
Roderick W. Smith
Linux®
ESSENTIALS
SYBEX
SERIOUS SKILLS.

CHAPTER 1

Selecting an Operating System

The fact that you're reading this book means you want to learn about the Linux operating system (OS). To begin this journey, you must first understand what Linux is and what an OS is. This chapter is therefore devoted to these basic issues. I describe what an OS is, how users interact with an OS, how Linux compares to other OSs with which you may be familiar, and how specific Linux implementations vary. Understanding these issues will help you make your way as you learn about Linux and switch between Linux-based systems and other computers.

- **What is an OS?**
- **Investigating user interfaces**
- **Where does Linux fit in the OS world?**
- **What is a distribution?**

What Is an OS?

An OS provides all the most fundamental features of a computer, at least from a software point of view. An OS enables you to use the computer's hardware devices, it defines the user interface standards, and it provides basic tools that begin to make the computer useful. Ultimately, many of these features trace their way back to the OS's *kernel*, which is described in more detail next. Other OS features are owed to additional programs that run atop the kernel, as described later in this chapter.

What Is a Kernel?

An OS kernel is a software component that's responsible for managing various low-level features of the computer, including:

- Interfacing with hardware devices (network adapters, hard disks, and so on)

- Allocating memory to individual programs
- Allocating CPU time to individual programs
- Enabling programs to interact with each other

When you use a program (say, a Web browser), it relies on the kernel for many of its basic functions. The Web browser can only communicate with the outside world by using network functions provided by the kernel. The kernel allocates memory and CPU time to the Web browser, without which it couldn't run. The Web browser may rely on plug-ins to display multimedia content; such programs are launched and interact with the Web browser through kernel services. Similar comments apply to any program you run on a computer, although the details vary from one OS to another and from one program to another.

In sum, the kernel is the software "glue" that holds the computer together. Without a kernel, a modern computer can do very little.

Many programs run on multiple kernels, but most need OS-specific tweaks. Programmers create *binaries*—the program files for a particular processor and kernel—for each OS.

Kernels are not interchangeable; the Linux kernel is different from the Mac OS X kernel or the Windows kernel. Each of these kernels uses a different internal design and provides different software interfaces for programs to use. Thus, each OS is built from the kernel up and uses its own set of programs that further define each OS's features.

Linux uses a kernel called *Linux*—in fact, technically speaking, the word *Linux* refers *only* to the kernel. Other features that you might associate with Linux are provided by non-kernel programs, most of which are available on other platforms, as described shortly, in "What Else Identifies an OS."

A student named Linus Torvalds created the Linux kernel in 1991. Linux has evolved considerably since that time. Today, it runs on a wide variety of CPUs and other hardware. The easiest way to learn about Linux is to use it on a desktop or laptop PC, so that's the type of configuration that's emphasized in this book. The Linux kernel, however, runs on everything from tiny cell phones to powerful supercomputers.

What Else Identifies an OS?

The kernel is at the core of any OS, but it's a component that most users don't directly manipulate. Instead, most users interact with a number of other software components, many of which are closely associated with particular OSs. Such programs include the following:

Certification Objective

Command-line shells Years ago, users interacted with computers exclusively by typing commands in a program (known as a *shell*) that accepted such

commands. The commands would rename files, launch programs, and so on. Although many computer users today don't use text-mode shells, they're still important for intermediate and advanced Linux users, so I describe them in more detail in Chapter 6, "Getting to Know the Command Line," and subsequent chapters rely heavily on your ability to use a text-mode shell. Many different shells are available, and which shells are available and popular differ from one OS to another. In Linux, a shell known as the Bourne Again Shell (`bash` or Bash) is popular.

Graphical user interfaces A graphical user interface (GUI) is an improvement on a text-mode shell, at least from the perspective of a beginning user. Instead of using typed commands, GUIs rely on icons, menus, and a mouse pointer. Windows and Mac OS both have their own OS-specific GUIs. Linux relies on a GUI known as the X Window System, or X for short. X is a very basic GUI, so Linux also uses *desktop environment* program suites, such as the GNU Object Model Environment (GNOME) or the K Desktop Environment (KDE), to provide a more complete user experience. It's the differences between a Linux desktop environment and the GUIs in Windows or OS X that will probably strike you most when you first begin using Linux.

Certification Objective

Utility programs Modern OSs invariably ship with a wide variety of simple utility programs—calculators, calendars, text editors, disk maintenance tools, and so on. These programs differ from one OS to another. Indeed, even the names and methods of launching these programs can differ between OSs. Fortunately, you can usually find the programs you want by perusing menus in the main desktop environment.

Libraries Unless you're a programmer, you're unlikely to need to work with libraries directly; nonetheless, I include them in this list because they provide critical services to programs. Libraries are collections of programming functions that can be used by a variety of programs. For instance, in Linux most programs rely on a library called `libc`. Other libraries provide features associated with the GUI or that help programs parse options passed to them on the command line. Many libraries exist for Linux, which helps enrich the Linux software landscape.

Productivity programs Major productivity programs—Web browsers, word processors, graphics editors, and so on—are the usual reason for using a computer. Although such programs are often technically separate from the OS, they are sometimes associated with certain OSs. Even when a program is available on many OSs, it may have a different "feel" on each OS because of the different GUIs and other OS-specific features.

