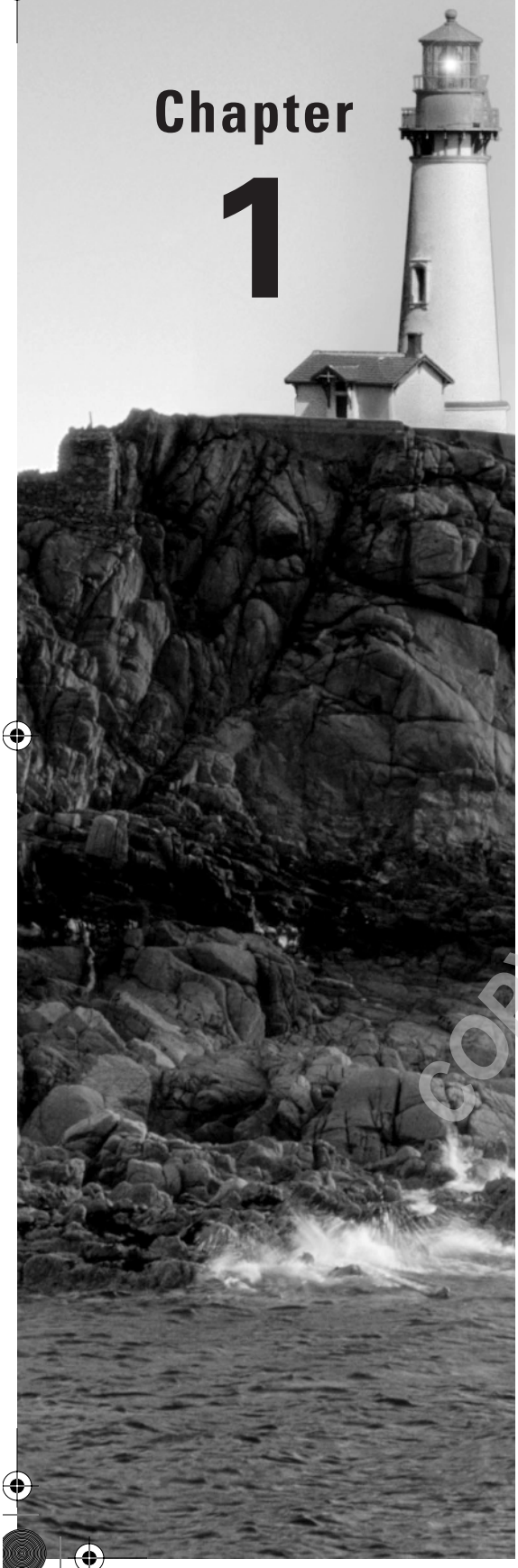
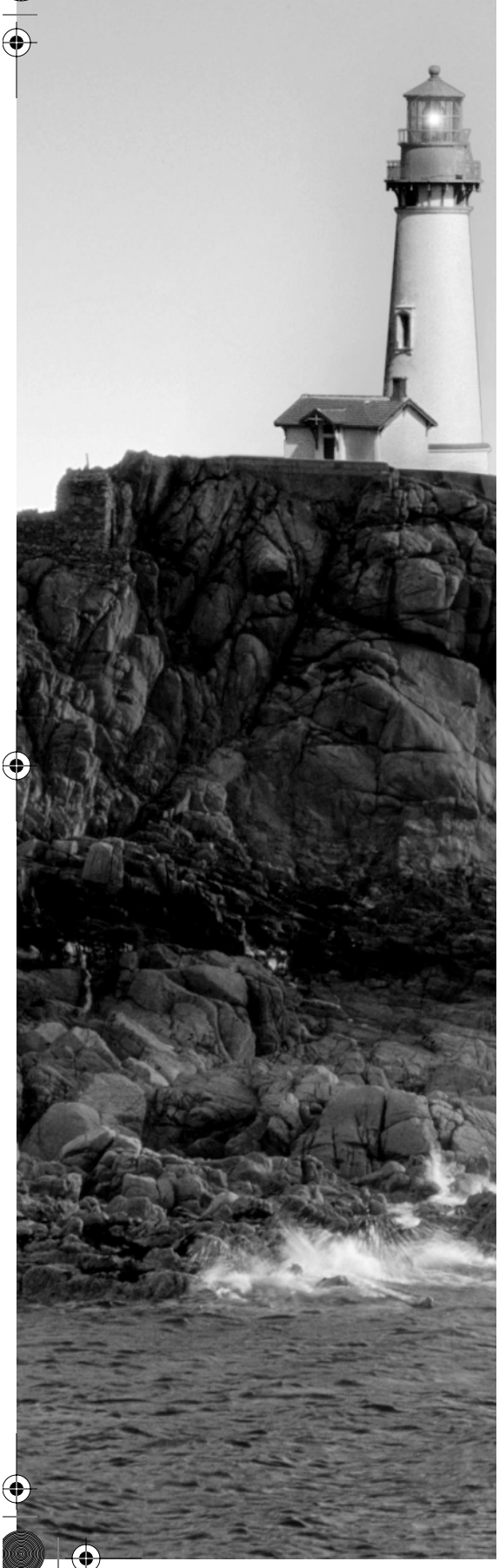


Chapter 1

Planning the Implementation

THE FOLLOWING COMPTIA OBJECTIVES ARE COVERED IN THIS CHAPTER:

- ✓ **1.1 Identify purpose of Linux machine based on predetermined customer requirements (e.g., appliance, desktop system, database, mail server).**
- ✓ **1.2 Identify all system hardware required and validate that it is supported by Linux (e.g., CPUs, RAM, graphics cards, storage devices, network interface cards, modem).**
- ✓ **1.3 Determine what software and services should be installed (e.g., client applications for workstation, server services for desired task), check requirements and validate that it is supported by Linux.**
- ✓ **1.4 Determine how storage space will be allocated to file systems (e.g., partition schemes).**
- ✓ **1.5 Compare and contrast how major Linux licensing schemes work (e.g., GNU/GPL, freeware, shareware, open source, closed source, artistic license).**
- ✓ **1.6 Identify the function of different Linux services (e.g., Apache, Squid, SAMBA, Sendmail, `ipchains`, BIND).**
- ✓ **1.8 Describe the functions, features, and benefits of a Linux solution as compared with other operating systems (e.g., Linux players, distributions, available software).**
- ✓ **1.10 Identify where to obtain software and resources.**
- ✓ **1.11 Determine customer resources for a solution (e.g., staffing, budget, training).**
- ✓ **7.1 Identify basic terms, concepts, and functions of system components, including how each component should work during normal operation and during the boot process.**



- ✓ **7.2 Assure that system hardware is configured correctly prior to installation (e.g., IRQs, BIOS, DMA, SCSI settings, cabling) by identifying proper procedures for installing and configuring ATA devices.**
- ✓ **7.3 Assure that system hardware is configured correctly prior to installation (e.g., IRQs, BIOS, DMA, SCSI settings, cabling) by identifying proper procedures for installing and configuring SCSI and IEEE 1394 devices.**
- ✓ **7.4 Assure that system hardware is configured correctly prior to installation (e.g., IRQs, BIOS, DMA, SCSI settings, cabling) by identifying proper procedures for installing and configuring peripheral devices.**
- ✓ **7.5 Assure that system hardware is configured correctly prior to installation (e.g., IRQs, BIOS, DMA, SCSI, cabling) settings by identifying available IRQs, DMAs, and I/O addresses and procedures for device installation and configuration.**

Most computers are not designed or sold with Linux in mind. This means that Linux doesn't always run properly on them, or it may not take full advantage of the computer's hardware. Therefore, if you need to buy or build a new computer, it's important to understand what Linux requires with respect to hardware so that you can buy a computer with the appropriate specifications.

Just as you should understand Linux's hardware requirements, you need to know something about the Linux software world. When you are determining what operating system (OS) to install on a computer, one of the most critical questions you should ask yourself is whether the software you need is available on the OS in question. Locating Linux software and understanding its licensing terms are also important aspects of software requirements for Linux.

Understanding these fundamental hardware and software features will help you in every subsequent aspect of Linux configuration and use because they lay the groundwork for additional Linux layers. Many of your installation choices (discussed in Chapter 2, "Installing Linux") depend on your hardware, for instance, and many details of system configuration and administration (discussed throughout the rest of the book) rely on your choice of Linux vendor.

Evaluating Computer Requirements

If you're building or buying a new computer, one of the first steps you must take is to lay out the system's general hardware requirements—the amount of RAM, the approximate *central processing unit (CPU)* speed, the amount of disk space, and so on. These characteristics are determined in large part by the role or roles the computer will play. For instance, a workstation for a graphics designer will require a large monitor and good video card, but an Internet server needs neither. Once you've decided the general outline of the hardware requirements, you can evaluate your resource limitations (such as your budget) and arrive at more specific hardware selections—specific brands and models for the individual components or for a prebuilt computer.

Workstations

A *workstation* is a computer that is used primarily or exclusively from that computer's own *console* (the keyboard and monitor attached directly to the computer). Workstations are sometimes also referred to as *desktop computers*, although some people apply the latter term to somewhat lower-performance computers without network connections, reserving the term "workstation" for systems with network connections.

Because they're used by individuals, workstations typically require fairly good input/output devices—a large display (typically 17-inch or larger), a high-quality keyboard, and a good 3-button mouse. (Linux, unlike Windows, uses all three buttons, so a two-button mouse is suboptimal.) Workstations also frequently include audio hardware (a sound card, speakers, and sometimes a microphone) and high-capacity removable media drives (Zip or LS-120 drives, perhaps CD-R or CD-RW burners, and often a DVD-ROM drive).

Cathode ray tube (CRT) displays have been the traditional favorite for desktop use, but in 2003 *liquid crystal display (LCD)* monitor sales surpassed sales of CRT displays. LCD display sizes are measured slightly differently than are CRT display sizes, so an LCD monitor is equivalent to a CRT monitor one to two inches larger.

CPU speed, memory, and hard disk requirements vary from one application to another. A low-end workstation that's to be used for simple tasks such as word processing can get by with less of each of these values than is available on new computers today. A high-end workstation that will be used for video rendering, heavy-duty scientific simulations, or the like may need the fastest CPU, the most RAM, and the biggest hard disk available. Likewise, low-end workstations are likely to have less cutting-edge network hardware than are high-end workstations, and the differing hard disk requirements dictate less in the way of backup hardware for the low-end workstation.

Servers

The word *server* can mean one of two things: a program that responds to network requests from other computers, or the computer on which the server program runs. When designing a computer, the latter is the appropriate definition. Servers usually have little or no need for user-oriented features such as large monitors or sound cards. Most servers make heavy use of their hard disks, however, so large and high-performance disks are desirable in servers. For the same reason, *Small Computer System Interface (SCSI)* disks are preferred to *Advanced Technology Attachment (ATA)* disks, also known as *Enhanced Integrated Device Electronics (EIDE)* disks—SCSI disks tend to perform better, particularly when multiple disks are present on a single computer. (This issue is covered more later in the chapter, in the "Hard Disk Space" section.) Likewise, servers by definition rely on the network, and busy servers may need top-notch network cards, and perhaps special dedicated network connections outside the computer itself.

Small servers, such as those handling a few users in a small office, don't need much in the way of CPU speed or RAM, but larger servers demand more of these quantities, especially RAM. Linux automatically buffers disk accesses, meaning that Linux keeps recent disk accesses in memory, and reads more than it requested from disk. These practices mean that when subsequent requests come in, Linux can deliver them from memory, which is faster than going back to the disk to obtain the data. Thus, a server with lots of RAM can often outperform an otherwise similar server with only a modest amount of RAM.

It's important to realize that server needs fall along a continuum; a very low-demand Web site might not require a very powerful computer, but a very popular Web site might need an extraordinarily powerful system. Many other types of servers are also available, including Usenet news servers, database servers, time servers, and more. (News and database servers are particularly likely to require very large hard disks.)

Dedicated Appliances

Some Linux systems function as dedicated appliances—as routers, print servers for just one or two printers, the OS in small robots, and so on. In some cases, as when the computer functions as a small router, Linux can enable recycling of old hardware that's otherwise unusable. Dedicated applications like these often require little in the way of specialized hardware. Other times, the application demands very specialized hardware, such as custom motherboards or touch-panel input devices. Overall, it's difficult to make sweeping generalizations concerning the needs of dedicated appliances.

Increasingly, Linux is being used in dedicated commercial devices—hardware sold as gadgets to perform specific functions but that happens to run Linux. For instance, some Sharp Zaurus palmtop computers, a growing number of broadband routers, and the TiVo digital video recorder all run Linux. In most cases, these embedded Linux systems are intended to be used by people who aren't trained in Linux, so these systems tend to mask their Linux innards from the user. If you dig into them, though, they're much like other Linux systems at their core. Their hardware tends to be unique, though, and they may use unusual software components and lack software that's popular on workstations and servers.

This book doesn't cover the unique aspects of embedded Linux.

Special Needs

Sometimes, the intended use of the computer requires specialized hardware of one variety or another. Common examples include the following:

Video input If the computer must digitize video signals, such as those from a television broadcast or a videotape, you will need a video input board. The Linux kernel includes drivers for several such products, and a variety of programs are available to handle such inputs. The Video4Linux project (`http://www.exploits.org/v4l/`) supports these efforts.

Scientific data acquisition Many scientific experiments require real-time data acquisition. This requires special timing capabilities, drivers for data acquisition hardware, and software. The Linux Lab Project (`http://www.llp.fu-berlin.de`) is a good starting point from which to locate appropriate information for such applications.

USB devices The *Universal Serial Bus (USB)* is a multipurpose external hardware interface. It's seeing increased use as an interface method for keyboards, mice, modems, scanners, digital cameras, printers, removable-media drives, and other devices. Linux added USB support in the 2.2.18 and later kernels. This support is good for many devices but weak or nonexistent for others. Check `http://www.linux-usb.org` to learn about support for specific devices. You'll also have to be sure to use a distribution with USB support, or at least upgrade the kernel to include this support.

IEEE-1394 devices *IEEE-1394* (also known as FireWire or i.Link) is a high-speed interface that's most commonly used for external hard disks and video input devices. As of the early 2.6.*x* kernel series, Linux's IEEE-1394 support is still weak, although some devices are supported, and the list of supported devices is growing. Check `http://www.linux1394.org` for details.

Determining Available Resources

Before you decide on the specific hardware you want for your new system, you should consider the resources available to you. These include any existing hardware that you can reuse, or with which a new system must integrate; the budget under which you must work; and the expertise available to you, both as it exists now and as it might exist after training. The following sections cover these factors.

Utilizing Existing Hardware

One resource you may have available is that of existing hardware. One of the reasons for Linux's popularity is that it can be stripped of graphical user interface (GUI) configuration tools and other resource hogs and still accomplish a great deal. This makes Linux an excellent OS with which to stretch the life of old hardware. Before you purchase a new system, you should also consider how it will integrate with your current infrastructure, such as your current network and existing printers.

Reusing Old Hardware

If you have much in the way of old hardware, you may be able to use it with Linux, particularly in specialized ways. For instance, an early Pentium, or even a 486 system, can make a good dedicated firewall for a small or medium-sized office. Such an application requires little CPU power, almost no disk space (some products for this purpose fit on a single floppy disk), and little RAM. Such a system can also be turned into a print server for two or three printers (this may require adding parallel port or USB cards); or, with the addition of more disk space and possibly RAM, it can be used as a light-duty file server. With a big enough monitor, such a system can function as a terminal (even a graphical X terminal) to other Linux or Unix computers.

Another approach for reusing old hardware is to scavenge parts for inclusion in an otherwise new system. Components like monitors, speakers, mice, keyboards, removable-media drives, hard disks, and many add-in cards can be moved from an old, decommissioned system to an otherwise new one, thus saving the cost of the new components. Of course, this is best done with

components that are in good condition. Also, new hardware is often faster or otherwise better than old hardware, so old components may not be useful in modern systems. A 1995 hard disk isn't likely to be large enough to be worth salvaging, for instance. Some technologies, such as those for RAM and CPUs, change so much that old components can't be used in new hardware after more than a year or two.

Integrating with Existing Infrastructure

When planning a new system, you must be aware of the environment in which it will be placed. This environment includes many components. Sometimes there's little you can do to make Linux compatible with this infrastructure; at other times, you can buy appropriate hardware or install software to make the system compatible. Here are a few examples:

Network connections In many offices, networking is critically important, so a Linux system must be able to work on the local network. If your office uses a Token Ring network, for instance, you'll need to track down and install Token Ring cards for which Linux drivers exist rather than using the more common Ethernet cards.

Printers If the computer must link directly to a printer, the two devices need compatible interfaces. Most *x*86 PCs sold today include one parallel, one or two serial, and two USB ports. You may need to add extra ports or buy a USB hub if you need to connect more printers than this. (Printer driver configuration is another important issue, which is discussed in Chapter 8, "Hardware Issues.")

Modems Linux works with most external RS-232 serial modems. External USB modems work if they follow the Communication Device Class/Abstract Control Model (CDC/ACM) protocol. Internal models may or may not work, depending on the model. Broadband (cable and DSL) modems work if they use Ethernet interfaces, but internal and USB devices both require explicit driver support, so check on that detail.

Scanners As a general rule, SCSI-interfaced scanners work well, while USB and parallel-port scanners may or may not work, depending on the model in question. Check out `http://www.sane-project.org` for information on Linux's support for scanners, which is provided by the Scanner Access Now Easy (SANE) project.

Removable-media devices Linux works well with most removable-media devices. Whether your office uses Zip, LS-120, magneto-optical, Jaz, CD-RW, or other media, Linux can use them. The main caveat relates to the interface. SCSI and ATA devices work well, as do many parallel-port and USB devices—but a few of the latter two types will cause problems. Also, some Windows CD-RW software creates discs that Linux can't read.

As a general rule, if you expect Linux to interface directly with a device, you should check on compatibility. You can usually find some way to get Linux working with your infrastructure, but you may need to buy extra hardware or use unusual drivers.

Balancing Budgetary Limitations

Whether you're working in an organization or purchasing a computer for yourself, it's a good idea to set a budget for a new computer before you buy. If you fail to do this, it's easy to fall

into a pattern of feature inflation, in which you decide to spend "just" $20 more on one upgrade, and "just" $30 more on another. Before long, you've added $500, $1000, or more to the cost of the computer. When setting your budget, remember that you may need to make some unexpected purchases after the fact. For instance, you may find that you need an adapter to connect an external SCSI device to your computer's external SCSI port.

There's a good chance you'll find that your ideal computer costs more than you've budgeted for it. If this happens, you have two choices: revise your budget or settle for a lesser computer. (You can, of course, do a little of both.) If you choose to trim, here are some suggestions:

- Today's hard disks frequently exceed 100GB in capacity, which is more than enough for most workstations and even many servers. You may be able to make do with a smaller hard disk than you'd planned. If necessary, you can add another hard disk in the future. Also, look for the hard disk capacity sweet spot—the point that carries the lowest price per gigabyte of storage. If you'd planned to buy at just above the sweet spot, reducing disk size by a little can reduce the cost by a lot.
- It's easy to list floppy, DVD-ROM, CD-RW, Zip, and Jaz drives as being required, but are they? You might be able to omit the DVD-ROM drive and use a CD-RW to both read and write CDs. An LS-120 drive can read ordinary floppy disks.
- Large monitors are very nice, but they may be overkill, particularly for low-end workstations and servers. LCD monitors are more expensive than CRTs capable of displaying a similar resolution. I do *not*, however, recommend that you cut costs by purchasing an unknown, low-cost monitor, particularly for a workstation. A monitor with a blurry screen, or one that flickers a lot, can cause eyestrain and headaches. Similarly, cheap keyboards can be a false economy on heavily used workstations if they contribute to carpal tunnel syndrome.
- CPU speed increases rapidly in the computer industry, and in 2003, CPUs are faster than required for many applications. If you truly need a fast CPU, by all means get one, but for many workstations, a mid-range CPU will do well. You may also want to look for the CPU price sweet spot. This often occurs just below the top of the line, so even if you need a speedy CPU, getting the model that's one or two notches below the top-speed model may make sense.
- Many of today's top video cards are targeted at game players, who want high-speed, three-dimensional effects. 3D effects are computationally expensive, and they also require a lot of RAM. For traditional office applications, a card with minimal or no 3D effects and 8MB of RAM is perfectly adequate. Some high-end graphics applications may need more, though.

One additional consideration, particularly if you're buying multiple Linux computers, is that you may be able to invest additional resources in a single computer rather than distribute the resources among a group of systems. For instance, a single computer with a large hard disk can function as a file server for dozens of computers with anemic hard disks. You can even give users accounts on a central system and have them run programs on that computer, using their own

systems as little more than terminals. In fact, this approach can be a great way to reuse old hardware—even a 486 can function as an adequate terminal (even for X-based GUI programs).

