Andrew Hudson
Paul Hudson

DVD Includes
Ubuntu 7.10
"Gutsy Gibbon"

Ubuntu 7.10 Linux®

UNLEASHED

SAMS

Andrew Hudson
Paul Hudson

Ubuntu 7.10 Linux®

UNLEASHED

SAMS | 800 East 96th Street, Indianapolis, Indiana 46240 USA

Ubuntu 7.10 Linux® Unleashed

ISBN-10: 0-672-32969-7
ISBN-13: 978-0-672-32969-2

Library of Congress Cataloging-in-Publication data is on file.

Printed in the United States of America

First Printing: December 2007

Trademarks

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Sams Publishing cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Warning and Disclaimer

Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an "as is" basis. The author(s) and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the DVD or programs accompanying it.

Bulk Sales

Sams Publishing offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales. For more information, please contact

U.S. Corporate and Government Sales
1-800-382-3419
corpsales@pearsontechgroup.com

For sales outside of the U.S., please contact

International Sales
international@pearsoned.com

Acquisitions Editor
Mark Taber

Development Editor
Michael Thurston

Managing Editor
Patrick Kanouse

Project Editor
Seth Kerney

Proofreader
Water Crest Publishing

Technical Editor
Dallas Releford

Publishing Coordinator
Vanessa Evans

Multimedia Developer
Dan Scherf

Book Designer
Gary Adair

Contents at a Glance

Table of Contents

About the Authors

Andrew Hudson is a freelance journalist who specializes in writing about Linux. He has significant experience in Red Hat and Debian-based Linux distributions and deployments and can often be found sitting at his keyboard tweaking various settings and config files just for the hell of it. He lives in Wiltshire, which is a county of England, along with his wife, Bernice, and their son, John. Andrew does not like Emacs. He can be reached at andy.hudson@gmail.com.

Paul Hudson is a recognized expert in open-source technologies. He is also a professional developer and full-time journalist for Future Publishing. His articles have appeared in *Mac Format*, *PC Answers*, *PC Format*, *PC Plus*, and *Linux Format*. Paul is passionate about free software in all its forms and uses a mix of Linux and BSD to power his desktops and servers. Paul likes Emacs. Paul can be contacted through http://hudzilla.org.

Dedication

To Bernice and John—the best supporters a man could ever wish for.

—Andrew Hudson

To World Peace—because this is about as close as I'm ever going to get to being in the Miss World competition.

—Paul Hudson

Acknowledgments

Thanks to our colleagues at Sams Publishing, whose patience and persistence made this book possible.

Thanks also to my family who have supported me during the writing of this book. My son John now has his own keyboard and mouse—the computer will come in a few years!

My wife Bernice has the patience of a saint, allowing me to lock myself away when I needed to, and being helpful when I've hit writer's block (and yes, it does happen!)

Finally, thanks to God who makes all things possible, including this book.

—Andrew Hudson

Thanks to Andrew, Shelley, Damon, Seth, Dallas, Mum and Dad, my wife, Ildiko; and, of course, God, who made all this possible. No book this big could be done without a lot of work from a dedicated team!

—Paul Hudson

We Want to Hear from You!

As the reader of this book, *you* are our most important critic and commentator. We value your opinion and want to know what we're doing right, what we could do better, what areas you'd like to see us publish in, and any other words of wisdom you're willing to pass our way.

You can email or write me directly to let me know what you did or didn't like about this book—as well as what we can do to make our books stronger.

Please note that I cannot help you with technical problems related to the topic of this book, and that due to the high volume of mail I receive, I might not be able to reply to every message.

When you write, please be sure to include this book's title and author as well as your name and phone number or email address. I will carefully review your comments and share them with the author and editors who worked on the book.

Email: opensource@samspublishing.com

Mail: Mark Taber
Associate Publisher
Sams Publishing
800 East 96th Street
Indianapolis, IN 46240 USA

Reader Services

Visit our website and register this book at www.samspublishing.com/register for convenient access to any updates, downloads, or errata that might be available for this book.

This page intentionally left blank

Introduction

Welcome to *Ubuntu 7.10 Linux Unleashed*! This book covers the free Linux distribution named Ubuntu and includes a fully functional and complete operating system produced by the Ubuntu Community, sponsored by Canonical Software.

Ubuntu directly descends from one of the oldest and most revered Linux distributions ever: Debian. Those of you who know nothing about Linux will likely not have heard of Debian; it is enough to know that it is considered to be one of the most stable and secure Linux distributions currently available. Ubuntu benefits directly from many contributions from free software developers across the world.

If you are new to Linux, you have made a great decision by choosing this book. Sams Publishing's *Unleashed* books offer an in-depth look at their subject, taking in both beginner and advanced users and moving them to a new level of knowledge and expertise. Ubuntu is a fast-changing distribution that can be updated at least twice a year. We have tracked the development of Ubuntu from early on to make sure that the information in this book mirrors closely the development of the distribution. A full copy of Ubuntu is included on the enclosed disc, making it possible for you to install Linux in less than an hour! No longer an upstart, Linux now has an enviable position in today's modern computing world. It can be found on machines as diverse as mobile phones and wristwatches, all the way up to supercomputers—in fact, Linux currently runs on more than half of the world's top 500 supercomputers.

Do not let the reputation of Linux discourage you, however. Most people who have heard of Linux think that it is found only on servers, looking after websites and email. Nothing could be further from the truth because Linux is making huge inroads in to the desktop market, too. Corporations are realizing the benefits of running a stable and powerful operating system that is easy to maintain and easy to secure. Add to that the hundreds of improvements in usability, and Linux becomes an attractive proposition that tempts many CIOs. The best part is that as large Linux vendors improve Linux, the majority of those improvements make it into freely available distributions, allowing you to benefit from the additions and refinements made. You can put Ubuntu to work today and be assured of a great user experience.

This book provides all the information that you need to get up and running with Ubuntu. It even tells you how to keep Ubuntu running in top shape and how to adapt Ubuntu to changes in your own needs. You can use Ubuntu at home, in the workplace, or, with permission, at your school or college. In fact, you might want to poke around your school's computer rooms: You will probably find that someone has already beaten you to the punch—Linux is commonly found in academic institutions. Feel free to make as many copies of the software as you want; because Ubuntu is freely distributable all over the world, no copyright lawyers are going to pound on your door.

After an introduction to Linux and Ubuntu, you will find out how to get started with Ubuntu, including installation and initial configuration. We also take you through installing software, managing users, and other common administrative tasks. For the more technically minded, we also cover some starting steps in programming across several languages—why not pick one and try it out? Throughout this book, you will also find information about multimedia applications, digital graphics, and even gaming (for after-hours when you are finished tinkering). After you make it through this book, you will be well equipped with the knowledge needed to use Linux successfully. We do assume that you are at least familiar with an operating system already (even if it is not with Linux) and have some basic computer knowledge.

Licensing

Software licensing is an important issue for all computer users and can entail moral, legal, and financial considerations. Many consumers think that purchasing a copy of a commercial or proprietary operating system, productivity application, utility, or game conveys ownership, but this is not true. In the majority of cases, the *end user license agreement (EULA)* included with a commercial software package states that you have paid only for the right to use the software according to specific terms. This generally means you may not examine, make copies, share, resell, or transfer ownership of the software package. More onerous software licenses enforce terms that preclude you from distributing or publishing comparative performance reviews of the software. Even more insidious licensing schemes (and supporting legislation, especially in the United States) contain provisions allowing onsite auditing of the software's use!

This is not the case with the software included with this book. You are entirely free to make copies, share them with friends, and install the software on as many computers as you want—we encourage you to purchase additional copies of this book to give as gifts, however. Be sure to read the README file on the disc included with this book for important information regarding the included software and disk contents. After you install Ubuntu, go to http://www.gnu.org/licenses/gpl.html to find a copy of the GNU GPL. You will see that the GPL provides unrestricted freedom to use, duplicate, share, study, modify, improve, and even sell the software.