You can search for Linux equivalents to popular Mac OS X or Windows programs on Web sites such as `http://www.linuxrsp.ru/win-lin-soft/table-eng` or `http://www.linuxalt.com`.

In addition to software that runs on an OS, several other features can distinguish between OSs, such as the details of user accounts, rules for naming disk files, and technical details of how the computer starts up. These features are all controlled by software that's part of the OS, of course—sometimes by the kernel and sometimes by non-kernel software.

Investigating User Interfaces

Earlier, I noted the distinction between text-mode and graphical user interfaces. Although most end users favor GUIs because of their ease of use, Linux retains a strong text-mode tradition. Chapter 6 describes Linux's text-mode tools in more detail, and Chapter 4, "Using Common Linux Programs," covers basic principles of Linux GUI operations. It's important that you have some grounding in the basic principles of both text-mode and graphical user interfaces now, since user interface issues crop up from time to time in intervening chapters.

Using a Text-Mode User Interface

Certification Objective

In the past, and even sometimes today, Linux computers booted in text mode. Once the system had completely booted, the screen would display a simple text-mode login prompt, which might resemble this:

```
Debian GNU/Linux 6.0 essentials tty1

essentials login:
```

The details of such a login prompt vary from one system to another. This example includes several pieces of information:

- The OS name and version—Debian GNU/Linux 6.0
- The computer's name—`essentials`
- The name of the hardware device being used for the login—`tty1`
- The login prompt itself—`login:`

To log in to such a system, you must type your username at the `login:` prompt. The system then prompts you for a password, which you must also type. If you entered a valid username and password, the computer is likely to display a login message, followed by a shell prompt:

```
rodsmith@essentials:~$
```

To try a text-mode login, you must first install Linux on a computer. Neither the Linux Essentials exam nor this book covers Linux installation; consult your distribution's documentation to learn more.

If you see a GUI login prompt, you can obtain a text-mode prompt by pressing Ctrl+Alt+F1 or Ctrl+Alt+F2. To return to the GUI login prompt, press Alt+F1 or Alt+F7.

In this book, I omit most of the prompt from example commands when they appear on their own lines. I keep the dollar sign ($) prompt, though, for ordinary user commands. Some commands must be entered as `root`, which is the Linux administrative user. I change the prompt to a hash mark (#) for such commands, since most Linux distributions make a similar change to their prompts for the `root` user.

Chapter 13, "Understanding Users and Groups," describes Linux accounts, including the root account, in more detail.

The details of this shell prompt vary from one installation to another, but you can type text-mode commands at the shell prompt. For instance, you could type **`ls`** (short for *list*) to see a list of files in the current directory. The most basic commands are shortened by removing vowels, and sometimes consonants, in order to minimize the amount of typing required to execute a command. This has the unfortunate effect of making many commands rather obscure.

Some commands display no information, but most produce some type of output. For instance, the `ls` command produces a list of files:

```
$ ls
106792c01.doc  f0101.tif
```

This example shows two files in the current directory: `106792c01.doc` and `f0101.tif`. You can use additional commands to manipulate these files, such as `cp` to copy them or `rm` to remove (delete) them. Chapter 6 ("Getting to Know the Command Line") and Chapter 7 ("Managing Files") describe some common file manipulation commands.

Some text-mode programs take over the display in order to provide constant updates or to enable you to interact with data in a flexible manner. Figure 1.1, for instance, shows the `nano` text editor, which is described in more detail in Chapter 11, "Editing Files." Once `nano` is working, you can use your keyboard's arrow keys to move the cursor around, add text by typing, and so on.

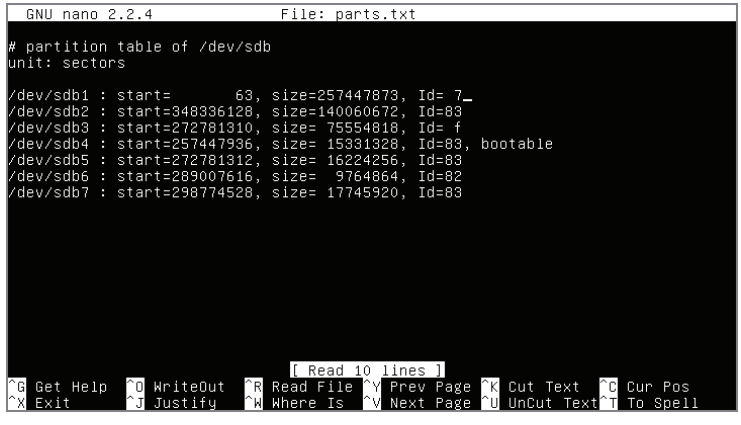


FIGURE 1.1 Some text-mode programs take over the entire display.

Even if you use a graphical login, you can use a text-mode shell inside a window, known as a *terminal*. Common Linux GUIs provide the ability to launch a terminal program, which provides a shell prompt and the means to run text-mode programs.

Using a Graphical User Interface

Certification Objective

Most users are more comfortable with GUIs than with text-mode commands. Thus, many modern Linux systems start up in GUI mode by default, presenting a login screen similar to the one shown in Figure 1.2. You can select your username from a list or type it, followed by typing your password, to log in.

