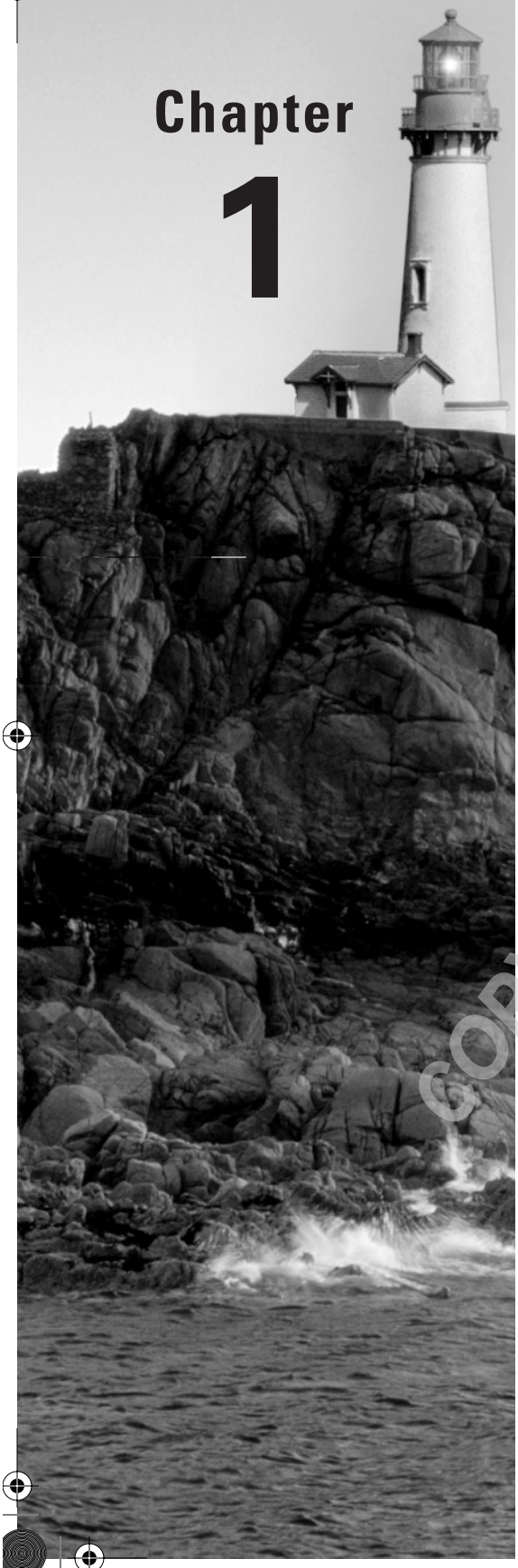
4389.book Page 1 Tuesday, January 11, 2005 9:35 PM



Chapter 1

Linux Installation

THE FOLLOWING COMPTIA OBJECTIVES ARE COVERED IN THIS CHAPTER:

- ✓ **1.1 Identify all system hardware required (e.g., CPU, memory, drive space, scalability) and check compatibility with Linux Distribution.**
- ✓ **1.2 Determine appropriate method of installation based on environment (e.g., boot disk, CD-ROM, network (HTTP, FTP, NFS, SMB)).**
- ✓ **1.3 Install multimedia options (e.g., video, sound, codecs).**
- ✓ **1.4 Identify purpose of Linux machine based on predetermined customer requirements (e.g., appliance, desktop system, database, mail server, web server, etc.).**
- ✓ **1.5 Determine what software and services should be installed (e.g., client applications for workstation, server services for desired task).**
- ✓ **1.6 Partition according to pre-installation plan using fdisk (e.g., `/boot`, `/usr`, `/var`, `/home`, Swap, RAID/volume, hotfix).**
- ✓ **1.7 Configure file systems (e.g., (ext2) or (ext3) or REISER).**
- ✓ **1.8 Configure a boot manager (e.g., LILO, ELILO, GRUB, multiple boot options).**
- ✓ **1.11 Select appropriate parameters for Linux installation (e.g., language, time zones, keyboard, mouse).**
- ✓ **3.11 Configure the X Window System**

COPYRIGHTED MATERIAL

Sometimes you'll encounter a system that's already running Linux—say, if you're hired to administer systems that are already up and running, or if you buy a system with Linux preinstalled on it. Frequently, though, you must install Linux before you can begin using or administering it. This task isn't really any more difficult than installing most other OSs, but OS installation generally can be intimidating to those who've never done it. Linux also has its own installation quirks, which you should understand before proceeding. In addition, installation options can have an impact on how you use a system. That is, installation choices help determine how a Linux system is configured, such as what servers are available and how the network is configured. Although you can change these details later, getting them right when you first install Linux is generally preferable to modifying them afterward.

Understanding your computer's role is important in determining how you install an OS on it. Thus, this chapter begins with a look at the needs of various types of computers—workstations, servers, and more specialized types of computers. This chapter continues with information on the hardware and software needs of both Linux and of various Linux roles. Understanding these factors will help you plan a Linux installation. The first of the actual installation tasks is partitioning your disk, so this topic is up next. You must then plan how you're going to install Linux—that is, what source media to use and how to interact with the computer. The actual installation process is described in broad strokes next, although details do vary substantially from one distribution to another. Finally, this chapter looks at configuring the X Window System—Linux's GUI environment.

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.

4389.book Page 3 Tuesday, January 11, 2005 9:35 PM

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 three-button mouse. (Linux, unlike Windows, uses all three buttons, so a two-button mouse is suboptimal.) Workstations also usually include audio hardware (a sound card, speakers, and sometimes a microphone) and high-capacity removable media drives (Zip or LS-120 drives, frequently 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 this 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.

4389.book Page 4 Tuesday, January 11, 2005 9:35 PM

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.

4389.book Page 5 Tuesday, January 11, 2005 9:35 PM

Real World Scenario

Linux Thin Clients

One use of Linux that's interesting in certain environments is using Linux as a *thin client* OS—that is, an OS for a computer that runs just enough software to provide input/output functions for another computer. This can be handy if an office has several workers who need to use a computer for functions that are not, by and large, CPU-intensive. You can set up a single login server computer and provide the individual users with thin client computers with which they access the main server. This approach can save money by enabling you to reuse old computers as thin clients. It can also reduce administrative effort compared to giving every user a full workstation system.

Thin clients often boot using network boot protocols such as the *Preboot Execution Environment (PXE)*, which is a BIOS feature that enables booting from files stored on a *Trivial File Transfer Protocol (TFTP)* server. PXE essentially turns a network card and TFTP server into a boot device.

Of course, the TFTP server must hold suitable boot files—essentially, a miniature Linux distribution with thin client software. Examples of such software include PXES (`http://pxes.sourceforge.net`) and the Linux Terminal Server Project (LTSP; `http://www.ltsp.org`). Once configured, a Linux thin client can use Linux, Windows, or other OSs as servers, provided they're equipped with appropriate software.

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 a popular 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. If you use an old distribution, it may lack USB support, but all current mainstream distributions provide good USB 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.

4389.book Page 6 Tuesday, January 11, 2005 9:35 PM

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 details. If you possess the necessary knowledge, I recommend indicating manufacturer and model numbers for every component, along with one or two backups for each. 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. 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.

4389.book Page 7 Tuesday, January 11, 2005 9:35 PM

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. 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 X, Linux's GUI package, that work with most cards, but not absolutely all of them.

Input devices The keyboard and mouse enable you to give commands to the computer. These devices are well standardized, although there are a few variants of each type. Linux provides standardized drivers for most common keyboards and mice (including trackballs and similar mouse alternatives).

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.

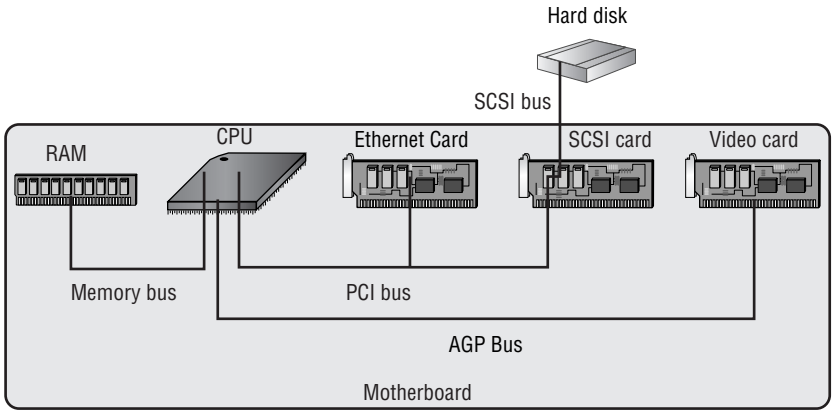
Audio hardware Many workstations include audio hardware, which lets the system play back sounds and digitize sounds using microphones or other audio input devices. These aren't critical to basic system functioning, though; Linux will boot quite well without a sound card.

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, lines of communication exist between the memory and *Peripheral Component Interconnect (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. The most common busses that connect to plug-in cards are the PCI bus and the *Advanced Graphics Port (AGP)* bus. The former is used for SCSI host adapters, Ethernet cards, sound cards, and most other card types. It comes in 32- and 64-bit varieties, the latter being faster, although it's still rare. The AGP bus is used only by video cards. Older busses, such as the *Industry Standard Architecture (ISA)* bus, have been largely abandoned, but you may run into them on older computers. The term "bus" can also refer to communication lines within the CPU and between the CPU and components that can't be removed.

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."

4389.book Page 8 Tuesday, January 11, 2005 9:35 PM

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



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.

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.

Traditional *x*86 systems use 32-bit internal registers, although Pentium systems and above have 64-bit links to memory. 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. The 64-bit variant of *x*86 is known as the *AMD64* or *x86-64* platform, and is available as the AMD Opteron, AMD Athlon-64, and some (but not all) Intel Xeon CPUs. (Intel uses the phrase "Extended Memory 64" to refer to the AMD64 architecture.) These CPUs can run both traditional 32-bit versions of Linux and 64-bit versions. When running a 64-bit version of Linux and applications compiled using a 64-bit compiler, you get a modest speed boost (about 10–30 percent). Most 32-bit binaries can run in an AMD64 environment, but a few don't.

4389.book Page 9 Tuesday, January 11, 2005 9:35 PM

Intel has also released another 64-bit *x*86 variant, known as IA-64. IA-64 CPUs are sold under the name Itanium, but this platform has not become popular. Most industry pundits predict that IA-64 will slowly fade away while AMD64 will take over the workstation and small server market.

In addition to *x*86 CPUs and their AMD64 and IA-64 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. 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) 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, and an interface to the keyboard. Linux works with most motherboards, although on occasion, Linux doesn't support all of a motherboard's features. 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 2004 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. 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

4389.book Page 10 Tuesday, January 11, 2005 9:35 PM

narrower. Motherboards host sockets for particular types of memory, so you must match your RAM purchases to your motherboard.

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 2004, 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.

RAM also varies in how well it copes with errors. Some memory modules incorporate a ninth bit (known as a parity bit) in each byte as an error-detection bit. This extra bit enables the motherboard's memory controller to detect, and often to correct, memory errors.

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 data in the cache, which results in a substantial performance increase.

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

4389.book Page 11 Tuesday, January 11, 2005 9:35 PM

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. This is because SCSI can support more devices per chain, SCSI handles multiple simultaneous transfers (from different devices) better than does ATA, and hard disk manufacturers tend to release their fastest and highest-capacity drives in SCSI format. 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 certain 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 2004, SATA is starting to become popular on new hardware, and SAS has yet to be released. 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.

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. (Some Linux SATA drivers also make them 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. 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.

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, although gigabit Ethernet is gaining in popularity.

4389.book Page 12 Tuesday, January 11, 2005 9:35 PM

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.

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.

Most 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 9, "Hardware." 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 the X Window System.