You can put your copy of Ubuntu to work right away in your home or at your place of business without worrying about software licensing, per-seat workstation or client licenses, software auditing, royalty payments, or any other type of payments to third parties. However, be aware that although much of the software included with Ubuntu is licensed under the GPL, some packages on this book's disc are licensed under other terms. There is a variety of related software licenses, and many software packages fall under a broad definition known as *open source*. Some of these include the Artistic License, the BSD License, the Mozilla Public License, and the Q Public License.

For additional information about the various GNU software licenses, browse to http://www.gnu.org/. For a definition of open-source and licensing guidelines, along with links

to the terms of nearly three dozen open-source licenses, browse to http://www.opensource.org/.

Who This Book Is For

This book is for anyone searching for guidance on using Ubuntu and primarily focuses on Intel-based PC platforms. Although the contents are aimed at intermediate to advanced users, even new users with a bit of computer savvy will benefit from the advice, tips, tricks, traps, and techniques presented in each chapter. Pointers to more detailed or related information are also provided at the end of each chapter.

If you are new to Linux, you might need to learn some new computer skills, such as how to research your computer's hardware, how to partition a hard drive, and (occasionally) how to use a command line. This book helps you learn these skills and shows you how to learn more about your computer, Linux, and the software included with Ubuntu. System administrators with experience using other operating systems can use the information in this book to install, set up, and run common Linux software services, such as the *Network File System (NFS)*, a *File Transfer Protocol (FTP)* server, and a web server (using Apache, among others).

What This Book Contains

Ubuntu Unleashed is organized into six parts, covering installation and configuration, Ubuntu on the desktop, system administration, programming and housekeeping, and a reference section. A disc containing the entire distribution is included so that you have everything you need to get started. This book starts by covering the initial and essential tasks required to get Ubuntu installed and running on a target system.

If you are new to Linux, and more specifically Ubuntu, first read the chapters in Part I, "Installation and Configuration." You will get valuable information on the following:

- Detailed steps that walk you through installation
- Critical advice on key configuration steps to fully install and configure Linux to work with your system's subsystems or peripherals, such as pointers, keyboards, modems, USB devices, power management, and—for laptop users—PCMCIA devices
- Initial steps needed by new users transitioning from other computing environments
- Configuration and use of the X Window System, the graphical interface for Linux

Part II, "Desktop Ubuntu," is aimed at users who want to get productive with Ubuntu and covers the following:

- Becoming familiar with the X Window System and looking at GNOME and KDE
- Discovering the many productivity applications that come with Ubuntu
- Surfing the Internet and working with email and newsgroups

- Using Ubuntu to listen to music and watch video
- Using Ubuntu to download and manipulate images from digital cameras
- Setting up local printers for Ubuntu
- Understanding the current state of gaming for Linux

Moving beyond the productivity and desktop areas of Ubuntu, Part III, "System Administration," covers the following:

- Managing users and groups
- Automating tasks and using shell scripts
- Monitoring system resources and availability
- Backup strategies and software
- Network connectivity, including sharing folders and securing the network
- Internet connectivity via dial-up and broadband connections
- Building and deploying web servers
- Database creation, management, and manipulation
- File and print servers
- Using FTP for serving files across the Internet and local networks
- Building and deploying email servers using Postfix and managing mailing lists
- Creating remote access gateways and services
- Configuring DNS for your network
- Using LDAP for storing information on users and security
- Configuring a local news server

Part IV, "Programming Linux," provides a great introduction to how you can extend Ubuntu capabilities even further using the development tools supplied with it. This part covers the following:

- Programming in Perl, using variables and scripting
- An introduction to the Python language
- Writing PHP scripts and linking them to databases
- C and C++ programming tools available with Ubuntu and how to use the GNU C Compiler (`gcc`)

Part V, "Ubuntu Housekeeping," looks at some of the more advanced skills you need to keep your system running in perfect condition, including the following:

- Securing your machine against attack from outsiders and viruses
- Performance tuning
- Command-line masterclass
- Advanced `apt`
- Kernel and module management and compilation

An extensive reference in Part VI, "Appendixes," gives you scope to explore in even more depth some of the topics covered in this book as well as providing historical context to Ubuntu and installation resources.

Conventions Used in This Book

A lot of documentation is included with every Linux distribution, and Ubuntu is certainly no exception. Although the intent of *Ubuntu Unleashed* is to be as complete as possible, it is impossible to cover every option of every command included in the distribution. However, this book offers numerous tables of various options, commands, and keystrokes to help condense, organize, and present information about a variety of subjects.

This edition is also packed full of screenshots to illustrate nearly all Ubuntu-specific graphical utilities—especially those related to system administration or the configuration and administration of various system and network services.

To help you better understand code listing examples and sample command lines, several formatting techniques are used to show input and ownership. For example, if the command or code listing example shows typed input, the input is formatted in boldface, as follows:

```
$ ls
```

If typed input is required, as in response to a prompt, the sample typed input also is in boldface, like so:

```
Delete files? [Y/n] y
```

All statements, variables, and text that should appear on your display use the same boldface formatting. In addition, command lines that require root or super user access are prefaced with the `sudo` command, as follows:

```
$ sudo printtool &
```

Command-line examples that any user can run are prefaced with a dollar sign ($), like so:

```
$ ls
```

The following elements provide you with useful tidbits of information that relate to the discussion of the text:

NOTE

A note provides additional information you might want to make note of as you are working; augments a discussion with ancillary details; or points you to an article, a whitepaper, or another online reference for more information about a specific topic.

TIP

A tip can contain special insight or a timesaving technique, as well as information about items of particular interest to you that you might not find elsewhere.

CAUTION

A caution warns you about pitfalls or problems before you run a command, edit a configuration file, or choose a setting when administering your system.

Sidebars Can Be Goldmines

Just because it is in a sidebar does not mean that you will not find something new here. Be sure to watch for these elements that bring in outside content that is an aside to the discussion in the text. You will read about other technologies, Linux-based hardware, and special procedures to make your system more robust and efficient.

Other formatting techniques used to increase readability include the use of italics for placeholders in computer command syntax. Computer terms or concepts are also italicized upon first introduction in text.

Finally, you should know that all text, sample code, and screenshots in *Ubuntu Unleashed* were developed using Ubuntu and open-source tools.

Read on to start learning about and using the latest version of Ubuntu. Experienced users will want to consider the new information in this edition when planning or considering upgrades. There are many different Linux distributions from different vendors, but many derive from, or closely mimic, the Debian distribution.

PART I

Installation and Configuration

IN THIS PART

This page intentionally left blank

CHAPTER 1

Installing Ubuntu

IN THIS CHAPTER

Not that long ago, the mere mention of installing Linux struck fear into the hearts of mortal men. Thanks to a campaign of fear, uncertainty, and doubt (commonly referred to as FUD), Linux garnered a reputation as something of an elitist operating system, only configurable by those in the know. Nowadays, it is a different story entirely, and Ubuntu is one of the easiest distros to install. In this chapter, we cover how to get started with the install disc, including booting into Ubuntu Live CD to test your system. Then we cover the actual installation of Ubuntu, looking at the various options available. The whole process is fairly pain-free under Ubuntu, as you are about to learn.

Before You Begin the Installation

Installing a new operating system is a major event, and you should make sure that you have properly thought through what is going to take place. The first thing to consider is how the hardware will be affected by the software that you propose to install. Although Ubuntu will run well on an extremely wide variety of hardware, it is worthwhile checking your hardware components out because there may be a banana skin waiting for you to slip up on. The following sections provide some areas for you to investigate and think about, and may even save you hours of frustration when something goes wrong. The sections are designed to complement the ideas and checklists presented in Appendix B, "Installation Resources."

You start by researching and documenting your hardware. This information will prove helpful later on during the installation.

Research Your Hardware Specifications

At the absolute minimum, you should know the basics of your system, such as how much RAM you have installed, what type of mouse, keyboard, and (importantly) monitor you have. Knowing the storage capacity of your hard drive is also important because it will help you plan how you will divide it up for Ubuntu. It is also a good idea to find out whether you are using SATA drivers or the more traditional PATA drives. A small detail such as whether your mouse uses the USB or PS/2 interface will ensure proper pointer configuration—something that should happen without fail, but you will be glad you knew in case something goes wrong! The more information you have, the better prepared you will be for any problems.