Considering Available Expertise

Ideally, once Linux is installed and running, the details of your hardware selections will be unimportant to the computer's ultimate users. Average users don't really care if you use an ATI or nVidia video card, and they won't need to manually reconfigure these components. Of course, the end user may care that you've selected a 17-inch rather than a 19-inch monitor to save money.

Local expertise becomes important in a few hardware devices, however. If you're installing a new network, you may need to arrange for training so that users can learn the necessary network applications. At anything but the smallest sites, it's also important to have somebody on hand to troubleshoot network problems as they arise. Removable disks occasionally require explanations, particularly warnings about proper care of the media. In Linux, recordable optical drives don't operate like other removable-media devices, so users must be told how to use appropriate Linux software. End users may also require training on tape backup devices.

End-user training and expertise are at least as important for Linux software as for hardware. Users who are used to Microsoft Windows or Mac OS can probably adapt to Linux without too much difficulty, but they'll find this task much easier if they're given a complete desktop environment, such as the K Desktop Environment (KDE) or the GNU Network Object Model Environment (GNOME). Both come with most Linux distributions, and are described briefly in Chapter 2. If you're migrating a large number of users from another OS to Linux, you may want to organize some introductory Linux orientation sessions in which you demonstrate Linux and highlight some of the differences between Linux and the old OS.

Deciding What Hardware to Use

Once you've decided on the approximate specifications for a computer and you've set a budget, you can begin deciding on exact specifications. If you possess the necessary knowledge, I recommend indicating manufacturer and model numbers for every component, along with one or two backups for each. (RAM, however, is close to being a commodity; few people shop for RAM by brand, although the *type* of RAM is important.) You can then take this list to a store and compare it to the components included in particular systems, or you can deliver your list to a custom-build shop to obtain a quote. If you don't have enough in-depth knowledge of specific components, you can omit the make and model numbers for some components, such as the hard disk, CD-ROM drive, monitor, and possibly the motherboard. You should definitely research Linux compatibility with video cards, network cards, SCSI host adapters (if you decide to use SCSI components), and sound cards (if the computer is to be so equipped). These components can cause problems for Linux, so unless you buy from a shop that's experienced in building Linux systems, a little research now can save you a lot of aggravation later when you try to get a component working in Linux.

A Rundown of PC Hardware

Computers are built from several components that must interact with one another in highly controlled ways. If a single component misbehaves or if the interactions go awry, the computer as a whole will malfunction in subtle or obvious ways. Major components in computers include the following:

Motherboard The *motherboard* (also sometimes called the mainboard) holds the CPU, RAM, and plug-in cards. It contains circuitry that "glues" all these components together. The motherboard determines what type of memory and CPU the computer can hold. It also includes the BIOS, which controls the boot process, and it usually has built-in support for hard disks, floppy disks, serial ports, and other common hardware.

CPU The CPU is the computer's brain—it performs most of the computations that result in a system's ability to crunch numbers in a spreadsheet, lay out text in a word processor, transform PostScript to printer-specific formats for a print queue, and so on. To be sure, some computations are performed by other components, such as some video computations by a video card, but the CPU does the bulk of the computational work.

Memory Computers hold various types of memory; the most common general classes of these are random access memory (RAM) and read-only memory (ROM). (RAM is volatile storage; it can be easily changed and holds current computations. ROM is difficult or impossible to change, and holds static information.) There are several varieties of each of these. Memory holds data, which can include Linux software and the data on which that software operates. Memory varies in access speed and capacity.

Disk storage Disk storage, like memory, is used to retain data. Disk storage is slower than memory, but usually higher in capacity. Typically, Linux itself resides on disk storage, and when the system boots, parts of Linux are loaded into RAM. In addition to the common hard disks, there are lower-capacity removable disks, CD-ROMs, and so on. Disks are controlled through ATA or SCSI circuitry on the motherboard or separate cards. As a general rule, Linux doesn't need specific drivers for disks, but Linux does need drivers for the controller.

Video hardware Video hardware includes the video card and the monitor. The video card may or may not literally be a separate card; sometimes it's built into the motherboard. Collectively, video hardware provides the primary means for a computer to communicate with its user, but Linux has the ability to do so through other computers' video hardware. Linux's video support is provided in two ways: through drivers in the kernel that work with just about any video card, at least in text mode; and through drivers in XFree86, Linux's GUI package, that work with most cards, but not absolutely all of them.

Input devices The keyboard and mouse allow you to give commands to the computer. These devices are well standardized, although there are a few variants of each type. Linux requires no unusual drivers for most common keyboards and mice (including trackballs and similar mouse alternatives), but if you use USB devices, you must ensure your kernel supports them. Kernels prior to 2.2.18 lacked USB support.

Network devices In most business settings, network hardware consists of an *Ethernet* card or a card for a similar type of computer network. Such networks link several computers together over a few tens or hundreds of feet, and they can interface to larger networks. Even many homes now use such a network. It's also possible to link computers via *modems*, which use telephone lines to create a low-speed network over potentially thousands of miles. These devices are usually quiescent until late in the boot process, when Linux may launch programs to begin network interactions. There are ways to boot a computer via network connections, though.

Audio hardware Many workstations include audio hardware, which lets the system play back sounds and digitize sounds using microphones or other audio input devices. These aren't critical to basic system functioning, though; Linux will boot quite well without a sound card.

To understand how these components interact, consider Figure 1.1, which shows a simplified diagram of the relationship between various system components. Components are tied together with lines that correspond to traces on a circuit board, chips on a circuit board, and physical cables. These are known as *busses*, and they carry data between components. Some busses are contained within the motherboard, but others are not. Components on a single bus can often communicate directly with one another, but components on different busses require some form of mediation, such as from the CPU. (Although not shown in Figure 1.1, there are lines of communication between the memory and PCI busses that don't directly involve the CPU.) A lot of what a computer does is coordinate the transfer of data between components on different busses. For instance, to run a program, data must be transferred from a hard disk to memory, and from there to the CPU. The CPU then operates on data in memory, and may transfer some of it to the video card. Busses may vary in speed (generally measured in megahertz, MHz) and width (generally measured in bits). Faster and wider busses are better than slower and narrower ones.

FIGURE 1.1 A computer is a collection of individual components that connect together in various ways.

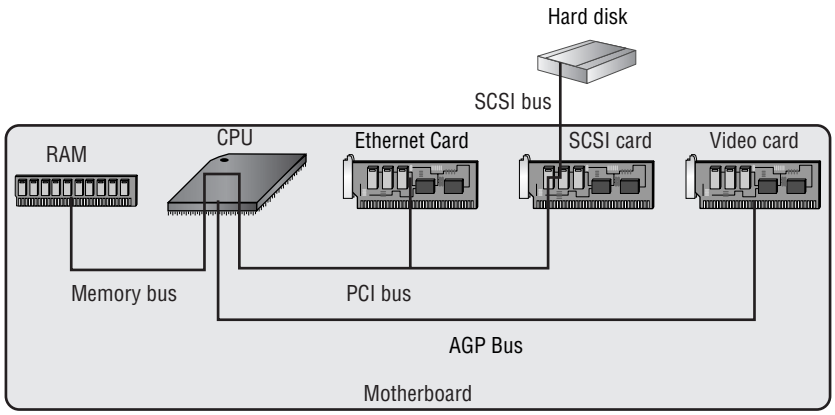


Figure 1.1 is *very* simplified. For instance, the link between the CPU and RAM passes through the motherboard's chipset and various types of cache, as described briefly in the upcoming section, "RAM."

The next few sections examine several critical system components in more detail.

CPU

Linux was originally developed for Intel's popular 80*x*86 (or *x*86 for short) line of CPUs. In particular, a 386 was the original development platform. (Earlier CPUs in the line lack features required by Linux.) Linux also works on subsequent CPUs, including the 486, Pentium, Pentium MMX, Pentium Pro, Pentium II, Pentium III, Pentium 4, and Celeron.

In addition to working on Intel-brand CPUs, *x*86 versions of Linux work on competitors' *x*86-compatible chips. Today, the most important of these are the AMD Athlon and Duron lines. VIA also sells a line of CPUs originally developed by Cyrix and IDT, but these lag substantially behind the offerings from Intel and AMD in speed. Transmeta sells *x*86-compatible CPUs with low power requirements, and Linux runs well on these CPUs. A few other companies have sold *x*86-compatible CPUs in the past, but these companies have failed or been consumed by others. (IBM and some other firms sell Cyrix or AMD designs under their own names, sometimes as part of a package to upgrade an existing motherboard beyond its originally designed maximum CPU speed.)

As a general rule, Linux has no problems with CPUs from any of the *x*86 CPU manufacturers. When a new CPU is introduced, Linux distributions occasionally have problems booting and installing on it, but such problems are usually fixed quickly.

CPU designs are often broadly described in terms of the width of their data registers and links to memory. These are busses much like the motherboard busses described earlier, but they're more closely tied to the CPU. Internal and external data busses determine the size of data chunks the CPU can use or transfer between the CPU and memory, respectively. Address busses determine the size of memory addresses, which in turn determines how much memory the computer can address.

Traditional *x*86 systems use 32-bit internal registers (that is, 32-bit internal data busses), although Pentium systems and above have 64-bit links to memory (that is, 64-bit external data busses). Some non-*x*86 systems use 64-bit internal registers, and both Intel and AMD have released 64-bit variants of the *x*86 architecture, which use 64-bit internal data busses and external address busses. Each company has taken a different path to 64-bit computing, though. The Intel variant is known as IA-64 and has been implemented in Intel's Itanium CPU line. IA-64 works best with code that has been specially designed for the IA-64 architecture. The Linux kernel works on IA-64, and some IA-64 Linux distributions are available. Although IA-64 is capable of running 32-bit *x*86 code, switching between 32-bit and 64-bit

modes isn't simple, so running 32-bit programs on these CPUs is impractical. Fortunately, recompiling all Linux programs to work on IA-64 CPUs, although not trivial, is practical.

AMD's 64-bit version of the *x*86 architecture is known as *x*86-64 or AMD64, and it's been released under the CPU names Opteron and Athlon 64. These CPUs do a better job of running 32-bit software than do IA-64 CPUs, and in fact they can often run entire 32-bit OSs, including Linux. This fact makes these CPUs appealing choices for a migration to 64-bit computing, particularly if you're stuck with some older commercial applications that aren't available in native 64-bit form.

In addition to *x*86 CPUs and their IA-64 and AMD64 derivatives, Linux runs on many unrelated CPUs, including the Apple/IBM/Motorola PowerPC (PPC), Compaq's (formerly DEC's) Alpha, and the SPARC CPU in Sun workstations. Linux is most mature on *x*86 hardware, and that hardware tends to be less expensive than hardware for other architectures; therefore, it's generally best to buy *x*86 hardware for Linux.

The best CPUs of some non-*x*86 lines sometimes perform slightly better than the best *x*86 CPUs, particularly in floating-point math, so you might favor alternative architectures for these reasons. You might also want to dual-boot between Linux and an OS that's available for some other architecture, such as Mac OS.

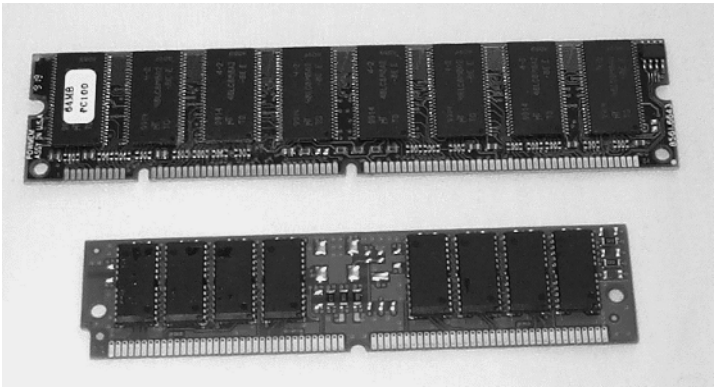
When comparing CPU performance, most people look at the chips' speeds in megahertz or gigahertz (GHz; 1GHz is 1,000MHz). This measure is useful when comparing CPUs of the same type; for instance, a 2.1GHz Celeron is slower than a 2.6GHz Celeron. Comparing across CPU models is trickier, though, because one model may be able to do more in a single CPU cycle than another can. What's worse, this comparison may differ according to the nature of the computation. For instance, in general, *x*86 CPUs have a reputation for poor floating-point math performance, although they've been improving on this measure in recent years. Thus, an Intel CPU might be the equal of an Alpha in most tasks, but the Alpha might have a substantial advantage in applications that require floating-point math, such as ray tracing and certain scientific applications. When comparing different CPUs (for instance, Pentium 4 to Athlon), you should look at a measure such as MIPS (millions of instructions per second) or a benchmark test that's relevant to your intended application. (The Linux kernel uses a measure called BogoMIPS as a calibration loop when it boots, but this is *not* a valid measure of CPU performance; it's used only to calibrate some internal timing loops.) The best measure is how quickly the software *you* use runs on both CPUs.

CPUs plug into specific motherboards, which are the main (and sometimes the only) major circuit board in a computer. The motherboard contains a *chipset*, which implements major functions such as an ATA controller, an interface between the CPU and memory, or an interface to the keyboard. Linux works with most motherboards, although on occasion, Linux doesn't support all of a motherboard's features. For instance, a motherboard may include an integrated video or audio chipset for which Linux drivers are immature or nonexistent. The key consideration in choosing a motherboard is that it is compatible with the CPU you buy—both its model and its speed. If you buy a preassembled system, this won't be a concern.

RAM

RAM comes in several forms, the most common of which in 2003 is the *dual inline memory module (DIMM)*. Older motherboards and some other components use the *single inline memory module (SIMM)* format, which comes in both 30-pin and 72-pin varieties. Figure 1.2 shows a DIMM and a 72-pin SIMM. A few motherboards use *RDRAM inline memory modules (RIMMs)*, which physically resemble DIMMs but use a special type of RAM known as *RAMbus dynamic RAM (RDRAM)*. Laptops and some compact computers use a *Small Outline (SO) DIMM*, which is similar to a SIMM or DIMM, but narrower.

FIGURE 1.2 Currently, DIMMs (top) are used in many computers; SIMMs (bottom) are used in older computers and some peripherals.



Motherboards host sockets for particular types of memory—30-pin SIMM sockets in many 486 and older motherboards, 72-pin SIMM sockets in some 486- and Pentium-class motherboards, DIMM sockets in some Pentium-class and later motherboards, and RIMM sockets in some Pentium II and later motherboards. Depending on the module and CPU type, you may need to add modules singly, in pairs, or in groups of four. Pentium and later systems take 72-pin SIMMs in pairs and DIMMs or RIMMs singly.

In addition to differences in physical interfaces, RAM varies in its electronic characteristics. RAM today is largely derived from *dynamic RAM (DRAM)*, which has spawned many improved variants, such as fast page mode (FPM) DRAM, extended data out (EDO) DRAM, synchronous DRAM (SDRAM), double data rate (DDR) SDRAM, and RDRAM. Most motherboards accept just one or two types of RAM, and with the exception of RDRAM and RIMMs, the physical format of the memory does not clearly indicate the RAM's electronic type. In 2003, most motherboards accept some combination of SDRAM, DDR SDRAM, or RDRAM, and possibly one or two lesser varieties. DDR SDRAM and RDRAM are the speed champions today. Each has its adherents. DDR SDRAM uses fairly conventional improvements to regular DRAM, delivering fast memory access by using a wide (64-bit) and moderately fast (66–133 MHz) bus. RDRAM uses a more unusual design in which the RIMM uses a narrow (16-bit) but unusually fast (800MHz) bus externally and a separate bus within the RIMM that uses a more conventional configuration.

RAM also varies in how well it copes with errors. Computer memory is composed of individual *bits*, which are *binary* (base 2) numbers—each digit is either 1 or 0. A *byte* is composed of 8 bits. If a single bit changes its value, say because of a cosmic ray hitting the memory, the data becomes corrupted. This can cause subtle or extreme errors in computations, or it can result in other data being corrupted. Some memory modules incorporate a ninth bit (known as a parity bit) in each byte as an error-detection bit. This bit is encoded to indicate whether an even or odd number of bits in the other 8 bits in the byte are set. If an error occurs, the motherboard's memory controller can detect this fact. Unfortunately, the usual result is a system crash, the idea being that it's better to crash the computer than to propagate bad data.

A more sophisticated approach is error correction. Like the error-detection process just described, error correction also requires one extra bit per byte (or, more precisely, 8 extra bits for every 64 bits in a Pentium-class or above system). In this case, though, a motherboard that supports error correction can correct 98 percent of the errors that occur, resulting in no disruption to the computer's operation. This is clearly a desirable characteristic, particularly in mission-critical systems such as important servers.

All of these characteristics apply to *main memory*, which, as you might imagine, is the main type of memory in a computer. Motherboards or CPUs also support another type of memory, though—*cache memory*. A computer has much less cache memory than main memory (typically about 1MB), but the cache memory is much faster. The system stores frequently used memory in the cache, which results in a substantial performance increase. Typically, two caches exist. The first, known as the L1 cache, resides in the main part of the CPU and is a few kilobytes in size. On Pentium-class and earlier systems, the second cache, known as L2, is on the motherboard and can sometimes be upgraded. On Pentium Pro, Athlon, and later systems, the L2 cache is on the CPU package, but it's not part of the same chip as the CPU. A few motherboards that take CPUs with an on-board L2 cache also provide a cache on the motherboard. In this configuration, the motherboard's cache is known as the L3 cache.

Linux itself is unconcerned with these details. To Linux, memory is memory, and the OS doesn't particularly care about what physical or electronic form the memory takes or whether it supports any form of error detection or correction. All these details are handled by the motherboard, which is why it's so important that your memory match the motherboard's requirements.

When upgrading a computer's memory, try to buy from a retailer that has a memory cross-reference tool. Such a tool may be a Web-based form or a printed book. You look up or enter your motherboard or computer model and find a specific model of memory that's compatible with your computer. If such a tool is unavailable, check your motherboard's manual for detailed specifications about the types of memory it accepts, and use those specifications when shopping.

Hard Disk Space

The great divide in hard disks is between ATA and SCSI devices. Both of these busses come in a variety of speeds, ranging from less than 10 megabytes per second (MB/s) to 640MB/s, with

higher speeds on the way. To achieve a given speed, both the hard disk and its interface must support the same speed. For instance, using an old 10MB/s Fast SCSI drive with an 80MB/s Ultra2 Wide SCSI host adapter will yield only 10MB/s speeds, not 80MB/s speeds.

It's important to distinguish between the speed of the *interface* and the speed of the *device*. Manufacturers typically emphasize the speed of the interface, but the mechanical device usually can't support these speeds. A hard disk might have an 80MB/s Ultra2 Wide SCSI interface but be capable of only 35MB/s sustained transfer rates. Manufacturers express the device's true maximum speed as an *internal transfer rate*, as opposed to the *external transfer rate* (of the interface). To further confuse matters, many manufacturers give the internal transfer rate in mega*bits* per second (Mbps), but the external rate in mega*bytes* per second (MB/s). If you fail to do the appropriate conversion (dividing or multiplying by 8), you'll erroneously believe that the interface is the bottleneck in data transfers to and from the device. Disks can transfer data at their external transfer rate only when they've previously stored data from the disk in their internal caches. For this reason, external speeds substantially higher than internal speeds can produce modest speed benefits, and disks with large caches are preferable to those with small caches.

As a general rule, SCSI devices are preferred in computers in which disk performance is important. There are several reasons for this:

- Depending on the variety of SCSI, each SCSI host adapter can support 7–15 devices on one hardware interrupt, or occasionally more if the adapter supports multiple SCSI busses. (There are only 15 interrupts available in the *x*86 architecture, and many are reserved for critical hardware such as the keyboard.) ATA, by contrast, supports just two devices per cable (and hence per interrupt), although most motherboards include support for two chains (using two interrupts), for a total of four devices.
- SCSI devices multitask better than ATA devices do. Given sufficient capacity on the SCSI host adapter, multiple SCSI devices can be engaged in data transfers at full speed. ATA, by contrast, dedicates its full capacity to one device per chain, even if that device can't use the ATA controller's full capacity.
- Hard disk manufacturers tend to release their fastest and highest-capacity drives in SCSI format. ATA drives tend to be slower and smaller.

These advantages are substantial, but for many situations, they're overwhelmed by one advantage of ATA: It's less expensive. As just mentioned, modern *x*86 motherboards ship with support for two ATA chains, so there's no need to buy an ATA controller. ATA hard disks are also typically less expensive than SCSI devices of the same capacity, although the ATA drives are often slower.