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`), X.org-X11 (`http://www.x.org`), or the Accelerated-X (`http://www.xig.com`) commercial X server. Because XFree86 or X.org-X11 ship with all major Linux distributions, it's best to use a board they support. (Prior to 2004, XFree86 was the preferred X server; but most distributions switched to X.org-X11 during 2004.) As a general rule of thumb, it's best to avoid the most recent video cards because drivers for XFree86 and X.org-X11 tend to lag a few months behind the release of the hardware. A few manufacturers do provide XFree86 and X.org-X11 drivers for their products, though, and Accelerated-X sometimes introduces drivers more rapidly than the open source developers do.

4389.book Page 13 Tuesday, January 11, 2005 9:35 PM

The Linux kernel includes a number of video drivers, known as frame buffer drivers. XFree86 and X.org-X11 include 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.

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.

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 Cases vary in quality—check for rough edges, a good fit, and easy access. There's nothing OS-specific about them.

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 4, "Disk Management").

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

4389.book Page 14 Tuesday, January 11, 2005 9:35 PM

hardware interface that matter. Mice using USB or PS/2 hardware use the PS/2 protocol or a variant of it that supports wheels.

RS-232 serial and parallel ports If you need to add extra RS-232 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 X. 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, and with any version of X.org-X11, the X server can sometimes obtain this information from the monitor. (X configuration is described in detail later in this chapter.)

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

USB devices Check `http://www.linux-usb.org` for information on what USB devices are 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 two sets of sound drivers: the original Open Sound System (OSS) drivers and the new Advanced Linux Sound Architecture (ALSA) drivers. Commercial variants of the OSS drivers are also available from `http://www.4front-tech.com`. 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/`; the `l` in `v4l` is a lowercase letter `l`, not a number `1`.) 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.

4389.book Page 15 Tuesday, January 11, 2005 9:35 PM

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.

Determining Software Needs

When you plan a Linux installation, you must know what software you'll need on the system. This task begins with picking the Linux *distribution*, which is a collection of software along with installation routines that enable you to install everything from scratch. Once this is done, you must decide what types of programs you need. 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.

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 2.6.8 Linux kernel and another might ship with 2.6.9, 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 5, "Package and Process Management."

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

4389.book Page 16 Tuesday, January 11, 2005 9:35 PM

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 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 nonprofit 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.

Fedora Linux This distribution is essentially the free version of Red Hat Linux. It's headquartered at `http://fedora.redhat.com`.

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, AMD64, 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.

Linspire This distribution, which is a Debian derivative, lies at the fringes of Linux. It's designed as a replacement for Windows on the desktop (and was once called "Lindows" to emphasize this fact). 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. Linspire is now included on some cut-rate retail PCs. Free downloads of Linspire are not available. You can learn more at `http://www.linspire.com`.

Lycoris Like Linspire, 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`.

4389.book Page 17 Tuesday, January 11, 2005 9:35 PM

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, AMD64, 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, AMD64, 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`; see also `http://www.novell.com`.) produces a distribution that's particularly popular in Europe. SuSE uses RPMs, but it's not otherwise based on Red Hat. 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, AMD64, IA-64, PPC, and Alpha are all available. Novell purchased SuSE in early 2004, although SuSE remains headquartered in Germany.

TurboLinux This distribution (`http://www.turbolinux.com`) began as a Red Hat derivative, but recent versions have lost much of this heritage. 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.

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.

4389.book Page 18 Tuesday, January 11, 2005 9:35 PM

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, "User Management," 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 and its Fedora subdistribution are unusually popular, so finding support for them 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 Easy Linux CDs (`http://www.easylinuxcds.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.

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 X.org-X11 package, although prior to 2004, XFree86 usually did this job. 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

4389.book Page 19 Tuesday, January 11, 2005 9:35 PM

(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. Markup languages like TeX and LaTeX (http://www.latex-project.org), 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.novell.com/linux/ximian.html) 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 a few aren't. Opera stands out in this respect.

4389.book Page 20 Tuesday, January 11, 2005 9:35 PM

For more information on network clients, refer to Chapter 6, "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 forgoing 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

4389.book Page 21 Tuesday, January 11, 2005 9:35 PM

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 "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 computer'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.

4389.book Page 22 Tuesday, January 11, 2005 9:35 PM

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 6. 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. Most major Linux distributions ship with all of these servers.

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.

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

4389.book Page 23 Tuesday, January 11, 2005 9:35 PM

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, enabling 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 that overview is 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 hostnames
- 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 3 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

4389.book Page 24 Tuesday, January 11, 2005 9:35 PM

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.

Programming Tools

Programming tools enable you to write programs for Linux. These tools can also be useful in getting Linux software you didn't write to run—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 have to install GCC, at a minimum, to compile most programs distributed as source code. 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. The Bash and `tcsh` shells also provide scripting features, which are described in Chapter 2, "Text-Mode Commands." 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.

4389.book Page 25 Tuesday, January 11, 2005 9:35 PM

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 5).

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. 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 5, 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. (One notable exception is the fact that most *x*86 programs can run on AMD64 CPUs.) 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.

4389.book Page 26 Tuesday, January 11, 2005 9:35 PM

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 5 describes 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 5 for information on Linux packages.

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 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

4389.book Page 27 Tuesday, January 11, 2005 9:35 PM

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- and AMD64-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.

Non-*x*86 distributions need boot loaders, too, but they're different from *x*86 boot loaders in various details. Sometimes a boot loader such as GRUB or LILO is ported or copied on non-*x*86 distributions. The IA-64 platform uses a boot loader called ELILO, for instance. Other times, a completely new boot loader is used, such as Yaboot for PowerPC systems.

4389.book Page 28 Tuesday, January 11, 2005 9:35 PM

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.

One partitioning strategy that's common on high-performance systems is a Redundant Array of Independent Disks (RAID). In a RAID configuration, partitions on separate physical hard disks are combined together to provide faster performance, greater reliability, or both. Some Linux distributions provide RAID options in their initial installation procedures, but others don't. RAID configuration is fairly advanced, and is covered in Chapter 4. If you're new to Linux, it's best to avoid RAID configurations on your first installation. After reading Chapter 4, you might try implementing a RAID configuration on subsequent installations.

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 are:

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.

4389.book Page 29 Tuesday, January 11, 2005 9:35 PM

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 1.1 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 4.

TABLE 1.1 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. Described in more detail in Chapter 4.
`/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.

4389.book Page 30 Tuesday, January 11, 2005 9:35 PM

TABLE 1.1 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.
/media	N/A	Holds subdirectories that may be used as mount points for removable media, much like /mnt or its subdirectories.

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.

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

4389.book Page 31 Tuesday, January 11, 2005 9:35 PM

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 1.1, 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.

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 4), but this isn't technically a filesystem because no files are stored within it.

4389.book Page 32 Tuesday, January 11, 2005 9:35 PM

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
- The Universal Disk Format (UDF), which is the up-and-coming successor to ISO-9660 for optical discs

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, Joliet, and UDF—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. All of the journaling filesystems except for ReiserFS support 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. 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.)

4389.book Page 33 Tuesday, January 11, 2005 9:35 PM

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 Symantec (`http://www.symantec.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.

4389.book Page 34 Tuesday, January 11, 2005 9:35 PM

Selecting an Installation Method

After you've decided on a distribution, the first choice you must make when installing Linux is what installation method you intend to use. Two classes of options are available: the installation media and the method of interaction during installation. In both cases, some distributions offer more or different options than do others, so in truth, your preferences in these matters may influence your distribution choice. For instance, Debian GNU/Linux doesn't support GUI installations, so if you strongly desire this feature, you can't use Debian.

Media Options

Linux can be booted and installed from any of several different media—floppy disks, CD-ROMs, network connections, and so on. For both booting and installing files, different media offer different advantages and disadvantages.

The Boot Method

Linux installer programs run within Linux itself. This means that in order to install Linux, you must be able to boot a small Linux system, which is provided by the distribution maintainer. This system is useful only for installing Linux and sometimes for doing emergency maintenance. It typically fits on one or two floppy disks, or can boot from a bootable CD-ROM.

Modern BIOSs include options for the selection of a boot medium. Typical choices include the floppy disk, CD-ROM drive, ATA hard disk, SCSI hard disk, and high-capacity removable-media drive (like a Zip or LS-120 disk). In addition, some network cards include BIOSs that enable a computer to boot from files stored on a server. In theory, any of these media can be used to boot a Linux installer. Additionally, some distributions provide a DOS or Windows program that can launch the installation from a working DOS or Windows system.

Although many boot methods are possible, the three most common are as follows:

Floppy Many boxed distributions come with one or more boot floppies. If you configure your BIOS to boot from floppy disks before any other working boot medium, you can insert the boot floppy and turn on the computer to start the installation process. Even if you download Linux or obtain it on a cut-rate CD-ROM without a boot floppy, you can create a boot floppy yourself from a file on the CD-ROM (often called `boot.img` or something similar), using a DOS program such as `RAWRITE`. Look for these files and instructions on how to use them on the installation CD-ROM. The floppy boot method may be necessary if you plan to install from a network server.

CD-ROM Modern Linux distributions almost always come on CD-ROMs or DVD-ROMs that are themselves bootable. On a computer that's configured to boot from CD-ROM before other bootable media, you can insert the CD-ROM in the drive, then turn on the computer, and the boot program automatically starts up. If you download and burn a Linux CD-R image file, you don't need to take any special steps to make this CD-R bootable. Some older BIOSs don't support CD-ROM boots, in which case you should make boot floppies, as just described.

4389.book Page 35 Tuesday, January 11, 2005 9:35 PM

Existing OS bootstrap Some distributions come with a DOS, Windows, or Mac OS program that shuts down that OS and boots up the Linux installer. These programs sometimes run automatically when you insert the Linux CD-ROM in the drive. Using them can be a good way to get started if you plan to install a dual-boot system, or if you plan to replace your current OS with Linux.

Ultimately, the boot method is unimportant, because the same installation programs run no matter what method you choose. Pick the boot method that's most convenient for your hardware and the form of installation medium you've chosen.

Installation Media

The installation medium is the physical form of the source of the Linux files. Linux is very flexible in its installation media. The most common choices are:

CD-ROM or DVD-ROM If you buy Linux in a store or from an online retailer, chances are you'll get a CD-ROM. In fact, most distributions come on multiple CD-ROMs. Some companies, such as SuSE, have begun shipping a DVD-ROM with some of their packages. (DVD-ROMs can store much more data than can CD-ROMs, so a single DVD-ROM is equivalent to multiple CD-ROMs.) CD-ROM installations tend to be quick. Most distribution maintainers offer CD-ROM image files that you can burn to CD-Rs yourself. To find CD-R image files, check `http://www.linuxiso.org`, `http://linux.tucows.com/distributions_default.html`, `ftp://sunsite.unc.edu/pub/linux/distributions`, or your chosen distribution's Web or FTP site.

Network If you have a fast network connection and don't want to be bothered with installation CD-ROMs, you can install many distributions via network connections. Download a boot floppy image, create a floppy disk from it, and boot the installer. Tell it you want to install via the network and point it to a public archive site for the distribution. This approach can also be useful if you've got a CD-ROM and a network but your target system doesn't have a CD-ROM drive. You can copy your installation CD-ROMs onto one computer on your network, configure that system to share the files, and use network installation tools to read the files over the network. The drawback to network installations is that they tend to be slower than installs from CD-ROMs. They require more information from the user, and so they can be more difficult for a new user to get working. They can also fail midway if a network connection goes down or a server stops responding. Network installations may use any of several protocols to transfer files, including FTP, HTTP (Web), SMB (Windows file sharing), and NFS (Unix/Linux file sharing). Precisely which protocols are supported varies from one distribution to another.

Hard disk It's possible to put the Linux files on a DOS or Windows partition and install Linux in another partition using those files. This approach used to be somewhat common among hobbyists who would download the files but who didn't have a CD-R burner. It's less common today but is still occasionally useful. You might use it if your CD-ROM drive doesn't seem to work in Linux, for instance; you could copy the files from the CD-ROM to the hard disk and then install from there. Because Linux treats high-capacity removable-media drives as if they were hard disks, you could also store installation files on something like a Jaz or Orb drive, which might be convenient for installing Linux on multiple systems in some environments.