Use the checklist shown in Table B.2 in Appendix B to inventory or at least record some basic features of your system. Items you need to know include the amount of installed memory, size of your hard drive, type of mouse, capabilities of the display monitor (such as maximum resolution), and number of installed network interfaces (if any).

Installation Options

Ubuntu is available in three forms: the Ubuntu distribution, the Ubuntu server distribution, and the Ubuntu alternative distribution. For most people, the main distribution should suffice; the alternate is mainly used for upgrading existing Ubuntu users to the latest version, as well as allowing installation on low-powered systems. As for the server installation, this gives you access to a LAMP server in about 20 minutes (Linux, Apache, MySQL, and PHP), but as you will learn in this book, all these components are available to the Ubuntu default distribution.

Planning Partition Strategies

Partitioning is a topic that can strike fear into the hearts of novice Linux users. Coming from a Microsoft world, where you might just be used to having one hard drive, it can seem a bit strange to use an operating system that makes partitioning important. Depending on your requirements, you may opt to have a single large partition to contain all your files or you may prefer to segment your installation across several partitions to match your individual needs. You also need to take into account such things as what you will use to back up your data. With the abundance of external hard drives and Flash-based memory sticks, you could use these; remember, however, to provision backup storage space equal to or in excess of your specific requirements. Thanks to the ever-decreasing prices of storage, you can buy a 500GB SATA drive for a little more than $100. You will thank yourself that you backed up your data when your primary hard drive goes down!

The needs of the business should be the primary concern when deciding to implement a Linux system. Be careful when specifying a system and ensure that you build in an adequate upgrade path that allows you to extend the life of the system and add any additional storage or memory.

Knowing how software is allocated on your hard drive for Linux involves knowing how Ubuntu organizes its file system, or layout of directories on storage media. This

knowledge will help you make the most out of hard drive space; and in some instances, such as planning to have user directories mounted via NFS or other means, can help head off data loss, increase security, and accommodate future needs. Create a great system, and you'll be the hero of information services.

To plan the best partitioning scheme, research and know the answers to these questions:

- How much disk space does your system require?
- Do you expect your disk space needs to grow significantly in the future?
- Will the system boot just Ubuntu, or do you need a dual-boot system?
- How much data will require backup, and what backup system will work best? (See Chapter 13, "Backing Up" for more information on backing up your system.)

CD-ROM Installation Jump-Start

To install Ubuntu from the disc included with this book, you must have at least a Pentium-class CPU, 3GB of hard drive space, and 256MB RAM. Most modern systems have significantly larger drives, and it is an idea to invest in more storage from your local computer store.

To begin the installation, you need to get into your computer's BIOS to set the boot sequence so that the CD/DVD drive is the first drive that is booted. Insert the DVD into the drive and let the system boot. When the menu appears, press the Enter key to boot into Ubuntu Live.

Double-click the Install icon on the desktop and follow the instructions. When the installer finishes, choose to restart the system immediately; eventually, the disc ejects, and Ubuntu starts to boot. After a few seconds, the login window appears. Enter the username and password specified during the installation. Welcome to Ubuntu!

The Boot Loader

During installation, Ubuntu automatically installs GRUB (Grand Unified Boot Loader) to the Master Boot Record (MBR) of your hard drive. Handily enough, it also detects any other operating systems such as Windows and adds entries in GRUB as appropriate. If you have a specific requirement not to install GRUB to the MBR, you need to install using the Alternate disc, which will allow you to specify the install location for GRUB.

Installing from CD or DVD

Most PCs' BIOS support booting directly from a CD or DVD drive, and enable you to set a specific order of devices (such as floppy, hard drive, CD-ROM, or USB) to search for bootable software. Turn on your PC and set its BIOS if required (usually accessed by pressing a Function or Del key after powering on); then insert your Ubuntu disc and boot to install Ubuntu.

To use this installation method, your computer must support booting from your optical drive. You can verify this by checking your BIOS and then booting your PC.

Older PCs might prove problematic when you desire to boot to an install using optical media. The good news is that this should no longer be a problem with most post-1995 personal computers.

Step-by-Step Installation

This section provides a basic step-by-step installation of Ubuntu from the install disc. The install process itself is fairly straightforward, and you should not encounter any real problems.

It is useful to have your computer ready to connect to the Internet so that you can download updates as soon as you have finished the installation. Although typically you would be recommended to not have your computer connected up, Ubuntu's unique security configuration means that it effectively blocks all incoming network ports, so it is fairly secure until you start opening these ports.

Starting the Install

To get started, insert the DVD into your drive and reboot your computer. You will first see a boot screen that offers a variety of options for booting (see Figure 1.1) among which is the option to check your media for errors (see Figure 1.2).

FIGURE 1.1 Select an option in this first Ubuntu boot screen.

FIGURE 1.2 You can check your CD-ROM or DVD media before installing Ubuntu.

After the bootup, Ubuntu's GNOME-based desktop appears ready for you to start experimenting (see Figure 1.3). At this point, you have a fully functional Ubuntu desktop in place with the exception that nothing has been installed to your hard drive. Live CDs are somewhat unique to Linux and are a great way to test out a distro's compatibility with your hardware. Take this opportunity to have a play around with Ubuntu to see whether any problems exist.

More often than not, your hardware will be fine, and you can proceed to install Ubuntu. You do this by double-clicking the Install icon helpfully located on your desktop (see Figure 1.3).

After a couple of seconds, the Installation Welcome screen appears and prompts you to select a language (see Figure 1.4). As you can see, Ubuntu supports a wide range of languages, even native Welsh! Click forward when you have selected your language.

The next screen asks you to specify your location in the world. This is used for setting time and region options, so find the city closest to you on the map and select it. Figure 1.5 shows that we have selected London, but it could be another city that is closer to you. Ubuntu automatically chooses the time zone and detects the offset from Greenwich mean time (GMT) for you.

FIGURE 1.3 The standard Ubuntu desktop, complete with the helpful Install icon ready for you to make the switch to Ubuntu.

FIGURE 1.4 Select a language to use when installing Ubuntu.

FIGURE 1.5 Choose the closest city to you in the world to allow Ubuntu to configure time and region settings correctly.

Click Forward to continue to the next screen.

Following on from the Time and Region screen is the keyboard layout. Ubuntu will give a good guess at what type of keyboard is connected to your computer, but it allows you to change it in case it should differ. Make sure to find the correct layout for your keyboard (see Figure 1.6). If in doubt, use the text box at the bottom of the screen to try out all the keys.

When you are happy that the keyboard layout is correct, click Forward to move on to the partitioning.

Partitioning under Linux used to be somewhat of a black art, but Ubuntu has made it easy. You have three main options (two of which are shown in Figure 1.7): resize the disk and use the freed-up space, erase and use the entire disc, or manually edit the partition table. Unless you have a requirement to do so, avoid the third option; it can seriously cause trouble with other operating systems. For now, we assume that you want to resize your partition.

Use the slider to choose how big you want your new partition to be. We recommend no less than 20GB, to give yourself plenty of breathing room on your hard drive. Click Forward to start the resizing process. This can take some time, so you may want to go off and read some more of this book to learn about what else you can do with Ubuntu.

Alternatively, you might decide that you want to erase all the data on your disk and start from scratch. In our opinion, this is the easiest and fastest option. If you're happy to resize your hard drive, click Continue to allow Ubuntu to start the resize operation. It may take some time for Ubuntu to carry out the resize operation, so make yourself comfortable!

