linux distributions everywhere.

Free Software

Commercial software, produced by businesses with the goal of making a profit, is obviously important to anyone who uses an off-the-shelf computer system. There's a tremendous amount of great commercial software available, enabling real people to get real work done: writing documents, creating spreadsheets, sending and receiving e-mail, creating fantastic artwork, laying out complex books and advertisements, and much, much more. However, most commercial software is *closed source* software, meaning that users can't see the source code, fix problems, or extend its capabilities. Closed source software therefore has some attendant problems. It does what its author(s) wrote it to do — which is not necessarily the same thing that a user may want it to do. If a user discovers the software does not meet his or her needs, the only recourse is to contact the vendor's customer support personnel and hope that they care enough or have the resources to fix the problem and/or release a new version.

In comparison, free software, which is freely distributed in the first place, is written by people who either need a tool that does whatever the package does, or who are simply interested in that type of software and are savvy enough to write it. There are essentially two types of free software: software that makes available the executables but not the source code and software that makes available both the executables and the source code. When I refer to free software in this book, I'm referring to the latter, because that is the free software model that underlies Linux, the GNU project, and thousands of other great software packages that run on top of the Linux kernel.

Earlier I mentioned that Linux and related utilities had a strong conceptual and philosophical component. That component is free software, which is based on the philosophy that no one should control your access to the source code for a program that you depend upon, thus opening the possibility that you might find yourself in the position where you can't get your work done because of a bug that you quite literally can't do anything about (regardless of your sophistication as a programmer). The Free Software Foundation's Web page (`www.fsf.org`) expresses this better than anyone:

> Free software is a matter of liberty not price. You should think of "free" as in "free speech."

Of course, just because software is free doesn't mean that everyone has the skills necessary to fix or enhance it. Free software simply guarantees that this option is always available. Another great thing about free software is that because you can freely examine the source code for an application, you can understand the format in which it produces its output files. Even if you encounter a problem with a piece of free software, the fact that its output formats are easily determined makes it possible for you to reuse and recover data in ways that no commercial software could possibly provide.

The distributors

The processes of building a kernel, partitioning disks, creating file systems on them, filling those file systems with correctly compiled software, and configuring all of your system's hardware is not for the weak of heart. For software to succeed, especially an operating system, it has to be easy to install, relatively easy to configure, and at least provide good defaults for all configuration settings. This was certainly not the case for Linux in the early days. I can remember spending days downloading the bits and pieces of Linux from various Web sites, reassembling them on a disk partition that I'd formatted after booting off a floppy that I'd downloaded and written as an image, spending a day or two writing a configuration file so that I could run graphical applications on my Linux system, and so on. This is hardly a recipe for commercial success, though I do indeed have some geek nostalgia for the process.

Luckily, some people who had actually seen commercial software before recognized that Linux in its initial, primitive form was too difficult for mere mortals to install, and that no one was going to be able to see how good it was if you couldn't even install it without at least a master's in computer science or electrical engineering.

As discussed near the beginning of this chapter, a Linux distribution is the sum of a precompiled version of the Linux kernel, a set of applications that run on top of it, an installer that makes it easy to install and configure your system, and (if you're really, really lucky) some documentation and support for at least the configuration and installation process. Linux distributions began to appear shortly after use of the Linux kernel and GNU utilities spread outside of the initial Linux development team. The developers of the first Linux distributions were just as interested in popularizing the operating system as in working on kernel or application code. Developing Linux distributions and making them widely available has played a critical role in the adoption of Linux as an operating system because these distributions have made it possible for people to actually install and use Linux, the GNU utilities, and so on.

Early Linux distributions included Owen Le Blanc's Manchester Computing Centre (MCC) Interim Linux, introduced in 1992, Texas A&M University's TAMU, and Peter MacDonald's Softlanding Linux System (SLS). SLS was the first distribution to add things that Linux users take for granted today, such as graphics capabilities and the X Window System. 1992 also saw the release of Patrick Volkerding's Slackware Linux distribution, which is the longest-lived Linux distribution and is still available and updated today. Other Linux distributions that are still active today followed quickly, most notably SUSE (1992), Red Hat (1993), and Debian (1993).