Some Linux GUI login screens don't prompt you for a password until after you've entered a valid username.

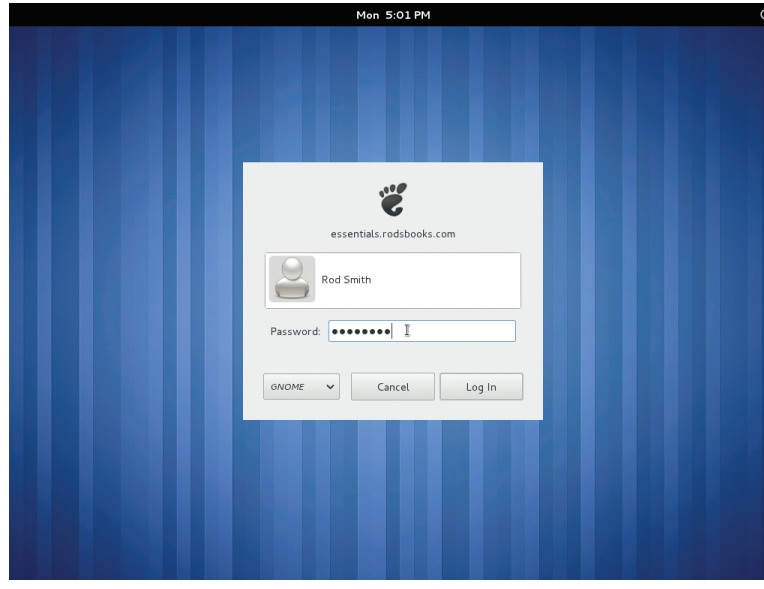


FIGURE 1.2 Graphical login screens on Linux are similar to those for Windows or Mac OS X.

Unlike Windows and Mac OS X, Linux provides a number of desktop environments. Which one you use depends on the specific variety of Linux you're using, what software options you selected at installation time, and your own personal preferences. Common choices include GNOME, KDE, Xfce, and Unity. Many other options are available as well. In Figure 1.2, you can see a selection option for the desktop environment in the lower-left corner of the central dialog box. It reads GNOME in Figure 1.2, meaning that if the item is left unchanged, the computer will launch GNOME when the user logs in.

Linux desktop environments can look quite different from one another, but they all provide similar functionality. Figure 1.3 shows the default KDE on an openSUSE 12.1 installation, with a couple of programs running. Chapter 4 describes common desktop environments and their features in more detail, but for now, you should know that they all provide features such as:

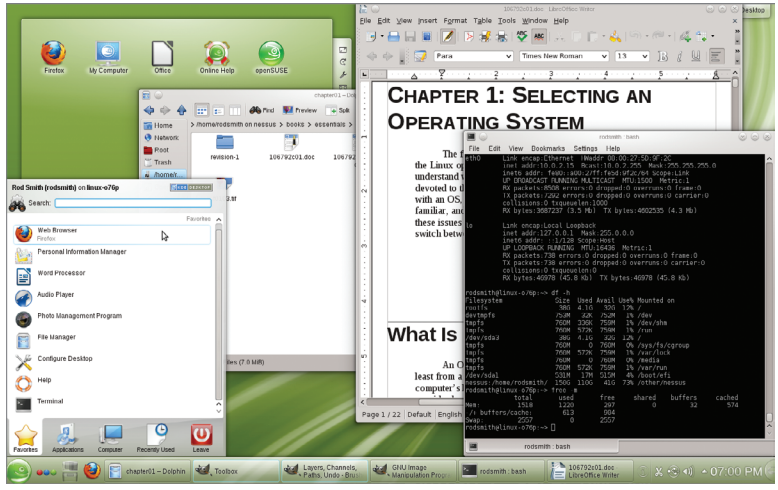


FIGURE 1.3 **Linux desktop environments provide the types of GUI controls that most users expect.**

Program launchers You can launch programs by selecting them from menus or lists. Typically, one or more menus exist along the top, bottom, or side of the screen. In Figure 1.3, you can click the openSUSE gecko icon in the bottom-left corner of the screen to produce the menu that appears in that figure.

File managers Linux provides GUI file managers similar to those in Windows or Mac OS X. A window for one of these is open in the center of Figure 1.3.

Window controls You can move windows by clicking and dragging their title bars, resize them by clicking and dragging their edges, and so on.

Multiple desktops Most Linux desktop environments enable you to keep multiple virtual desktops active, each with its own set of programs. This feature is very handy to keep the screen uncluttered while you run many programs simultaneously. Typically, an icon in one of the menus enables you to switch between virtual desktops.

Logout options You can log out of your Linux session, which enables you to shut down the computer or let another user log in.

Logging out is very important in public computing environments. If you fail to log out, a stranger might come along and use your account for malicious purposes.

You may need to install extra desktop environments to use them. This topic is not covered in this book.

As you learn more about Linux, you'll discover that its GUI environments are quite flexible. If you find you don't like the environment that's the default for your distribution, you can change it. Although they all provide similar features, some people have strong preferences about desktop environments. Linux gives you a choice in the matter that's not available in Windows or Mac OS X, so feel free to try multiple desktop environments.

Where Does Linux Fit in the OS World?