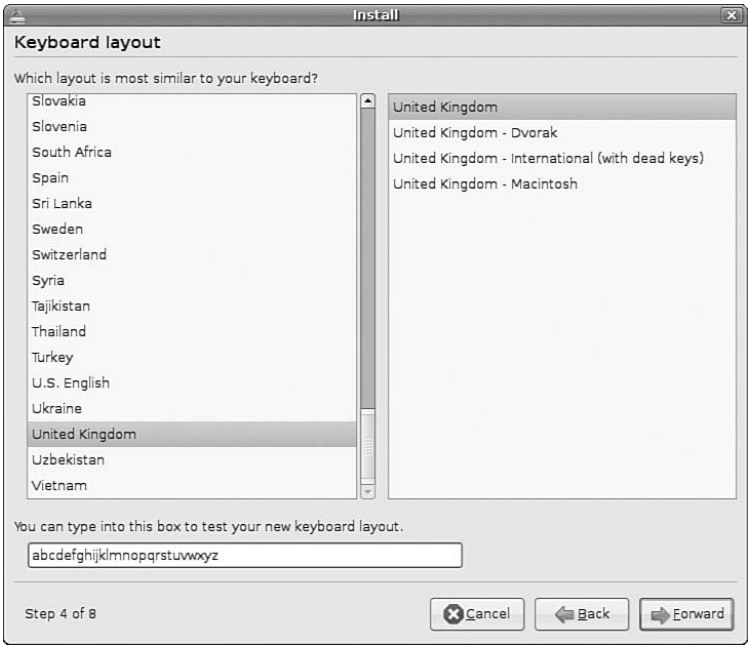


FIGURE 1.6 Choose the most appropriate keyboard layout for your computer, and then rigorously test it!

After your partition settings have been completed, you'll be presented with a dialog box asking you for your identity, so that Ubuntu can create its first user. Here you need to input your full name, your required username and password, and the name of the computer that you are installing onto. Once you've entered all the information, click Next to proceed to the summary screen where you can click OK to start the install.

CAUTION

When you set your password, be sure to remember what you entered! If you forget it, you will not be able to do much with your new system because you will not be able to log on to it.

When setting a password, make sure that it has a mixture of letters and numbers to make it more secure. For instance, a good example of a password is T1a5c0p. Although this may seem like garbage at first glance, the easy way to remember it is by remembering the phrase This Is A Good Choice Of Password, shortened to Tiagcop, and finally substituting some of the letters with similar-looking numbers. Experiment with some phrases to see what you can come up with.

After about 10 minutes, the installation finishes (see Figure 1.9) and you are asked whether you want to remain in the Live CD or reboot into your new system (see Figure 1.10). Click Restart Now for Ubuntu to eject the disc and reboot back into the full desktop system.

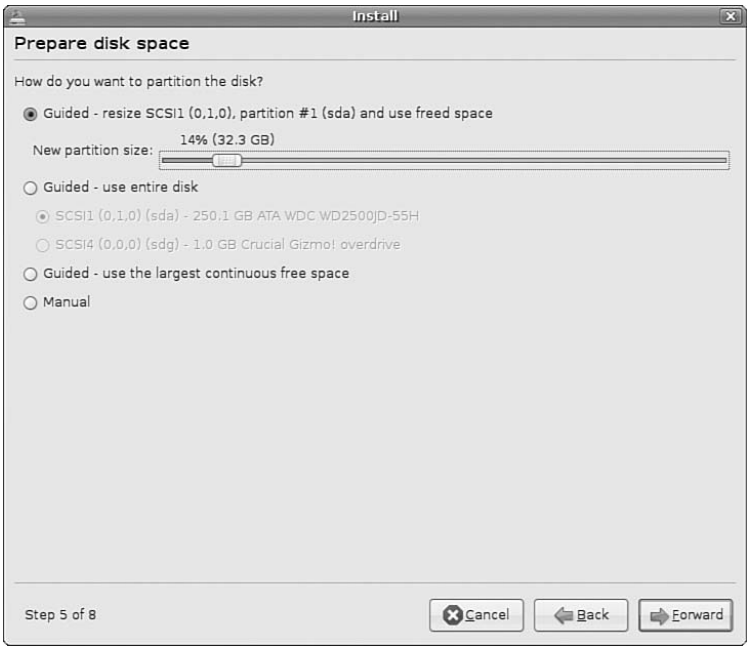


FIGURE 1.7 Tread carefully. One false move could obliterate your Windows partition. Wait, you wanted to do that?

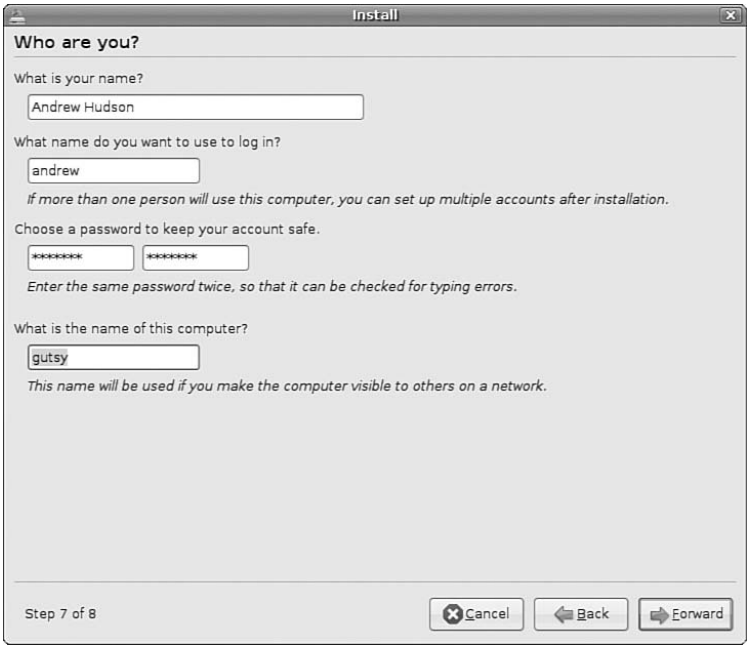


FIGURE 1.8 Fill out all the fields on this screen to give you and your computer an identity.

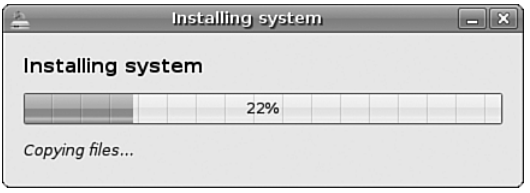


FIGURE 1.9 Sit back as Ubuntu takes over and installs itself to your hard drive.

At this point, get ready to connect your machine to the Internet. Log in to Ubuntu at the GDM welcome page (see Figure 1.11) and you will arrive at the default Ubuntu desktop (see Figure 1.12).

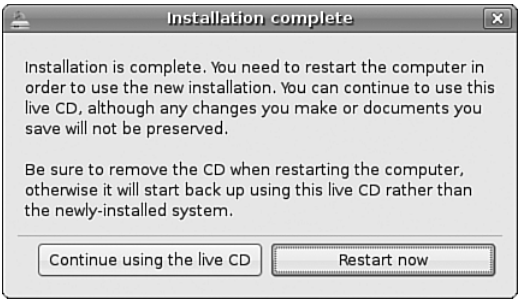


FIGURE 1.10 All done. Now all you have to do is reboot and watch as your system appears.

FIGURE 1.11 Enter your username and password to log on to your new system.

TIP

Ubuntu works well with other operating systems, and you should see your other operating system listed on the GRUB screen. If you do not, head on over to http://www.tldp.org to get a host of tips for configuring GRUB.

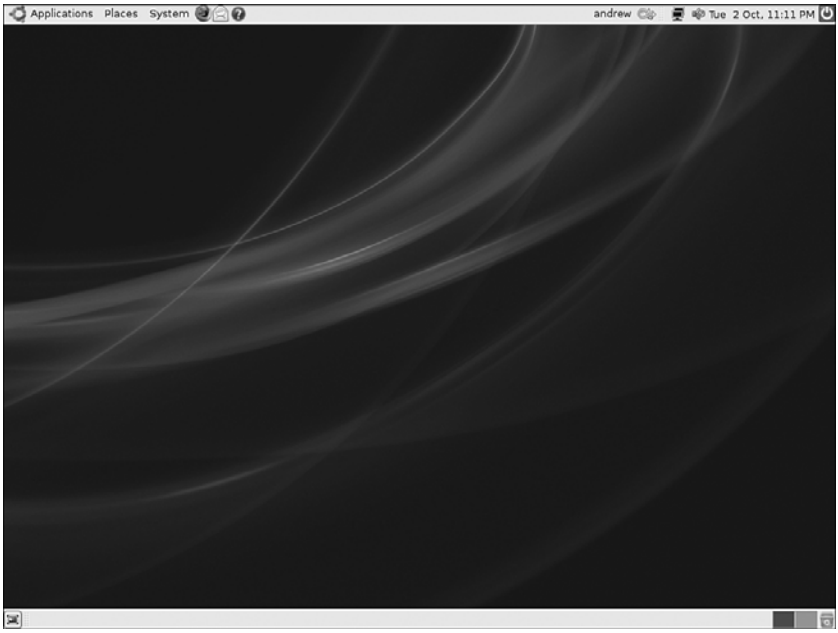


FIGURE 1.12 All ready for you to jump right in, the Ubuntu desktop welcomes you!