4389.book Page 36 Tuesday, January 11, 2005 9:35 PM

Floppy disks Early Linux distributions came as floppy disk sets. With today's major distributions commonly exceeding 1GB compressed, floppy disks aren't a very appealing distribution medium. A few specialized distributions, however, are entirely floppy-based. Tom's Root/Boot disk (`http://www.toms.net/rb/`), for instance, is a single-floppy Linux distribution intended for emergency recovery use.

Monolithic files It's possible to distribute an entire Linux system as a single file. One example along these lines is an image file of a demo Linux CD-ROM, which can boot directly from the CD-ROM drive and run Linux without installing it on the computer. Another example is the ZipSlack distribution, which is a stripped-down version of Slackware (`http://www.slackware.com`). This distribution uses extensions to the DOS or Windows File Allocation Table (FAT) filesystem so that you can store the distribution on an ordinary FAT partition or high-capacity removable-media drive, such as a Zip or LS-120 drive. Once this is done, you can boot ZipSlack using a floppy disk.

Not all distributions support all of these installation options. All mainstream distributions support installation from CD-ROM, and most support at least one form of network installation. Beyond this, you should check the documentation for the distribution.

Even if a system lacks a CD-ROM drive, you can temporarily install a drive from another computer in order to install Linux. This is usually not the most efficient course of action if the system has a network connection, but it can be handy for installing Linux in an isolated system.

Methods of Interaction during Installation

Most methods of Linux installation require you to make decisions during the process. You may need to tell the system how to partition your hard disk, what your network settings are, and so on. To handle such interactions, distribution maintainers have developed three methods of data entry: GUI-based, text-based, and scripted. The first two are most suitable for customized individual installations, while scripts are best used when you are configuring large numbers of nearly identical systems.

GUI Installations

As a general rule, Linux distributions are shifting toward GUI installer programs. These tools run the X GUI environment in a basic 640×480 (VGA) mode that works on most modern video cards. (Some installers can run at 800×600 or higher.) The system can then use familiar mouse-based point-and-click operations to obtain input from the user. Because the display is a bitmapped graphics image, it's also possible to display graphical representations of information such as partition sizes. These displays can be very useful because people often find it easier to interpret graphs than the numbers that are more often used by text-based utilities.

GUI installations are most popular on CD-based installations. X and its related libraries are fairly large, so implementing an X-based installation over a network or floppy-based connection is tedious at best. Also, GUI installers don't work on all systems because some have unusual video

4389.book Page 37 Tuesday, January 11, 2005 9:35 PM

hardware that's incompatible with the GUI installer. This problem is particularly acute with laptop computers, whose LCD screens sometimes don't work with the video modes used by GUI installers. If you're faced with such a situation, you may need to use a text-based installer.

Text-Based Installations

A few distributions (most notably Gentoo and Slackware) don't provide GUI tools, so you *must* use a text-based installer if you want to install one of these distributions. In principle, a text-based installation works just like a GUI one. Specifically, you must enter the same types of information—Linux partition information, TCP/IP network configuration options, package selections, and so on. Text-based tools require you to select options using your keyboard, though, and they can't display graphics that are nearly as sophisticated as can a GUI installer. Some text-based programs can produce crude progress bars and the like, though, and some use text-based menus in which you tab through options to select the one you want. A few even enable you to use the mouse to select options from textual menus.

Most Linux distributions offer a text-based installation option. Typically, an early screen gives you the choice of running a GUI or text-based install, or you can type a special command to switch into a text-based mode if the default is a GUI installer. Consult your distribution's documentation if you don't see an obvious way to start a text-based installer.

Scripted Installations

With an automatic scripted installation, you typically create a configuration file that includes the information you'd normally enter with the keyboard or mouse—partition sizes, networking options, packages to install, and so on. Early in the installation process, you tell the system to read the configuration file from a floppy disk or from the network. The system then proceeds with the installation without further intervention from you.

To create the configuration file, you must normally install Linux manually on one system. The installer gives you the option of saving the configuration file. When you install Linux on the next system, you use this file to create a system that's configured identically to the first.

Scripted installations work best when you need to install Linux on many identical or nearly identical computers. If the systems vary in important details like hard disk size or intended function (workstation, server, and so on), a scripted install won't give you the opportunity to change those details on the various systems, so you'll end up spending *more* time correcting matters after the installation than you'll save by using the scripting features. You can also save your configuration options so that you can quickly reinstall a distribution on a single computer, should the need arise.

If you have many nearly identical systems to install, invest time in getting the installation just right when you create a set of installation parameters. For instance, you might want to use a custom package selection option to fine-tune what packages are installed. You'll invest time in the initial installation, but you'll save time reconfiguring all the systems after they're installed.

Not all distributions include scripted installation options. Consult your distribution's documentation for details.

4389.book Page 38 Tuesday, January 11, 2005 9:35 PM

Installing Linux

The process of installing Linux varies from one distribution to another, so you may want to consult your distribution's documentation for details. Generally speaking, though, this process guides you through several stages in which you enter critical information about your system and its desired role:

Language options Many Linux installers today support multiple languages, so you may need to select one.

Keyboard and mouse options Keyboard options sometimes appear alongside language options, because keyboards vary with language. Even within a language, keyboard options can vary, such as 101-key versus 104-key keyboards. Mouse options are sometimes presented later in the installation process, along with X configuration, but sometimes this information is gathered early to support GUI installers. You must tell the system how the mouse is connected (via a USB port, PS/2 port, and so on) and what protocol the mouse uses (most today use the PS/2 protocol or a variant of that).

Partition creation Most Linux installers give you the option of creating partitions for Linux. Sometimes the installer can automate this process, but this involves making assumptions about your system. Partition options were described earlier, in the section "Planning Disk Partitioning."

Network configuration You can usually tell Linux about your network hardware and how it should be configured as you install the OS. This topic is described in more detail in Chapter 6. If you're uncertain of how to proceed, either select no networking support or consult your local network administrator. Ideally, you'll be able to select an option to use DHCP, which will automatically configure basic network settings, but not all networks support DHCP. You may also be able to select which network servers you want to run. I recommend taking a minimalist approach; don't run any server unless you know what it does and you know that you need it. Running servers unnecessarily can be a security risk.

Time and date options Most installation procedures give you the ability to set the computer's time and date. One unusual feature of Linux is that it enables you to store the time in the hardware clock either in local time or in *Coordinated Universal Time (UTC)*, which is essentially the same as *Greenwich Mean Time (GMT)*—the time in Greenwich, England, unadjusted for daylight saving. Linux, like Unix, uses UTC internally, so setting your hardware clock to UTC is preferable on a Linux-only system. If the computer dual-boots Linux and an OS, such as Windows, that assumes the hardware clock stores the local time, you may need to pick that option.

Package selection One of the more tedious parts of some installations is picking software packages to be installed. Some distributions provide a few very high-level options, either by default or as part of a simplified process. On these distributions, you pick a package set such as "workstation" or "server," and the software installs whole clusters of packages. On other distributions, you pick packages individually. Most provide a middle ground in which you pick clusters of packages, and can sometimes fine-tune package sets.

4389.book Page 39 Tuesday, January 11, 2005 9:35 PM

X configuration If the computer is to be used as a workstation, chances are you'll want to configure X. Most desktop-oriented distributions today make this very easy; they feature tools that can detect your video card make and model, your mouse, and perhaps even your optimal screen resolution. Other times, you'll need information on your hardware at hand to enter in the X configuration screens. Your monitor's horizontal and vertical refresh rates are particularly important. You may be able to enter these by selecting your monitor from a list, but you may need to enter the information manually from your monitor's manual. Thus, you should have this manual handy when you perform the installation.

Miscellaneous hardware detection Distributions vary in precisely what hardware they detect and configure during installation. For instance, some enable you to configure a modem at system installation, but others don't. Other hardware that might or might not be detected includes the sound card, a second video card, and a printer.

Account creation Almost all distributions require you to enter a `root` password as part of the installation process. This password should be unique and hard to guess. (Chapter 3 provides pointers on picking good passwords.) Some distributions also enable you to create one or more ordinary user accounts. Generally speaking, creating at least one such account is a good idea. Some distributions support configuring Linux to use a remote account database, such as one maintained by a Windows NT domain controller, a Network Information System (NIS) server, or a Lightweight Directory Access Protocol (LDAP) server. These systems can be very handy, but you shouldn't attempt to use one unless you know you should do so.

Boot loader configuration Linux relies on a boot loader in order to boot, and boot loaders must be configured. Typically, you can select a few options from point-and-click menus, and these will work.

Most Linux installers provide text-mode or GUI menus that guide you through the process, as illustrated in Figure 1.2, which shows the package selection menu from a Fedora installation. Pick the appropriate package groups and click Next to move on to the next screen. Some text-based installers are less user-friendly, though. Gentoo requires you to enter text-mode commands at a command prompt, for instance. Gentoo makes up for this by providing very explicit installation instructions on its Web site, though.

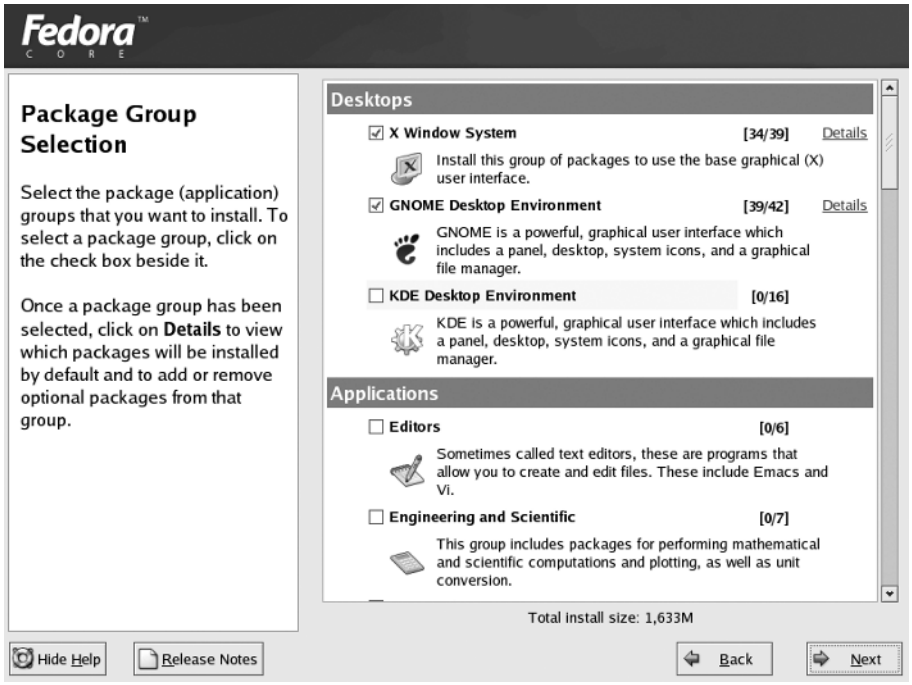
However it's done, by the time you finish the process you should have a bootable Linux system. Follow the installer through to the end and, when prompted, reboot the computer as instructed. If all goes well, Linux will boot up.

Configuring Boot Loaders

The Linux kernel is at the heart of a Linux computer; in fact, technically speaking, the kernel *is* Linux—everything else is support programs. Because the kernel must run before Linux is completely booted, the kernel must be loaded into memory in a unique way. A program known as a *boot loader* handles this task. Several boot loaders are available, some of which can boot a Linux kernel directly, and others of which require help to do the job.

4389.book Page 40 Tuesday, January 11, 2005 9:35 PM

FIGURE 1.2 Most Linux installation tools provide options with at least minimal explanations to help guide you through the process.



This section describes boot loaders for *x*86 and AMD64 systems. If you're using Linux on another architecture, such as PowerPC (Macintosh) or Alpha, the available boot loaders will be different. Consult your distribution's documentation for details.

The Role of the Boot Loader

When it's first powered up, an *x*86 CPU checks a specific area of memory for code to execute. This code is the BIOS. You're probably familiar with the BIOS through your computer's BIOS setup screens, which enable you to configure features such as RAM timing and whether or not on-board ports are active. The BIOS also provides code that allows the computer to boot. The BIOS checks the first sector of your hard disk (or of your floppy disk, CD-ROM, or other disk devices, depending on the BIOS's capabilities and configuration) for a small boot loader program. This program normally resides on the MBR of a hard disk or the boot sector of a floppy disk. The MBR resides on the first sector of a hard disk and controls the boot process. A boot sector is the first sector of a floppy or of a hard disk partition and also controls the boot process. (In the case of a partition's boot sector, it's used after the MBR.)