As described later, in "What Is a Distribution?," Linux can be considered a family of OSs. Thus, you can compare one Linux version to another one.

This chapter's title implies a comparison, and as this book is about Linux, the comparison must be to non-Linux OSs. Thus, I compare Linux to three other OSs or OS families: Unix, Mac OS X, and Microsoft Windows.

Comparing Linux to Unix

Certification Objective

If you were to attempt to draw a "family tree" of OSs, you would end up scratching your head a lot. This is because OS designers often mimic each other's features, and sometimes even incorporate each other's code into their OSs' workings. The result can be a tangled mess of similarities between OSs, with causes ranging from coincidence to code "borrowing." Attempting to map these influences can be difficult. In the case of Linux and Unix, though, a broad statement is possible: Linux is modeled after Unix.

Open source software is software that you can not only run, but modify and redistribute yourself. Chapter 2, "Investigating Linux's Principles and Philosophy," covers the philosophy and legal issues concerning open source software.

Unix was created in 1969 at AT&T's Bell Labs. Unix's history is complex and involves multiple *forks* (that is, splitting of the code into two or more independent projects) and even entirely separate code rewrites. Modern Linux systems are, by and large, the product of open source projects that clone Unix programs, or of original open source code projects for Unix generally. These projects include:

The Linux kernel Linus Torvalds created the Linux kernel as a hobby programming project in 1991, but it soon grew to be much more than that. The Linux kernel was designed to be compatible with other Unix kernels, in the sense that it used the same software interfaces in source code. This made using open source programs for other Unix versions with the Linux kernel easy.

The GNU project The GNU's Not Unix (GNU) project is an effort by the Free Software Foundation (FSF) to develop open source replacements for all the core elements of a Unix OS. In 1991, the FSF had already released the most important such tools, with the notable exception of the kernel. (The GNU HURD kernel is now available but is not as popular as the Linux kernel.) Alternatives to the GNU tools include proprietary commercial tools and open source tools developed for the BSD Unix variants. The tools used on a Unix-like OS can influence its overall

"flavor," but all of these tool sets are similar enough to give any Unix variety a similar feel compared to a non-Unix OS.

Xorg-X11 The X Window System is the GUI environment for most Unix OSs. Most Linux distributions today use the Xorg-X11 variety of X. As with the basic text-mode tools provided by the GNU project, choice of an X server can affect some features of a Unix-like OS, such as the types of fonts it supports.

Desktop environments GNOME, KDE, Xfce, and other popular open source desktop environments have largely displaced commercial desktop environments even on commercial versions of Unix. Thus, you won't find big differences between Linux and Unix in this area.

Server programs Historically, Unix and Linux have been popular as server OSs—organizations use them to run Web servers, e-mail servers, file servers, and so on. Linux runs the same popular server programs as do commercial Unix versions and the open source BSDs.

User productivity programs In this realm, as in server programs, Linux runs the same software as do other Unix-like OSs. In a few cases, Linux runs more programs, or runs them better. This is mostly because of Linux's popularity and the vast array of hardware drivers that Linux offers. If a program needs advanced video card support, for example, it's more likely to find that support on Linux than on a less popular Unix-like OS.

On the whole, Linux can be thought of as a member of the family of Unix-like OSs. Although Linux is technically *not* a Unix OS, it's similar enough that the differences are unimportant compared to the differences between this family as a whole and other OSs, such as Windows. Because of its popularity, Linux offers better hardware support, at least on commodity PC hardware. Some Unix varieties offer specific features that Linux lacks, though. For instance, the Zettabyte File System (ZFS), available on Solaris, FreeBSD, and some other OSs, provides advanced filesystem features that aren't yet fully implemented in Linux.

GNU is an example of a recursive acronym—an acronym whose expansion includes the acronym itself. This is an example of geek humor.

Mac OS X, described shortly, is a commercial Unix that eschews both X and the desktop environments that run on it in favor of Apple's own GUI.

A ZFS add-on for Linux is available, but it's not fully integrated into the OS. A Linux filesystem known as Btrfs offers many ZFS features, but Btrfs isn't yet complete.

Code Types

Human beings write programs in a form known as *source code*. Although source code can seem arcane to the uninitiated, it's crystal clear compared to the form a program must take for a computer to run it: *binary code*. A program known as a *compiler* translates source code to binary code. (Alternatively, some programming languages rely on an *interpreter*, which converts source code to binary code "on the fly," eliminating the need to compile source code.)

(Continues)

CODE TYPES *(Continued)*

The term *open source* refers to the availability of source code, which is generally withheld from the public in the case of commercial programs and OSs. A programmer with access to a program's source code can fix bugs, add features, and otherwise alter how the program operates.

Comparing Linux to Mac OS X

Certification Objective

Mac OS X is a commercial Unix-based OS that borrows heavily from the BSDs and discards the usual Unix GUI (namely X) in favor of its own user interface. This makes OS X both very similar to Linux and quite different from it.

You can open an OS X Terminal window and type many of the same commands described in this book to achieve similar ends. If a command described in this book isn't present, you may be able to install it in one way or another. OS X ships with some popular Unix server programs, so you can configure it to work much like Linux or another Unix-like OS as a network server computer.

The X in X server is a letter X, but the X in OS X is a Roman numeral (10), denoting the tenth version of Mac OS.