Both ATA and SCSI have traditionally been parallel busses, meaning that they consist of several data lines—enough to transfer an entire byte at once. Timing issues make it hard to boost the speed of a parallel interface past a given point, though, so both ATA and SCSI are moving toward newer serial hardware interfaces. For ATA, the serial variant is known as *Serial ATA (SATA)*; for SCSI, it's *Serial Attached SCSI (SAS)*. In 2003, SATA is rare in the installed base, although it's starting to appear on new hardware, and SAS has yet to be released. These standards are likely to dominate the internal drive market by mid-decade. The groups working on these standards are now merging them; the result may eventually be called SATA-2, but such

devices don't yet exist. Other competing formats include IEEE-1394 and USB 2.0, both of which are popular for external hard drives.

On the whole, SCSI is worthwhile when disk performance is important or when you need to support a large number of storage devices (including CD-ROM, DVD-ROM, removable disk, and tape drives). For most low-end and even mid-range workstations, though, ATA's lower cost makes it appealing, and ATA performance is adequate for many such systems.

Fortunately, Linux's support for both ATA and SCSI adapters is excellent. Most ATA controllers can be run in an old-style (and slow) mode using generic drivers, but faster speeds often require explicit driver support. Therefore, you may want to check on Linux's ATA drivers for your motherboard or ATA controller. There is no generic SCSI host adapter support, so you must have support for your specific SCSI host adapter. Serial variants require their own drivers, so check on Linux support before buying. Likewise, look into Linux drivers for IEEE-1394 or USB drives before buying one. Linux's IEEE-1394 and USB support makes these disks look like SCSI disks.

Once you configure Linux to work with an ATA controller or a SCSI host adapter, you don't need to worry about support for specific models of disk. (If you recompile your kernel, you need to explicitly include support for hard disks or any other devices attached to your adapter, but this support is present by default in all major Linux distributions.) You can purchase hard disks and other storage devices on the basis of capacity, speed, and the reputation for quality of a manufacturer or model.

Hard disks consist of spinning platters with read/write heads reading or writing data from those platters as the platters move under the heads. You'll usually see ads that list the rotational velocity of platters—such as 7200 revolutions per minute (rpm)—and the latency or seek time—such as 9 milliseconds (ms). A faster rotational velocity translates into data passing under the heads faster. The latency is the time it takes to move a head to a new position from the center of the disk and to begin retrieving data. Unfortunately, these two figures aren't the only important ones in determining the speed of a hard disk. The data density—how much data can be packed into a given amount of space—interacts with the rotational velocity to determine disk speed. A 7200-rpm disk might actually be faster than a 10,000-rpm disk, if the former has a substantially higher data density. For this reason, you should look for the actual transfer rates, expressed in MB/s or Mbps.

The data transfer rate varies between inner and outer tracks—outer tracks contain more data than inner tracks do, so outer tracks produce higher transfer rates.

Network Hardware

Ethernet is the most common type of network today. There are several varieties of Ethernet, including 10Base-2 and 10Base-5 (which use thin and thick coaxial cabling, respectively); 10Base-T, 100Base-T, and 1000Base-T (which use twisted-pair cabling similar to telephone wires); and 1000Base-SX (which uses fiber-optic cabling). In any of these cases, the first number (10, 100, or 1000) represents the maximum speed of the network in Mbps. 1000Mbps Ethernet

is often called gigabit Ethernet. Of these classes of Ethernet, 100Base-T is currently the most popular choice for new installations, although gigabit Ethernet is gaining in popularity.

Most 100Base-T network cards also support 10Base-T speeds. This fact can help you migrate a network from 10Base-T to 100Base-T; you can install dual-speed cards in new systems and eventually replace older 10Base-T hardware with dual-speed hardware to upgrade the entire network. Similarly, many 1000Base-T cards also support 100Base-T and even 10Base-T speeds.

Linux's support for Ethernet cards is, on the whole, excellent. Linux drivers are written for particular chipsets rather than specific models of network card. Therefore, the driver names often bear no resemblance to the name of the card you've bought, and you may use the same driver for boards purchased from different manufacturers. Fortunately, most distributions do a good job of auto-detecting the appropriate chipset during installation, so you probably won't have to deal with this issue if the card is installed when you install Linux.

Chapter 8 covers adding new hardware, should you need to add a network card after the fact.

If you're faced with the choice, purchase a *Peripheral Component Interconnect (PCI)* card rather than an *Industry Standard Architecture (ISA)* card. PCI cards tend to be easier to configure, and they support higher transfer rates. The ISA bus tops out at a theoretical maximum speed of 64Mbps—less than that of 100Base-T Ethernet. A 32-bit PCI card has a theoretical maximum speed of 1056Mbps, which is barely enough for gigabit Ethernet. In practice, PCI can't handle this full speed because the needs of other devices and deviations from theoretical maximum performance will degrade performance. (Rare 64-bit PCI variants are better able to sustain full gigabit Ethernet speeds.) Some motherboards ship with Ethernet hardware built in. This hardware generally uses a PCI bus, although it's built into the motherboard and lacks a physical PCI connector.

Linux supports networking standards other than Ethernet, but these devices are less well supported overall. Linux includes support for some Token Ring, Fiber Distributed Data Interface (FDDI), LocalTalk, Fibre Channel, and wireless products, among others. If your existing network uses one of these technologies, you should carefully research Linux's support for specific network cards before buying one.

Networking hardware outside the computer doesn't require Linux-specific drivers. Network cables, hubs, switches, routers, and so on are all OS-independent. They also generally work well with one another no matter what their brands, although brand-to-brand incompatibilities occasionally crop up.

One partial exception to the rule of needing no specific Linux support is in the case of network-capable printers. If you buy a printer with a network interface, you must still have appropriate Linux printer drivers to use the printer, as described in Chapter 8. Fortunately, network-capable printers usually understand PostScript, which is ideal from a Linux point of view.

Video Hardware

Linux works in text mode with just about any video card available for *x*86 systems. This means that you can log in, type commands, use text-based utilities, and so on. Such operation is probably adequate for a system intended to function as a server, so the selection of a video card for a server need not occupy too much of your time. Workstations, though, usually operate in GUI mode, which means they run XFree86 or a commercial X Window System (X for short) server.

Unlike most other drivers, the drivers necessary to operate a video card in the bitmapped graphics modes used by X do not reside in the kernel; they're part of the X server. Therefore, you should research the compatibility of a video card with XFree86 (`http://www.xfree86.org`) or the commercial X servers, Accelerated-X (`http://www.xig.com`) and Metro-X (`http://www.metrolink.com`). Because XFree86 ships with all major Linux distributions, it's best to use a board it supports. As a general rule of thumb, it's best to avoid the most recent video cards because drivers for XFree86 tend to lag a few months behind the release of the hardware. A few manufacturers do provide XFree86 drivers for their products, though, and the commercial X servers sometimes introduce drivers more rapidly than the XFree86 team does.

The Linux kernel includes a number of video drivers, known as frame buffer drivers. XFree86 includes a driver to interface to these kernel-level drivers. This approach is particularly common outside the *x*86 world, but it usually produces poorer performance than using a native XFree86 driver.

One important question when you're deciding on a video card is how much memory it should contain. The video card uses on-board memory to store a copy of the image displayed on the screen. Because of this, the video card must have enough memory to hold this image. The formula for determining this value is

$$R = x \times y \times b \div 8{,}388{,}608$$

In this equation, R is the RAM in megabytes, x and y are the width and height of the screen, respectively, and b is the color depth in bits (typically 8, 16, 24, or 32). For instance, to support a 1024×768 display at 16-bit color depth requires 1.5MB of RAM.

Most video cards have at least 8MB of RAM, which is more than enough to handle a 1600×1200 display with a 32-bit color depth—a very high resolution and color depth. Cards with more memory than this typically use it in conjunction with 3D effects processors, which are useful in games and certain types of 3D rendering packages. 3D acceleration is still rare in Linux, and few Linux programs take advantage of these effects. If you need them, you should research 3D support carefully before settling on a product to buy.

In addition, most video cards use a special video bus, known as the *Advanced Graphics Port (AGP)* bus. Some motherboards include video hardware connected via the AGP bus; others provide a special socket into which you plug an AGP video card. The AGP bus produces better video performance than do other busses, such as PCI and ISA. Some older video cards use these or more exotic busses, though.

Miscellaneous Hardware

Some hardware is so well standardized that there's no reason to give it much thought for Linux compatibility. The following are included in this category:

Cases Computer cases are hunks of plastic and metal shaped to hold other components. Usually, they also include power supplies, fans, and a few wires. Cases do vary in quality—check for rough edges, a good fit, and easy access. There's nothing OS-specific about them, though.

Floppy drives Standard floppy drives are very standardized. A few variant technologies exist, though, such as LS-120 drives, which typically interface via the ATA port. These may need to be treated like hard disks in the `/etc/fstab` configuration file (described in Chapter 6, "Managing Files and Services").

CD-ROM drives Today, most CD-ROM drives use either the ATA or the SCSI interface, and the devices are very well standardized. (ATA drives use a software extension, known as the ATA Packet Interface, or ATAPI.) The main exceptions are drives that use USB or IEEE-1394 interfaces. Even DVD-ROM drives are well standardized. Recordable and rewriteable CDs (CD-R and CD-RW drives) and recordable DVD drives are also well standardized.

Tape drives Most tape drives use a standard ATAPI or SCSI interface. These drives almost always respond to a standardized set of commands, and therefore don't require a special configuration in Linux. There are a few older floppy-interfaced drives that work with the Linux ftape drivers, which are part of the kernel. Some old parallel-interfaced drives can cause problems, and newer USB-interfaced drives are as yet rare and not well tested.

Keyboards Standard PC keyboards are well supported by Linux and require no special configuration. Some keyboards include special keys that may not be recognized by Linux, though, such as volume-control keys or keys that launch specific applications. USB keyboards are also available. They are supported in 2.4.*x* and later kernels, but they aren't as well tested.

Mice Most mice today use USB or PS/2 interfaces, but some older mice used RS-232 serial or various exotic interfaces. All are well supported, although USB support prior to the 2.4.*x* kernels was poor. Note that the tracking technology (conventional wheeled mouse, optical mouse, trackball, touchpad, and so on) is unimportant; it's only the interface protocols and the type of hardware interface that matter. Mice using USB or PS/2 hardware use the PS/2 protocol or a variant of it that supports wheels.

Serial and parallel ports If you need to add extra serial or parallel ports, you can do so with plug-in cards. These cards are fairly well standardized, so they'll seldom pose serious problems with Linux itself, although they can sometimes conflict with other hardware. USB-to-serial and USB-to-parallel adapters are also available and well supported in Linux.

Monitors Monitors don't require drivers, although you may have to know certain features of a monitor to configure it in XFree86. Specifically, you may need to know the monitor's maximum horizontal and vertical refresh rates (expressed in kHz and Hz, respectively). With

XFree86 4.0 and later, the X server can sometimes obtain this information from the monitor. (Chapter 2 covers XFree86 configuration in detail.)

Some other types of hardware require special consideration. These devices may require unusual drivers or configuration in Linux. Examples include the following:

USB devices Linux needs drivers for each USB device; a single USB driver isn't sufficient to handle all USB devices. The 2.2.18 and later kernels support many—but by no means all—USB devices. Check `http://www.linux-usb.org` for information on what's currently supported.

Internal modems In years gone by, internal modems seldom caused problems in Linux, because they were essentially composed of ordinary modem hardware linked to an ordinary serial port, all on one card. Today, though, internal modems are more likely to be *software modems*—devices that rely on the CPU to do some of the modem's traditional chores. Such devices require special drivers, which sometimes don't exist for Linux. Check `http://www.linmodems.org` for information on what's supported and what's not.

Sound cards Linux supports most sound cards. (Sound hardware is increasingly being integrated on the motherboard, but this fact is unimportant from a Linux software perspective.) The standard kernel includes drivers for many cards. Commercial variants of these drivers (often called the Open Sound System, or OSS) are available from `http://www.4front-tech.com`. An entirely separate project, the Advanced Linux Sound Architecture (ALSA; `http://www.alsa-project.org`), aims to replace the standard kernel drivers and supports a different (but overlapping) set of cards. The ALSA drivers are being integrated into the Linux kernel with the 2.5.*x* kernel series. You can also check to see whether the sound card vendor provides drivers, which may be unique or work along with the kernel or ALSA core.

Video acquisition boards Video acquisition hardware includes cameras (which typically interface via the parallel, USB, IEEE-1394, or RS-232 serial ports) and internal cards that accept television input signals. The Video4Linux project (`http://www.exploits.org/v4l/`) is devoted to developing tools for such devices, and the standard kernel includes many of the requisite drivers—but be sure to check for supported hardware if this is important.

Aside from trivial components such as cables, you should be cautious about adding hardware to a Linux computer without checking its compatibility with Linux. It's easy to forget that computer hardware often requires drivers, and if nobody has written appropriate drivers for Linux, that hardware simply will not work. These drivers can also vary in quality, which partially explains why one device may work well while another works poorly.

Unreliable drivers can be a major cause of system instability. Most drivers have privileged access to the computer's hardware as well as to kernel data structures. As a result, a bug in a driver is unusually likely to crash the system or cause other major problems.

Checking Hardware Configuration Before Installation

One of the reasons to buy a preassembled computer is so that you won't have to worry about all the pesky little details of hardware configuration. Unfortunately, life doesn't always work out that way. Prebuilt computers often come with one or more components configured suboptimally, so even if you buy such a system, it's wise to review the hardware configuration before you install Linux. Some settings, if incorrect, can cause problems during Linux installation, or soon thereafter.

Because most hardware is inside the computer's case, you must open that case to check the hardware's status. This poses three dangers. First, you might suffer an electrical shock if the computer is plugged into a wall outlet. Some power supplies have power switches independent of the computer's main switch; turning these off can reduce this risk. Second, static charges built up in your own body (say, from shuffling across a carpet in dry weather) can damage computer components. You can reduce this risk by grounding yourself frequently—for instance, by wearing a wrist strap designed for that purpose or by frequently touching a water faucet, radiator, or the computer's power supply if it's plugged into the wall. Finally, some computers (particularly notebooks and other small or specialized devices) aren't meant to be opened, so opening the case may void your warranty.

Checking Cabling

Several types of devices use cables, typically to link a device to the motherboard or to a controller card of some type. These cables can be entirely internal or external, depending on the device type. Particular types of cable have specific requirements, which are described in the following sections.

Power Cables

The most obvious power cable to most users is the one that stretches from a wall outlet, power strip, or uninterruptible power supply (UPS) to the computer. This cable is much like power cables on many other devices, and it should be fully inserted into the computer and its power source.

A second class of power cables resides inside the computer case. These cables stretch from the power supply (a rectangular metal box inside the computer to which the external power cable attaches) to the motherboard and various disk devices (hard disk, floppy disk, CD-ROM drive, and so on). Several types of internal power connectors are available. Most power supplies have about half a dozen connectors of various forms, each of which connects to just certain types of devices—the motherboard, hard disk devices, or floppy devices. You should check that power

connectors are all inserted firmly in their respective devices because they sometimes work loose during shipping.

So-called AT-style motherboards (used on many Pentium and earlier computers) used two motherboard power connectors, rather than the integrated connector used in later ATX systems. These AT connectors *must* be inserted side by side, with the black wires next to each other. These connectors can be inserted in each other's sockets, which will *destroy* the motherboard!

Some motherboards have connectors that supply power to fans—typically CPU fans, but sometimes extra case fans. Other systems rely on connectors direct from the case power supply to drive internal fans. In any event, be sure these power connectors are firmly attached to their appropriate supply points.

Internal Data Cables

The second major form of internal cabling is data cables. These carry data between components—typically between a disk or tape device and a motherboard or controller. The most common form of data cable is a *ribbon cable*, so called because the cable resembles a ribbon. Ribbon cables differ in their widths and in the exact forms of their connectors. Some also have unique characteristics, such as a twisted portion on floppy cables. Common ribbon cables include 34-pin floppy, 40-pin ATA, 50-pin SCSI, and 68-pin Wide SCSI.

You should check that all cable connectors are inserted firmly and correctly. Most cables include notches or asymmetrical connectors so that they cannot be inserted backward, but some cheap cables lack these safeguards. If some of your cables are so crippled, pay careful attention to the cable's orientation. Most cables include a colored stripe on one edge, which indicates the location of the first signal line. The matched connector on the device or board should indicate the location of pin #1, probably in tiny type by the connector. This pin also usually has a square (as opposed to a round) solder joint. Be sure to plug the cable in so that the stripe is next to pin #1.

Some types of ribbon cable can have more connectors than devices. For instance, it's possible to use a SCSI cable with four connectors when you have just two SCSI drives, leaving one connector unused (two connectors attach to the SCSI drives and one to the host adapter). For most types of cable, you should ensure that the end connectors are both used. Normally, one of these attaches to the motherboard or controller card, and the other end attaches to one of the devices.

Particularly on older systems, ribbon cables sometimes link internal to external connectors. For instance, a motherboard might have an internal connector for its parallel port, so a ribbon cable ties this to an external parallel-port connector. Such cables are rare on modern motherboards, which integrate the connector into the motherboard in a standard location so that it's directly accessible from outside the case. You might still find such cables on a few designs—for instance, if they are being used to link a USB port to a front-panel USB connector.

Ribbon cables aren't the only type of internal data cable. CD-ROM drives frequently sport three-wire cables to tie the CD-ROM drive's audio output to a sound card. There are also

two-to-four-wire connectors that link the motherboard to front-panel computer components, such as the power button, the reset button, and the hard disk activity LEDs.

LED cables must be connected in the correct orientation, but the cables aren't keyed, so you have a 50/50 chance of getting it wrong unless you pay careful attention to the positive and negative markings on the cables and motherboard. This detail isn't important for the power or reset switches on modern computers.

External Cables

External cables connect the computer to its keyboard, mouse, and monitor. Printers, scanners, network connections, and so on also use external cables. (A few wireless devices exist, but even these often use short cables to link from a standard port to a radio or infrared transmitter.)

In all cases, for a device to function properly it's important that the associated cable be inserted firmly into its matching socket. Some cable types, such as Ethernet (RJ-45) cables, snap into place and cannot be removed unless you push a small lever or similar locking mechanism. Others, such as parallel, RS-232 serial, and some varieties of external SCSI connectors, have thumbscrews that can be tightened to ensure a snug connection (some of these require an actual screwdriver to tighten and loosen). Others, such as USB and keyboard connectors, have no locking or tightening mechanism, so you must be sure these connectors are fully and firmly inserted.

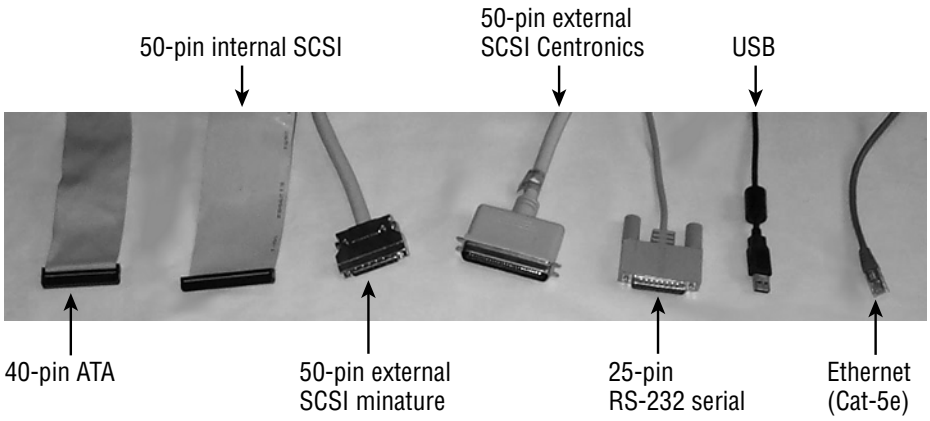
Some cable types should not be routinely connected or disconnected when the computer is in operation. These include SCSI, RS-232 serial, and parallel connectors. When attaching or detaching such a cable, a short can damage the device or the computer. Other connectors, such as those for USB and Ethernet, are designed for *hot swapping*—attachment and detachment when the computer is in operation.

Because you'll be plugging external devices in yourself, you should be sure you do this job correctly. It's easy to mistakenly connect a device to the wrong port. This is particularly true for RS-232 serial devices since many computers have two such ports; for speakers, microphones, and audio inputs on sound cards; and for PS/2-style mice and keyboards. USB ports are interchangeable on most computers; it doesn't matter which one you use.