The situation today

The Slackware, SUSE (now owned by Novell), Red Hat, and Debian Linux distributions are still going strong after more than a decade. However, many other Linux distributions are available — more than ever before — and the number keeps growing. Some very popular Linux distributions are specific to certain hardware — the best example of this is Yellow Dog Linux (`www.yellowdoglinux.com`), which has specialized in producing high-quality Linux distributions for Apple's Macintosh hardware since 1999. Many other Linux distributions are lovingly crafted by Linux fans with a special goal in mind: minimizing disk space requirements, creating distributions targeted towards playing multimedia, providing the ability to boot and run from a CD without requiring installation, and many more. There are over 300 Linux distributions available today. You can find information about most of these and their relative popularity at the Distribution Watch Web site, `www.distrowatch.com`. Some of the newer and most popular distributions for x86 desktop systems today, aside from those mentioned earlier, are Ubuntu (`www.ubuntulinux.com`), Mandriva (www.mandriva.com), Gentoo (www.gentoo.org), and TurboLinux (www.turbolinux.com). You can find more detail about most of these distributions, their focus and history, and their availability in Chapter 2.

Some Linux distributions have a specific geographic or international focus, even though most Linux distributions provide built-in support for internationalization and alternate character sets. There are Polish Linux distributions (`www.pld-linux.org`), Russian Linux distributions (`www.altlinux.com`), and Chinese Linux distributions (`www.redflag-linux.com`). Some of these are commercial, some are simply labors of love, but all share the desire of their authors to popularize Linux and help their audiences make the most of it.

Hundreds of special-purpose Linux distributions are also available that are rarely seen or recognized unless you are working in certain industries. The best example of this phenomenon is embedded computing, which refers to operating systems and applications that run on computers that are built into anything from factory assembly lines to telecommunications switches. Linux distributions are available for every processor architecture and family in the embedded computing market, each taking full advantage of the capabilities of the advanced single-board computers used in embedded computing today. Custom Linux distributions are also available and used in many popular consumer electronic devices, such as cell phones, home gateways, and the popular TiVo digital video recorder.

Today's Linux distributions provide better installers, sophisticated package management (to make it easy to install, update, and even remove applications and associated files), and easy-to-use system update capabilities, but the concept of a Linux distribution is still the same as it's always been. They're a bit larger now — the most widely used x86 Linux distributions now come on several CDs or a DVD — and they offer multiple configuration options (desktop, server, workstation, and so on) both to customize the type of software that is installed and to reduce the amount of space that an installation requires.

What Makes Linux Different?

With all the excitement about Linux and its penetration of almost every area of modern computing, it's natural to step back and think, "Why should I care? It's just another operating system." The following is a list of the most significant advantages of Linux over other operating systems:

- **Multiprocessing, multiuser operating system:** Other desktop computer operating systems, such as Microsoft Windows, started out as small operating systems that could only run one program at a time, and have been retrofitting the ability to run multiple programs at the same time ever since. Linux draws upon decades of computer science research in Unix, the effective use of hardware, how to run and service multiple programs at the same time, and general computer performance, and was created with these things in mind. No retrofitting necessary.
- **Always free and available source code:** This includes the Linux kernel and all of the components of a core Linux distribution. For many home computer users who just want an off-the-shelf operating system that "does the right thing," this doesn't matter directly. However, by making it possible for many different companies and groups to create, distribute, and maintain Linux distributions that are easy to install and use, freely available source code is the cornerstone of Linux availability. Linux distributions may include closed source applications, such as RealPlayer, that run on Linux, but the source code for the Linux kernel and the vast majority of Linux software is and will always be readily available.
- **No lock-in to a specific vendor:** Regardless of the operating system that you're currently using, you may have encountered problems with applications or the way things worked. However, if you're using an off-the-shelf operating system from Microsoft or Apple, you can get it only from Microsoft or Apple, it will work only the way that Microsoft and Apple think

that it should, and it will always cost what Microsoft and Apple think that it should cost. On the other hand, Linux is open and free, so if you don't like the way that Red Hat's Linux works, how much it costs, or the type of customer support that's available, you can always switch to Novell's SUSE Linux. Or to Mandriva Linux. Or to Ubuntu Linux. Or to . . . I think you see where I'm going with this.