OS X differs from Linux in its user interface, though. The OS X user interface is known as *Cocoa* from a programming perspective, or *Aqua* from a user's point of view. It includes elements that are roughly equivalent to both X and a desktop environment in Linux. Because Cocoa isn't compatible with X from a programming perspective, applications developed for OS X can't be run directly on Linux (or on other Unix-like OSs), and porting them (that is, modifying the source code and recompiling them) for Linux is a non-trivial undertaking. Thus, native OS X applications seldom make the transition to Linux.

OS X includes an implementation of X that runs under Aqua. This makes the transfer of GUI Linux and Unix programs to OS X relatively straightforward. The resulting programs don't entirely conform to the Aqua user interface, though. They may have buttons, menus, and other features that look out of place compared to the usual appearance of OS X equivalents.

Apple makes OS X available for its own computers. Its license terms forbid installation on non-Apple hardware, and even aside from licensing issues, installing OS X on non-Apple hardware is a non-trivial undertaking. A variant of OS X, known as iOS, runs on Apple's iPad and iPhone devices, and is equally non-portable to other devices. Thus, OS X is largely limited to Apple hardware. Linux, by contrast, runs on a wide variety of hardware, including most PCs. You can even install Linux on Macintosh computers.

Comparing Linux to Windows

Certification Objective

Most desktop and laptop computers today run Microsoft Windows. Thus, if you're considering running Linux, the most likely comparison is to Windows. Broadly speaking, Linux and Windows have similar capabilities; however, there are significant differences in details. These include the following:

Licensing Linux is an open source OS whereas Windows is a proprietary commercial OS. Chapter 2 covers open source issues in greater detail, but for now you should know that open source software gives you greater control over your computer than does proprietary software—at least in theory. In practice, you may need a great deal of expertise to take advantage of open source's benefits. Proprietary software may be preferable if you work for an organization that's only comfortable with the idea of software that's sold in a more traditional way. (Some Linux variants, though, are sold in a similar way, along with service contracts.)

Costs Many Linux varieties are available free of charge, and so are appealing if you're trying to cut costs. On the other hand, the expertise needed to install and maintain a Linux installation is likely to be greater, and therefore more expensive, than the expertise needed to install and maintain a Windows installation. Different studies on the issue of total cost of ownership of Linux vs. Windows have gone both ways, but most tend to favor Linux.

Hardware compatibility Most hardware components require OS support, usually in the form of drivers. Most hardware manufacturers provide Windows drivers for their devices, or work with Microsoft to ensure that Windows includes appropriate drivers. Although some manufacturers provide Linux drivers, too, for the most part the Linux community as a whole must supply the necessary drivers. This means that Linux drivers may take a few weeks or even months to appear after a device becomes available. On the other hand, Linux developers tend to maintain drivers for old hardware for much longer than manufacturers continue to support their own old hardware. Thus, a modern Linux may run better than a recent version of Windows on old hardware. Linux also tends to be less resource-intensive, so you can be productive on older hardware when using Linux.

Software availability Some popular desktop applications, such as Microsoft Office, are available on Windows but not on Linux. Although Linux alternatives, such as OpenOffice.org or LibreOffice, are available, they haven't caught on in the public's mind. In other realms, the situation is reversed. Popular server programs, such as the Apache Web server, were developed first for Linux or Unix.

Although many such servers are available for Windows, they run more efficiently on Linux. If you have a specific program you must run, you may want to research its availability and practicality on any platforms you're considering.

Microsoft is making major changes to its user interface with Windows 8. The new user interface, Metro, works the same on everything from cell phones to desktop computers.

User interfaces Like Mac OS X, Windows uses its own unique user interface. This fact contributes to poor inter-OS portability. (Tools exist to help bridge the gap, though; X Window System implementations for Windows are available, as are tools for running Windows programs in Linux.) Some users prefer the Windows user interface to any Linux desktop environment, but others prefer a Linux desktop environment.

Configurability Linux is a much more configurable OS than is Windows. Although both OSs provide means to run specific programs at startup, change user interface themes, and so on, Linux's open source nature means you can tweak any detail you want. Furthermore, you can pick any Linux variant you like to get a head start on setting up the system as you see fit.

Security Advocates of each OS claim it's more secure than the other. They can do this because they focus on different security issues. Many of the threats to Windows come from viruses, which by and large target Windows and its huge installed user base. Viruses are essentially a non-issue for Linux; in Linux, security threats come mostly from break-ins involving misconfigured servers or untrustworthy local users.

For over a decade, Windows has dominated the desktop arena. In both homes and offices, users have become familiar with Windows and are used to popular Windows applications, such as Microsoft Office. Although Linux *can* be used in such environments, it's a less popular choice for a variety of reasons—its unfamiliarity, the fact that Windows comes pre-installed on most PCs, and the lack of any compelling Linux-only applications for most users.

Unix generally, and Linux in particular, on the other hand, have come to dominate the server market. Linux powers the Web servers, email servers, file servers, and so on that make up the Internet and that many businesses rely on to provide local network services. Thus, most people use Linux daily even if they don't realize it.