Some connectors are electrically compatible but come in different sizes or shapes. This is particularly true of RS-232 serial connectors (which come in 9- and 25-pin varieties), keyboard connectors (which come in large AT-style and small PS/2-style connectors), and external SCSI connectors (which come in several varieties, such as 25-pin, 50-pin, Centronics-style, 50-pin miniature, and 68-pin miniature). Adapters for these are available, but be cautious with them—the adapters can add enough weight to the connector so that it's likely to fall out. This is particularly true of one-piece keyboard adapters and some types of SCSI adapters.

Figure 1.3 shows several common internal and external cable types. The 40- and 50-pin ribbon cables are hard to tell apart by sight except by width. The external cables' connectors are more varied in appearance, although some can be easily confused at first glance. The ends of parallel printer cables that connect to printers look like slightly narrower versions of the 50-pin Centronics-style SCSI cable shown in Figure 1.3, for instance.

FIGURE 1.3 Internal and external cables come in a wide variety of shapes and sizes, although some resemble each other.



Checking IRQ, DMA, and I/O Settings

Most plug-in boards use various hardware resources that are in limited supply in the *x*86 architecture. Of particular interest are the board's *interrupt request (IRQ) number*, its *direct memory access (DMA)* channel, and its *input/output (I/O)* port. The *x*86 architecture supports just 15 interrupts (0–15, with IRQs 2 and 9 being the same), each of which permits a device to signal that it needs attention from the CPU. There are also just a handful of DMA channels, which enable devices to transfer blocks of data to and from memory with little CPU intervention. I/O ports are in less short supply, but still occasionally produce conflicts; these are memory areas that devices and CPUs use to exchange data. Boards use an interrupt to tell the CPU that something important is happening that requires the CPU's attention. DMA channels and I/O ports are used to transfer data from the board to the computer's memory or CPU. Table 1.1 summarizes common IRQ, DMA, and I/O settings for some popular device types. Most of these settings can be changed, however, at least within limited ranges. Some devices, such as SCSI host adapters and Ethernet cards, don't have standardized resource settings. Also, not all devices use all of these resource types, so some cells in Table 1.1 are empty.

TABLE 1.1 Common Hardware Resource Settings

Device	Common IRQs	Common DMA Channels	Common I/O Ports (Hexadecimal)
System Timer	0		0040–005F
Keyboard Controller	1		0060–006F
Second Interrupt Controller (for IRQs 8–15)	2	4	
Real-Time Clock	8		
Math Coprocessor	13		00F0-00FF
PS/2 Mouse Port	12		
RS-232 Serial Port 1 (`/dev/ttyS0`)	4		03F8–03FF
RS-232 Serial Port 2 (`/dev/ttyS1`)	3		02F8–02FF
Parallel Port 1 (`/dev/lp0`)	7	3	0378–037F or 03BC–03BF or 0778–077F
Parallel Port 2 (`/dev/lp1`)	5		0278–027F or 0678–067F
USB Port	9 or 10		FF80–FF9F
SoundBlaster-Compatible Sound Card	5	1, 5	0220–0233, 0240–0253, or 0260–0263
Floppy Disk Controller	6	2	03F0–03F5
ATA Controller 1	14		01F0–01F7, 03F6, FFA0–FFA7
ATA Controller 2	15		0170–0177, 0376, FFA8–FFAF

With ISA, it's important that two devices don't attempt to use the same IRQ, DMA channel, or I/O port. Doing so can result in one board being unavailable, and in extreme cases, it can crash the computer. Of particular interest, note that both SoundBlaster-compatible sound cards and second parallel ports use the same IRQ, which can cause problems with these devices. Fortunately, most modern sound cards are flexible in their IRQ use, and multiple parallel ports are

becoming rare as USB becomes more popular. PCI boards may be able to share an IRQ with another PCI board, but even this sometimes causes the hardware to work slowly or behave strangely.

The motherboard uses several IRQs for its own devices. In fact, most of the devices specified in Table 1.1 reside on the motherboard on modern computers.

If you have any old ISA boards, you can check their IRQs by examining jumper settings on the boards themselves. Consult the board's documentation for details. Newer ISA boards use software configuration, as described in Chapter 8. PCI boards are auto-configured by the computer's BIOS or by the Linux kernel. In both of these latter cases, it's impossible to tell what hardware resources a board will use without booting the computer. Unfortunately, if the resources cause a conflict, the computer may not boot completely. If you suspect this may be happening, consult Chapter 8 for hardware troubleshooting information.

Checking ATA Devices

Most *x*86 computers use ATA for hard disks, CD-ROMs, and often other types of disk and tape devices. Several variants on ATA are available, ranging in speed from 8MB/s to 133MB/s, with faster speeds in the works. In 2003, 66MB/s is considered low end, 100MB/s is common, and 133MB/s is the interface of choice. These different interface types are referred to by various names, which usually include the speed, such as "UltraDMA/66" or "ATA/133," although the official names do not include these speeds. The more capable parallel ATA interfaces can communicate with less-capable devices, and vice versa, so you can mix and match if you need to—but each chain runs at just one speed, so you can seriously degrade a fast disk's performance by attaching it to the same cable as a slow CD-ROM or the like. Table 1.2 summarizes ATA hardware types.

Each parallel ATA chain can support the controller and up to two devices. Traditionally, you must configure each device to be either the *master* or the *slave*. In Linux, the master device takes on a lower device letter in its `/dev/hdx` device filename, where *x* is the device letter. Configuring master/slave status is done through a switch or jumper on the device itself; consult your documentation for details. Most modern devices and controllers support an auto-configuration protocol, typically enabled by setting jumpers to a setting called Cable Select.

If you need more than two devices, or if you want to separate fast and slow devices, you must use multiple ATA chains, each of which corresponds to one physical ATA cable. Most motherboards support two chains (hence four devices total), and you can add more by adding plug-in ATA controller cards. You can use similar cards to upgrade to faster forms of ATA.

Normally, one ATA device will be the master on the first (or primary) chain. A second device might be the slave on the same chain or the master on a second chain. The former configuration preserves IRQs, which may be desirable if you have lots of other devices, but the second is likely to produce better performance.

TABLE 1.2 ATA Hardware Types

Official Name	Unofficial Names	Maximum Speed	Added PIO Modes	Added DMA Modes	Cable Type
ATA-1	IDE	8.3MB/s	0, 1, 2	0, 1, 2, Multiword 0	40-wire parallel
ATA-2	EIDE	16.6MB/s	3, 4	Multiword 1, Multiword 2	40-wire parallel
ATA-3		16.6MB/s			40-wire parallel
ATA-4	UltraDMA/33, ATA/33	33.3MB/s		UltraDMA 0, UltraDMA 1, UltraDMA 2	40-wire or 80-wire parallel
ATA-5	UltraDMA/66, ATA/66	66.6MB/s		UltraDMA 3, UltraDMA 4	80-wire parallel
ATA-6	UltraDMA/100, ATA/100	100MB/s		UltraDMA 5	80-wire parallel
ATA-7[1]	UltraDMA/133, ATA/133	133MB/s		UltraDMA 6	80-wire parallel
Serial ATA		150MB/s			7-wire serial

[1]As of late 2003, ATA-7 had not been officially approved, but many manufacturers produce hardware that complies with the draft ATA-7 specifications.

Modern ATA controllers and drives all support both DMA-driven access and *Programmed Input/Output (PIO)* access. DMA access uses direct board-to-memory transfers, whereas PIO access uses the device's I/O ports and CPU to transfer data to memory. DMA access is therefore less CPU intensive than PIO access is. Table 1.2 lists the DMA and PIO modes used by each type of device, but you don't need to be concerned with this detail at this point; you can adjust the drive's DMA and PIO modes after you've installed and configured Linux. Chapter 8 describes Linux's tools for making these adjustments.

Checking SCSI Devices

SCSI is an unusually capable and complex interface bus. For this reason, SCSI busses can sometimes be difficult to configure correctly, particularly when they're loaded down with many devices. Factors you should consider when planning or checking a SCSI bus include the following:

SCSI variant Many versions of SCSI are available, ranging from the original 5MB/s SCSI-1 to the 640MB/s Ultra640 SCSI. Most of these versions are compatible with one another, but adding a less-capable device to an otherwise more-capable SCSI chain can degrade performance. Also, the more different two devices are, the less likely they are to get along on one chain. Adding a SCSI-1 device to an Ultra3 SCSI chain, for instance, is likely to cause problems. Table 1.3 summarizes the features of current SCSI variants.

SCSI IDs SCSI devices are differentiated by their ID numbers. Older SCSI variants (those that use a bus that's 8 bits wide) use ID numbers that range from 0 to 7, while Wide variants (which use 16-bit busses) have IDs that range from 0 to 15. The SCSI host adapter itself consumes one number, so this is the source of the 7- or 15-device limit on SCSI chains. SCSI IDs are generally set with jumpers on internal devices, or via some sort of switches or dial on external devices. Check your documentation for details. If two devices share an ID, it's likely that one will mask the other, or they'll both malfunction quite seriously. New devices can often use the SCSI Configured Automatically (SCAM) protocol, which allows devices to acquire IDs automatically.

Termination A SCSI bus can be thought of as a one-dimensional chain of devices. The devices on both ends of the chain must be terminated, which keeps signals from bouncing back from the end of the chain. Several types of termination are associated with different SCSI variants, ranging from passive to active to low-voltage differential (LVD). Most SCSI devices include termination that can be activated by setting a jumper, or even automatically. Sometimes you need to add a separate SCSI terminator. Be sure this detail is set correctly because incorrect termination can lead to bizarre errors, which can crash a Linux system.

Cable quality SCSI—and particularly high-speed SCSI variants—is quite susceptible to problems caused by low-quality cables. Particularly if your SCSI chain has many devices, it can be worthwhile to purchase high-quality cables. These are, unfortunately, likely to be expensive—often $50 or more.

Cable length Maximum SCSI cable lengths range from 1.5 to 12 meters (m), depending on the SCSI version. SCSI cable length limits apply to the *entire* SCSI chain. If you have two external SCSI devices, for instance, you sum the lengths of the external cables, along with any internal cables, to determine your SCSI chain's cable length.

Chapter 8 includes information on troubleshooting a SCSI chain.

TABLE 1.3 SCSI Hardware Types

SCSI Type	Speed	Termination	Cable Type	Maximum Cable Length
SCSI-1	5MB/s	Single-ended	25- or 50-pin	6m
SCSI-2	5MB/s	Single-ended	50-pin	6m
Fast SCSI-2	10MB/s	Single-ended	50-pin	3m
Fast/Wide SCSI-2	20MB/s	Single-ended	68-pin	3m
UltraSCSI	20MB/s	Single-ended	50-pin	3m or 1.5m[1]
UltraWide SCSI	40MB/s	Single-ended	68-pin	3m or 1.5m[1]
Ultra2 Wide SCSI	80MB/s	LVD	68-pin	12m
Ultra3 SCSI or Ultra160 SCSI	160MB/s	LVD	68-pin	12m
Ultra320 SCSI	320MB/s	LVD	68-pin	12m
Ultra640 SCSI	640MB/s	LVD	68-pin	12m

[1]Maximum cable length is 3m for four or fewer devices and 1.5m for five or more devices.

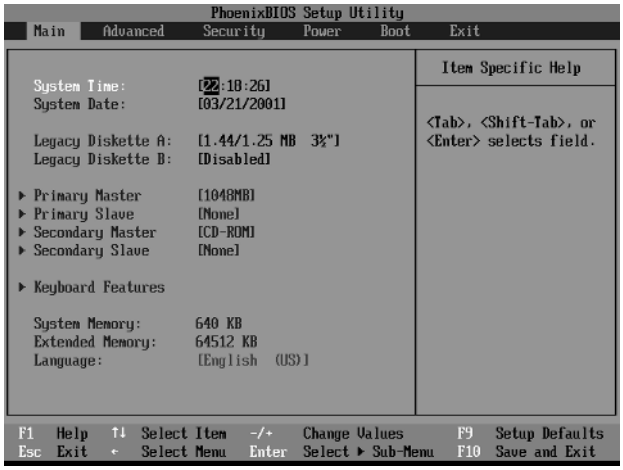
Checking BIOS Settings

The *Basic Input/Output System (BIOS)* is the lowest-level software component in a computer. The CPU runs BIOS code as part of its startup procedure. As a result, the BIOS configures many fundamental aspects of the computer before Linux has a chance to boot. The BIOS also provides tools that the computer uses to load the Linux kernel into memory.

Although the *x*86 BIOS provides some standard features, it's not entirely standardized. In particular, modern BIOSs provide a setup tool, often referred to as the *Complementary Metal Oxide Semiconductor (CMOS) setup utility*, that you can use to set various low-level options. The options available in a computer's CMOS setup utility differ from one computer to another, both because of differences in hardware and because of different BIOS designs.

Most computers display a prompt at boot time that tells you how to get into the CMOS setup utility. This is usually done by pressing a key, such as Delete or F2, at a critical point during the boot process. Once you've done this, you'll see a BIOS setup screen, such as the one shown in Figure 1.4. This screen allows you to select and set various options, typically by moving through menus by pressing the arrow keys on the keyboard.

FIGURE 1.4 CMOS setup utilities use menu-driven displays to let you adjust a computer's built-in hardware.



SCSI host adapters often include their own BIOSs and setup utilities, which are separate from the motherboard BIOS. The SCSI setup utilities usually have setup options that you can adjust by pressing a key sequence at a particular point in the boot process. Watch your boot displays or consult your SCSI adapter's documentation for details.

Most systems come with reasonable default BIOS settings, but you may want to check, and possibly adjust, a few. These include the following:

Disk settings You may need to adjust two common hard disk settings. The first specifies the size of the disk. An auto-detection feature normally works well for this. The second setting determines how the BIOS interprets the disk's cylinder/head/sector (CHS) addresses. On most BIOSs, a linear block addressing (LBA) mode is the best choice. If you use SCSI hard disks, the main motherboard BIOS won't detect them. This is normal; the SCSI BIOS provides the necessary support.

On-board ports Modern motherboards include RS-232 serial, parallel, USB, ATA, and frequently other types of ports. Some motherboards include video or audio hardware as well. You can enable or disable these or change their settings (for instance, you can change the IRQs used by the devices). Disabling unused ports can free up resources for other devices.

PCI settings Some BIOSs allow you to specify how the system treats PCI devices. Most commonly, you can choose from two or more rules for how the BIOS assigns IRQs to PCI devices. Sometimes, one rule results in IRQ conflicts and another doesn't, so such a setting is worth investigating if you have problems booting and suspect IRQ conflicts.

Passwords In a high-security environment, you may want to set a BIOS password. This prevents the system from booting unless the correct password is entered. It can slow down intruders who have physical access to the computer and boot with their own boot disk, but if intruders have physical access to the computer, they can bypass this feature in various ways. Setting a BIOS password also prevents automatic reboots in the event of a power failure. Nonetheless, slowing down an intruder may be worthwhile in some environments.

Memory settings BIOSs can be configured to copy parts of themselves, or of BIOSs stored on other devices, to RAM. This practice, which is known as *shadowing*, speeds up access to the BIOS, and it is useful in DOS, which relies on the BIOS for input/output. Linux doesn't use the BIOS as much, so it's generally best to disable all shadowing in Linux, which can result in slightly more memory available in Linux. Some BIOSs also allow you to control one or more memory holes—regions of the CPU's memory map that are unusable. These sometimes cause Linux to incorrectly detect the amount of RAM installed in the computer, so you may want to experiment with different memory hole settings.

Boot devices Modern BIOSs support booting from a wide variety of disk and disk-like devices, including floppy disks, ATA disks, SCSI disks, CD-ROM drives, and high-capacity removable disks like Zip or LS-120 disks. You can usually set the system to boot from some subset of these devices in any order you like. The BIOS tries each medium in turn, and if it's not present or isn't bootable, it tries the next one. For highest security, set the system to boot from your ATA or SCSI hard disk first; for convenient booting of installation or emergency media, set it to boot from a CD-ROM, floppy, or other removable media drive first.

In practice, you may need to experiment with a particular computer's CMOS settings to determine which work best. It's generally not a good idea to try random changes on a working system, though; experiment with these settings only if you're having trouble. Making changes without cause can produce an unbootable system, although if you remember what you changed, you can usually recover your system to a working state.

Most CMOS setup utilities include an option that restores the settings to the factory default values. These options may not always produce optimal results, but they'll usually work.

Planning Disk Partitioning

Hard disks can be broken into logical chunks known as *partitions*. In Windows, partitions correspond to drive letters (`C:`, `D:`, and so on). In Linux, partitions are mounted at particular points in the Linux directory tree, so they're accessible as subdirectories. Before actually installing Linux, it's a good idea to give some thought to how you'll partition your hard disk. A poor initial partitioning scheme can become awkward because you'll run out of space in one partition when another has lots of available space or because the partition layout ties your hands in terms of achieving particular goals.

The PC Partitioning System

The original *x*86 partitioning scheme allowed for only four partitions. As hard disks increased in size and the need for more partitions became apparent, the original scheme was extended in a way that retained compatibility with the old scheme. The new scheme uses three partition types:

- *Primary partitions*, which are the same as the original partition types
- *Extended partitions*, which are a special type of primary partition that serves as a placeholder for the next type
- *Logical partitions*, which reside within an extended partition

For any one disk, you're limited to four primary partitions, or three primary partitions and one extended partition. Many OSs, such as DOS, Windows, and FreeBSD, *must* boot from primary partitions, and because of this, most hard disks include at least one primary partition. Linux, however, is not so limited, so you could boot Linux from a disk that contains no primary partitions, although in practice few people do this.

The *x*86 partitioning scheme isn't the only one around. Linux includes support for many alternatives, but *x*86-based Linux systems generally use the PC partitioning scheme. Linux systems running on other architectures tend to use the partitioning systems native to those architectures. From an administrative point of view, these systems are almost always simpler than the PC system because there aren't any distinctions between primary, extended, and logical partitions.

A disk's primary partition layout is stored in a data structure known as the *partition table*, which exists on the first sector of the hard disk. This sector is known as the *master boot record (MBR)* because it also contains some of the first code to be run by the computer after the BIOS initializes. The locations of the logical partitions are stored within the extended partition, outside of the MBR. Although they are not a part of the MBR, these data are sometimes considered to be part of the partition table because they do define partition locations.

Linux Partition Requirements

To Linux, there's very little difference between the partition types. Linux numbers partitions on a disk, and the primary and extended partitions get the numbers from 1 to 4 (such as `/dev/hda1` or `/dev/sdc3`), while logical partitions get numbers from 5 up. This is true even if there are fewer than four primary and extended partitions, so partitions might be numbered 1, 2, 4, 5, and 6 (omitting partition 3). Primary partition numbers are like fixed slots, so when a disk uses just 1–3 of these slots, any of the four numbers may go unused. Logical partitions, by contrast, are always numbered sequentially, without any missing numbers, so a system with precisely three logical partitions *must* number them 5, 6, and 7.

Some administrators use a primary Linux boot partition because a conventional *x*86 MBR can boot only from a primary partition. When the computer does so, it runs code in the boot sector of the boot partition. Typically, Linux places a special boot loader program in this location. The Grand Unified Boot Loader (GRUB) and the Linux Loader (LILO) are the two boot loaders most

commonly found on *x*86 Linux systems. Alternatively, GRUB or LILO can reside directly in the MBR, which is more direct but leaves the boot loader more vulnerable to being wiped out should some other utility rewrite the MBR. (Chapter 3, "Software Management," covers the boot process, GRUB, and LILO in more detail.)

At a bare minimum, Linux needs a single partition to install and boot. This partition is referred to as the *root partition*, or simply as `/`. This partition is so called because it holds the root directory, which lies at the "root" of the directory "tree"—all files on the system are identified relative to the root directory. The root partition also stores directories, such as `/etc` and `/bin`, that fall off the root directory and in which other files reside. Some of these directories can serve as *mount points*—directories to which Linux attaches other partitions. For instance, you might mount a partition on `/home`.

One important directory in Linux is `/root`, which serves as the system administrator's home directory—the system administrator's default program settings and so on go here. The `/root` directory is not to be confused with the root (`/`) directory.

Common Optional Partitions

In addition to the root partition, many system administrators like creating other partitions. Some advantages that come from splitting an installation into multiple partitions rather than leaving it as one monolithic root partition include:

Multiple disks When you have two or more hard disks, you *must* create separate partitions—at least one for each disk. For instance, one disk might host the root directory and the second might hold `/home`. Also, removable disks (floppies, CD-ROMs, and so on) must be mounted as if they were separate partitions.