4389.book Page 41 Tuesday, January 11, 2005 9:35 PM

In the case of a PC that runs nothing but Windows, the boot loader in the MBR is hard-coded to check for a secondary boot loader in the active primary partition. This secondary boot loader directly loads the Windows kernel. The approach in Linux is similar, but standard Linux boot loaders are somewhat more complex. LILO and GRUB are the most common Linux boot loaders. Both programs enable you to boot the Linux kernel or to redirect the boot process to another OS. (GRUB can directly boot several non-Linux OSs, as well.)

In some cases, a system uses multiple boot loaders. One resides in the MBR, and another resides in the boot sector of an individual disk partition. (OSs on different partitions can each have their own boot sector–based boot loaders.) In this configuration, the MBR-based boot loader is the *primary boot loader*, and the one in a partition's boot sector is a *secondary boot loader*. Some boot loaders work in only one of these positions. It's often possible for a secondary boot loader to redirect the boot process to a different partition, in which case that partition's boot loader becomes the tertiary boot loader, although the configuration is the same as for secondary status.

Available Boot Loaders

Many OSs ship with their own boot loaders, and others are available from third parties. Some of the most common boot loaders are:

LILO This boot loader can directly boot a Linux kernel, and it can function as either a primary or a secondary boot loader. It may also be installed on a floppy disk, which is unusual for a boot loader. When used as a secondary boot loader, LILO should *only* be installed in a Linux partition; it will damage the contents of most non-Linux filesystems. Installing LILO in a swap partition is also inadvisable since it will be wiped out by swap activity. LILO can redirect the boot process to non-Linux partitions, and so it can be used to select Linux or Windows in a dual-boot system.

GRUB This boot loader is on the way to becoming the standard Linux boot loader. GRUB was the first boot loader that could directly boot Linux from above the 1024th cylinder of a hard disk, which gained it some popularity. LILO has since achieved similar capabilities, though. GRUB can be installed in the same locations as LILO—a floppy disk, the MBR, or the boot sector of a Linux partition.

OS Loader This is one name by which Windows NT/200*x*/XP's boot loader goes. Another is `NTLDR`. This is a secondary boot loader that cannot directly boot Linux, but it can boot a disk file that can contain LILO or GRUB, and hence boot Linux indirectly. It's common on some dual-boot installations.

System Commander This boot loader, from V Communications (`http://www.v-com.com`), is the Cadillac of boot loaders, with some very advanced features. It cannot directly boot Linux, but like many others, it can direct the boot process to a Linux partition on which LILO or GRUB is installed.

`LOADLIN` This is an unusual boot loader in that it's neither a primary nor a secondary boot loader. Rather, it's a DOS program that can be used to boot Linux after DOS has already loaded. It's particularly useful for emergency situations because it enables you to boot a Linux kernel using a DOS boot floppy, and you can also use it to pass kernel parameters to influence the booted system's behavior. `LOADLIN` comes with most Linux distributions, generally in a directory on the main installation CD-ROM.

4389.book Page 42 Tuesday, January 11, 2005 9:35 PM

After installing Linux, create a DOS boot floppy with `LOADLIN` and a copy of your Linux kernel. You can then use this boot floppy to boot Linux if LILO misbehaves or your kernel is accidentally overwritten.

Many additional third-party boot loaders are available, most of which, like System Commander, cannot directly boot a Linux kernel but can boot a partition on which LILO or GRUB is installed. For this reason, this chapter emphasizes configuring LILO and GRUB—these boot loaders can be used to boot Linux, whether they function as primary, secondary, or tertiary boot loaders. If you opt to use LILO or GRUB as a secondary boot loader, you'll need to consult the documentation for your primary boot loader to learn how to configure it.

On a Linux-only system, there's no need to deal with an advanced third-party boot loader; LILO or GRUB can function as a primary boot loader without trouble on such systems. Third-party boot loaders are most useful when you have two or more OSs installed, and particularly when LILO or GRUB has trouble redirecting the boot process to the other OSs, which is rare.

The usual configuration for LILO or GRUB is to place them in the MBR. Even in a Linux-only situation, however, it's sometimes desirable to place LILO or GRUB in the Linux boot partition. Used in this way, a standard DOS/Windows MBR will boot Linux *if* the Linux boot partition is a primary partition that's marked as active. This configuration can be particularly helpful in DOS/Linux or Windows/Linux dual-boot configurations because DOS and Windows tend to overwrite the MBR at installation. Therefore, putting LILO or GRUB in the Linux boot sector keeps it out of harm's way, and you can get LILO or GRUB working after installing or reinstalling DOS or Windows by using the DOS or Windows `FDISK` program and marking the Linux partition as active. If LILO or GRUB is on the MBR and is wiped out, you'll need to boot Linux in some other way, such as by using `LOADLIN`, and then rerun the `lilo` program to restore LILO to the MBR or rerun `grub-install` to restore GRUB to the MBR.

Configuring LILO

LILO is the traditional Linux boot loader, and some distributions still install it by default. Unlike most programs, LILO is really three things. First, it's a program you can run from within Linux, called `lilo`. This side of LILO is really an installer program; its job is to copy the boot loader code itself to the MBR, boot partition, or floppy disk. It's this boot loader that's the second side of LILO. LILO's third part is its configuration file, `/etc/lilo.conf`, which specifies the kernels and OSs LILO can boot. When you run the `lilo` utility, it modifies the code it copies to its ultimate destination to support the options you specify in this file.

An Overview of LILO Configuration

The `/etc/lilo.conf` file consists of lines with general configuration options, followed by one or more stanzas—groups of lines that define a single OS to be booted. For instance, Listing 1.1 shows a simple `lilo.conf` file that defines a system that can boot either Linux or Windows.

4389.book Page 43 Tuesday, January 11, 2005 9:35 PM

The 1024-Cylinder Limit

One bane of the PC world that reared its ugly head twice in the 1990s was the so-called *1024-cylinder limit*. This limit is derived from the fact that the *x*86 BIOS uses a three-number scheme for addressing hard disk sectors. Each sector is identified by a cylinder number, a head number, and a sector number, known collectively as the sector's *CHS address*. The problem is that each of these values is limited in size. The cylinder number, in particular, is allotted only 10 bits and so cannot exceed 2^{10}, or 1024, values. In conjunction with the limits for sectors and heads, this restricted addressable ATA hard disk size to precisely 504MB in the early 1990s.

When disks larger than 504MB became common, BIOSs were adjusted with *CHS translation* schemes, which allowed them to juggle numbers between cylinders, heads, and sectors. This increased the effective limit to just under 8GB. A similar scheme abandoned CHS addressing for BIOS-to-disk communications but retained it for BIOS-to-software communications. This was known as *linear block addressing (LBA)* mode.

These limits never affected Linux once it had booted, because Linux could handle more than 10-bit cylinder values, and it could access disks directly using LBA mode. The Linux boot process was limited, however, because LILO (this was pre-GRUB) relied on CHS addressing via the BIOS to boot the kernel. Therefore, the Linux kernel has traditionally had to reside below the 1024-cylinder mark.

Today, all new BIOSs include support for so-called *extended INT13* calls, which bypass the CHS addressing scheme. These BIOSs support booting an OS from past the 1024-cylinder mark on a hard disk, but only if the boot loader and OS support this feature. Recent versions of LILO and GRUB support extended INT13 calls, so new Linux distributions can be installed anywhere on a hard disk—if the BIOS supports this feature.

Listing 1.1: Sample `lilo.conf` File

```
boot=/dev/hda
prompt
delay=40
map=/boot/map
install=/boot/boot.b
default=linux
lba32
message=/boot/message
image=/boot/bzImage-2.6.10
        label=linux
        root=/dev/hda9
        append="mem=256M"
        read-only
```

4389.book Page 44 Tuesday, January 11, 2005 9:35 PM


```
other=/dev/hda3
        label=windows
        table=/dev/hda
```

Each line contains a command that defines some aspect of LILO's operation. The following entries describe some of the important general configuration options shown in Listing 1.1:

Boot device The `boot=/dev/hda` option tells LILO that it will install itself to `/dev/hda`—the MBR of the first physical ATA disk. To install LILO as a secondary boot loader, put it in a Linux partition, such as `/dev/hda9`.

Prompt the user The `prompt` option tells LILO to prompt the user. By default, LILO uses a simple text-mode prompt, such as `lilo:`. Many modern distributions include additional parameters that add menu-based and graphical prompts.

Boot delay LILO boots a default OS after a configurable delay, expressed in tenths of a second. The `delay=40` line sets the delay to 4 seconds.

Set default configuration The `default=linux` option specifies the default OS or kernel to boot; it refers to the `label` line in the stanza in question. If this option is omitted, LILO uses the first stanza as the default.

Enable LBA mode The `lba32` option enables the ability to boot kernels located past the 1024th cylinder of the disk (about 8GB on most modern hard drives).

Other options are present, but you're not likely to need to change them unless you need to customize how LILO appears for users or enable advanced features.

Each stanza begins with its own line—`image=` for Linux kernels or `other=` for other OSs, such as DOS or Windows. Listing 1.1 shows all but the first line of each stanza indented, but this isn't required; it simply helps distinguish the stanzas from each other. Important options for specific stanzas include the following:

OS label The `label` option sets the name by which an OS or kernel will be known. In the default LILO configuration, the user types this name at the `lilo:` prompt. If LILO is configured to use a menu, the name appears in that menu. Every stanza must have a `label` definition. Although Listing 1.1 shows labels named after the OSs in question, these labels are, in fact, arbitrary.

Linux root filesystem The `root` option sets the root (`/`) filesystem for a Linux system. Once booted, the Linux kernel looks here for startup scripts, `/etc/fstab` (for the locations of other filesystems), and so on.

Kernel options The `append` option line lets you pass parameters to the kernel. These parameters influence the way the kernel treats hardware. Listing 1.1 includes an `append` option that tells the kernel that the system has 256MB of RAM. (Linux usually detects this correctly, but some BIOSs throw Linux off.) You can also tell Linux what settings (IRQs and DMA channels) to use for hardware, if the drivers are built into the kernel.

Boot read-only Linux normally starts up by booting the root filesystem in read-only mode, and later switches it to full read-write mode. The `read-only` option tells Linux to behave in this way; it's a standard part of a Linux boot stanza.

4389.book Page 45 Tuesday, January 11, 2005 9:35 PM

Partition table The `table` option allows LILO to pass the location of the boot disk's partition table to a non-Linux OS. This is required for some OSs to boot. Its normal value is `/dev/hda` for ATA disks or `/dev/sda` for SCSI disks.

LILO is a complex program that has many additional options. Consult the `lilo.conf` man page for more information on these options.

It's important to realize that there are three aspects to LILO:

- The LILO configuration file, `/etc/lilo.conf`
- The installed boot loader, which resides in the MBR or boot sector
- The `lilo` program, which converts a `lilo.conf` file into an installed boot loader

After you've edited `/etc/lilo.conf`, you *must* type **`lilo`** as `root` to activate your changes. If you omit this step, your system will continue to use the old boot loader.

Adding a New Kernel to LILO

It's possible to configure LILO to boot either of two or more kernels using the same distribution. This can be very convenient when you want to test a new kernel. Rather than eliminate your old working configuration, you install a new kernel alongside the old one and create a new `lilo.conf` entry for the new kernel. The result is that you can select either the old kernel or the new one at boot time. If the new kernel doesn't work as expected, you can reboot and select the old kernel. This procedure allows you to avoid otherwise ugly situations should a new kernel not boot at all.