In most cases, it's possible to use either Linux or Windows on a computer and have it do an acceptable job. Sometimes, though, specific needs dictate use of one OS or another. You might need to run a particular exotic program, for instance, or your hardware might be too old for a modern Windows or too new for Linux. In other cases, your own or your users' familiarity with one OS or the other may favor its use.

What Is a Distribution?

Up until now, I've described Linux as if it were a single OS, but this isn't really the case. Many different Linux *distributions* are available, each consisting of a Linux kernel along with a set of utilities and configuration files. The result is a complete OS, and two Linux distributions can differ from each other as much as either differs from OS X or even Windows. I therefore describe in more detail what a distribution is, what distributions are popular, and the ways in which distribution maintainers keep their offerings up to date.

Creating a Complete Linux-Based OS

Certification Objective

I've already described some of what makes up a Linux OS, but some details need reiteration or elaboration:

A Linux kernel A Linux kernel is at the core of any Linux OS, of course. I've written this item as *a* Linux kernel because the Linux kernel is constantly evolving. Two distributions are likely to use slightly different kernels. Distribution maintainers also often *patch* kernels—that is, they make small changes to fix bugs or add features.

Core Unix tools Tools such as the GNU tool set, the X Window System, and the utilities used to manage disks are critical to the normal functioning of a Linux system. Most Linux distributions include more or less the same set of such tools, but as with the kernel, they can vary in versions and patches.

Supplemental software Additional software, such as major server programs, desktop environments, and productivity tools, ships with most Linux distributions. As with core Unix software, most Linux distributions provide similar options for such software. Distributions sometimes provide their own "branding," though, particularly in desktop environment graphics.

Startup scripts Much of a Linux distribution's "personality" comes from the way it manages its startup process. Linux uses scripts and utilities to launch the dozens of programs that link the computer to a network, present a login prompt, and so on. These scripts and utilities vary between distributions, which means that they have different features and may be configured in different ways.

An installer Software must be installed to be used, and most Linux distributions provide unique installation software to help you manage this important task. Thus, two distributions may install in very different ways, giving you different options for key features such as disk layouts and initial user account creation.

The UNetbootin tool (`http://unetbootin.sourceforge.net`) can copy the files from a Linux installation disc image file to a USB flash drive.

Typically, Linux distributions are available for download from their Web sites. You can usually download a CD-R or DVD image file that you can then burn to an optical disc. When you boot the resulting disc, the installer runs and you can install the OS. You can sometimes download an image that can be copied to a USB flash drive if your computer lacks an optical drive.

Some Linux installers come complete with all the software you're likely to install. Others come with only minimal software and expect you to have a working Internet connection so that the installer can download additional software. If your computer isn't connected to the Internet, be sure to get the right type of installer.

A Summary of Common Linux Distributions

Certification Objective

Depending on how you count, there are about a dozen major Linux distributions for desktop, laptop, and small server computers, and hundreds more that serve specialized purposes. Table 1.1 summarizes the features of the most important distributions.

TABLE 1.1 Features of major Linux distributions

Distribution	Availability	Package format	Release cycle	Administrator skill requirements
Arch	Free	pacman	Rolling	Expert
CentOS	Free	RPM	approximately 2-year	Intermediate
Debian	Free	Debian	2-year	Intermediate to Expert
Fedora	Free	RPM	approximately 6-month	Intermediate
Gentoo	Free	ebuild	Rolling	Expert
Mandriva	Free	RPM	1-year	Intermediate
openSUSE	Free	RPM	8-month	Intermediate
Red Hat Enterprise	Commercial	RPM	approximately 2-year	Intermediate

(Continues)

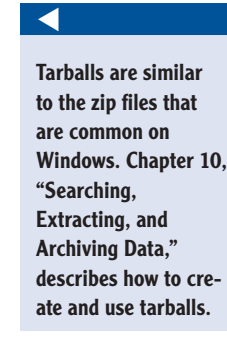
TABLE 1.1 *(Continued)*

Distribution	Availability	Package format	Release cycle	Administrator skill requirements
Slackware	Free	tarballs	Irregular	Expert
SUSE Enterprise	Commercial	RPM	2–3 years	Intermediate
Ubuntu	Free	Debian	6-month	Novice to Intermediate

These features require explanation:

Availability Most Linux distributions are entirely open source or free software; however, some include proprietary components and are sold for money, typically with a support contract. Red Hat Enterprise Linux (RHEL) and SUSE Enterprise Linux are the two most prominent examples of this type of distribution. Both have completely free cousins. For RHEL, CentOS is a near-clone that omits the proprietary components, and Fedora is an open version that serves as a testbed for technologies that may eventually be included in CentOS. For SUSE Enterprise, openSUSE is a free alternative.

Package format Most Linux distributions distribute software in *packages*, which are collections of many files in one. Package software maintains a local database of installed files, making upgrades and uninstallations easy. The RPM Package Manager (RPM) system is the most popular one in the Linux world, but Debian packages are very common, too. Other packaging systems work fine but are distribution-specific. Slackware is unusual in that it uses *tarballs* for its packages. These are package files created by the standard `tar` utility, which is used for backing up computers and for distributing source code, among other things. The tarballs that Slackware uses for its packages contain Slackware-specific information to help with package management. Gentoo is unusual because its package system is based on compiling most software from source code. This is time-consuming but enables experienced administrators to tweak compilation options to optimize the packages for their own hardware and software environments.