Better security options By breaking important directories into separate partitions, you can apply different security options to different partitions. For instance, you might make `/usr` read-only, which reduces the chance of accidental or intentional corruption of important binary files.

Data overrun protection Some errors or attacks can cause files to grow to huge sizes, which can potentially crash the system or cause serious problems. Splitting key directories into separate partitions guarantees that a runaway process in such a directory won't cause problems for processes that rely on the ability to create files in other directories. This makes it easier to recover from such difficulties. On the downside, splitting partitions up makes it more likely that a file will legitimately grow to a size that fills the partition.

Disk error protection Disk partitions sometimes develop data errors, which are data structures that are corrupted, or a disk that has developed a physically bad sector. If your system consists of multiple partitions, such problems will more likely be isolated to one partition, which can make data recovery easier or more complete.

Backup If your backup medium is substantially smaller than your hard disk, breaking up your disk into chunks that fit on a single medium can simplify your backup procedures.

Ideal filesystems A *filesystem* is a set of low-level data structures that regulate how the computer allocates space on the disk for individual files, as well as what types of data are associated with files, such as file creation times and filenames. Sometimes, one filesystem works well for some purposes but not for others. You might therefore want to break the directory tree into separate partitions so that you can use multiple filesystems.

So, what directories are commonly split off into separate partitions? Table 1.4 summarizes some popular choices. Note that typical sizes for many of these partitions vary greatly depending on how the system is used. Therefore, it's impossible to make recommendations on partition size that will be universally acceptable.

For more information, see Chapter 7, "Managing Partitions and Processes."

TABLE 1.4 Common Partitions and Their Uses

Partition (mount point)	Typical size	Use
Swap (not mounted)	1.5–2 times system RAM size	Serves as an adjunct to system RAM; is slow, but enables the system to run more or larger programs. Discussed in more detail in Chapter 8.
`/home`	200MB–200GB	Holds users' data files. Isolating it on a separate partition preserves user data during a system upgrade. Size depends on number of users and their data storage needs.
`/boot`	5–50MB	Holds critical boot files. Creating as a separate partition allows for circumventing limitations of older BIOSs and boot loaders on hard disks over 8GB.
`/usr`	500MB–6GB	Holds most Linux program and data files; this is frequently the largest partition.
`/usr/local`	100MB–3GB	Holds Linux program and data files that are unique to this installation, particularly those that you compile yourself.
`/opt`	100MB–3GB	Holds Linux program and data files that are associated with third-party packages, especially commercial ones.

TABLE 1.4 Common Partitions and Their Uses *(continued)*

Partition (mount point)	Typical size	Use
/var	100MB–200GB	Holds miscellaneous files associated with the day-to-day functioning of a computer. These files are often transient in nature. Most often split off as a separate partition when the system functions as a server that uses the /var directory for server-related files like mail queues.
/tmp	100MB-20GB	Holds temporary files created by ordinary users.
/mnt	N/A	/mnt isn't itself a separate partition; rather, it or its subdirectories are used as mount points for removable media like floppies or CD-ROMs.

Some directories—/etc, /bin, /sbin, /lib, and /dev—should *never* be placed on separate partitions. These directories host critical system configuration files or files without which a Linux system cannot function. For instance, /etc contains /etc/fstab, the file that specifies what partitions correspond to what directories, and /bin contains the mount utility that's used to mount partitions on directories.

The 2.4.*x* and later kernels include support for a dedicated /dev filesystem, which obviates the need for files in an actual /dev directory, so in some sense, /dev can reside on a separate filesystem, although not a separate partition.

Real World Scenario

When to Create Multiple Partitions

One problem with splitting off lots of separate partitions, particularly for new administrators, is that it can be difficult to settle on appropriate partition sizes. As noted in Table 1.4, the appropriate size of various partitions can vary substantially from one system to another. For instance, a workstation is likely to need a fairly small /var partition (say, 100MB), but a mail or news server might need a /var partition that's gigabytes in size. Guessing wrong isn't fatal, but it is annoying. You'll need to resize your partitions (which is tedious and dangerous) or set up symbolic links between partitions so that subdirectories on one partition can be stored on other partitions.

For this reason, I generally recommend that new Linux administrators try simple partition layouts first. The root (/) partition is required, and swap is a very good idea. Beyond this, `/boot` can be very helpful on hard disks of more than 8GB with older distributions or BIOSs, but is seldom needed with computers or distributions sold since 2000. An appropriate size for `/home` is often relatively easy for new administrators to guess, so splitting it off generally makes sense. Beyond this, I recommend that new administrators proceed with caution.

As you gain more experience with Linux, you may want to break off other directories into their own partitions on subsequent installations, or when upgrading disk hardware. You can use the `du` command to learn how much space is used by files within any given directory.

Linux Filesystem Options

Linux supports many filesystems. Linux's standard filesystem for most of the 1990s was the *second extended filesystem (ext2 or ext2fs)*, which was the default filesystem for most distributions. Ext2fs supports all the features required by Linux (or by Unix-style OSs in general), and is well tested and robust.

Ext2fs has one major problem, though: If the computer is shut down improperly (because of a power outage, system crash, or the like), it can take several minutes for Linux to verify an ext2fs partition's integrity when the computer reboots. This delay is an annoyance at best, and it is a serious problem on mission-critical systems such as major servers. The solution is implemented in what's known as a *journaling filesystem*. Such a filesystem keeps a record of changes it's about to make in a special journal log file. Therefore, after an unexpected crash, the system can examine the log file to determine what areas of the disk might need to be checked. This design makes for very fast checks after a crash or power failure—a few seconds at most, typically. The four journaling filesystems for Linux are:

- The *third extended filesystem (ext3fs)*, which is derived from ext2fs and is the most popular journaling filesystem for Linux
- *ReiserFS* (`http://www.namesys.com`), which was added as a standard component to the 2.4.1 kernel
- The Extent Filesystem or XFS (`http://linux-xfs.sgi.com/projects/xfs`), which was originally designed for Silicon Graphics' IRIX OS
- The Journaled Filesystem, or JFS (`http://oss.software.ibm.com/developerworks/opensource/jfs`), which IBM developed for its AIX and OS/2

Of these four, XFS and JFS are the most advanced, but ext3fs and ReiserFS are the most stable and popular. A derivative of the current ReiserFS, Reiser4, is under development.

The Linux swap partition doesn't use a filesystem per se. Linux does need to write some basic data structures to this partition in order to use it as swap space (as described in Chapter 8), but this isn't technically a filesystem because no files are stored within it.

Linux also supports many non-Linux filesystems, including:

- The File Allocation Table (FAT) filesystem used by DOS and Windows
- The New Technology Filesystem (NTFS) used by Windows NT/200*x*/XP
- The High-Performance Filesystem (HPFS) used by OS/2
- The Unix Filesystem (UFS; also known as the Fast Filesystem, or FFS) used by various versions of Unix
- The Hierarchical Filesystem (HFS) used by Mac OS
- ISO-9660 and Joliet filesystems used on CD-ROMs

Most of these filesystems are useful mainly in dual-boot configurations—for instance, to share files between Linux and Windows. Some—particularly FAT, ISO-9660, and Joliet—are useful for exchanging files between computers on removable media. As a general rule, these filesystems can't hold critical Linux files because they lack necessary filesystem features. There are exceptions, though—Linux sports extensions for cramming necessary information into FAT and HPFS partitions, UFS was designed for storing Unix filesystem features in the first place, and the Rock Ridge extensions add the necessary support to ISO-9660.

It's usually best to use a journaling filesystem for Linux partitions. As a general rule, any of the current crop of journaling filesystems works well, at least with recent (late 2.4.*x* or later) kernels. The best tested under Linux are ext3fs and ReiserFS. ReiserFS versions of 3.5 and earlier have a 2GB file-size limit, but this limit is raised to 16TB for ReiserFS 3.6 and later. XFS and JFS are both well tested under other OSs, but are not as well tested under Linux. XFS and ext3fs have the widest array of filesystem support tools, such as versions of `dump` and `restore` for creating and restoring backups. XFS supports a flexible security system known as access control lists (ACLs), which are particularly important if your system functions as a Samba server to Windows NT/200*x*/XP clients. (ACL extensions for other journaling filesystems and for ext2fs are available, but they require special configuration.) All Linux distributions support ext2fs out of the box, and most released since 2001 support ReiserFS as well. Support for others is spottier, but increasing. Use non-Linux filesystems for data exchange with non-Linux systems.

Partitioning Tools

In order to create partitions, you use a partitioning tool. Dozens of such tools are available, but only a few are reasonable choices when you're installing a Linux system:

DOS's `FDISK` Microsoft's DOS and Windows ship with a simple partitioning tool known as `FDISK` (for fixed disk). This program is inflexible and uses a crude text-based user interface, but it's readily available and can create partitions that Linux can use. (You'll probably have to modify the partition type codes using Linux tools in order to use DOS-created partitions, though.)

Linux's `fdisk` Linux includes a partitioning tool that's named after the DOS program, but the Linux tool's name is entirely lowercase, whereas the DOS tool's name is usually written in uppercase. Linux's `fdisk` is much more flexible than DOS's `FDISK`, but it also uses a text-based user interface. If you have an existing Linux emergency disk, you can use it to create partitions for Linux before installing the OS.

Linux install-time tools Most Linux installation utilities include partitioning tools. Sometimes the installers simply call `fdisk`, but other times they provide GUI tools that are much easier to use. If you're installing a Linux-only system, using the installer's tools is probably the best course of action.

PowerQuest's PartitionMagic PowerQuest (`http://www.powerquest.com`) makes an unusually flexible partitioning program known as PartitionMagic. This commercial program provides a GUI interface and can create partitions that are prepared with ext2fs, ext3fs, FAT, NTFS, or HPFS filesystems. (HPFS support is missing from the latest versions, though.) This makes it an excellent tool for configuring a disk for a multi-OS computer. PartitionMagic can also resize a partition without damaging its contents. The main program is Windows-based, but the package comes with a DOS version that can run from a floppy, so it's possible to use it on a system without Windows.

GNU Parted GNU Parted (`http://www.gnu.org/software/parted/`) is an open source alternative to PartitionMagic. It can create, resize, and move various partition types, such as FAT, ext2fs, ext3fs, ReiserFS, and Linux swap. GNU Parted runs from Linux and provides a text-only user interface, though, which makes it intimidating and less than ideal for preparing a new disk for Linux installation. Nonetheless, you can prepare a Linux boot disk that runs Parted if you like.

QTParted This program, headquartered at `http://qtparted.sourceforge.net`, provides a GUI front-end to GNU Parted. This GUI control system is similar to the one used by PartitionMagic, but QTParted runs in Linux and supports the filesystems that GNU Parted supports.

FIPS The First Nondestructive Interactive Partition Splitting (FIPS) program comes with many Linux distributions. It's a fairly specialized partitioning tool that splits a single primary FAT partition into two partitions. It's designed to make room for Linux on computers that already have Windows installed—you run FIPS, delete the second (empty) partition that FIPS creates, and create Linux partitions in that empty space.

In theory, partitions created by any tool may be used in any OS, provided the tool uses the standard *x*86 partition table. In practice, though, OSs sometimes object to unusual features of partitions created by certain partitioning tools. Therefore, it's usually best to take one of two approaches to disk partitioning:

- Use a cross-platform partitioning tool like PartitionMagic. Such tools tend to create partitions that are inoffensive to all major OSs.
- Use each OS's partitioning tool to create that OS's partitions.

Chapter 2 includes information on partitioning during installation of a Linux system. Although other distributions' partitioning tools differ somewhat, the basic principles remain the same across distributions. Before you invest too much effort in partitioning, though, you need to study Linux software issues, not the least of which is whether Linux is the proper tool to use.

Linux and Non-Linux Solutions

Why use Linux? This may seem like an odd question to ask in a book on Linux, but it's an important one, and one that you should ask yourself whenever you begin configuring a new Linux computer. If you find that another OS is better suited to a particular task, you can save yourself time and effort in the long run by using that other OS. When Linux is really the best choice, asking yourself why you should use it will give you a ready answer should somebody else ask you the question. By weighing Linux against other OSs, you will also increase your confidence that you're doing the right thing by choosing Linux. This activity may also lead you to think about the computer's purpose in a way that helps you decide precisely how to configure it.

Another important aspect to consider is *which* flavor of Linux to use. Many companies and organizations package Linux. Each of the resulting distributions has its own mix of features. In the end, you can configure any distribution to do what any of the others does, but some distributions are better suited to particular tasks right out of the box. Therefore, it's helpful to know something about the range of what's available before you begin configuring a new Linux system.

Linux vs. Proprietary OSs

Linux competes, to a greater or lesser extent, with many OSs. One of the great divides is between Linux and *proprietary* OSs—operating systems that use a source code base that's closed to public scrutiny. In particular, Linux competes against Microsoft Windows in many peoples' minds. In many respects, though, Linux is more properly pitted against commercial versions of Unix.

Most proprietary OSs are at a cost disadvantage when compared to Linux. Price tags on commercial OSs range from around $50 up to thousands of dollars. Most Linux distributions can be downloaded for free from the Internet, obtained on CD-ROM for less than $10 if you don't need official support or a printed manual, or for $100 or less with these extras. Some extra-deluxe Linux packages cost more than this, often because they come bundled with some costly commercial packages or extended support options.

Linux vs. Microsoft OSs

Microsoft makes two OS product lines: Windows 9*x*/Me and Windows NT/200*x*/XP. Each OS is targeted at a particular market, although they overlap to some extent. Linux competes against both, although more directly against the NT/200*x*/XP line.

Microsoft is now promoting Windows XP as its entry-level and desktop OS. Nonetheless, Windows 9*x*/Me enjoys a huge installed base, and can still be found in stores. Thus, although this line is a technological dead-end, I refer to it in the present tense.

The Windows 9*x*/Me line is intended for home and relatively low-powered business desktop systems. This OS is derived from the old MS-DOS and Windows 3.1 products, and this legacy influences Windows 9*x*/Me, in terms of filesystem support, multitasking power (the ability to

run multiple programs at once), and so on. One of the drawbacks of Windows 9*x*/Me is that it's saddled with the need for backward compatibility with its outmoded predecessors. This fact limits the stability of Windows 9*x*/Me and its suitability for advanced network server functions. Many desktop users, however, aren't particularly bothered by these limits, and they are drawn to the easy-to-use Windows user interface. Although Linux is far more stable than Windows 9*x*/Me, Linux's user interface is slightly less polished than that of Windows. Linux has made great strides in this area since 2000, though, and Linux distributions released since 2002 might arguably be as good as Windows Me in this respect. Overall, Windows 9*x*/Me is best used on existing installations or as an OS for older systems that can't handle Windows 200*x*/XP or a modern Linux distribution.

Microsoft markets Windows NT/200*x* as its competition to Unix and Linux, and Windows XP as the next step up from Windows Me. This OS branch uses a newer kernel with better support for features that are important today, such as filesystem security and multitasking. This makes Windows NT/200*x* much more suited to function as a network server than Windows 9*x*/Me. These improvements also benefit the desktop use of Windows XP in that it's more stable than its predecessors. One of the key differences between Windows NT/200*x*/XP and Linux is that the former is much more tightly tied to its graphical user interface (GUI). This fact makes Windows NT/200*x*/XP easier for new administrators to pick up, but at the same time, it reduces the OS's flexibility. With Linux, on the other hand, you can customize configuration files in ways a GUI doesn't allow, and you can do other things that are difficult or impossible to do using Windows NT/200*x*/XP. The fact that Linux can be configured through text-based tools also means that it's easy to administer remotely, using nothing more than common remote login tools such as Secure Shell (SSH), which allows you to run text-based Linux programs from another computer.

Although Microsoft no longer markets it, OS/2 is another OS in the same family as DOS and Windows. This OS is still sold by IBM and, under the name eComStation, by a company called Serenity Systems. OS/2 has, however, been sliding in market share since the mid-1990s. In many respects, OS/2 is similar to Windows NT—it uses its own updated kernel that abandons much of the DOS baggage still carried by Windows 9*x*/Me. OS/2 lacks compatibility with today's popular Windows software, however. OS/2 may still be worth considering in environments where it's still heavily used, such as many banks, but its declining market share argues against its adoption in new environments without some compelling cause.

One of the strongest arguments in favor of any Microsoft or Microsoft-related OS is the application base for desktop use. In particular, the Microsoft Office application group is extraordinarily popular, and it is not available for Linux (or OS/2, for that matter). Although some Linux programs can import and export Office files, these operations are necessarily imperfect. Likewise, emulators like Windows Emulator/WINE Is Not an Emulator (WINE), VMware, and Win4Lin enable you to run Windows software under Linux, but if the primary reason to use a computer is to use Windows software, there's seldom an advantage to running that software in an emulator under Linux, as opposed to running it directly.

Likewise, the principal advantage for Linux in a comparison with Windows is its software base. Linux supports a wide range of open source (and therefore generally low-cost) applications. Because Linux is modeled after Unix, administrators with Unix experience should have no trouble handling a Linux system—and a person who learns Linux can pick up other flavors of Unix quite quickly.

Linux vs. Unix

Originally, Linux was developed as a clone of Unix. Linus Torvalds set out to write an open source kernel around which existing open source Unix replacement parts could converge; the result was the Linux distribution as we know it today. Because of this history, modern Linux systems bear a very strong resemblance to modern Unix systems, and in fact, the two can often be used similarly.

In many cases, the most important difference between Linux and a commercial Unix is cost. As noted earlier in this section, Linux is a very low-cost OS, but commercial Unix OSs cost much more. (The most common *x*86 Unix OSs are now available at low cost for personal use, but most commercial users must still pay hundreds of dollars for a license.) Most Linux software is available in open source form, and it can be compiled on commercial Unix machines—indeed, most Linux software is developed as Unix software generically, with Linux as just one of many Unix-like platforms on which it works. Many commercial Unix programs have been ported to (that is, recompiled on) Linux.

Where commercial Unix OSs hold an edge is in very high-performance computing. OSs such as Silicon Graphics' (SGI) IRIX and Sun's Solaris run on very fast non-*x*86 hardware and support advanced features that Linux supports poorly, if at all. Also, the hardware used by high-end systems is often superior to that used on the *x*86 PCs on which Linux usually runs. Linux has been ported to many non-*x*86 platforms, including many of those on which its Unix "big brothers" run, but these ports often lag behind the *x*86 version in terms of overall polish and general usability.

As Linux improves, the gap between Linux and commercial versions of Unix is shrinking. Even today, Linux is an excellent platform for workstations and small to mid-size servers. Because of its similarity to more advanced systems, it's possible to deploy Linux today and move to a higher-end commercial Unix system in the future, with minimal changes to configuration and administrators' training. This is certainly an advantage of the Linux/Unix family as a whole over Windows.

Individual Linux systems can be linked using high-speed network connections and function as a huge supercomputer. In fact, some of the fastest supercomputers today are Linux clusters. Thus, Linux spans the range of computing devices from tiny embedded systems to fast supercomputers. Nonetheless, commercial Unix systems continue to be the best choice for some of the highest-powered individual computers.

Linux vs. Other Open Source OSs

Linux is not the only open source OS in existence. Most competing open source OSs are, like Linux, clones of Unix. In fact, the main competing family—FreeBSD, NetBSD, and OpenBSD—is derived directly from mainstream Unix.

BSD stands for *Berkeley Standard Distribution*. BSD grew as a component-by-component open source replacement for AT&T's original Unix. During and after this process, the development forked several times, producing several variant products, including the three major open source BSDs. Today, the term BSD refers either to an entire OS that shares the Berkeley heritage or to specific OS components that are so derived, such as the BSD printing system described in Chapter 8.

In most situations, one of the open source BSDs will function as well as Linux. These OSs have more-or-less the same base of applications, and they're administered in largely the same ways. The main differences between Linux and its BSD cousins come down to three factors:

Kernel licenses The Linux kernel is released under the General Public License (GPL), which tends to encourage greater public participation in the development of the kernel than does the BSD license used for the BSDs. The culture that's emerged around each OS has furthered this distinction. The end result is that the Linux kernel has developed more quickly than the BSD kernels, and Linux has more support for more hardware, more filesystems, and so on.

Commercial software availability Commercial software developers seem to be more willing to port software to Linux than to the open source BSDs. The BSDs include ways to run Linux programs, and most software for both systems is open source and available on both. Therefore, this factor isn't critical for most people, but it is, nonetheless, a plus in the Linux column.

Support network The Linux support network, as embodied in the Linux newsgroups, Web sites, and support from commercial Linux vendors, is generally more active than is the support network for the BSDs. This helps new Linux users get started with Linux, and it helps even experienced administrators resolve problems. Some support forums, though, are OS-neutral, such as mailing lists and newsgroups devoted to programs that run on both platforms.