Assuming you don't need to change kernel `append` options or other features, one procedure for adding a new kernel to LILO is as follows:

1. Install the new kernel file, typically in `/boot`. Ensure that you *do not* overwrite the existing kernel file, though. If you compile your own kernel, remember to install the kernel modules (with **`make modules_install`**) as well.
2. Copy the stanza for the existing kernel file in `/etc/lilo.conf`. The result is two identical stanzas.
3. Modify the name (`label`) of one of the stanzas to reflect the new kernel name. You can use any arbitrary name you like, even a numeric one, such as `2610` for the 2.6.10 kernel.
4. Adjust the `image` line in the new kernel's stanza to point to the new kernel file.
5. If you want to make the new kernel the default, change the `default` line to point to the new kernel.
6. Save your `/etc/lilo.conf` changes.
7. Type **`lilo`** to install LILO in the MBR or boot partition's boot sector.

It's generally best to hold off on making the new kernel the default until you've tested it. If you make this change too early and then can't get around to fixing problems with the new kernel for a while, you might find yourself accidentally booting the bad kernel. This is normally a minor nuisance.

4389.book Page 46 Tuesday, January 11, 2005 9:35 PM

Once you've done this, you can reboot the computer to load the new kernel. Be sure to select the new kernel at the `lilo:` prompt, or you'll boot the old one. If everything works, you can go back to step 5 if you skipped it initially (remember to repeat steps 6 and 7 as well). If the new kernel doesn't work properly, you can reboot the computer and select the old kernel in LILO to boot it.

Adding a New OS to LILO

Adding a new OS to LILO works much as does adding a new Linux kernel. There are two basic conditions for doing this:

Multiple Linux OSs You may want to install two or more Linux OSs on one computer—say, to have a small emergency system for disaster recovery or to be able to run and test multiple distributions on one computer. When doing this, the procedure is basically the same as that for adding a new kernel, except that you must also specify the correct root partition (with the `root` parameter). In many cases, you'll need to mount your alternate Linux's root partition within the first one's filesystem and point to the alternate system's kernel on this mount point. For instance, when installing an emergency boot system and configuring it from the main Linux system, you might mount the emergency installation's root filesystem at `/emerg`, so the `image` line might read `image=/emerg/boot/bzImage-2.4.22`.

Linux and another OS LILO can boot most non-Linux OSs using the `other` line in `/etc/lilo.conf`, as shown in Listing 1.1. Model the entry for your non-Linux OS after this, pointing to the correct boot partition for the alternate OS.

In either case, once you've saved your changes, you must remember to type **`lilo`**. This action writes a new customized LILO to the MBR or Linux boot partition. If you fail to do this, you'll continue to use the old configuration the next time you boot.

Real World Scenario

Naming Kernel Files

A good practice when adding a new kernel is to give it a name that includes its version number or other identifying information. For instance, Listing 1.1's kernel is called `bzImage-2.6.10`, identifying it as a 2.6.10 kernel. If you had such a kernel and wanted to try adding a new feature (say, XFS support), you might call this new kernel `bzImage2.6.10-xfs`. There are no hard-and-fast rules for such naming, so use whatever system you like. As a general rule, though, the base of the name begins with `vmlinux` (for a "raw" kernel file), `vmlinuz` (for a kernel compressed with `gzip`), `zImage` (another name for a kernel compressed with `gzip`), or `bzImage` (for a kernel compressed in a way that supports booting larger kernel images). Most distributions use `vmlinuz` for their kernels, but locally compiled kernels usually go by the `bzImage` name.

4389.book Page 47 Tuesday, January 11, 2005 9:35 PM

Configuring GRUB

Configuring GRUB is similar to configuring LILO in many respects, although there are several important differences. Like LILO, GRUB is a collection of several components, including the boot loader code proper, a configuration file, and a set of utilities for installing and manipulating the boot loader code. Unlike LILO, the boot loader code itself can read the configuration file, so there's no need to reinstall the boot loader code whenever you change your GRUB configuration. You can even place the configuration file on a non-Linux partition, which can be handy for quickly reconfiguring GRUB from another OS.

GRUB wasn't developed exclusively for Linux. It can be installed from, and used to boot, a wide variety of OSs. Its home Web page is `http://www.gnu.org/software/grub/`.

An Overview of GRUB Configuration

The traditional location for the GRUB configuration file is `/boot/grub/menu.lst`. Fedora, Gentoo, and Red Hat, though, ship with a version of GRUB that uses `/boot/grub/grub.conf` as the configuration file. Whatever the name, the GRUB configuration file has the same basic form, as illustrated in Listing 1.2.

Listing 1.2: Sample `menu.lst` File

```
default=0
timeout=4
splashimage=(hd0,3)/grub/splash.xpm.gz
title Linux (2.6.10)
    root (hd0,3)
    kernel /bzImage-2.6.10 ro root=/dev/hda9 mem=256M
    boot
title Windows
    rootnoverify (hd0,1)
    chainloader +1
    boot
```

Because GRUB wasn't designed exclusively for Linux, it introduces a new way of referring to hard disks and their partitions. Rather than Linux device files, such as `/dev/hda` and `/dev/hda9`, GRUB uses strings of the form `(hd`*x*`,`*y*`)`, where *x* is a disk number and *y* is a partition number. (The *y* and preceding comma may be omitted to refer to an entire disk or its MBR.) Both the *x* and the *y* are numbered starting from 0, which contrasts with Linux's numbering partitions starting with 1. Thus, Linux's `/dev/hda9` is GRUB's `(hd0,8)`. GRUB doesn't distinguish between ATA and SCSI disks; `hd0` is the first disk recognized by the BIOS, `hd1` is the second disk, and so on.

4389.book Page 48 Tuesday, January 11, 2005 9:35 PM

The first three lines of Listing 1.2 set global options:

Default OS The `default=0` line tells GRUB to boot the first OS defined in the file by default. If this line read `default=1`, the default would be the second OS, and so on.

Timeout period The `timeout=4` line sets the timeout before booting the default OS to 4 seconds. (Note that LILO uses tenths of a second, but GRUB uses full seconds.)

Splash image The third line in Listing 1.2 sets a splash image—an image that's displayed as part of the boot process. Many Linux distributions ship a splash image with their GRUB files to make for a fancier boot loader menu, but you can omit this line if you like. This example uses a GRUB-style hard disk specification to point to the image file. In this case, it's the `grub/splash.xpm.gz` file on the fourth partition on the first disk (probably `/dev/hda4`). Depending on where this partition is mounted, that could be `/grub/splash.xpm.gz`, `/boot/grub/splash.xpm.gz`, or some other location.

The two OS definitions in Listing 1.2 both begin with the keyword `title`, which provides a label for the OS that's displayed by GRUB when it boots. Subsequent lines may be indented to help distinguish between the OS definitions, but this indentation is optional. Important features of OS definitions include:

Root partition The `root` option identifies the GRUB root partition, which is the partition on which the GRUB configuration files reside. If you did *not* set aside a separate partition for `/boot` when you installed Linux, this line will identify the Linux root (`/`) partition, and subsequent file references will be relative to the Linux root partition. If you used a separate `/boot` partition, though, chances are the GRUB root partition will be the Linux `/boot` partition, and GRUB references to files in Linux's `/boot` directory will omit that directory name. Listing 1.2 identifies the GRUB root partition as `(hd0,3)`, which is `/dev/hda4` on an ATA system.

GRUB can read files from several filesystems, including ext2fs, ext3fs, ReiserFS, FAT, and FFS. You can use any of these filesystems as your GRUB root partition. If you want to use another filesystem, such as the JFS or XFS, as your Linux root partition, you should split off your GRUB root partition from the Linux root partition.

Linux kernel The `kernel` option identifies a Linux kernel or a kernel for certain other Unix-like OSs, such as a GNU Hurd kernel. This reference is relative to the GRUB root partition, as defined by `root`. You can also pass kernel options on this line. (LILO uses separate lines for options.) Note that the `root` option passed to the Linux kernel identifies the Linux root partition using a Linux device filename, but the `root` option in the GRUB OS definition identifies the GRUB root partition. The two might be the same, but they might not be. In the case of Listing 1.2, they aren't the same—the GRUB root partition is `(hd0,3)`, or `/dev/hda4`, whereas the Linux root partition is `/dev/hda9`. Chances are `/dev/hda4` is the Linux `/boot` partition. The `ro` option passed on the `kernel` line tells the kernel to mount the root partition in read-only mode initially, just as the `read-only` line does in `lilo.conf`.

4389.book Page 49 Tuesday, January 11, 2005 9:35 PM

Root partition without verification The `rootnoverify` option works just like the `root` option, except that it tells GRUB it shouldn't try to access files on the partition in question. It's most often found when booting non-Linux and non-Unix OSs, such as DOS or Windows.

Specify a chain loader The `chainloader +1` line in Listing 1.2 tells the system to load the first sector of the root partition and pass execution to it. This option is common when booting DOS, Windows, or other OSs that place boot loader code in their boot sectors.

Start the boot The `boot` line tells GRUB to actually boot the kernel or boot sector for the OS in this definition. In practice, it can often be omitted.

In order to boot, the GRUB boot loader code must reside in the MBR, the boot partition's boot sector, or a floppy disk. You can do this by using the `grub` utility:

```
# grub
grub> root (hd0,3)
grub> setup (hd0)
grub> quit
```

These commands set the GRUB root partition (the same as the one defined in your `menu.lst` or `grub.conf` file), install the boot loader code to the MBR of the hard disk (that is, to `hd0`), and exit from the utility. If you want to install the boot loader to a partition, you'd use **`setup (hd0,3)`** or some other partition identifier rather than **`setup (hd0)`**. The `grub-install` program provides a simplified method of performing these steps:

```
# grub-install (hd0)
```

This command installs GRUB to the MBR of the first disk. It should be able to locate the GRUB root partition automatically.

If you installed a distribution that uses GRUB by default, you shouldn't have to perform any of these steps; GRUB should already be installed and working. You might need to reinstall GRUB from an emergency boot system if it becomes corrupted, though, and you might want to replace the installed system if you learn of a serious GRUB bug. If you just want to add a new kernel or OS to your existing GRUB installation, you do *not* need to reinstall the boot loader code; you need only edit the `menu.lst` or `grub.conf` file.

Adding a New Kernel or OS to GRUB

You can add a new kernel or OS to GRUB much as you do for LILO—by copying an existing entry (or using one in Listing 1.2 as a model) and modifying it to suit your needs. When trying a new kernel, don't replace your old kernel; instead, add the new kernel to the `/boot` directory and add a description of the new kernel to the GRUB configuration file. Remember to change the `title` line so that you can tell your two kernels apart. When you reboot the computer, you should be able to select the new kernel or OS from the list; there's no need to reinstall the GRUB boot loader code using the `grub` or `grub-install` tool.

4389.book Page 50 Tuesday, January 11, 2005 9:35 PM

Post-Installation X Configuration

Once you've installed Linux, you may need to take additional steps to get it working at even a minimally acceptable level. The item that's most likely to cause problems is X configuration. You may find that you've installed Linux but that X doesn't work correctly. You might also want to modify your X configuration to work in a way that's more to your liking, such as running in a different resolution. You'll also need to change your X configuration if you replace your video card with an incompatible model. For all of these cases, Linux provides X configuration tools, or you can manually edit the X configuration file. The first task you may need to undertake is selecting an X server; only then can you move on to configuring it.

Selecting an X Server

X is a network-enabled GUI system. It consists of an *X server*, which displays information on its local monitor and sends back user input from a keyboard and mouse; and an *X client*, which is a program that relies on the X server for user interaction. Although these two programs frequently run on the same computer, they don't need to. Chapter 6 includes additional information on using X over a network. The rest of this chapter assumes you'll be running X programs on the same system that runs the X server, but you don't install X differently if you'll be running X programs remotely.

The X server includes the driver for your video card, as well as support for your mouse and keyboard. Therefore, it's important that you know something about your video card when you install and configure your X server.

Determining Your Video Card Chipset

To properly configure X for your system, you must know what video card chipset your system uses. Unfortunately, this information isn't always obvious from the video card's box or manual because many manufacturers use other companies' chipsets, and they don't always make the chipset manufacturer obvious. You have several ways of approaching this problem, including:

Auto-detection Linux can often auto-detect the chipset, either during system installation or by running an X configuration tool after installation.