First Update

The first thing that you need to do with your new system is update it to the latest package versions. You do this mainly to ensure that you have the latest security updates available.

You can do this in a couple of ways in Ubuntu, but the easiest way to do it is to click the Updates icon in the panel to open the Update Manager. Your screen will darken, and you will be asked for a password. This password is the same as the one you used to log in to your system and is used in this case to authorize Ubuntu to make a systemwide change (install software, on this occasion).

Figure 1.13 shows the Update Manager in action.

The application is easy to use because it automatically retrieves information on all the possible updates available to you. Just click Install Updates to start downloading and installing the necessary package updates, as shown in Figure 1.14.

If you want more information, just click the Show Details arrow, or if you want Ubuntu to check again, click the Check button to force Ubuntu to refresh the updates.

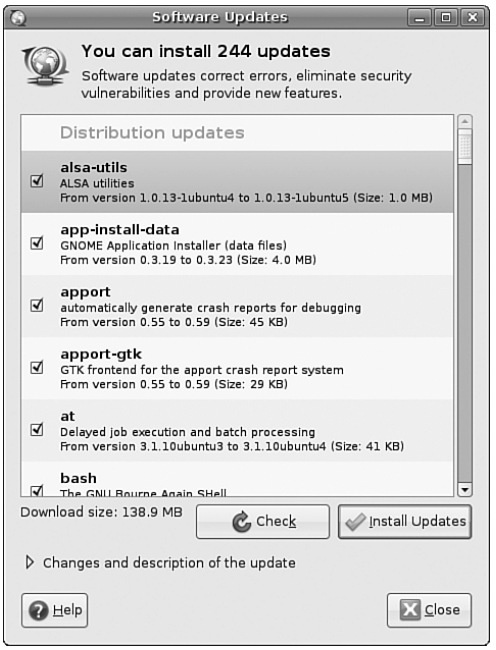


FIGURE 1.13 Get the latest security and bug-fix updates for your installed software by using Update Manager.

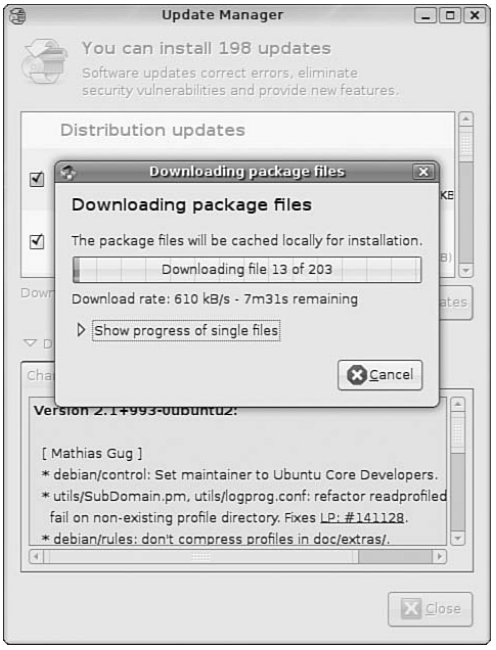


FIGURE 1.14 Update Manager takes the hard work out of keeping your software current.

Shutting Down

At some point, you are going to want to shut your computer down. As with most things in Linux, there are different ways to do it. You can use the Quit icon located in the upper-right corner of your screen or use the same button located in the System menu. Either way, you can choose to shut down or restart your system. You can also choose to hibernate (saves a copy of the running state to disk and shutdowns) or suspend (saves the current running state to memory). We would recommend using either shutdown or reboot to ensure you get a clean system when it comes back up.

If you are working at the command line, you can immediately shut down your system by using the `shutdown` command like this:

```
$ sudo shutdown -h now
```

You can also use the `shutdown` command to restart your computer, as follows:

```
$ sudo shutdown -r now
```

For new users, installing Ubuntu is just the beginning of a new and highly rewarding journey on the path to learning Linux. For Ubuntu system administrators, the task ahead is to fine-tune the installation and to customize the server or user environment.

Reference

- http://www.ubuntu.com—The place to start when looking for news, information, and documentation about installing, configuring, and using Ubuntu.
- http://tinyurl.com/c2x5u—Symantec's PartitionMagic utility includes BootMagic, which can be used to support booting of Linux or, regrettably, other less-capable operating systems, such as Windows XP.
- http://www.v-com.com/product/System_Commander_Home.html—V Communications, Inc.'s System Commander, a commercial 4.2MB download that can be used to support booting of any operating system capable of running on today's PCs. An intelligent partitioning utility, Partition Commander, is included.
- http://www.nwc.com/columnists/1101colron.html—How to use Intel's Pre-execution Environment (PXE) protocol to remote boot workstations.
- http://www.gnu.org/software/grub/—Home page for the GRUB boot loader.
- http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/BootPrompt-HOWTO.html—The BootPrompt-HOWTO, a guide to using the boot prompt for passing kernel arguments.
- http://www.tldp.org/HOWTO/Installation-HOWTO/index.html—The Linux Installation-HOWTO, a guide to installing Linux, by Eric S. Raymond.

This page intentionally left blank

CHAPTER 2

Post-Installation Configuration

Now that the hard bit is out of the way (installing Ubuntu) you can begin to customize your new operating system. By default, Ubuntu presents you with a pretty blank canvas with which to personalize your experience. Some people choose to stick with the chocolate colored background, while others go for a full make over.

In this chapter, we'll take a look at getting up and running with Ubuntu, including a quick tour around the desktop. We will also take a look at some of the ways in which you can customize your Ubuntu installation, such as ensuring that date and time settings are correct, as well as show you how to do some basic administration tasks, such as identifying any problems with hardware. By the end of this chapter, you should feel comfortable enough to move on through the rest of the book.

NOTE

Throughout this chapter, we introduce you to several key applications relevant also to other chapters of this book. We include them here because they are essential to getting your system to work as you intend. You will find it worthwhile referring back to this chapter as you work your way through the book.

IN THIS CHAPTER

Troubleshooting Post-Installation Configuration Problems

A lot of work has gone into Ubuntu to make it as versatile as possible, but sometimes you may come across a piece of hardware that Ubuntu is not sure about. Knowing what to do in these situations is important, especially when you are working with Ubuntu for the first time.

Because Ubuntu (and Linux in general) is built on a resilient UNIX foundation, it is much more stable than other operating systems. You might find this surprising if you are used to the Blue Screens of Death found on a certain operating system from Redmond, Washington. However, even though things might seem to be working fine, Ubuntu could have a problem that might not affect the appearance of the system. Perhaps kernel modules for devices will not load, for example, or services cannot start for some reason. In this section, you learn how to examine some of Ubuntu's built-in error logs to help you diagnose any unseen faults. Ubuntu has a command that enables you to see detailed messages that are output directly by the operating system: the `dmesg` command, which is commonly used with the `grep` command to filter output. The `dmesg` command takes its output directly from the `/var/log/messages` file, so you can choose to either run `dmesg` directly or enter `less /var/log/messages` instead. The output is fairly detailed, so be prepared for some initial shock when you see how much information is generated. You might find it easier to generate a file with the `dmesg` output by using the following command:

```
$ dmesg > dmesg.txt
```

This takes the output from the `dmesg` command and stores it in a new text file called `dmesg.txt`. You can then browse it at your leisure using `vi` or `emacs`, depending on your taste. You can even use the `less` command, like so:

```
$ less dmesg.txt
```

The messages are generated by the kernel, other software run by `/etc/init.d`, and Ubuntu's runlevel scripts. You might find what appear to be errors at first glance, but some errors are not really problems (for example, if a piece of hardware is configured but not present on your system).