These three factors produce a positive feedback cycle—more users creates a better volunteer support network, more potential kernel developers, and more incentive for commercial software vendors to port their products to Linux. Each of these factors in turn creates an environment that will attract even more users.

In the end, you're probably best off using whichever Unix-like open source OS is most familiar to you. For new users, the broader support network for Linux can be a major point in its favor. If you can access personal support for a BSD more easily, though, or if a BSD version supports hardware that you need that's not supported in Linux, then a BSD may be a better choice for you. In either case, moving from Linux to a BSD or vice versa is fairly straightforward, so the time you spend learning one system is not wasted should you decide to change.

A Rundown of Linux Distributions

Within the Linux world, several *distributions* exist. A distribution is a compilation of a Linux kernel, startup scripts, configuration files, and critical support software. Distributions also

include some type of installation routine so that you can get a working Linux system. Any two distributions may use different versions of any or all of these components, which will produce distinctly different feels. Critical components, though, such as the kernel and certain support software, come from the same line in all distributions. For instance, one distribution might use the 2.4.21 Linux kernel and another might ship with 2.4.22, but they're both Linux kernels.

One important distinguishing characteristic of Linux distributions is which packaging methods they use. RPM Package Manager (RPM), Debian packages, and tarballs are the three most common package formats. The details of using these three package formats are covered in Chapter 3.

Depending on your definition of "major," there are anywhere from two or three to over a dozen or more major Linux distributions. In addition, less popular and specialized distributions are available. Many Linux distributions are derived from either Debian or Red Hat. Some common Linux distributions include the following:

Conectiva Linux This distribution is one of three current distributions based on UnitedLinux. It's targeted at users in South and Central America, and is limited to running on *x*86 systems. You can learn more at `http://www.conectiva.com`.

Debian GNU/Linux This distribution, headquartered at `http://www.debian.org`, is built by a non-profit organization, rather than by a for-profit company, as are most other distributions. Debian eschews many of the GUI configuration tools used by most other distributions, and instead it aims to be a very stable and flexible distribution. For these reasons, it's well liked by open source hard-liners and those who like tinkering with the underlying text-based configuration files. Because it favors stability, Debian has a long release cycle and may not ship with the latest versions of many of its components. Debian is available on a very wide array of CPUs, including *x*86, IA-64, PowerPC, Alpha, SPARC, and 680*x*0.

Gentoo Linux Most distributions ship as collections of precompiled binary packages. To be sure, source code is available, but most distributions don't provide any simple means to recompile the entire distribution. Gentoo Linux is the exception to this rule. Although precompiled versions for *x*86, PowerPC, and SPARC are available, much of the benefit of this distribution is that it supports recompiling everything with optimizations to suit your own hardware. (This feature is similar to the BSD ports system.) In theory, this ability should make a properly recompiled Gentoo faster than competing distributions. In practice, the effect is small, and the time spent recompiling everything can measure in the days. Like Debian, Gentoo is a noncommercial distribution. You can learn more about Gentoo at `http://www.gentoo.org`.

Libranet GNU/Linux Debian has spawned a number of derivative distributions, and this is one of them. Libranet adds improved GUI system administration tools, but keeps many of Debian's core components and system administration defaults. Thus, you can easily install most Debian packages in Libranet. Libranet doesn't make its latest version available for free download; you must buy a CD-ROM or pay for a download. This distribution is headquartered at `http://www.libranet.com` and is available only for *x*86 CPUs.

Lindows This distribution, which is a Debian derivative, lies at the fringes of Linux. It's designed as a replacement for Windows (hence the name) on the desktop. The original Lindows plan was to make heavy use of WINE to enable the system to run Windows programs more-or-less seamlessly. This emphasis has been toned down, however, because Windows emulation is a very difficult task. Lindows is now included on some cut-rate retail PCs. Free downloads of Lindows are not available. You can learn more at `http://www.lindows.com`.

Lycoris Like Lindows, Lycoris aims to be Linux for the desktop. Lycoris has never emphasized Windows compatibility, though, and it's an RPM-based distribution. The latest version is available only on CD-ROM from the company or preinstalled, although earlier versions can be downloaded from the Internet. The Lycoris home page is `http://www.lycoris.com`.

Mandrake Linux This distribution is a French-based offshoot of Red Hat Linux. Originally developed as a Red Hat with integrated K Desktop Environment (KDE), Mandrake has since developed more of its own personality, which includes a good GUI installer and some unusual choices in standard server software, such as Postfix rather than the more popular sendmail for a mail server. Its English Web page is `http://www.linux-mandrake.com/en/`. Mandrake is available for *x*86, IA-64, SPARC, Alpha, and PowerPC CPUs.

Red Hat Linux Red Hat (`http://www.redhat.com`) is one of the oldest major distributions today, and one of the most influential. Red Hat developed the RPM format that's used by many other distributions, including some that aren't otherwise based on Red Hat. The distribution includes GUI installation and configuration tools that are unusually complete. Red Hat is or has been available on *x*86, IA-64, SPARC, and Alpha CPUs, although the company has ceased SPARC development with version 6.2 and Alpha with 7.2. In late 2003, Red Hat split its distribution into Fedora Linux, which is freely available and developed by the community, and Red Hat Enterprise, which is an expensive product aimed at large businesses.

Slackware Linux Slackware is the oldest of the surviving Linux distributions. Like Debian, Slackware favors manual text-based configuration over GUI configuration tools, so it's often recommended for those who want the "Unix experience" without GUI "crutches." Slackware is the only major distribution to rely on tarballs for package management. You can read more at `http://www.slackware.com`. This distribution is available for *x*86, Alpha, and SPARC CPUs.

SUSE Linux The German company SUSE (`http://www.suse.com`) produces a distribution that's particularly popular in Europe. SUSE uses RPMs, but it's not otherwise based on Red Hat. SUSE is one of the UnitedLinux members. Some SUSE packages use a DVD-ROM for software distribution, which is very helpful if your system has a DVD-ROM drive—SUSE ships with an unusually large number of packages, so juggling the half-dozen CD-ROMs can be awkward, compared to using a single higher-capacity DVD-ROM. This distribution includes GUI installation and configuration tools. Versions of SUSE for *x*86, IA-64, PPC, and Alpha are all available. Like Libranet, SUSE doesn't offer free CD-ROM images for download, although you can download the installation boot disk images and install from the network.

TurboLinux This distribution (`http://www.turbolinux.com`) began as a Red Hat derivative, but recent versions are based on UnitedLinux. This distribution includes unusually strong support for Asian languages, and is targeted at the server market. TurboLinux is available for *x*86 and AMD64 CPUs.

UnitedLinux This is a sort of meta-distribution—a set of core packages and configuration standards created by a consortium of four Linux distributors: Conectiva, the Santa Cruz Organization (SCO; formerly Caldera), SUSE, and TurboLinux. Each produces or did produce a distribution based on the UnitedLinux core. The idea was to create Linux standards strong enough to compete with Red Hat's growing dominance of the Linux market. Since the start of the UnitedLinux effort, SCO has dropped its Linux distribution and fallen out of favor in the Linux community because of lawsuits it's filed against IBM and allegations it's made of improper use of Unix source code in Linux. The remaining three UnitedLinux members continue to produce Linux distributions, though. You can learn more about UnitedLinux at `http://www.unitedlinux.com`.

Xandros Linux Xandros (`http://www.xandros.com`) picked up an earlier and discontinued distribution from Corel, which based its distribution on Debian GNU/Linux. Xandros Linux adds a very user-friendly installation routine and GUI configuration tools. In implementing these features, though, Xandros has become less easily configured through traditional Linux command-line methods. This distribution is targeted at new Linux users who want to use the OS as a desktop OS to replace Windows. Xandros is an *x*86-only distribution.

Yellow Dog Linux This distribution is available exclusively for PPC systems, but is based on Red Hat. Yellow Dog (`http://www.yellowdoglinux.com`) uses its own unique installer, but once set up, it is quite similar to Red Hat.

When deciding on a Linux distribution, you'll find that some of these will fall out of the running for very basic reasons. For instance, there's no point in considering Yellow Dog for an *x*86 system, or Xandros for an Alpha CPU. The RPM and Debian package management systems are, on the whole, quite similar in overall features and capabilities, so if you're not already familiar with either, there's little reason to favor one over the other. (Chapter 3 covers both systems in more detail.) Any of these distributions can be configured to do anything that another can do, with the exception of running on an unsupported CPU.

As a practical matter, you *do* need to decide between distributions. As a general rule, Lycoris, Mandrake, SUSE, and Xandros are probably the best suited as delivered to function as workstations, particularly for new Linux users. Debian and SUSE both ship with an unusually wide array of software (for SUSE, this is particularly true of the Professional package, which ships with a DVD-ROM and half a dozen CD-ROMs). Red Hat is unusually popular, so finding support for it on newsgroups and the like is particularly easy. TurboLinux is specifically marketed for the server market, but others can fill that role just as easily. Some distributions come in variants that include additional software, such as secure servers, third-party partition managers, and so on.

If you have a fast Internet connection and a CD-R drive, and you want to experiment with several Linux distributions, check out the Linux ISO Web site at `http://www.linuxiso.org`. This site includes links to CD-R image files for most Linux distributions. You can also obtain distributions on no-frills CD-ROMs (with no manual and no support) for less than $10 from the likes of CheapBytes (`http://www.cheapbytes.com`) or Linux Buy (`http://www.linuxbuy.com`). Official boxed sets typically cost $20 to $100, or occasionally more for the most feature-packed versions. The boxed sets generally include printed manuals, support, and occasionally, a commercial software product or two.

Determining Software Needs

When you plan a Linux installation, it's important that you know what software you'll need on the system. For each program class, you'll have to decide what particular package you want to run. For instance, if you want to configure a word processing workstation, you must decide if you want to use OpenOffice.org, KWord, AbiWord, LyX, or something else. Most of these packages come with most distributions of Linux, but sometimes you must obtain software from another source. In the case of downloadable software, if it doesn't accompany the distribution you use, you may want to download it before installing Linux. Depending on your available hardware, you can usually put a package on floppy disk, a high-capacity removable disk (like a Zip or LS-120 disk), or a CD-R to have it ready for installation once you've installed the main distribution. Doing this from Windows works just fine, if this is your first Linux installation.

Common Workstation Programs

Workstations don't usually need much in the way of server software. Workstations may include such software to provide local services, though—for instance, Linux workstations usually include mail servers to handle mail for the administrator that is generated by automatic scripts and the like. The most important workstation programs are designed to help an individual get work done. Such software includes the X Window System, office tools, network clients, audio/visual programs, personal productivity tools, and scientific tools.

The X Window System

The X Window System (or X for short) is Linux's GUI environment. It's usually implemented through the XFree86 package. Although Linux can be used without this GUI, most workstation users expect a GUI environment, and an increasing number of workstation programs require X in order to function.

X itself is a fairly spare environment, so it's frequently supplemented by additional tools, such as *window managers* (which provide borders and controls around windows) and *desktop environments* (which include a window manager and an assortment of utility programs to help make for a comfortable working environment). In particular, the K Desktop Environment (KDE; `http://www.kde.org`) and the GNU Network Object Model Environment (GNOME; `http://www.gnome.org`) are two popular desktop environments for Linux. Most Linux distributions ship with both, but some install one or the other by default. Red Hat, for instance, favors GNOME, whereas SUSE favors KDE.

Office Tools

Office tools are the workhorses of computer use in offices; they are primarily made up of word processors, spreadsheets, and databases, but they may also contain various other applications, such as personal contact managers, calendar programs, and so on. Sun's (`http://www.sun.com`) StarOffice and its open source twin, OpenOffice.org (`http://www.openoffice.org`), are available in both Linux and Windows, and so they can be good choices in a mixed Linux/Windows environment.

Corel used to make WordPerfect available for Linux, but it's been discontinued and is hard to find; however, if you need to exchange WordPerfect documents with others, it's worth tracking down a copy. You're more likely to have luck with WordPerfect 8. Although it requires old libc5 libraries, WordPerfect 8 is easier to install and use on modern Linux distributions than the more recent WordPerfect Office 2000, which relies on an obsolete version of WINE that's almost impossible to get working on modern distributions.

All of these products also include import/export filters for Microsoft Office documents, but as noted earlier, this approach is imperfect at best. (StarOffice is generally considered to have the best of these filters.) Both the GNOME (`http://www.gnome.org`) and KDE (`http://www.kde.org`) projects are building open source office suites.

Various singleton packages are also available. For instance, LyX (`http://www.lyx.org`), KWord (part of KDE), and AbiWord (`http://www.abisource.com`) are three popular What You See Is What You Get (WYSIWYG) Linux word processors. There are also markup languages like TeX and LaTeX (`http://www.latex-project.org`) that, in conjunction with editors like Emacs, can do much the same job. Gnumeric (`http://www.gnome.org/projects/gnumeric`) is a popular Linux spreadsheet. Ximian (`http://www.ximian.com`) produces an integrated mail reader/address book/calendar program called Evolution. Some of these tools are being integrated as part of the GNOME Office suite.

Network Clients

Users run network client programs to access network resources. Examples include Web browsers like Netscape (`http://www.netscape.com`), its open source twin Mozilla (`http://www.mozilla.org`), and Opera (`http://www.opera.com`); mail readers like Mutt (`http://www.mutt.org`) and KMail (part of KDE); and FTP clients like gFTP (`http://gftp.seul.org`). All major Linux distributions ship with a wide variety of network clients, but if you need a specific program, you should check whether it's included in your distribution. If it's not, track it down and install it. Most Linux network clients are open source, but there are a few that aren't. Opera stands out in this respect.

For more information on network clients, please refer to Chapter 5, "Networking."

Audio/Visual Programs

Audio/Visual programs cover quite a wide range of products. Examples include graphics viewers and editors like XV (`http://www.trilon.com/xv/`) and the GIMP (`http://www.gimp.org`); ray tracing programs like POV-Ray (`http://www.povray.org`); MP3 players like the X Multimedia System (XMMS; `http://www.xmms.org`); multimedia players like XAnim (`http://smurfland.cit.buffalo.edu/xanim/`); audio/video editors like Cinelerra (`http://heroines.sourceforge.net/cinelerra.php3`) and Linux Video Studio

(http://ronald.bitfreak.net); digital video recorder (DVR) software like MythTV (http://www.mythtv.org); and games like FreeCiv (http://www.freeciv.org) and Tux Racer (http://tuxracer.sourceforge.net). Some audio/visual programs are serious tools for work and are on a par with office utilities for some users. Somebody whose work involves graphics design, for instance, may need tools like the GIMP or POV-Ray. Other audio/visual programs fall more in the realm of entertainment, like games.

Linux's support for audio/visual programs has traditionally been weak. This has changed substantially since the mid-1990s, however, with the development of powerful programs like the GIMP and increasingly sophisticated multimedia players and editors. Even Linux games have come a long way, thanks in part to companies that specialize in porting other companies' games to Linux.

Personal Productivity Tools

Personal productivity tools are programs that individuals use to better their own lives. Examples include personal finance programs like GnuCash (http://www.gnucash.org) and slimmer versions of office programs (word processors for writing letters, for instance). As with audio/visual programs, personal productivity applications have traditionally been lacking in Linux, but that situation is improving. GnuCash, in particular, fills a niche that many users find important for personal use of Linux.

Personal productivity tools need not be restricted to the home, however. For instance, although big word processors like StarOffice and WordPerfect are very useful in some situations, many office users don't need anything nearly so powerful. Slimmer tools like Maxwell (http://sourceforge.net/projects/maxwellwp) suit some users' needs just fine. By foregoing the resource requirements of a larger package, a company may find that using such programs can help save it money by allowing its employees to use less powerful computers than might otherwise be required.

Scientific Programs

Unix systems have long been used in scientific research, and Linux has inherited a wealth of specialized and general scientific tools. These include data-plotting programs such as the GNU plotutils package (http://www.gnu.org/software/plotutils/plotutils.html) and SciGraphica (http://scigraphica.sourceforge.net), data processing programs like Stata (http://www.stata.com), and many very specialized programs written for specific studies or purposes. Linux's software development tools (described shortly, in the section titled "Programming Tools") let you or your users write scientific programs, or compile those written by others.

Common Server Programs

A *server program* is one that provides some sort of service, usually to other systems via a network connection. Typically, a server runs in the background, unnoticed by the computer's users. In fact, many computers that run server programs don't have ordinary login users; instead, the system's users are located at other systems, and they use the computer only for its

servers. A Web server computer, for instance, may not have any local users aside from those who maintain the computer and its Web pages. Other servers include mail servers, remote login servers, file access servers, and miscellaneous servers.

The term *server* is sometimes applied to an entire computer, as in "the Web server needs a bigger hard disk." Context is usually sufficient to distinguish this use from the use of the term in reference to a specific software product.

Web Servers

One very popular use of Linux is as a platform for running a Web server. This software uses the *Hypertext Transfer Protocol (HTTP)* to deliver files to users who request them with a Web client program, more commonly known as a Web browser. The most popular Web server for Linux by far is Apache (`http://httpd.apache.org`), which is an open source program included with Linux. Other Linux Web servers are available, however, including Zeus (`http://www.zeus.com`), Roxen (`http://www.roxen.com/products/webserver`), and thttpd (`http://www.acme.com/software/thttpd`). Zeus is a high-powered commercial Web server, Roxen is a high-powered open source Web server, and thttpd is a minimalist open source program suitable for small Web sites or those that don't need advanced features.

Some Linux distributions install Web servers even on workstations because the distributions use the Web servers to deliver help files to the local users. Such a configuration chews up resources, though, and can at least potentially be a security problem.

Mail Servers

Mail servers handle e-mail delivery. All major Linux distributions ship with a mail server, such as sendmail (`http://www.sendmail.org`), Exim (`http://www.exim.org`), or Postfix (`http://www.postfix.org`). These servers all handle the Simple Mail Transfer Protocol (SMTP), which is used to deliver mail between mail servers on the Internet at large, and can also be used as part of a local network's e-mail system. All major Linux distributions also ship with Post Office Protocol (POP) and Internet Message Access Protocol (IMAP) servers. These are used to deliver mail to end-user mail reader programs, which typically reside off the mail server. Most Linux SMTP, POP, and IMAP servers are open source, although commercial servers are available as well.

Disabling the SMTP server on a system that doesn't function as a mail server may seem like a good idea, but many Linux systems rely on this functionality to deliver important system status reports to the system administrator. Because of this, it's generally best to ensure that the mail server is configured in a secure way, which it normally is by default, and leave it running.

Remote Login Servers

A remote login server allows a user to log into the computer from a remote location. The traditional remote login protocol is Telnet, which is handled by a server called `telnetd` or `in.telnetd` in Linux. This server is open source and comes with all Linux distributions, although it's not always active by default.

Unfortunately, Telnet is an insecure protocol. Data passing between the Telnet client and server can be intercepted at points in-between the two, leading to compromised data. For this reason, it's best to disable the Telnet server on any Linux system and instead use a more secure protocol. Secure Telnet variants are available, but an alternative protocol, known as the *Secure Shell (SSH),* is more popular. SSH encrypts all data passing between two systems, making intercepted data useless. The most popular SSH implementation for Linux is the open source OpenSSH (`http://www.openssh.com`).

Telnet and SSH are basically text-based tools. SSH can be configured to tunnel X sessions through its connections, however. With this configuration, you can run X programs remotely. You can do the same by setting various parameters from a Telnet login, as described in Chapter 5. More direct GUI remote login tools (the X Display Manager [XDM], GNOME Display Manager [GDM], and K Display Manager [KDM]) are also available and come with all major distributions. Finally, the VNC package (`http://www.realvnc.com`) allows direct remote X logins as well. With the exception of VNC, these tools come with all major Linux distributions.

File Access Servers

A file access server lets users read, write, and otherwise manipulate files and directories from a remote location. The traditional remote access protocol is the File Transfer Protocol (FTP), which is still in common use. Many local networks use *file-sharing protocols*, which allow programs on one computer to treat files on another system as if those files were local. Sun's Network Filesystem (NFS) is used for file sharing between Linux or Unix systems; the *Server Message Block (SMB)*, also known as the *Common Internet Filesystem (CIFS)*, is used to share files with DOS, Windows, and OS/2 systems; Novell's IPX/SPX (most strongly associated with the NetWare OS) is another PC file sharing protocol; and Apple's AppleShare is the protocol used for Macintosh file sharing. Linux supports all of these protocols—NFS with standard kernel tools and various NFS servers; SMB/CIFS with the Samba package; IPX/SPX with the `mars_nwe` and `lwared` packages; and AppleShare through Netatalk (`http://rsug.itd.umich.edu/software/netatalk.html`).