Video card documentation Although some manufacturers attempt to hide the true identity of their products' chipsets, many do not. Because of this, it's worthwhile to check the product's documentation. This documentation might not use the word "chipset," though; it could use a phrase such as "powered by" or "based on."

Windows driver report If the computer dual-boots to Windows, or if you've just bought a Windows system and intend to convert it to Linux, you can use the System tool in Windows to find out what driver (and thus, perhaps, what chipset) is installed. Double-click the System icon in the Windows Control Panel, then click the Hardware tab and the Device Manager button. (In Windows 9*x*/Me, click the Device Manager tab to achieve a similar effect.) Click the plus sign next to the Display Adapters item. This will produce a list of the

4389.book Page 51 Tuesday, January 11, 2005 9:35 PM

video cards installed in the computer. (Normally, there'll be just one.) Double-click the entry for more information; this produces the Properties dialog box for the video card, as shown in Figure 1.3. The driver and manufacturer name may be that of the video card or of the chipset.

Visual inspection You can examine your video card for signs of the chipset manufacturer. Most video cards are dominated by just one large chip. This chip may have markings identifying the manufacturer and model number, as shown in Figure 1.4. Normally, the first line or two of text contains the relevant information; the remaining lines specify the revision number, place of manufacture, and so on.

FIGURE 1.3 The Windows Properties dialog box for the video card may provide information on the video chipset manufacturer.

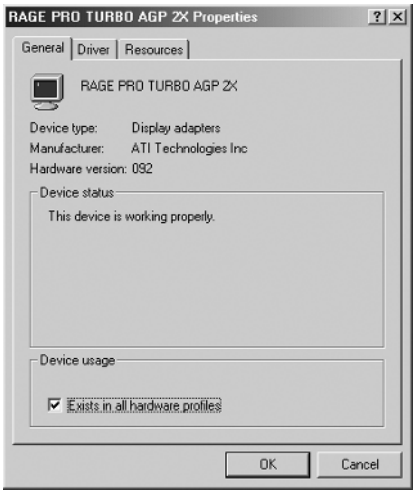
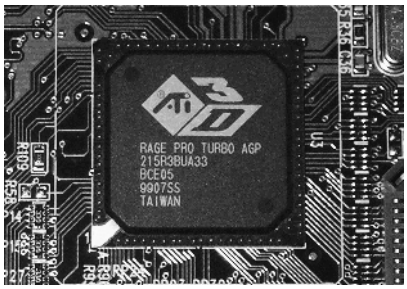


FIGURE 1.4 Markings on chips can help identify the chipset for X.



4389.book Page 52 Tuesday, January 11, 2005 9:35 PM

Increasingly, high-performance video card chipsets generate a great deal of heat, and for reliability, that heat must be dissipated by means of a heat sink—a finned metallic device that draws heat away from the chip so that it can be radiated into the surrounding air. Some boards also place a fan atop the heat sink. *Do not* attempt to remove a heat sink that's glued to a chip; doing so can damage the chip. Some manufacturers cover their chips with paper labels; these can be safely removed.

If you examine Figures 1.3 and 1.4, you'll see that they identify the chipset in the same way—as that of an ATI Rage Pro Turbo AGP. You won't always find consistency, however; sometimes a chipset may go by more than one name, or one identification method or another may reveal the board manufacturer's name rather than the chipset name. These situations need not be too troublesome, though; they just mean that you'll have to look for a driver under more than one name.

One point to keep in mind when identifying the video card chipset is that some manufacturers produce both video cards and the chipsets that go on them (ATI and Matrox both fall into this category). Other companies produce just one or the other; for instance, Trident produces chipsets, and ELSA produces video cards. Thus, if you find that the name you uncover matches your card manufacturer's name, that's not necessarily a sign that you've failed to turn up the correct chipset manufacturer.

X Server Options for Linux

All major Linux distributions ship with a free X server. In the past, a server known as XFree86 was common, but most distributions have switched to X.org-X11 instead, because of changes to the XFree86 licensing terms. These two servers are very similar, though; X.org-X11 6.7.0 was based on XFree86 4.3.99. You can learn more about XFree86 at `http://www.xfree86.org`, and X.org-X11 is headquartered at `http://www.x.org`. One particularly important subpage on the XFree86 site is `http://www.xfree86.org/current/Status.html`. This page hosts information about XFree86 compatibility with various chipsets, so it's a good place to go once you've discovered what chipset your board uses. You may find notes here on how to work around problems such as using an older or newer version of XFree86 than was shipped with your distribution.

Linux distributions from 2001 and before used XFree86 3.3.6 or earlier, but more recent distributions use XFree86 4.*x* or X.org-X11. Some major architectural modifications marked the change to XFree86 4.*x*, and some configuration files changed with this release. By the time X.org-X11 was forked off of the XFree86 project, XFree86 3.3 had become largely obsolete. Thus, I don't cover this old version of XFree86. If you encounter it or must use it because of poor support for an obscure video card in more recent X servers, though, you should be aware that some configuration options changed between XFree86 3.3.6 and 4.0.

Some video card and chipset manufacturers have made XFree86- and X.org-X11-compatible drivers available for their products. Thus, it's worth checking the Web sites maintained by your board and chipset manufacturers to see if drivers are available. This is definitely true if the main XFree86 or X.org-X11 release doesn't include appropriate drivers, and it may be true even if there

4389.book Page 53 Tuesday, January 11, 2005 9:35 PM

are drivers—a few standard drivers are not accelerated, meaning that they don't support some of the video card's features for improving the speed of drawing or moving images. If the video card manufacturer has accelerated drivers but the main XFree86 or X.org-X11 distribution ships with unaccelerated drivers, you'll see a substantial improvement in video performance by installing the accelerated drivers.

XFree86 or X.org-X11 occasionally doesn't support a device at all. You have three choices in this case:

Use the frame buffer device. The Linux kernel has some video drivers of its own. These can be accessed via the *frame buffer* XFree86 driver. For this to work, your kernel must include frame buffer support for your video chipset.

Use another X server. As X.org-X11 and XFree86 diverge, they may develop different driver strengths and weaknesses, so you might want to check the other project for drivers. In addition, a company called Xi Graphics (`http://www.xig.com`) produces a commercial X server for Linux, known as Accelerated-X. This server occasionally works on hardware that's not supported by XFree86 or X.org-X11, or produces better speed.

Replace the hardware. If you have a recalcitrant video card, the final option is to replace it. You may be able to swap with a Windows system that uses a different card, or you may need to buy a new card. Unfortunately, this isn't always an option; you can't replace the video card on a notebook computer, for instance.

Installing an X Server

Actually installing an X server is usually not very difficult; it's a matter of using your distribution's package management tools to install the software—much as you would any other software (described in Chapter 5). In most cases, this will be done during system installation, as described earlier in this chapter. You'll only have to manually install a server if you failed to install X during system installation or if you need to install a new server.

X normally comes in several packages. Only one package contains the X server proper; others provide support libraries, fonts, utilities, and so on.

One server package supports all video chipsets. The name of this package varies from one distribution to another, but it's likely to be called `XFree86`, `XFree86-server`, `xserver-xfree86`, or something similar for XFree86; or `xorg-x11` or something similar for X.org-X11. You might install it using a command similar to the following in a distribution that uses RPMs:

```
# rpm -Uvh xorg-x11-6.8.0-2.i386.rpm
```

The result is the installation of a program called `Xorg`, which is usually stored in `/usr/X11R6/bin`. This program is a generic X server. It relies on separate driver modules, which are installed along with the main package in most cases. These driver modules probably reside in `/usr/X11R6/lib/modules/drivers`.

4389.book Page 54 Tuesday, January 11, 2005 9:35 PM

If you're using an X driver provided by a video card manufacturer, follow the manufacturer's directions for installing the driver. Chances are you'll be required to copy a driver file to the X drivers directory, although the driver may come as an RPM or Debian package that will do this automatically.

If your card isn't supported by XFree86 4.*x* or X.org-X11 but it is supported by XFree86 3.3.6, you'll need to install an old XFree86 3.3.6 X server. These come in files that typically include the name of the chipset, such as `XFree86-S3-3.3.6-19.i386.rpm`. This file provides an X server for various chipsets made by S3, some of which aren't supported in more recent versions of X. If you had one of these chipsets, you could install the 3.3.6 server file, which would install an X server called `XF86_S3`. Running this server program rather than the `Xorg` executable would let you use your video card. (The upcoming section "Choosing the Server or Driver" specifies how to have the system launch a particular X server program.)

Configuring X

XFree86 is configured through the `XF86Config` file, which is usually located in `/etc` or `/etc/X11`. For XFree86 4.*x*, this file is sometimes called `XF86Config-4`. X.org-X11 calls its configuration file `xorg.conf`; it's located in the same location and has the same format. (For simplicity, I refer to both files as `xorg.conf` from now on.) Accelerated X has its own configuration file, but its format differs from that described here for XFree86 and X.org-X11. Consult the Accelerated X documentation for configuration details.

When you configure X, you provide information on the input devices (the keyboard and mouse), the video card, and the monitor. Particularly important is information on the monitor's maximum horizontal and vertical refresh rates; if this information is wrong or missing, you might not get a display. This information can be obtained from the monitor's manual.

Methods of Configuring X

XFree86 can be configured via either of two methods: by using configuration tools and by configuring manually. Configuration tools are programs that prompt you for information or obtain it directly from the hardware and then write the `xorg.conf` file, which is a standard plain-text file like other Linux configuration files. Because this file is relatively complex, it's usually wise to begin with an automatic configuration, even if it's a flawed one. Manual configuration involves opening `xorg.conf` in a text editor and changing its settings using your own know-how. You can use this method to tweak a working configuration for better performance or to correct one that's not working at all. Either way, you may need to configure X, test it, reconfigure X, test it, and so on for several iterations until you find a configuration that works correctly.

The X Configure-and-Test Cycle

If your X configuration isn't working correctly, you need to be able to modify that configuration and then test it. Many Linux distributions configure the system to start X automatically; however, starting X automatically can make it difficult to test the X configuration. To a new Linux administrator, the only obvious way to test a new configuration is to reboot the computer.

4389.book Page 55 Tuesday, January 11, 2005 9:35 PM

A better solution is to kick the system into a mode in which X is *not* started automatically. On most distributions, this goal can be achieved by typing **`telinit 3`**. This action sets the computer to use runlevel 3, in which X normally doesn't run. Chapter 6 covers runlevels in more detail, but for now, know that setting the system to a runlevel of 3 normally shuts down the X session that launched automatically at system startup.

Debian and Gentoo don't use runlevels as a signal for whether or not to start X. With these distributions, you must shut down the GUI login server by typing **`/etc/init.d/xdm stop`**. (You may need to change `xdm` to `gdm` or `kdm`, depending on your configuration.)

Once the X session is shut down, you can log in using a text-mode login prompt and tweak your X settings manually, or you can use text-based X configuration programs, as described shortly. You can then type **`startx`** to start the X server again. If you get the desired results, quit from X and type **`telinit 5`** (**`/etc/init.d/xdm start`** in Debian or Gentoo) to restore the system to its normal X login screen. If after typing **`startx`** you don't get the results you want, you can try modifying the system some more.

If X is working minimally but you want to modify it using X-based configuration tools, you can do so after typing **`startx`** to get a normal X session running. Alternatively, you can reconfigure the system before taking it out of the X-enabled runlevel.

Another approach to restarting X is to leave the system in its X-enabled runlevel and then kill the X server. The Ctrl+Alt+Backspace keystroke does this on many systems, or you can do it manually with the `kill` command, after finding the appropriate process ID with the `ps` command, as shown here:

```
# ps ax | grep X
1375 ?   S  6:32 /etc/X11/X -auth /etc/X11/xdm/authdir/
# kill 1375
```

This approach works better on systems that don't map the running of X to specific runlevels, such as Debian and its derivatives.

X Configuration Tools

Several utilities can help in X configuration, although not all distributions ship with all of them:

The X server The `XFree86` or `Xorg` server itself includes the capacity to query the hardware and produce a configuration file. To do so, type **`XFree86 -configure`** or **`Xorg -configure`** when no X server is running. The result should be a file called `/root/XF86Config.new` or `/root/xorg.conf.new`. This file might not produce optimal results, but it is at least a starting point for manual modifications.