Release cycle I describe release cycles in more detail shortly, in "Understanding Release Cycles." As a general rule, distributions with short release cycles aim to provide the latest software possible, whereas those with longer release cycles strive

to provide the most stable environments possible. Some try to have it both ways; for instance, Ubuntu releases long-term support (LTS) versions in April of even-numbered years. Its other releases aim to provide the latest software.

Don't be scared off by the "intermediate" classification of most distributions. This book's purpose is to help you manage the essential features of such distributions.

Administrator skill requirements The final column in Table 1.1 provides my personal estimation of the skill level required to administer a distribution. As you can see, I've described most Linux distributions as requiring "intermediate" skill to administer. Some, however, provide less in the way of user-friendly GUI administrative tools, and so require more skill. Ubuntu aims to be particularly easy to use and administer.

Most Linux distributions are available for at least two platforms—that is, CPU types: *x*86 (also known as IA32, i386, and several variants) and *x*86-64 (also known as AMD64, EM64T, and *x*64). Until about 2007, *x*86 computers were the most common variety, but more recently, *x*86-64 computers have become the standard. If you have an *x*86-64 computer, you can run either an *x*86 or an *x*86-64 distribution on it, although the latter provides a small speed improvement. More exotic platforms, such as PowerPC, Alpha, and SPARC, are available. Such platforms are mostly restricted to servers and to specialized devices (described shortly).

In addition to the mainstream PC distributions, several others are available that serve more specialized purposes. Some of these run on regular PCs, but others run on their own specialized hardware:

Android is best known as a cell phone OS, but it can be used on other devices. Some e-book readers, for instance, run Android.

Android Many cell phones today use a Linux-based OS known as *Android*. Its user interface is similar to that of other smart phones, but underneath lies a Linux kernel and a significant amount of the same Linux infrastructure you'll find on a PC. Such phones don't use X or typical desktop applications, though; instead, they run specialized applications for cell phones.

Network appliances Many broadband routers, print servers, and other devices you plug into a local network to perform specialized tasks run Linux. You can sometimes replace the standard OS with a customized one if you want to add features to the device. Tomato (`http://www.polarcloud.com/tomato`) and OpenWrt (`https://openwrt.org`) are two examples of such customized Linux distributions. Don't install such software on a whim, though; if done improperly, or on the wrong device, they can render the device useless!

I recommend you download Parted Magic, or a similar tool, to have on hand in case you run into problems with your main Linux installation.

TiVo This popular digital video recorder (DVR) uses a Linux kernel and a significant number of standard support programs, along with proprietary drivers and DVR software. Although many people who use them don't realize it, they are Linux-based computers under the surface.

Parted Magic This distribution, based at `http://partedmagic.com`, is a Linux distribution for PCs that's intended for emergency recovery operations. It runs

from a single CD-R and you can use it to access a Linux or Windows hard disk if the main installation won't boot.

Android, Linux-based network appliances, and TiVo are examples of *embedded systems* that use Linux. Such devices typically require little or no administrative work from users, at least not in the way such tasks are described in this book. Instead, these devices have fixed basic configurations and guided setup tools to help inexperienced users set critical basic options, such as network settings and your time zone.

Certification Objective

Understanding Release Cycles

Certification Objective

Table 1.1 summarized the release cycles employed by a number of common Linux distributions. The values cited in that table are the time between releases. For instance, new versions of Ubuntu come out every six months, like clockwork. Most other distributions' release schedules provide some "wiggle room"; if a release date slides a month, that may be acceptable.

After its release, a distribution is typically supported until sometime *after* the next version's release—typically a few months to a year or more. During this support period, the distribution's maintainers provide software updates to fix bugs and security problems. Once the support period has passed, you can continue to use a distribution, but you're on your own—if you need updated software, you'll have to compile it from source code yourself or hope that you can find a compatible binary package from some other source. As a practical matter, therefore, it's generally a good idea to upgrade to the latest version before the support period ends. This fact makes distributions with longer release cycles appealing to businesses, since a longer time between installations minimizes disruptions and costs associated with upgrades.

Two of the distributions in Table 1.1 (Arch and Gentoo) have *rolling* release cycles. Such distributions have no version numbers in the usual sense; instead, upgrades occur in an ongoing manner. Using such a distribution makes it unnecessary to ever do a full upgrade, with all the hassles that creates; however, you'll occasionally have to do a disruptive upgrade of one particular subsystem, such as a major upgrade in your desktop environment.

Certification Objective

Prior to the release of a new version, most distributions make pre-release versions available. *Alpha software* is extremely new and very likely to contain serious bugs, while *beta software* is more stable but nonetheless more likely to contain bugs than is the final release software. As a general rule, you should avoid using such software unless you want to contribute to the development effort by reporting bugs or unless you're desperate to have a new feature.

The Essentials and Beyond

Linux is a powerful OS that you can use on everything from a cell phone to a supercomputer. At Linux's core is its *kernel*, which manages the computer's hardware. Built atop that are various utilities (many from the GNU project) and user applications. Linux is a clone of the Unix OS, with which it shares many programs. Mac OS X is another Unix OS, although one with a unique user interface. Although Windows shares many features with Unix, it's an entirely different OS, so software compatibility between Linux and Windows is limited. Linux comes in many varieties, known as distributions, each of which has its own unique "flavor." Because of this variety, you can pick a Linux version that best suits your needs, based on its ease of use, release cycle, and other unique features.