Thanks to Google, troubleshooting is no longer the slow process it used to be. You can simply copy and paste error messages into Google and click Find to bring up a whole selection of results similar to the problem you face. Remember, Google is your friend, especially http://www.google.com/linux, which provides a specialized search engine for Linux. You can also try http://marc.info, which browses newsgroup and mailing list archives. Either way, you are likely to come across people who have had the same problem as you.

It is important to only work on a solution to one problem at a time; otherwise, you may end up getting no work done whatsoever. You should also get into the habit of making backup copies of all files that you modify, just in case you make a bad situation worse.

Use the copy command like this:

```
$ cp file file.backup
```

You should never use a `.bak` extension because this could get overwritten by another automatic process and will leave you frustrated when you try to restore the original file.

If something breaks as a result of you changing the original file, you can always copy the original back into place using the command like this:

```
$ cp file.backup file
```

(Something as simple as this can really save your bacon, especially when you are under pressure when you've changed something you shouldn't have on a production system. That is, if you are daft enough to make sweeping changes on a production system!)

The `sudo` command

If you have come across Linux, you are probably aware of the command line. As you will find as you work through this book, Ubuntu puts a lot of reliance upon the `sudo` command while working at the command line. This command is used in front of other commands to tell Ubuntu that you want to run the specified command with super user powers. This sounds really special, and it actually is. When you work using the `sudo` command, you can make wide-ranging changes to your system that impact the way it runs. Be extra careful when running any command prefixed with `sudo`, however; a wrong option or incorrect command can have devastating consequences.

The use of `sudo` is straightforward. All you have to do is enter it like this:

```
$ sudo command commandoptions
```

Just replace the word *command* with the command that you want to run, along with any options. For example, the following command opens your `xorg.conf` file in `vi` and enables you to make any changes as the super user before being able to save it:

```
$ sudo vi /etc/X11/xorg.conf
```

Whenever you execute a command using `sudo`, you are prompted for your password. This is the same password that you use to log in to Ubuntu, so it is important that you remember it.

Sometimes, however, you may want to work with a classic root prompt instead of having to type `sudo` in front of every command (perhaps if you have to work with lots of commands at the command line that require super-user access, for example). `sudo` enables you to do this by using the `sudo -i` command. Again, you are prompted for your password, which you should enter, after which Ubuntu gives you the standard root prompt, as follows:

```
#
```

From here, you can execute any command without having to keep entering `sudo`.

First Update

As discussed in Chapter 1, you should update your software as soon as you log in to Ubuntu for the first time to benefit from any available security and bug fixes. In that example, we used the Update Manager, which is really a GUI wrapper for the command `apt-get`. In the background, Ubuntu automatically polls the software repositories configured as standard to determine whether any updates are available. When it detects that new versions of installed software are available, a pop-up message appears in the upper-right corner of your screen (see Figure 2.1). By clicking the Updates icon in the panel, you automatically open Update Manager, but not before you are prompted for your password (because you are about to make a systemwide change).

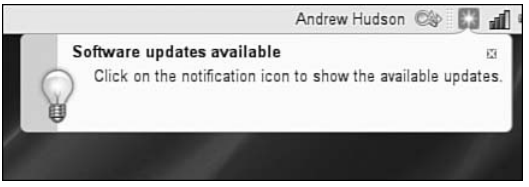


FIGURE 2.1 A quick and easy way to get updates installed is to look for the update notification.

Another way of updating your system, and one that can be quicker than Update Manager, is to use the command line. If you go to the Applications, Accessories menu and select the Terminal option, a blank screen displays. This is the command line (commonly referred to as the *terminal*) and is one of the most powerful features of Linux. It is covered in more detail in Chapter 30, "Command Line Masterclass," so we won't delve too deeply here.

You are greeted with a prompt similar to the one here:

```
andrew@optimus:~$
```

A blinking cursor also displays. Ubuntu is awaiting your first command. Here we want to issue the following command:

```
$ sudo apt-get update
```

This command tells the package management utility `apt-get` to check in with the configured repositories and check for any updates for installed software. In a matter of seconds, Ubuntu completes all of this and your screen should look something like this:

```
$ sudo apt-get update
Password:
Get: 1 http://security.ubuntu.com gutsy-security Release.gpg [189B]
Get: 2 http://security.ubuntu.com gutsy-security Release [30.9kB]
Get: 3 http://gb.archive.ubuntu.com gutsy Release.gpg [189B]
Get: 4 http://gb.archive.ubuntu.com gutsy-updates Release.gpg [189B]
```

```
Get: 5 http://gb.archive.ubuntu.com gutsy-backports Release.gpg [189B]
Hit http://gb.archive.ubuntu.com gutsy Release
Get: 6 http://gb.archive.ubuntu.com gutsy-updates Release [30.9kB]
Get: 7 http://security.ubuntu.com gutsy-security/main Packages [25.1kB]
Get: 8 http://gb.archive.ubuntu.com gutsy-backports Release [19.6kB]
Hit http://gb.archive.ubuntu.com gutsy/main Packages
Get: 9 http://gb.archive.ubuntu.com gutsy/restricted Packages [4571B]
Hit http://gb.archive.ubuntu.com gutsy/universe Packages
Get: 10 http://gb.archive.ubuntu.com gutsy/multiverse Packages [95.2kB]
Get: 11 http://security.ubuntu.com gutsy-security/restricted Packages [4253B]
Get: 12 http://security.ubuntu.com gutsy-security/universe Packages [5271B]
Get: 13 http://security.ubuntu.com gutsy-security/multiverse Packages [1677B]
Get: 14 http://security.ubuntu.com gutsy-security/main Sources [6227B]
Get: 15 http://security.ubuntu.com gutsy-security/restricted Sources [974B]
Get: 16 http://security.ubuntu.com gutsy-security/universe Sources [639B]
Get: 17 http://security.ubuntu.com gutsy-security/multiverse Sources [533B]
Hit http://gb.archive.ubuntu.com gutsy/main Sources
Get: 18 http://gb.archive.ubuntu.com gutsy/restricted Sources [1478B]
Hit http://gb.archive.ubuntu.com gutsy/universe Sources
Hit http://gb.archive.ubuntu.com gutsy/multiverse Sources
Get: 19 http://gb.archive.ubuntu.com gutsy-updates/main Packages [37.7kB]
Get: 20 http://gb.archive.ubuntu.com gutsy-updates/restricted Packages [14B]
Get: 21 http://gb.archive.ubuntu.com gutsy-updates/universe Packages [8363B]
Get: 22 http://gb.archive.ubuntu.com gutsy-updates/multiverse Packages [866B]
Get: 23 http://gb.archive.ubuntu.com gutsy-updates/main Sources [22.1kB]
Get: 24 http://gb.archive.ubuntu.com gutsy-updates/restricted Sources [14B]
Get: 25 http://gb.archive.ubuntu.com gutsy-updates/universe Sources [1823B]
Get: 26 http://gb.archive.ubuntu.com gutsy-updates/multiverse Sources [427B]
Get: 27 http://gb.archive.ubuntu.com gutsy-backports/main Packages [14B]
Get: 28 http://gb.archive.ubuntu.com gutsy-backports/restricted Packages [14B]
Get: 29 http://gb.archive.ubuntu.com gutsy-backports/universe Packages [14B]
Get: 30 http://gb.archive.ubuntu.com gutsy-backports/multiverse Packages [14B]
Get: 31 http://gb.archive.ubuntu.com gutsy-backports/main Sources [14B]
Get: 32 http://gb.archive.ubuntu.com gutsy-backports/restricted Sources [14B]
Get: 33 http://gb.archive.ubuntu.com gutsy-backports/universe Sources [14B]
Get: 34 http://gb.archive.ubuntu.com gutsy-backports/multiverse Sources [14B]
Fetched 299kB in 2s (140kB/s)
Reading package lists... Done
```

Now you need to issue the command to upgrade your software by entering the following:

```
$ apt-get dist-upgrade
```

Because you have already checked for updates, Ubuntu automatically knows to download and install only the packages it needs. The `dist-upgrade` option intelligently works with

newer packages to ensure that any dependencies that are needed can be satisfied. You can also use the option `upgrade` instead, but it isn't as smart as `dist-upgrade`.