Most of these file-sharing servers have printer-sharing features as well, so you can provide network access to printers connected to Linux. NFS is an exception to this rule, but NFS's lack of printer sharing is offset by the fact that Linux's standard printing tools include this feature themselves.

Because of its excellent support for so many different file-sharing protocols, Linux makes an outstanding file- and printer-sharing platform in a cross-platform office. In an office that supports Windows, Mac OS, OS/2, and Unix or Linux desktop systems, for instance, a single Linux computer can provide file- and printer-sharing services for all of these OSs, allowing users to move freely from one client platform to another or to collaborate with users of other platforms.

Miscellaneous Servers

The preceding sections cover many of the most popular server types, but it's far from complete. Many servers fall into less-used categories or simply defy categorization. Examples include:

- Proxy servers, such as Squid (`http://www.squid-cache.org`), which improve network performance or security by buffering Internet access attempts

- Dynamic Host Configuration Protocol (DHCP) servers, which keep track of network configurations and help automate the configuration of DHCP client systems
- Domain Name System (DNS) servers, such as BIND (also known as `named`), which convert between numeric IP addresses and hostname
- Remote configuration tools like Webmin (`http://www.webmin.com`), which enable you to change a system's configuration from another computer

Most Linux distributions ship with a wide range of such servers, some of which are active by default and some of which aren't.

Although not a server per se, the `ipchains` and `iptables` tools are extremely useful when configuring a system as a firewall, or in protecting an individual workstation with firewall-like rules. These programs can block access to your system based on IP addresses or network ports (numbers associated with specific servers or runs of client programs). The `ipchains` tool fills this role with the 2.2.*x* kernel series, while `iptables` works with the 2.4.*x* and later kernels.

Useful Software on Any System

Whether a computer is to be used as a workstation or a server, certain classes of programs are extremely useful. These programs help users handle common user tasks and help administrators administer a system. Libraries are particularly important because they're the foundation on which most other programs are built.

Text Editors

A *text editor*, as you might imagine, is a program used to edit text. Most system administrators need to be familiar with Vi, which is a small and ubiquitous Unix and Linux text editor. (Chapter 7 includes an overview of Vi operation.) If you need to do emergency maintenance, there's a good chance your emergency tools will include Vi as the text editor, or a close relative, such as Vi Improved (VIM). A couple of other small text editors are jed and pico. These tools are designed to be similar to the popular Emacs program, which is an extremely large and flexible text editor.

Vi, jed, pico, and Emacs are all text-based programs, although some of them have at least some X extensions. In particular, XEmacs (`http://www.xemacs.org`) is an X-enhanced version of Emacs. Other text editors, such as Nedit (`http://www.nedit.org`), gEdit (part of GNOME), and KEdit (part of KDE), are designed from the ground up as GUI text editors. Although you may prefer to use one of these in day-to-day operation, you *will* occasionally need to use a text-based editor, so you should familiarize yourself with at least one of them, as well.

Programming Tools

Programming tools enable you to write programs for Linux. These tools can also be useful in getting Linux software to run or for other purposes—some programs are distributed in a form that requires you to have programming tools available. Therefore, installing certain key programming tools on a Linux system is often necessary even if you don't know a thing about programming.

A *compiler* is a tool for converting a program's source code (its human-readable form, written by a programmer) into binary form (the machine-readable form, which users run). All major Linux distributions ship with a wide array of compilers, the most important of these being the GNU Compiler Collection (GCC). The Linux kernel is written mostly in C, as are many Linux programs, and GCC is best known for its C compiler. Some installations require other programming languages. If your users will be doing programming, ask them what tools they'll need. You'll need to install GCC, at a minimum, to compile most programs distributed as source code into a useable form. Most programming languages are available with major Linux distributions, and the rest can be found in open source and, occasionally, commercial forms.

Some programming languages aren't compiled; they're interpreted. In an interpreted language, the computer translates from human-readable form to machine code on the fly. This reduces execution speed, but it can speed development since there's no need to explicitly compile the software. Many interpreted languages are known as *scripting languages*, because they're used to create simple programs known as scripts. Java, Python, and Perl are popular interpreted languages. Some Linux or cross-platform programs are distributed in these forms, so installing them (particularly Perl and Python) may be necessary on many systems.

Many developers like to work with an integrated development environment (IDE). IDEs provide GUI front-ends to editors, compilers, linkers, debugging utilities, and other programming tools. Some software companies make money selling IDEs for Linux development, such as Metrowerks CodeWarrior (`http://www.metrowerks.com`). Other IDEs are open source projects, such as Code Crusader (`http://www.newplanetsoftware.com/jcc/`) and KDevelop (`http://www.kdevelop.org`). Chances are you won't need to install an IDE just to use software distributed in source code form; IDEs are most useful for active program development efforts.

It's generally unwise to leave programming tools on a server system. If the system is ever compromised by crackers (those who break into computer systems), the programming tools can be turned against you to compile the cracker's own utilities. Nonetheless, compilers are useful in administering servers. Typically, you'll compile software on a system that's configured much like the server, and then you'll transfer the compiled software to the server system.

Libraries

A *library* isn't a program per se; rather, it's a collection of software routines that may be used by programs. Placing commonly used code in libraries saves both disk space and RAM. All Linux systems rely on a library known as the *C library (libc)* because it provides routines that are necessary for any C program to run in Linux. (The version of libc shipped with major distributions today is known as *glibc*.) Any but the most trivial Linux system will use a number of additional libraries as well. You must ensure that you install the appropriate libraries. If you fail to do so, your package system will probably tell you about the problem, expressed as a *failed dependency* (dependencies are described in more detail in Chapter 3).

Validating Software Requirements

Computer software is highly interdependent. Programs rely on others, which in turn rely on still others. This cycle ultimately leads to the Linux kernel—the "heart" of a Linux system. Even the kernel relies on other software—namely, the BIOS, which the kernel needs to start up, as described in Chapter 3. This web of dependencies and requirements sometimes poses a problem because you may need to install a dozen new programs in order to install a single package you want to use.

If a program comes with your Linux distribution, that program will most likely work well with that distribution. In some cases, you may need to install additional packages. Most distributions use package-management systems that support dependency checking, as described in Chapter 3, so you'll be told what files or packages you're missing when you try to install a new program.

For programs that don't ship with a distribution—and even for those that do—you can usually find a list of requirements on the program's Web site or in its documentation. This requirement list may include several components:

Supported OSs Most Linux software works on many Unix-like OSs. It's usually best to check that a package explicitly supports Linux. This is particularly true of binary-only packages, such as those that are common in the commercial world. A binary package for IRIX won't do you any good in Linux, for instance. Unix programs that come with source code can often be compiled without trouble on Linux, but the larger the program, the more likely you'll run into a snag if the author doesn't explicitly support Linux.

Supported distributions Some packages' documentation refers to specific Linux distributions. As a general rule, what works on one distribution can be made to work on another. Sometimes the conversion process is trivial, but sometimes you'll need to wade through a tangled mess of unfulfilled dependencies to get a program working on a distribution its author doesn't explicitly support.

CPU requirements Software that comes in source code form can usually be compiled on any type of CPU. Binary-only programs, though, usually work only on one CPU family, such as *x*86 or PowerPC. This problem afflicts many commercial packages. Even some programs that come with source code don't compile properly on all CPUs, although this problem is rare.

Library requirements The vast majority of programs rely on specific libraries, such as libc and GTK+. Check the requirements list and try to determine if the libraries are installed in your system. If your distribution uses the RPM or Debian package system, you can usually check for a library of the specified name. Chapter 3 discusses software management, including RPM and Debian package utilities.

Development tools and libraries If you intend to compile a program yourself, pay attention to any development tools or libraries the package uses. For instance, if a program is written in C++, you'll need a C++ compiler. Also, many libraries have matching development libraries. These include additional files needed to compile programs that use the libraries but that aren't needed merely to run such programs once compiled.

If your system seems to meet all the requirements specified by the program's author, try installing the package according to the provided instructions. If you have trouble, read any error messages you get when you try to install or run the program; these often contain clues. You may also want to check Chapter 3 for information on Linux packages, and Chapter 9, "Troubleshooting," for package-installation troubleshooting tips.

Understanding Software Licenses

Most computer software is copyrighted. This status gives the copyright owner the legal right to restrict distribution of the software, and even (with the addition of a license) to limit how it may be used. In the Linux world, *open source* licenses dominate the landscape. These licenses give users unusually broad rights. Commercial software for Linux is also available, though, and some products fall somewhere in-between the two.

One of the problems with Linux software licensing is that the culture surrounding open source is often perceived as hostile toward commercial software and even commercial users of software. Although this perception is often overblown, it's important that you be aware of it and how you may and may not use software with the various licenses in Linux. Understanding these issues can help you arrive at a decision regarding the correct software licenses for your environment, and it may help you overcome any misconceptions you may encounter among your co-workers.

Open Source Software Licenses

Open source software and its culture have evolved substantially over time. One early influence was the Free Software Foundation (FSF) and its GNU's Not Unix (GNU) project, which developed many critically important components that are now used in Linux, such as GCC. The FSF developed the *General Public License (GPL)*, which Linus Torvalds used for the Linux kernel. The FSF is a leader in what's known as the free software movement, which advocates software freedom—the ability of users to distribute and modify program source code.

By the mid-1990s, the free software community began to see the need for some changes. For one thing, some felt that the term "free software" was inappropriate because it tended to deter development by for-profit companies. In fact, the primary meaning of *free* in free software refers to the freedom to do what one likes with the code; it does not mean that it is a zero-cost distribution policy. In 1997, a collection of movers in the free software community (notably lacking Richard Stallman, the founder of the FSF) created a formal definition of what they termed open source software. This definition includes 10 components, summarized here:

1. Free redistribution The user must have the right to redistribute the software without paying royalties.

2. Source availability The program must have source code, either as part of the main package or readily available via the Internet.

3. Derived works The license must allow the user to modify the source code and distribute these modifications under the same terms as the original.

4. Source code integrity An open source license may restrict distribution of modified code, but only if patch files (files used to modify, or "patch," original source code) are permitted. This condition is essentially a weakening of point #3 in order to let the software's original author control a primary package but still allow third-party modifications.

5. No discrimination against persons or groups The license may not discriminate against any person or group.

6. No discrimination against fields of endeavor The license may not restrict rights based on fields of endeavor (such as business use or genetic research).

7. License distribution The license terms must apply automatically, without requiring you to sign a form or the like.

8. Not specific to a product The license terms may not be contingent on the program being part of another product.

9. No contamination The license must not "contaminate" other software by placing restrictions on other software distributed with the product.

10. Technology neutrality The license must not be tied to any specific technology or interface style. For instance, the license can't prevent a user from modifying a compiler for Intel *x*86 CPUs to work with AMD's AMD64 platform.

Today, the Linux community as a whole has embraced the open source definition, although some influential individuals and groups still prefer to use other terms. In particular, the FSF continues to use the term "free software" for software distributed under the GPL and some other licenses. The FSF's GPL includes language stipulating that any changed version of a program must be distributed under the GPL. Such a requirement is certainly allowed by the open source definition, but it's not required by this definition. In fact, some open source licenses allow a person to modify the source code and distribute it under another license. (Note that the open source requirement #3 allows, but does not require, modifications to be distributed under the original license.)

Open source software, and particularly software distributed under the GPL, is frequently referred to as falling under *copyleft*. This play on the word "copyright" suggests a use of copyright laws to achieve goals that are in many ways the opposite of copyright—to ensure the free availability of software, rather than to limit the right to copy.

Every open source license has its own unique characteristics. These are mostly of interest to developers who might want to contribute to a software project, but on occasion they may be important to a system administrator. The major open source licenses include the following:

GNU GPL and LGPL The GNU GPL is the license used by the Linux kernel. As noted earlier, it contains language that requires modifications to be made available under the GNU GPL. This

ensures that there will never be a proprietary version of the Linux kernel, which may be a good or bad thing, depending on your point of view. A variant on this license is the Lesser GPL (LGPL), formerly known as the Library GPL. This is intended to be applied to libraries. The LGPL explicitly allows software that uses LGPLed code to do so even if the software does not follow the LGPL or GPL. This loophole is important for libraries; a library licensed with the GPL would require all programs that use the library to also use the GPL.

BSD The BSD license is used by the open source BSD OSs, and by various software components developed for them. Unlike the GPL, the BSD license allows modifications to be distributed under other licenses. The latest versions of this license are very similar to the MIT license.

MIT The Massachusetts Institute of Technology (MIT) was the original moving force behind the X Window System, and the MIT license (sometimes called the X11 license) continues to be used for XFree86—the implementation of X included with all major Linux distributions. The MIT license is unusually short.

Artistic The Artistic license was originally developed for the Perl programming language, but it has been used with other programs. It's filled with requirements and loopholes for those requirements. Most software that uses the Artistic license is shipped with the stipulation that this license is optional; the user may elect to follow the terms of some other license (usually the GPL) instead.

The Qt Public License Trolltech (`http://www.trolltech.com`) developed a cross-platform GUI library, Qt, that was used as the core of KDE, among other programs. Qt was originally licensed in a manner that did not qualify it as open source, although it was freely available in Linux. Trolltech has since modified its license so that it does qualify as open source, although some free software purists dislike it because it retains more rights for the owner than most open source licenses do.

NPL and MPL The Netscape Public License (NPL) and Mozilla Public License (MPL) were developed by Netscape when they brought their Netscape Web browser into the open source field. Like the Qt Public license, the NPL reserves some rights for the copyright holder, but the MPL is more open. Numerous little-used licenses are modeled after these licenses.

Additional licenses exist that meet the open source requirements. You can find a complete list, and additional discussion of just what an open source license is, on the Open Source Initiative Web site (`http://www.opensource.org`).

The details of the various open source licenses are probably not terribly important to most system administrators. You may use and redistribute any open source program as you like. If you modify a program, though, you should be aware of redistribution requirements, particularly if you want to merge two or more programs or distribute a program under a modified license. You should also be aware that some Linux distributions (particularly those that ship in official boxes) may include software that doesn't qualify as open source. Some of this is commercial software, and some of it falls into a variant category.

Open source software is *not* the same as public domain software, although the two are similar in some ways. The section "Miscellaneous Software Licenses" touches on public domain software later in this chapter.

Commercial Software Licenses

Commercial software isn't as common in Linux as it is in Windows or commercial Unix systems, but it still exists. Commercial programs for Linux are frequently large productivity applications, such as StarOffice, or major servers, such as Zeus. You're less likely to find small utilities sold under commercial licenses.

Although the Linux kernel is distributed under the GPL, there are a few commercial kernel module packages. In particular, the Open Sound System (OSS; `http://www.4front-tech.com`) package is a commercial set of sound drivers for Linux. OSS can distribute commercial kernel modifications because they're separate modules that aren't compiled into the kernel proper.

Commercial software is not as clearly defined as open source software. Some people consider anything that's not open source to be commercial, but it's not as simple as that, as illustrated shortly in the section "Miscellaneous Software Licenses." As a general rule, though, commercial software has several characteristics not shared with open source software:

Closed source The source code to commercial software tends to be unavailable, or is available only with serious restrictions, such as the recipient signing a nondisclosure agreement (NDA), which prevents its redistribution.

Distribution limitations Often, the user cannot legally redistribute commercial software. This limitation sometimes doesn't apply, though. For instance, many commercial software packages today may be downloaded from the Internet and redistributed, but they don't work fully unless you enter a license key. (In this case, you wouldn't be allowed to distribute your license key.)

Payment You must usually pay to use commercial software. To be sure, you may also pay to obtain open source software, but you can usually find open source software for little or no cost, particularly if you don't count media or Internet charges. The developer of a commercial product usually expects a monetary return, though. Sometimes a stripped-down version of commercial software is given away.

If all these restrictions apply, you can be pretty secure in applying the term "commercial software" to a package. If some of these conditions don't apply, it may be a gray area, but commercial packages often relax one or even all of these restrictions. For instance, Corel made a stripped-down version of its WordPerfect 8 available for download from the Internet for free, but most people still considered it to be commercial software. (WordPerfect 8 has since been discontinued, but you may still be able to find the free version on old archive CD-ROMs or the like.)

When you use commercial software in Linux, make sure that you comply with any license restrictions. These may include a limit on the number of total or simultaneous users of the software, an expiration date for a license, or other factors. If you use much commercial software, tracking these limitations can be a headache.

Some people use the term "closed source" instead of "commercial software." This usage has the merit of being more precise—it focuses on the unavailability of source code, de-emphasizing the redistribution rights and payment issues. Something that's not open source is not necessarily closed source, though, as described in a moment.

Miscellaneous Software Licenses

Some software falls in a limbo-land somewhere between open source and fully commercial software. There are also licenses (or a lack thereof) that simply aren't either open source or commercial. A few specific examples include the following:

Public domain The term *public domain* refers to a work for which the author has eliminated the copyright, or for which the copyright has lapsed. Public domain software is distributed with *no* licensing or copyright restrictions. As such, it's neither open source nor commercial, although somebody who makes modifications could copyright those changes in either way.

A lack of a copyright notice does not necessarily mean that a product is in the public domain. If you can't find a copyright notice or statement that a package is public domain, contact the author to learn its status. The author may have neglected to explicitly address the copyright issue, or the copyright notice may have been lost somewhere between the author and you.

Not-quite-open-source Some software is distributed under a license that's fairly free but not quite fully open source. For instance, the qmail mail server uses a modified version of the GPL that forbids redistribution of modified binary versions of the product. This clause violates the open source definition, but the software is far from commercial in nature.

Shareware The term *shareware* is applied to software that's freely redistributable but for which the author requests a payment. Sometimes this software is uncrippled in its freely available form, but other times it requires a license key to unlock some of its features. The line between shareware and commercial products has blurred as the Internet has allowed major software houses to distribute software electronically. There have also been cases of people distributing software under an open source license while simultaneously requesting (but not requiring) payment.

Demoware Commercial software houses sometimes distribute crippled versions of their products for free. This software is often called demoware or crippleware. It's generally considered fairly commercial in nature. Sometimes demoware can be converted into a full product by paying for it and entering a product key, making the difference between it and shareware foggy.

The distinctions between these licenses can become quite blurry. In all cases, though, what's important is not so much whether you call the software open source, commercial, or something

else; it's what the license does and does not allow you to do. If you're in doubt, contact the author of the software.

One final term that deserves consideration is *freeware*. This word generally applies to any software that's distributed free of charge, whether it's open source or not. This is not the same as free software, the term favored by the FSF for GPLed and some other forms of open source software. The free version of WordPerfect 8 is an example of freeware that is not open source.

Using Licensed Software in Linux

Some people believe that Linux's open source nature means that the OS may only run other open source software. This isn't true. Many commercial applications are available for Linux. Most libraries use the LGPL or other open source licenses that allow developers to write software that uses the library without becoming encumbered by the open source license themselves. If this weren't the case, programs like StarOffice and Zeus would not be available in Linux.

As with any environment, though, you should check the licensing terms of the software you intend to run. Software licenses occasionally include peculiar terms, and in principle, a program could include a requirement that the software not be run from Linux, or something else that would prevent you from legally running the software. I've seen licenses that restrict the use of the software geographically or in certain professions, for instance (such practices spurred the open source requirements #5 and #6).

Linux Distributions' Licenses

One final concern when discussing software licenses is the license for Linux as a whole. When you download a CD-ROM image file or buy a Linux package, the software you obtain uses many different licenses—the GPL, the BSD license, the MIT license, and so on. Most of these licenses are open source, but some aren't. Many distributions ship with a few shareware or not-quite-open-source packages, such as the shareware XV. Retail packages sometimes include outright commercial software. For this reason, you shouldn't copy a retail Linux package's CD-ROM. (An image file downloaded from the Internet is probably safe to copy, but check any accompanying file called `COPYING` or `COPYRIGHT` to be sure.)

Linux distributions include installation programs, configuration programs, and the like. These tools are usually all that a distribution packager can lay claim to, in terms of copyright. Most distribution maintainers have made their installation and configuration routines available under the GPL or some other open source license, but this isn't always the case. For instance, SUSE's GUI configuration tools, YaST and YaST2, are not open source. Such details can turn what might seem like an open source OS into something that's not quite fully open source. Debian maintains a policy of using only open source software in its main package set, although it lets freely redistributable but non-open source programs into its "non-free" package set. Gentoo maintains a similar policy.

Because a complete Linux distribution is composed of components using many different licenses, it's not useful to speak of a single copyright or license applying to the entire OS.

Instead, you should think of a Linux distribution as being a collection of different products that comes with a unifying installation utility. The vast majority of all the programs use one open source license or another, though.