`Xconfigurator` A program called `Xconfigurator` can produce and modify the X configuration file format. Red Hat (`Xconfigurator`'s developer) has abandoned this tool in favor of GUI utilities, though.

4389.book Page 56 Tuesday, January 11, 2005 9:35 PM

Distribution-specific tools Many modern distributions ship with their own custom X configuration tools. These include Red Hat's (and Fedora's) Display Settings tool (accessible from the default desktop menu or by typing **`system-config-xfree86`** in an xterm) and SuSE's YaST and YaST2. These tools frequently resemble the distribution's install-time X configuration tools, which can vary substantially.

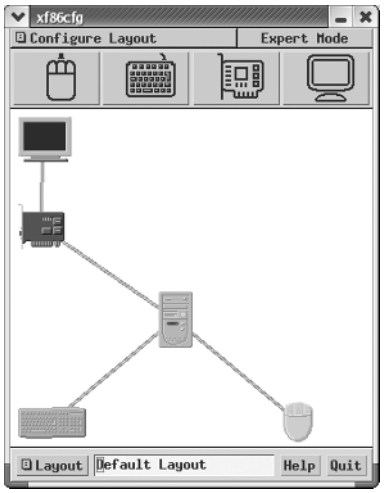
`xf86cfg` This program is another that works only once X is already running. Its user interface (shown in Figure 1.5), like that of `XF86Setup`, enables you to jump around to configure different elements in whatever order you like. In `xf86cfg`, you right-click an icon and select the Configure option to configure the element, or you can select other options (Remove, Disable, and so on) to perform other actions.

Manually Editing the *xorg.conf* File

The `xorg.conf` file consists of a number of labeled sections, each of which begins with the keyword `Section`, followed by the section name in quotes, and ends with the keyword `EndSection`. Between these two lines are lines that define features relevant to the configuration of that feature. There may also be comments, which are lines that begin with hash marks (#). For instance, here's a section that defines where the computer can find certain critical files:

```
Section "Files"
    RgbPath   "/usr/X11R6/lib/X11/rgb"
    # Multiple FontPath entries are allowed
    FontPath  "/usr/X11R6/lib/X11/fonts/75dpi"
    FontPath  "/usr/X11R6/lib/X11/fonts/Type1"
EndSection
```

FIGURE 1.5 The `xf86cfg` program lets you configure X using point-and-click operations.



4389.book Page 57 Tuesday, January 11, 2005 9:35 PM

The pages that follow tell you what sections and critical options within these sections exist to modify X's operation. You should then be able to edit the `xorg.conf` file directly or use a configuration utility to do the job. (The configuration utilities tend to use terminology that's similar to that used in the configuration file, so figuring out what to change with a utility isn't difficult if you know for what option you're looking.)

If you have a working configuration, be sure to back up `xorg.conf` before modifying it. If you mistakenly delete or modify some critical line, you can easily end up with a system that won't start X at all, and without a backup or a perfect memory of what you changed, it can be difficult to restore even a partially functioning system.

Setting Miscellaneous Options

Some sections of the `xorg.conf` file relate to miscellaneous options or those that require just a handful of lines to set properly. (The big video sections often boast dozens of lines of configuration options.) Nonetheless, getting these settings right is important to a functioning X system.

Configuring Paths

The `Files` section hosts information on the locations of important files. The entries you're most likely to change relate to the locations of X's fonts. These are handled through the `FontPath` option line. Examples of the use of this line include the following:

```
FontPath   "/usr/share/fonts/Type1"
FontPath   "unix/:-1"
FontPath   "tcp/fontserver.example.com:7101"
```

The first of these lines indicates a directory in which fonts may be found. The second refers to a *font server* that runs locally, and is not accessible to other systems. The final line points to a font server that runs on another computer (*`fontserver.example.com`*) on port 7101. A font server is a program that delivers fonts to local or remote computers. Some Linux distributions use font servers for local font handling, and networks sometimes use them to reduce the effort of administering fonts. You don't need to use a font server if you don't want to, but if your distribution uses a local font server by default, you should leave its reference intact in `xorg.conf`. A single `xorg.conf` file can have multiple `FontPath` lines; X searches for fonts in each of the specified locations in order.

An important difference between XFree86 and X.org-X11 is in their default font directories. X.org-X11 uses subdirectories of `/usr/share/fonts`, whereas XFree86 uses subdirectories of `/usr/X11R6/lib/X11/fonts`.

4389.book Page 58 Tuesday, January 11, 2005 9:35 PM

Configuring the Keyboard

The `Keyboard` input device section defines the operation of the keyboard in XFree86. In most cases, there's little need to modify most `xorg.conf` keyboard settings, which typically look like this:

```
Section "InputDevice"
  Driver       "Keyboard"
  Identifier   "Keyboard[0]"
  Option       "MapName" "Generic keyboard [ pc101 ]"
  Option       "Protocol" "Standard"
  Option       "XkbLayout" "us"
  Option       "XkbModel" "pc101"
  Option       "XkbRules" "xfree86"
  Option       "AutoRepeat" 500 200
EndSection
```

One setting that you might want to change, however, is the `AutoRepeat` line. (This line may not even be present on a default installation, but you can add it if you like.) When you press and hold a key, the system begins repeating it, as if you were repeatedly pressing the key. This line controls the rate at which keys repeat when running X.

The first number on this line (`500` in the preceding example) is the time in milliseconds (ms), thousandths of a second, before the system begins repeating a key, and the second number (`200` in the preceding example) is the interval between repeats once they begin. For instance, in the preceding example, the system waits 500ms after the key is first pressed, and thereafter produces another character every 200ms (five per second) until you release the key.

Users can override the default keyboard repeat rate by setting this option using a desktop environment's control utilities or various other programs.

In some cases, you might also want to adjust the `XkbModel` and `XkbLayout` lines. These lines set the keyboard model and layout. The model relates to the number of keys and their placement, and the layout determines what character each key produces. The layout is used to specify keyboards for different nationalities or languages, which often contain slightly different key selections.

Configuring the Mouse

A second `InputDevice` section defines the mouse. This section is typically quite short, as shown here:

```
Section "InputDevice"
    Identifier "Mouse1"
    Driver      "mouse"
    Option "Protocol"    "PS/2"
    Option "Device"      "/dev/psaux"
    Option "Emulate3Buttons"
    Option "Emulate3Timeout"    "50"
EndSection
```

4389.book Page 59 Tuesday, January 11, 2005 9:35 PM

Chances are you won't need to modify the `Identifier` or `Driver` options. The `Protocol` is the software protocol used by mice. It's often `PS/2`, but it may be something else (such as `Microsoft` or `Logitech`), particularly for older serial mice. Scroll mice frequently set `Protocol` to `IMPS/2`, which is the Microsoft IntelliMouse PS/2 protocol variant. The `Device` option points to the Linux device file with which the mouse is associated. This is sometimes `/dev/mouse`, which is a symbolic link to the real device file, such as `/dev/psaux` (for PS/2 mice), `/dev/usb/usbmouse` (for USB mice), or `/dev/ttyS0` or `/dev/ttyS1` (for serial mice). `Emulate3Buttons` tells X to treat simultaneous presses of the two buttons of a two-button mouse as if they were the third button, and `Emulate3Timeout` tells the system how close (in milliseconds) those two presses must be. If the system has a three-button mouse to begin with, these options should be commented out or deleted.

X programs frequently use the middle button; for instance, text editors use it for pasting text. Therefore, any Linux workstation should be equipped with a genuine three-button mouse rather than a two-button device. Scroll wheels on mice that are so equipped can usually function as a middle button, as well as handling wheel duty. Although the `Emulate3Buttons` option enables you to use a two-button mouse in Linux, doing so is awkward.

Setting Monitor Options

Some of the trickiest aspects of X configuration relate to the monitor options. You set these in the `Monitor` section, which has a tendency to be quite large, particularly in XFree86 3.3.6. A shortened `Monitor` section looks like this:

```
Section "Monitor"
    Identifier "Iiyama"
    ModelName  "VisionMaster Pro 450"
    HorizSync  27.0-115.0
    VertRefresh 50.0-160.0
    # My custom 1360x1024 mode
    Modeline "1360x1024" 197.8 \
              1360 1370 1480 1752 \
              1024 1031 1046 1072 -HSync -VSync
EndSection
```

The `Identifier` option is a free-form string that contains information that's used to identify a monitor in a later section. This later section links together various components of the configuration. `Identifier` can be just about anything you like. Likewise, the `ModelName` option also can be anything you like; it's used mainly for your own edification when reviewing the configuration file.

As you continue down the section, you'll see the `HorizSync` and `VertRefresh` lines, which are extremely critical; they define the range of horizontal and vertical refresh rates that the monitor

4389.book Page 60 Tuesday, January 11, 2005 9:35 PM

can accept, in kilohertz (kHz) and hertz (Hz), respectively. Together, these values determine the maximum resolution and refresh rate of the monitor. Despite the name, the `HorizSync` item alone doesn't determine the maximum horizontal refresh rate. Rather, this value, the `VertRefresh` value, and the resolution determine the monitor's maximum refresh rate. X selects the maximum refresh rate that the monitor will support, given the resolution you specify in other sections. Some X configuration utilities show a list of monitor models or resolution and refresh rate combinations (such as "800 × 600 at 72 Hz") to obtain this information. This approach is often simpler to handle, but it's less precise than entering the exact horizontal and vertical sync values.

Don't set random horizontal and vertical refresh rates; particularly on older hardware, setting these values too high can actually damage a monitor. (Modern monitors ignore signals presented at too high a refresh rate.)

To settle on a resolution, X looks through a series of *mode lines*, which are specified via the `Modeline` option. Computing mode lines is tricky, so I don't recommend you try it unless you're skilled in such matters. The mode lines define combinations of horizontal and vertical timing that can produce a given resolution and refresh rate. For instance, a particular mode line might define a 1024 × 768 display at a 90Hz refresh rate, and another might represent 1024 × 768 at 72Hz.

Some mode lines represent video modes that are outside the horizontal or vertical sync ranges of a monitor. X can compute these cases and discard the video modes that a monitor can't support. If asked to produce a given resolution, X searches all the mode lines that accomplish the job, discards those that the monitor can't handle, and uses the remaining mode line that creates the highest refresh rate at that resolution. (If no mode line supports the requested resolution, X drops down to another specified resolution, as described shortly, and tries again.)

As a result of this arrangement, you'll see a large number of `Modeline` entries in the `XF86Config` file for XFree86 3.3.*x*. Most end up going unused because they're for resolutions you don't use or because your monitor can't support them. You can delete these unused mode lines, but it's usually not worth the bother.

XFree86 4.*x* and X.org-X11 support a feature known as *Data Display Channel (DDC)*. This is a protocol that enables monitors to communicate their maximum horizontal and vertical refresh rates and appropriate mode lines to the computer. The `XFree86 -configure` or `Xorg -configure` command uses this information to generate mode lines, and on every start, the system can obtain horizontal and vertical refresh rates. The end result is that an XFree86 4.*x* or X.org-X11 system can have a substantially shorter `Monitor` section than is typical with XFree86 3.3.*x*.

Setting Video Card Options

Your monitor is usually the most important factor in determining your maximum refresh rate at any given resolution, but X sends data to the monitor only indirectly, through the video card. Because of this, it's important that you be able to configure this component correctly. An incorrect configuration of the video card is likely to result in an inability to start X.

4389.book Page 61 Tuesday, January 11, 2005 9:35 PM

Choosing the Server or Driver

XFree86 4.*x* and X.org-X11 use driver modules that are stored in separate files from the main X server executable. The server can't determine what module is required automatically, however. Instead, you must give it that information in the `xorg.conf` file. In particular, the driver module is set by a line in the `Device` section, which resembles the following:

```
Driver "ati"
```

This line sets the name of the driver. The drivers reside in the `/usr/X11R6/lib/modules/drivers/` directory. Most of the drivers' filenames end in `_drv.o`, and if you remove this portion, you're left with the driver name. For instance, `ati_drv.o` corresponds to the `ati` driver.