Suggested Exercises

- Make a list of the programs you run as an ordinary user, including everything from a calculator applet to a major office suite. Look for equivalents at `http://www.linuxrsp.ru/win-lin-soft/table-eng` or `http://www.linuxalt.com`. Is there anything you can't find? If so, try a Web search to find an equivalent.
- Read more about two or three Linux distributions by perusing their Web pages. Which distribution would you select for running a major Web server? Which distribution sounds most appealing for use by office workers who do word processing and email?

Review Questions

1. Which of the following is *not* a function of the Linux kernel?
   - **A.** Allocating memory for use by programs
   - **B.** Allocating CPU time for use by programs
   - **C.** Creating menus in GUI programs
   - **D.** Controlling access to the hard disk
   - **E.** Enabling programs to use a network
2. Which of the following is an example of an embedded Linux OS?
   - **A.** Android
   - **B.** SUSE
   - **C.** CentOS
   - **D.** Debian
   - **E.** Fedora

(Continues)

The Essentials and Beyond *(Continued)*

3. Which of the following is a notable difference between Linux and Mac OS X?

- **A.** Linux can run common GNU programs, whereas OS X cannot.
- **B.** Linux's GUI is based on the X Window System, whereas OS X's is not.
- **C.** Linux cannot run on Apple Macintosh hardware, whereas OS X can run only on Apple hardware.
- **D.** Linux relies heavily on BSD software, whereas OS X uses no BSD software.
- **E.** Linux supports text-mode commands, but OS X is a GUI-only OS.

4. True or false: The Linux kernel is derived from the BSD kernel.

5. True or false: If you log into a Linux system in graphical mode, you cannot use text-mode commands in that session.

6. True or false: CentOS is a Linux distribution with a long release cycle.

7. A Linux text-mode login prompt reads _____(one word).

8. A common security problem with Windows that's essentially nonexistent on Linux is _____.

9. Pre-release software that's likely to contain bugs is known as _____ and _____.

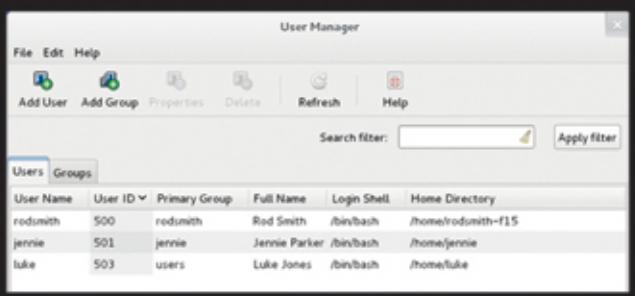
Contents

Learn Linux Quickly and Easily

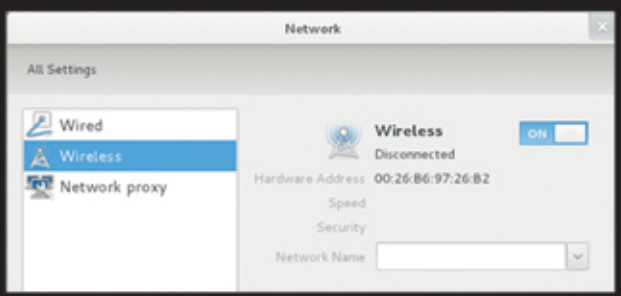
This full-color guide thoroughly covers Linux fundamentals. You'll begin with basic principles and the Linux way of doing things, then move on to common user programs such as the command line and text editors. With these skills in hand, you can tackle system administration tasks, such as user management and network configuration. Whether you're prepping for the LPI Linux Essentials certification or just brushing up on the basics for your professional work, this is the fast and thorough grounding you need.

Learn these Linux essentials—and more:

- Discovering where Linux fits into the OS world
- Managing hardware in Linux
- Using Bash, the Linux command line
- Using file management tools and commands
- Installing and controlling programs
- Finding and archiving data on your computer
- Editing text files with common Linux editors
- Automating tasks with the help of scripts
- Managing accounts and groups
- Setting file ownership and permissions



Manage your computer's users and groups



Configure your network connections

This striking *Essentials* book maximizes skill acquisition and knowledge retention with:

- Chapter-opening learning topics
- Step-by-step tutorials
- Four-color screenshots and illustrations
- Essentials and Beyond—summaries and additional suggested exercises
- Chapter review questions

About the Author

Roderick W. Smith, Linux+, LPIC-1, LPIC-2, is a recognized Linux open source programmer and networking expert. He is the author of over a dozen books on open source technologies. His books include *CompTIA Linux+ Complete Study Guide*; *LPIC-1: Linux Professional Institute Certification Study Guide, 2nd Edition*; *LPIC-2: Linux Professional Institute Certification Study Guide*, and *Linux Administrator Street Smarts*, all from Sybex.

www.sybex.com
www.sybex.com/go/linuxessentials

Sybex®
An Imprint of
WILEY

COMPUTERS/
Operating Systems/Linux

ISBN 978-1-118-10679-2
53999
9 781118 106792