`apt-get` Alternatives

Ubuntu has an alternative to `apt-get` called `aptitude`. It works in pretty much the same way as `apt-get` with the exception that `aptitude` also includes any recommended packages on top of the requested packages when installing or upgrading.

Configuring Software Repositories

Ubuntu uses software repositories to get information about available software that can be installed onto your system. By default, it only allows access to a small portion of software (even though this software is officially supported by Ubuntu). However, Ubuntu is based on a much older Linux distribution called Debian. Debian has access to more than 17,000 different packages, which means that Ubuntu can have access to these packages, too.

To do this, you need to use one of Ubuntu's GUI tools, found under System, Administration, Software Sources and shown in Figure 2.2. On the first tab (Ubuntu Software), you have five options to choose from, depending on your specific requirements. It is entirely up to you which options you check, but make sure that as a minimum the first check box is checked to allow you to select "official" software with Canonical support for Ubuntu. The more boxes you check, the wider your selection of software. It's also a good idea to make sure that the Proprietary Drivers box is checked in order to benefit from drivers that could enhance your system's performance.

FIGURE 2.2 Enable both Universe and Multiverse repositories to allow access to a huge variety of software for Ubuntu.

Open Source Versus Proprietary

You will hear a lot of arguments about using Proprietary drivers within Ubuntu. Some people feel that it goes against what Open Source stands for, in that the underlying code that is used for the drivers cannot be viewed and modified by the wider community (as opposed to the actual driver developers). However, there is also a strong argument that says users should have to undergo the least amount of work for a fully functional system. This is certainly the case for graphics cards, although at the time of writing AMD has announced the open sourcing of the ATI graphics driver.

Ubuntu takes a middle of the road stance on this and leaves it up to the user to either enable or disable the repository that gives access to proprietary drivers. Why not give the open source drivers a chance before plumping for a proprietary one?

Once you are happy with your selections, switch to the Updates tab to configure Ubuntu's behavior when updates are available (see Figure 2.3). By default both the important security updates and recommended updates are checked to ensure that you have the latest bug fixes and patches. You can also choose to receive proposed updates and *back-ports* (software that is released for a newer version of Ubuntu but re-programmed to be compatible with 7.10), but we'd only recommend this if you are happy to carry out testing for the community as any updated software from these repositories can have an adverse effect on your system.

FIGURE 2.3 Configure which updates you want, and you want them to be handled in the Updates tab of Software Sources.

Ubuntu also allows you to configure how often it checks for updates, as well as how they are installed. By default Ubuntu checks daily for updates and, if there are any available,

will notify you. However, you can change the frequency (something which is recommended if you want to have a structured update policy) and the actions Ubuntu carries out when it finds available updates. We recommend keeping the notification only option as this allows you to see what updates are available prior to them being installed. If you want to save time then choose Download All Updates in the Background to allow Ubuntu to silently download the updates prior to you choosing to install them.

CAUTION

We don't recommend selecting the option which automatically installs security updates. It's important that you have the option of choosing to install updates as there is always the chance that an update may cause problems. The last thing you want is for your system to suddenly stop working because an update was installed that has broken something without your knowledge. After configuring your update options, click the Close button. Ubuntu will prompt you to say that the software information is out of date and needs to be refreshed, so click Reload to retrieve the very latest update information. After a few seconds you will be returned to your desktop and can carry on working.

Installing Graphics Drivers

Ubuntu is extremely good at detecting and configuring graphics cards. By default it ships with a number of proprietary drivers to allow graphics cards to work and will alert you to this by using the Restricted Drivers Manager. You can then choose to enable or disable the drivers as appropriate, depending on your personal preference. (See the "Open-Source Versus Proprietary" note earlier in this chapter.)

You can find the Restricted Drivers Manager under the System, Administration menu, and it is shown in Figure 2.4.

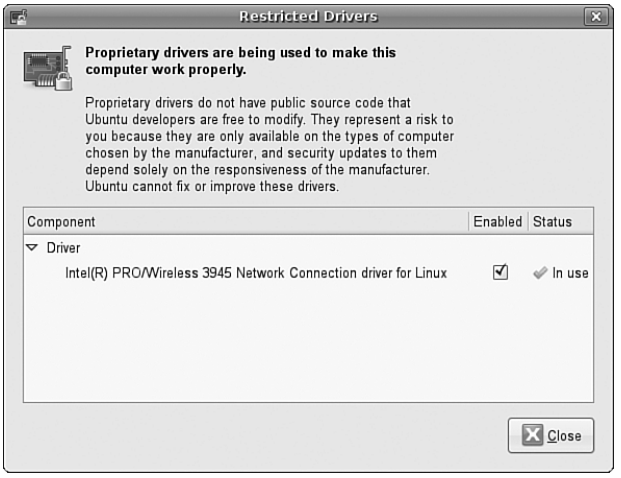


FIGURE 2.4 Toggle the usage of restricted drivers with a simple point-and-click interface. Here you can see that an Intel network driver has been enabled for use.

If you elect to use a listed proprietary driver, Ubuntu will first confirm that you are happy to proceed and then automatically download and install the driver. This may require you to log out and back in again in order for the driver to take effect.

For the most part, Ubuntu will detect and configure the majority of graphics cards from the start, and even if it has problems, it will attempt to give you a display of some sort. This feature is known as BulletProof X.

Changing Ubuntu's Look and Feel

GNOME, the default window manager for Ubuntu, has a number of options to change how it looks and feels. The default theme is Human; this takes a blend of the GNOME Clearlooks theme, mixes it with icons from the Tango Project, and adds a little spice direct from Ubuntu. However, it is not to everyone's taste, so in this section we look at changing the visual style of Ubuntu.

Changing the Desktop Background

Perhaps the easiest and most dramatic change you can make is to change the default desktop background. It is as simple as right-clicking on the desktop and selecting the option to Change Desktop Background to see the dialog box shown in Figure 2.5. Ubuntu comes with a small selection of wallpapers to start with, but we recommend going to the Web to find a great selection. Our favorite site is http://www.gnome-look.org, where you can find a great selection of desktop wallpapers.

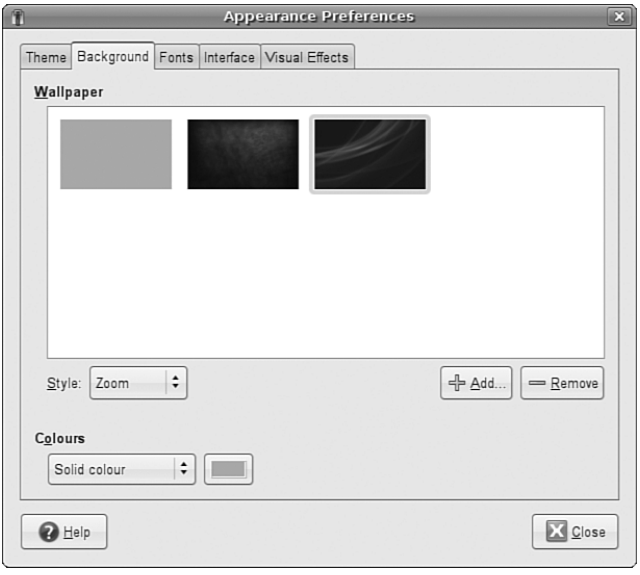


FIGURE 2.5 Choose one of the default wallpapers, or use the Add Wallpaper option to select your own image file.

As you click on a wallpaper, Ubuntu automatically applies it so that you can quickly see whether you like it. When you are happy with your selection, keep the dialog box open so we can change other aspects of the desktop.

Changing Colors

Next up is the colors that Ubuntu defaults to. When Ubuntu was originally launched in October 2004, it came with a predominantly brown theme, leading to some users questioning the style choices of the Ubuntu development team. With Ubuntu 7.10, the brown has been replaced with a warm caramel color (which, to be honest, is only slightly better than the brown).

Thankfully, GNOME offers an array of tools to modify the defaults. Just head to the System, Preferences, Appearance option to quickly change the entire theme (see Figure 2.6). Or if you've kept the dialog box open as suggested earlier, click the Theme tab. You will see a list of predefined themes that you can choose from.