The `xf86cfg` utility provides a large list of chipsets and specific video card models, so you can select the chipset or board from this list to have the utility configure this detail.

Setting Card-Specific Options

The `Device` section of the `xorg.conf` file sets various options related to specific X servers. A typical `Device` section resembles the following:

```
Section "Device"
    Identifier  "ATI Mach64"
    VendorName  "ATI"
    BoardName   "Xpert 98"
    Driver      "ati"
    VideoRam    8192
EndSection
```

The `Identifier` line provides a name that's used in the subsequent `Screen` section to identify this particular `Device` section. (`xorg.conf` files frequently host multiple `Device` sections—for instance, one for a bare-bones VGA driver and one for an accelerated driver.) The `VendorName` and `BoardName` lines provide information that's useful mainly to people reading the file.

The `VideoRam` line is unnecessary with many servers and drivers because the driver can detect the amount of RAM installed in the card. With some devices, however, you may need to specify the amount of RAM installed in the card, in kilobytes. For instance, the preceding example indicates a card with 8MB of RAM installed.

Many drivers support additional driver-specific options. They may enable support for features such as hardware cursors (special hardware that enables the card to handle mouse pointers more easily) or caches (using spare memory to speed up various operations). Consult the `xorg.conf` man page or other driver-specific documentation for details.

4389.book Page 62 Tuesday, January 11, 2005 9:35 PM

Setting Screen Options

The `Screen` section ties together the other sections. A short example is:

```
Section "Screen"
    Identifier "screen1"
    Device     "ATI Mach64"
    Monitor    "Iiyama"
    DefaultDepth 16
    Subsection "Display"
        Depth       8
        Modes       "1280x1024" "1024x768" "640x400"
    EndSubsection
    Subsection "Display"
        Depth       16
        Modes       "1024x768" "800x600" "640x480"
        Virtual     1280 1024
        ViewPort    0 0
    EndSubsection
EndSection
```

Several key points in this section should be emphasized:

- The `Identifier` specifies an overall configuration. A configuration file can hold multiple `Screen` sections, as described shortly.
- The `Device` and `Monitor` lines point to specific `Device` and `Monitor` sections, respectively.
- The `DefaultDepth` line specifies the number of bits per pixel to be used by default. For instance, the preceding example sets this value to `16`, so a 16-bit color depth is used, resulting in 2^{16}, or 65,536, possible colors.
- Each `Subsection` defines a particular display type. They have associated color depths (specified by the `Depth` line) and a series of resolutions (specified by the `Modes` line). The system tries each resolution specified by the `Modes` line in turn, until it finds one that works. There are also various optional parameters, such as `Virtual` (which defines a virtual screen that can be larger than the one that's actually displayed) and `ViewPort` (a point within that virtual display at which the initial display is started).

One final section is required: the `ServerLayout` section. This section consists of lines that identify the default `Screen` section and link it to mouse and keyboard definitions. For instance, a typical configuration will include a `ServerLayout` section resembling the following:

```
Section "ServerLayout"
    Identifier  "layout1"
    Screen      "screen1"
    InputDevice "Mouse1" "CorePointer"
    InputDevice "Keyboard1" "CoreKeyboard"
EndSection
```

4389.book Page 63 Tuesday, January 11, 2005 9:35 PM

Although I describe the `ServerLayout` section last because it ties together all the other sections, it can appear earlier in the file—perhaps even first. The order of sections in the `xorg.conf` file is arbitrary.

Normally, an `xorg.conf` file will have just one `ServerLayout` section, but by passing the `-layout` parameter to the server program, you can tell the server to use a different `ServerLayout` section, if one is present. You might use this to start X using a different mouse, for instance—say, a USB mouse on a notebook rather than the built-in PS/2 touch pad.

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 determining which Linux distribution to use, and continues with planning what software packages to install.

Actually installing Linux begins with planning the disk partitioning. Typically, you can perform the partitioning during the install process, but you should have an idea of how to proceed before you begin. You can then select installation media and perform the installation. Most distributions guide you through this process.

After installing Linux, you may need to attend to certain details. One of these is boot loader configuration. Although the installer usually gets this detail correct, particularly for single-OS systems, you may want to tweak the settings or add other OSs to the boot loader. You'll also need to understand this process when you install a new kernel down the road. Another common post-installation configuration detail is getting X working. Again, Linux distributions usually configure X correctly during installation, but you may need to tweak the settings or change them at a later date.

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.

Summarize some common workstation and server software. Workstation software includes word processors, spreadsheets, mail readers, Web browsers, graphics editors, and other programs used by individuals on the local system. Server software includes Web servers,

4389.book Page 64 Tuesday, January 11, 2005 9:35 PM

mail servers, file servers, time servers, news servers, login servers and other programs that are often accessed remotely.

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.

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 when it's most appropriate to use CD-ROM and network installations. CD-ROM installations are most convenient when installing to systems with poor network connectivity or when you have a CD-ROM and want to install quickly. Network installations are convenient when you are installing several systems simultaneously or when you don't have a Linux CD-ROM or a CD-ROM drive on the target system.

Ascertain what type of interaction is most appropriate during installation. GUI- and text-based installations are good for when you are installing a single system or when you are preparing a template for scripted installations with some distributions. Automatic scripted installations are convenient when you are installing nearly identical systems on multiple computers.

Describe why you might pick particular filesystems for Linux installation. Ext3fs is a popular choice, and generally a good one. ReiserFS, XFS, and JFS are also good choices on distributions that support them, but many don't. The older ext2fs can be a good choice for small partitions, but is better avoided for large partitions.

Determine what video chipset your system uses. Many manufacturers document the video card chipset in their manuals or on the product boxes. You can also check the Microsoft Windows System Control Panel or visually inspect the board, if the manufacturer did not make the information readily available.

Summarize how X determines the monitor's refresh rate. X uses the monitor's maximum horizontal and vertical refresh rates and a series of fixed mode lines, which define particular timings for various video resolutions. X picks the mode line that produces the highest refresh rate supported by the monitor at the specified resolution.

4389.book Page 65 Tuesday, January 11, 2005 9:35 PM

Commands in This Chapter

Command	Description
`Xconfigurator`	Text- or GUI-based XFree86 3.3.x and 4.0.x configuration program
`Xorg`	X.org-X11 server that can automatically produce its own configuration file
`XFree86`	XFree86 4.0.x server that can automatically produce its own configuration file
`xf86cfg`	GUI-based XFree86 4.0.*x* and X.org-X11 configuration program

4389.book Page 66 Tuesday, January 11, 2005 9:35 PM

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. 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

4. 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.

5. 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 server.
   - **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.

4389.book Page 67 Tuesday, January 11, 2005 9:35 PM

6. 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.

7. 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.

8. 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.

9. 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.

4389.book Page 68 Tuesday, January 11, 2005 9:35 PM

10. 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

11. 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

12. 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

13. Which of the following best describes a typical Linux distribution's method of installation?

A. The installation program is a small Linux system that boots from floppy, CD-ROM, or hard disk to install a larger system on the hard disk.

B. The installation program is a set of DOS scripts that copies files to the hard disk, followed by a conversion program that turns the target partition into a Linux partition.

C. The installation program boots only from a network boot server to enable installation from CD-ROM or network connections.

D. The installation program runs under the Minix OS, which is small enough to fit on a floppy disk but can copy data to a Linux partition.

14. Which of the following is an advantage of a GUI installation over a text-based installation?

A. GUI installers support more hardware than do their text-based counterparts.

B. GUI installers can provide graphical representations of partition sizes, package browsers, and so on.

C. GUI installers can work even on video cards that support only VGA graphics.

D. GUI installers better test the system's hardware during the installation.

15. Which of the following tools may you use when creating partitions for Linux? (Choose all that apply.)

A. Linux's `fdisk` from an emergency disk, run prior to the system installation

B. PowerQuest's PartitionMagic or similar third-party utilities

C. Distribution-specific install-time utility

D. The DOS `FORMAT` utility, run prior to the system installation

4389.book Page 69 Tuesday, January 11, 2005 9:35 PM

16. What mount point should you associate with swap partitions?
    - **A.** `/`
    - **B.** `/swap`
    - **C.** `/boot`
    - **D.** None

17. Which of the following is the *most* useful information in locating an X driver for a video card?
    - **A.** The interrupt used by the video card under Microsoft Windows
    - **B.** Markings on the video card's main chip
    - **C.** Whether the card uses the ISA, VLB, PCI, or AGP bus
    - **D.** The name of the video card's manufacturer

18. When you configure an X server, you need to make changes to configuration files and then start or restart the X server. Which of the following can help streamline this process?
    - **A.** Shut down X by switching to a runlevel in which X doesn't run automatically, then reconfigure it and use `startx` to test X startup.
    - **B.** Shut down X by booting into single-user mode, then reconfigure X and use `telinit` to start X running again.
    - **C.** Reconfigure X, then unplug the computer to avoid the lengthy shutdown process before restarting the system, and X along with it.
    - **D.** Use the `startx` utility to check the X configuration file for errors before restarting the X server.

19. Which of the following summarizes the organization of the `xorg.conf` file?
    - **A.** The file contains multiple sections, one for each screen. Each section includes subsections for individual components (keyboard, video card, and so on).
    - **B.** Configuration options are entered in any order desired. Options relating to specific components (keyboard, video card, and so on) may be interspersed.
    - **C.** The file begins with a summary of individual screens. Configuration options are preceded by a code word indicating the screen to which they apply.
    - **D.** The file is broken into sections, one or more for each component (keyboard, video card, and so on). The end of the file has one or more sections that define how to combine the main sections.

20. In what section of `XF86Config` do you specify the resolution that you want to run?
    - **A.** In the `Screen` section, subsection `Display`, using the `Modes` option
    - **B.** In the `Monitor` section, using the `Modeline` option
    - **C.** In the `Device` section, using the `Modeline` option
    - **D.** In the `DefaultResolution` section, using the `Define` option

4389.book Page 70 Tuesday, January 11, 2005 9:35 PM

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. 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.

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

5. 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.

6. 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.

7. 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.

8. 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.

9. 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.

4389.book Page 71 Tuesday, January 11, 2005 9:35 PM

10. 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.

11. 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.

12. 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.

13. A. Most Linux distributions use installation programs written in Linux, not in DOS or Minix. The system usually boots from floppy or CD-ROM, although other boot media (such as hard disk or even network) are possible.

14. B. A bitmapped display, as used by a GUI installer, can be used to show graphical representations of the system's state that can't be done in a text-mode display. Text-based installers actually have an edge in hardware support because they can run on video cards that aren't supported by X.

15. A, B, C. You can usually define partitions using just about any tool that can create them, although with some tools (such as DOS's `FDISK`), you may need to change the partition type code using Linux tools. The DOS `FORMAT` utility is used to create a FAT filesystem, not define a partition.

16. D. Swap partitions aren't mounted in the way filesystems are, so they have no associated mount points.

17. B. Markings on the video card's main chip typically include a name or number for the chipset; this is what you need in order to locate an X driver for the card. The video card's manufacturer name might or might not be useful information. If it proves to be useful, you'd also need a model number. The interrupt used by the video card in Windows is irrelevant. The card's bus can narrow the range of possibilities, but it isn't extremely helpful.

18. A. On most Linux systems, some runlevels don't run X by default, so using one of them along with the `startx` program (which starts X running) can be an effective way to quickly test changes to an X configuration. The `telinit` program changes runlevels, which is a lengthy process compared to using `startx`. Unplugging the computer to avoid the shutdown process is self-defeating since you'll have to suffer through a long startup (if you use a non-journaling filesystem), and it can also result in data loss. The `startx` utility doesn't check the veracity of an X configuration file; it starts X running from a text-mode login.

19. D. The `xorg.conf` file design enables you to define variants or multiple components and easily combine or recombine them as necessary.

4389.book Page 72 Tuesday, January 11, 2005 9:35 PM

20. A. The `Modeline` option in the Monitor section defines *one* possible resolution, but there are usually several `Modeline` entries defining many resolutions. The `Modeline` option doesn't exist in the `Device` section, however, nor is that section where the resolution is set. There is no `DefaultResolution` section.