- **Thousands of free, powerful applications:** Need a word processor? Download and install OpenOffice Writer, AbiWord, Kwrite, or dozens of others. Need a database? Download and install MySQL, PostgreSQL, or many others. Need to create graphics or manipulate digital photographs? It doesn't get much better than the GNU Image Manipulation Program (GIMP). If anything, a problem with Linux can be that you have too many choices — which is a good thing for the people who write books like this one.
- **Interoperability with other operating systems:** Companies that write and sell an operating system (no names mentioned) are primarily interested in furthering that operating system and interoperating with other copies of itself. Windows networking and support for sharing disk drives over a network is a good example. Microsoft grudgingly added TCP/IP support after the Internet took off and added support for networked file sharing largely because other companies were already doing that (Artisoft, Banyan, Novell) and they were missing out on potential sales. Linux is written and supported by people who want to use it but can't ignore the fact that other operating systems exist. Linux has extensive built-in support for interoperating with networked drives on systems running Windows, Apple Mac OS and OS X, and Novell Netware. You can always get there from here on a Linux system.
- **Support for standards:** Linux and Linux applications are designed to support standards because these are the language of free intellectual commerce. Linux supports computing standards such as POSIX, FHS, and many more. Linux applications support modern application and data formats for audio, multimedia, document formatting, spreadsheet data, and many more. Because Linux is open and free, there can be no such thing as a proprietary Linux data or application format. This not only fosters data exchange between Linux applications but also guarantees that you'll always be able to get to your data.
- **Lower total cost of ownership:** If you want to use Linux on your desktop or throughout your business, it's free to obtain and there are legions of Linux wizards available who can help you do whatever you want with it. There are no licensing fees — if you need to pay for something, you can pay for updates and support from the vendor of your Linux distribution. I can have 50 Linux wizards at your doorstep faster than Microsoft can say "Please advertise for an MCP with MCTS, MCITP, MCSA, MCSE, and MCDBA certifications who actually understands enterprise computing requirements." Call me.
- **Stable, powerful, and virus-free:** Linux is a mature, multiuser system that is dependable, stable, and rarely needs to be rebooted (and then usually due to power failures or hardware moves). It is designed to support running hundreds of simultaneous processes and multiple users. It has built-in security and is immune to viruses except through system administration slip-ups. In comparison, Microsoft Windows is a petri dish for virus development and cultivation.

- **Wide range of hardware support:** Microsoft Windows runs on x86 boxes. Period. (Full stop for those of you in the U.K.) Apple's Mac OS and OS X run on PPC hardware (x86 hardware coming real soon now). Novell Netware runs on x86 boxes. Up-to-date Linux distributions are available for x86 hardware, PPC hardware, and on ARM, MIPS, Super-H, XScale, and other architectures in the embedded computing market. This may not matter much to you if you're just looking for a desktop operating system for home or business use, but if you're writing software or creating products with a built-in operating system, Linux gives you a wider range of modern hardware choices than any other operating system.

The discussion of some of the advantages of Linux may have wandered into evangelism, but they're all true. It's easy to get excited about an operating system that is as powerful, inexpensive, standards-conformant, and generally open as Linux is. The vendors of other operating system are not stupid — they're aware of the value and increasing popularity of Linux and other Unix-like operating systems. Apple's OS X operating system uses a version of Unix to provide its customers with the power and stability that it needs to run modern Internet services and do true multiprocessing on Macs. Microsoft has mailed me enough copies of its Unix Services CD (Unix commands for the Windows environment) that I now have coasters for up to 50 visitors who are having drinks at my home.

Summary

Linux is a stable, powerful, modern operating system with deep roots in computer science research. It is also eminently usable by anyone as a desktop operating system. Linux has significant advantages over other operating systems from a technical and conceptual point of view — and it is free. As I discuss in the next chapter, the many vendors who provide stable, supported, and up-to-date Linux distributions make it easy to install, configure, and use Linux. As you can see throughout the rest of this book, different vendors may do things differently, but the core concepts of Linux are the same in any Linux distribution — including ease of use, configuration, updating, and reconfiguration. The days when Linux came with a pocket protector are long gone. Today's Linux is the equal of any other operating system.