Locating Linux Software

Modern Linux distributions are fairly complete entities as delivered. Most fill one to eight 650MB CD-ROMs—and most of this content is stored in a compressed form, so the result is a system that can easily fill 1–10GB when completely installed and uncompressed. In practice, though, a Linux installation can be much sparer than this; you're unlikely to install every package that comes with your distribution, after all. Some of this space is also consumed by source code, which you probably don't need.

On the other hand, it's entirely possible that your installed Linux system will be missing a few programs that you want. This is particularly likely if you use one of the smaller distributions or want to run some commercial software products. When this happens, it's necessary that you know where to go to find what you want.

Locating Open Source Software

You can obtain open source software in any of several ways:

The Linux CD-ROM Linux installation media, as just noted, typically include a wide array of software. If you know the name of the package you want, you can search for it on the CD-ROM. For instance, to find the Nedit editor on a CD-ROM mounted at `/mnt/cdrom`, you might type **`find /mnt/cdrom -name "nedit*"`**. Such searches work best when the filename is something obvious, given the package name.

File archive collections Companies that distribute inexpensive Linux CD-ROMs also often distribute CD-ROMs that contain the archives of popular open source FTP sites. These can be a good way to obtain extra software if you don't have a fast Internet connection.

File archive sites Many Web and FTP sites host collections of open source software. Notable examples include `http://sourceforge.net`, `http://www.ibiblio.org/pub/Linux`, and `ftp://sunsite.unc.edu`.

Package maintainer's site Most open source projects have homepages, often named after the application itself (usually in the `.org` domain). If you can't find such a site by using the program's name, try doing a Web search.

If you don't know the name of a package but do know the type of software you want to find (such as a Pascal compiler or alternatives to the Apache Web server), you can try searching at `http://sourceforge.net` or `http://www.linux.org`. Both sites include categorized lists of open source or Linux programs. The GNU Project Web site, `http://www.gnu.org`, includes a

listing of the FSF's software and some related packages. A Web search and a search on Google Groups (`http://groups.google.com`) are also well worth trying.

Locating Commercial Software

Commercial software for Linux can be obtained in ways that often parallel those for obtaining open source software. Specifically, consider the following:

Linux packages Retail Linux packages sometimes include demos of commercial products. Some ship with one or two fully functional commercial programs as well.

Linux retailers You can obtain many commercial Linux programs from Linux retailers, such as CheapBytes (`http://www.cheapbytes.com`). Some mainstream Web retailers also carry Linux products. In addition to Web-based retailers, Linux software is increasingly finding its way onto computer, electronics, office supply, and even department store shelves.

Software publishers The software publishers themselves often sell their packages directly. In fact, some offer downloadable versions; you can download the software and then buy a license key from the publisher's Web site.

If you don't know the name of the software you want, searching on Linux Web sites or doing a Web search, as with open source software, will often turn up useful information. Linux magazines, such as *Linux Journal* and *Linux Magazine*, often carry reviews of commercial software products. These publications frequently have monthly themes in which some topic is covered in depth, often including multiple or head-to-head product reviews.

Summary

Before installing Linux, you should take some time to plan the implementation. Although Linux works with a wide variety of hardware, you should consider this detail carefully, both to get a system with the features you need within your budget and to be sure that you don't have any components that are unsupported in Linux. Checking the hardware before you install Linux can also save you a great deal of aggravation, should some component be installed incorrectly or conflict with another device.

Planning your software configuration is also important. This begins with planning disk partitions to suit the needs of the system. Several Linux distributions are available, and you may even want to consider non-Linux OSs for some purposes. Occasionally, selecting software is dependent on software licenses. Most Linux packages use an open source license, but some programs use commercial and other types of licenses. Depending on the role of the computer, you'll need to install different sets of software when it comes time to install Linux, as described in the next chapter.

Exam Essentials

Describe the difference between a workstation and a server. Individuals use workstations for productivity tasks; servers exchange data with other computers over a network.

Suggest ways to stretch a limited budget when buying or building a Linux computer. Upgrade an existing computer rather than buy a new one; incorporate existing components into an otherwise new computer; prioritize the system's hardware needs, and eliminate or use inexpensive hardware for low-priority functions.

Describe how CPU speed, available RAM, and hard disk characteristics influence performance. Faster CPUs result in faster computations, and thus faster speed in computationally intensive tasks, while plentiful RAM gives the computer room to perform computations on large data sets. Hard disks vary in capacity and speed, which affect your ability to store lots of data and your ability to rapidly access it.

Describe Linux's partitioning needs. Linux requires a single root partition, and may require a separate swap partition. Additional partitions, corresponding to directories such as `/boot`, `/home`, and `/var`, are desirable on some systems, but aren't usually required.

Summarize the concept of a Linux distribution. A distribution is a collection of software developed by diverse individuals and groups, bound by an installation routine. Linux distributions can differ in many details, but they all share the same heritage and the ability to run the same programs.

Determine how a computer will be used. Workstations serve as productivity tools for individuals, whereas servers respond to data transfer requests from many clients. Which role a computer will fill determines the types of programs you'll install on it.

Describe the most important characteristics of open source software licenses. Open source licenses ensure the availability of source code and your ability to change and redistribute that source code and the resulting binary programs.

List the most important characteristics of commercial software licenses. Commercial licenses usually don't allow users to distribute the software or see the source code. Commercial software copyright holders usually expect payment for their software.

Summarize the *x*86 boot process. The CPU executes code stored on the BIOS, which redirects the CPU to load and execute a boot loader from the MBR. This boot loader may load the OS kernel or redirect the boot process to another boot loader, which in turn loads the kernel and starts the OS running.

Describe important configuration concerns when adding ATA components. ATA supports up to two devices per chain, and each device must be configured as a master or a slave; only one of each type is permitted per chain.

Describe important configuration concerns when adding SCSI components. SCSI devices are configured with unique SCSI ID numbers, which range from 0 to 7 or 15, depending on the SCSI variant. The devices on the end of each SCSI chain must be properly terminated, and those in between must *not* be terminated.

Summarize procedures you should follow when installing new internal hardware. Be sure to ground yourself and turn off the computer. Older ISA devices must have their jumpers set for proper use of IRQs, DMAs, I/O ports, and possibly other features; consult their documentation.

Commands in This Chapter

In other chapters, you will find a list of commands that were used in the chapter here. You should make sure you are familiar with these commands and how to use them.

Key Terms

Before you take the exam, be certain you are familiar with the following terms:

Advanced Graphics Port (AGP)
Advanced Technology Attachment (ATA)
Basic Input/Output System (BIOS)
binary
bits
bus
byte
C library (libc)
cache memory
cathode ray tube (CRT)
central processing unit (CPU)
chipset
Common Internet Filesystem (CIFS)
compiler
library
liquid crystal display (LCD)
logical partition
main memory
master
master boot record (MBR)
modem
motherboard
mount point
open source
partition table
partition
Peripheral Component Interconnect (PCI)
primary partition

Complementary Metal Oxide Semiconductor (CMOS) setup utility
console
copyleft
desktop computer
desktop environment
direct memory access (DMA)
distribution
dual inline memory module (DIMM)
dynamic RAM (DRAM)
Enhanced Integrated Device Electronics (EIDE)
Ethernet
extended partition
external transfer rate
failed dependency
file-sharing protocol
filesystem
freeware
General Public License (GPL)
glibc
hot swapping
Hypertext Transfer Protocol (HTTP)
IEEE-1394
Industry Standard Architecture (ISA)
input/output (I/O)
internal transfer rate
interrupt request (IRQ) number
journaling filesystem
Programmed Input/Output (PIO)
public domain
RAMbus dynamic RAM (RDRAM)
RDRAM inline memory modules (RIMMs)
ReiserFS
ribbon cable
root partition
scripting language
second extended filesystem (ext2 or ext2fs)
Secure Shell (SSH)
Serial ATA (SATA)
Serial Attached SCSI (SAS)
server
Server Message Block (SMB)
server program
shareware
single inline memory module (SIMM)
slave
Small Computer System Interface (SCSI)
Small Outline (SO) DIMM
software modem
text editor
third extended filesystem (ext3fs)
Universal Serial Bus (USB)
window manager
workstation

Review Questions

1. Which of the following are typical workstation tasks? (Choose all that apply.)
   - **A.** Word processing
   - **B.** Routing between networks
   - **C.** Running a Web site
   - **D.** Running scientific simulations

2. A computer is to be used to capture 640×480 images of a room every 10 minutes and then store them for a day on hard disk. Which of the following components might you research before building such a computer?
   - **A.** A 21-inch monitor for viewing the images
   - **B.** A high-end SCSI disk to store the images quickly
   - **C.** A 3D graphics card to render the image of the room
   - **D.** USB support for a USB-interfaced camera

3. You're designing a computer as a workstation to be used primarily for word processing. Which of the following cost-saving measures is *least* appropriate for this system?
   - **A.** Buying a keyboard that costs $10 rather than one that costs $50
   - **B.** Buying an 80GB hard disk rather than a 160GB model
   - **C.** Buying a CD-ROM drive rather than a DVD-ROM drive
   - **D.** Buying a 1.5GHz system rather than a 2GHz one

4. Linux runs on many different types of CPUs. Which of the following measures is most useful when comparing the speed of CPUs from different families?
   - **A.** The BogoMIPS measures reported by the kernel
   - **B.** The CPU speeds in MHz
   - **C.** The number of transistors in the CPUs
   - **D.** How quickly each CPU runs your programs

5. Which of the following is *not* an advantage of SCSI hard disks over ATA hard disks?
   - **A.** SCSI supports more devices per IRQ.
   - **B.** SCSI hard disks are less expensive than their ATA counterparts.
   - **C.** SCSI allows multiple simultaneous transfers on a single chain.
   - **D.** The highest-performance drives come in SCSI format.

6. As a general rule, which of the following is most important in order for a video card to be used in a Linux business workstation?
   - **A.** The card should be supported by the commercial Accelerated-X and Metro-X servers.
   - **B.** The card should have much more than 8MB of RAM for best speed.
   - **C.** The card should be supported by XFree86.
   - **D.** The card should be the most recent design to ensure continued usefulness in the future.

7. When installing an ATA hard disk, what feature might you have to set by changing a jumper setting on the disk?
   - **A.** The drive's bus speed (33, 66, 100, or 133MB/s)
   - **B.** The drive's termination (on or off)
   - **C.** The drive's master or slave status
   - **D.** The drive's ID number (0–7 or 0–15)

8. Why might you want to check the motherboard BIOS settings on a computer before installing Linux?
   - **A.** The BIOS lets you configure the partition to be booted by default.
   - **B.** You can use the BIOS to disable built-in hardware you plan not to use in Linux.
   - **C.** The motherboard BIOS lets you set the IDs of SCSI devices.
   - **D.** You can set the screen resolution using the motherboard BIOS.

9. You want to attach an old 10MB/s SCSI-2 scanner to a computer, but the only SCSI host adapter you have available is a 20MB/s UltraSCSI device. The system has no other SCSI devices. Which of the following is true?
   - **A.** You can attach the scanner to the UltraSCSI host adapter; the two are compatible, although you may need an adapter cable.
   - **B.** You must set an appropriate jumper on the UltraSCSI host adapter before it will communicate with the SCSI-2 scanner.
   - **C.** You must buy a new SCSI-2 host adapter; SCSI devices aren't compatible across versions, so the UltraSCSI adapter won't work.
   - **D.** You can attach the scanner to the UltraSCSI host adapter, but performance will be poor because of the incompatible protocols.

10. A new Linux administrator plans to create a system with separate `/home`, `/usr/local`, and `/etc` partitions. Which of the following best describes this configuration?
    - **A.** The system won't boot because `/etc` contains configuration files necessary to mount non-root partitions.
    - **B.** The system will boot, but `/usr/local` won't be available because mounted partitions must be mounted directly off their parent partition, not in a subdirectory.
    - **C.** The system will boot only if the `/home` partition is on a separate physical disk from the `/usr/local` partition.
    - **D.** The system will boot and operate correctly, provided each partition is large enough for its intended use.

11. Which of the following best summarizes the differences between DOS's `FDISK` and Linux's `fdisk`?

A. Linux's `fdisk` is a simple clone of DOS's `FDISK`, but written to work from Linux rather than from DOS or Windows.

B. The two are completely independent programs that accomplish similar goals, although Linux's `fdisk` is more flexible.

C. DOS's `FDISK` uses GUI controls, whereas Linux's `fdisk` uses a command-line interface, but they have similar functionality.

D. Despite their similar names, they're completely different tools—DOS's `FDISK` handles disk partitioning, whereas Linux's `fdisk` formats floppy disks.

12. Which of the following characteristics differ between Linux and commercial Unix systems? (Choose all that apply.)

A. The ability to run open source software

B. Cost

C. The history of the kernel source code base

D. Underlying principles of OS design

13. In what ways do Linux distributions differ from one another? (Choose all that apply.)

A. Package management systems

B. Kernel development history

C. Installation routines

D. The ability to run popular Unix servers

14. Which of the following packages are *most* likely to be needed on a computer that functions as an office file server?

A. Samba and Netatalk

B. Apache and StarOffice

C. Gnumeric and Postfix

D. XV and BIND

15. What type of software is it most important to *remove* from a publicly accessible server?

A. Unnecessary kernel modules

B. Unused firewall software

C. Uncompiled source code

D. Software development tools

16. Which of the following is *not* required in order for software to be certified as open source?
    - **A.** The license must not discriminate against people or groups of people.
    - **B.** The license must not require that the software be distributed as part of a specific product.
    - **C.** The license must not require that changes be distributed under the same license.
    - **D.** The program must come with source code, or the author must make it readily available on the Internet.

17. Which of the following is true of commercial software licenses in Linux? (Choose all that apply.)
    - **A.** They must conform to the terms of the LGPL.
    - **B.** They may restrict distribution, require payments, or have other terms common to commercial licenses in commercial OSs.
    - **C.** They are uncommon compared to open source licenses.
    - **D.** They necessarily prevent distribution of the commercial package with a Linux distribution.

18. How do you set IRQs on PCI boards?
    - **A.** You don't; PCI boards don't use IRQs.
    - **B.** You don't; they're set automatically by the BIOS or kernel.
    - **C.** By adjusting jumpers on the board.
    - **D.** By editing the `/etc/isapnp.conf` file and running `isapnp`.

19. How can you expect your Linux distribution to arrive?
    - **A.** It should come with enough software that some systems don't need additional packages.
    - **B.** It will invariably require additional software package installation.
    - **C.** Generally, it will consist of at least 50 percent commercial software.
    - **D.** It cannot be obtained from the same stores that make commercial software available.

20. Possible sources of both commercial and open source Linux software include which of the following? (Choose all that apply.)
    - **A.** The program author's Web site
    - **B.** Linux retail boxes
    - **C.** Internet retailers
    - **D.** Brick-and-mortar retailers

Answers to Review Questions

1. A, D. Workstations are used by individuals to perform productivity tasks, such as word processing, drafting, scientific simulations, and so on. Routing is a task that's performed by a router—typically a dedicated-appliance task. Web sites are run on servers.

2. D. Many digital cameras use USB interfaces, so Linux's support for USB, and for specific USB cameras, may be important for this application. (Some cameras use parallel-port, IEEE-1394, or specialized PCI card interfaces as well.) A 21-inch monitor is overkill for displaying 640 × 480 images, and a 3D graphics card isn't required, either. Likewise, a 10-minute pause between captures is slow enough that a high-end hard disk (SCSI or ATA) isn't necessary for speed reasons, although a large hard disk may be required if the images are to be retained for any length of time.

3. A. As a word processing workstation, this system's keyboard quality is important. An 80GB hard disk and a 1.50GHz CPU are almost certainly more than adequate for this application, as is a CD-ROM drive.

4. D. The ultimate measure of a CPU's speed is how quickly it runs *your* programs, so the best measure of CPU performance is the CPU's performance when running those programs. The BogoMIPS measure is almost meaningless; it's used to calibrate some internal kernel timing loops. CPU speed in MHz is also meaningless across CPU families, although it is useful *within* a family. Likewise, the number of transistors in a CPU is unimportant per se, although more sophisticated CPUs are often faster.

5. B. SCSI hard disks usually cost more than ATA drives of the same size, although the SCSI disks often perform better.

6. C. XFree86 comes with all full Linux distributions, so having XFree86 support is important to getting Linux working in GUI mode. Support in Accelerated-X and Metro-X can work around a lack of support in XFree86 or provide a few features not present in XFree86, but in most cases, XFree86 support is more important. More than 8MB of RAM is important if you want to use a card's 3D features, but few Linux programs use these today. The most recent designs are often incompatible with XFree86 because drivers have yet to be written.

7. C. ATA drives can be configured for one of two positions on an ATA chain, master or slave. (Modern drives often support autoconfiguration through a "cable select" or similar option, and sometimes a single-drive configuration, but these are just different ways of setting the same feature.) Termination and ID number are characteristics of SCSI devices, not ATA devices. The drive's bus speed adjusts automatically depending on the maximum of the drive and the ATA controller.

8. B. Motherboards with built-in RS-232 serial, parallel, ATA, audio, and other devices generally allow you to disable these devices from the BIOS setup utility. The BIOS does *not* control the boot partition, although it *does* control the boot device (floppy, CD-ROM, hard disk, and so on). SCSI host adapters have their own BIOSs, with setup utilities that are separate from those of the motherboard BIOS. (They're usually accessed separately even when the SCSI adapter is built into the motherboard.) You set the screen resolution using X configuration tools, not the BIOS.

9. A. SCSI devices are compatible from one version of the SCSI protocols to another, with a few exceptions such as LVD SCSI devices. Several types of SCSI connectors are available, so a simple adapter may be required. No jumper settings should be needed to make the UltraSCSI adapter communicate with the SCSI-2 scanner. Performance will be at SCSI-2 levels, just as if you were using a SCSI-2 host adapter.

10. A. The `/etc/fstab` file contains the mapping of partitions to mount points, so `/etc` must be an ordinary directory on the root partition, not on a separate partition. Options B and C describe restrictions that don't exist. Option D would be correct if `/etc` were not a separate partition.

11. B. Although they have similar names and purposes, Linux's `fdisk` is not modeled after DOS's `FDISK`. DOS's `FDISK` does *not* have GUI controls. Linux's `fdisk` does *not* format floppy disks.

12. B, C. Linux generally costs less than commercial Unix systems, and its source code (particularly the kernel) is not derived from the same base as that of commercial Unixes. Both Linux and commercial Unix systems can run most of the same open source software, though. As a clone of Unix, Linux uses the same underlying OS design principles.

13. A, C. Different Linux distributions use different package management systems and installation routines. Although they may ship with slightly different kernel versions, they use fundamentally the same kernel. Likewise, they may ship with different server collections, but can run the same set of servers.

14. A. Samba is a file server for SMB/CIFS (Windows networking), while Netatalk is a file server for AppleShare (Mac OS networking). Apache is a Web server, and StarOffice is a workstation package. Gnumeric is a spreadsheet, and Postfix is a mail server. XV is a graphics package, and BIND is a name server. Any of these last six *might* be found on a file server computer, but none fills the file serving or any other necessary role, and so each is superfluous on a system that's strictly a file server.

15. D. System crackers can use compilers and other development tools to compile their own damaging software on your computer. Unnecessary kernel modules don't pose a threat. You may want to begin using unused firewall software, but removing it is unlikely to be necessary or helpful. Uncompiled source code may consume disk space, but it isn't a threat unless a compiler is available and the source code is for network penetration tools.

16. C. The open source definition specifies that users be able to distribute changes, but it doesn't require that the license allow distribution under the terms of another license. Options A, B, and D all paraphrase actual open source license term requirements.

17. B, C. Linux's licensing terms don't restrict the rights of commercial software vendors to apply their own licensing terms. Most Linux software is open source in nature. The LGPL allows commercial software to link to LGPLed software, but the LGPL does not impose its terms on commercial packages. Commercial packages can be and sometimes are distributed with Linux packages.

18. B. PCI was designed so that the BIOS or OS could set IRQs for these devices automatically, so it's not normally necessary to explicitly adjust these features. In the event of a conflict, you can sometimes change the algorithm the BIOS uses to assign IRQs, though.

19. A. Linux distributions come on 1–8 CD-ROMs, which include wide assortments of open source and occasionally commercial software—filling all the needs of some systems. Sometimes—but not invariably—additional software is required. All Linux distributions sold today include far less than 50 percent commercial software. Many retailers sell both Linux distributions and commercial Linux software.

20. A, B, C, D. Open source software may be distributed in any way that's technologically possible. Distribution of commercial software is dependent on license terms, but as a whole, commercial software developers have embraced a wide array of distribution methods, including all the options listed here.