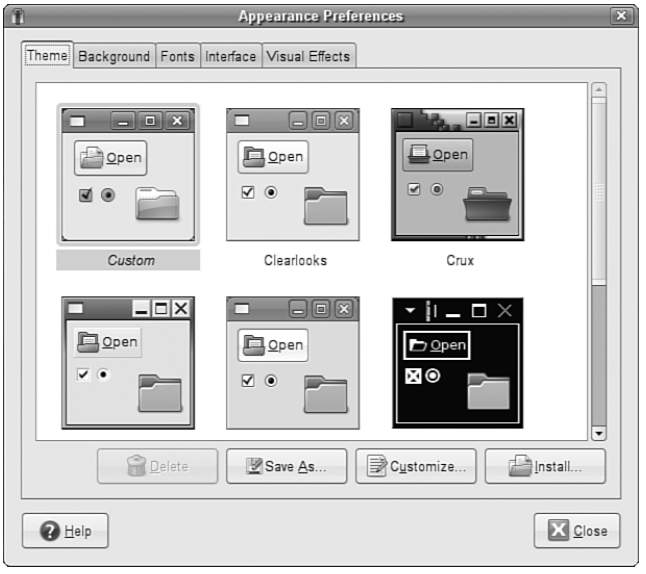


FIGURE 2.6 Either use the default themes or mix and match elements of them to match your specifications.

Alternatively, you can click the Customize button to mix up your own look and feel, as shown in Figure 2.7. You can choose to change the window decorations (title bar, minimize, maximize buttons, and so on), the general color scheme and the icon theme. As you select an option, Ubuntu automatically applies it to the desktop so you can get an idea of how it will look. When you're happy with your selections click Close to return to the main Theme dialog box and move on to the next section.

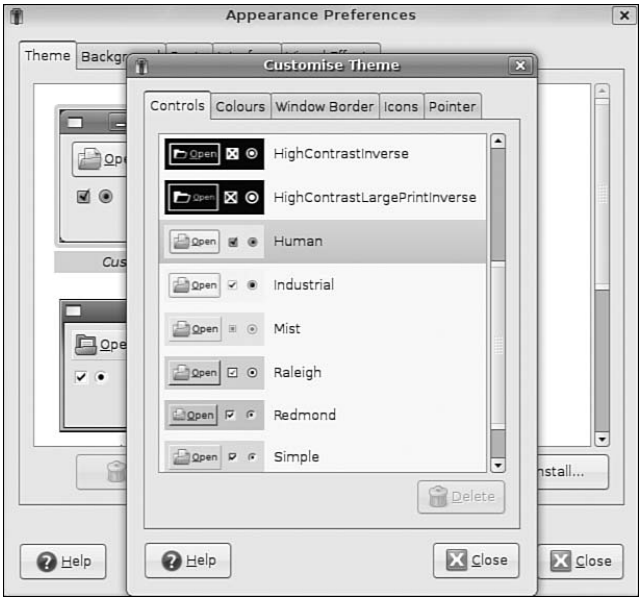


FIGURE 2.7 Use any number of combinations of icons, colors, and window decorations to give you a truly personalized look and feel.

Modifying System Fonts

GNOME also enables you to change the fonts used throughout Ubuntu. If you have difficulty reading one font, just exchange it for another.

In the Appearance dialog box you will see a tab called Fonts. Click on this tab to see the options available for changing system fonts, which are shown in Figure 2.8. Simply click on each font name to customize the font used for that function.

For instance, you may prefer your Window Title Font to be italicized, so click the font name and select the Italic option. Clicking OK immediately applies the change, again giving you a good idea of whether it works. Sometimes a font can look good within the Font Preferences screen, but when it comes to using it you wonder why on earth you chose it. Choose your fonts wisely!

Changing How Menus Look

You can also change the look of menus, including the location of text descriptions for each icon, by selecting the Interface tab. To be honest, there's not much to look at here,

and you would only change any of these options if you had a burning desire to replace your icons with text.

FIGURE 2.8 Keep your fonts relatively simple for ease of use, or if you prefer, indulge your love of calligraphic fonts!

Visual Effects

Perhaps the most exciting part of Ubuntu 7.10 is that by default it comes with some pretty snazzy visual effects to enhance the look and feel of your desktop. Click the Visual Effects tab to see the options available for configuring the level of effects, shown in Figure 2.9. In keeping with Ubuntu's philosophy of keeping things simple, there are three options to choose, from which you can elect to turn off all visual effects, use a basic set of effects, or use a lot of visual effects. If you choose this last option you will need to have a fast computer (that is, purchased after 2004) to ensure that the computer doesn't grind to a halt while rendering effects.

FIGURE 2.9 Wow your friends and colleagues by activating visual effects to enhance your user experience. Alternatively, switch them all off if you get a bit overwhelmed by windows flying everywhere!

Preferred Behaviors

Ubuntu can detect and adapt to certain events that happen when you plug something into your computer or if you click on a link. Sometimes you may want Ubuntu to open one application rather than another, or sometimes to not do anything at all. This is called Ubuntu's behavior, and you can modify it to work as you want it to.

You can find the two main tools you need to do this under the System, Preferences menu, and you need use either Preferred Applications or Removable Drives and Media.

Preferred Applications

Preferred applications are the applications that Ubuntu calls upon when it wants to open an Internet site, an email, or a terminal (see Figure 2.10). Ubuntu makes it easy for you by automatically detecting the available options for you. Alternatively, if you want to specify your own particular application that may not appear in the list by default, select the Custom option within the application type and enter the command used to launch the application into the field along with any options, arguments, or switches that you want to use.

FIGURE 2.10 Setting preferred Internet applications.

Removable Drives and Media

In an increasingly connected world, you will find yourself plugging all sorts of devices and widgets into your computer. For the most part, it is usually multimedia files that you want to launch in a specific way. Ubuntu provides a great tool to configure the consequences of everything from plugging a digital camera into your computer, inserting a film DVD, and even plugging in a graphics tablet.

This handy tool goes by the catchy name of Removable Drives and Media, or to call it by its proper (package) name, `gnome-volume-properties`. With this tool, you can select the default command to run when Ubuntu detects the presence of a new drive or piece of hardware. Unlike the Preferred Applications tool, this tool does not detect the options for you, instead relying on you knowing the command needed to execute the correct application.

For example, in Figure 2.11, you can see the options for handling digital images in Ubuntu. Rather than using gthumb to import pictures from a digital camera, I want Ubuntu to use F-Spot, a cool photo management and editing application available through `synaptic`. In this case, I need to enter the `f-spot` command in the Command field and make sure to check mark the Import Digital Photos When Connected option.

Now when I connect my digital camera after a hard day's snapping, GNOME will detect its presence and launch F-Spot, ready for it to import my pictures directly into its library.

Other options include GNOME's behavior when removable drives (USB pen drives and the like) are connected, what to do in the event of a PDA or scanner being connected, and even more obscurely, what to do when a USB mouse is plugged in.

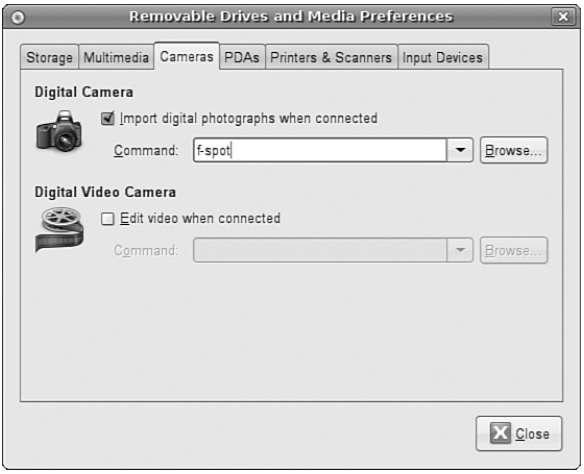


FIGURE 2.11 Take the hard work out of importing digital images by configuring `gnome-volume-properties` to work the way you want.

Input Devices

The primary interface between you and your computer is your keyboard and mouse. Both of these are essential to the correct usage of your computer, so it is important that they are configured correctly. Windows users will be familiar with the basic options available, so it could come as somewhat of a shock to learn that you can do a few more nifty things with a keyboard and mouse in Ubuntu.

Keyboard Shortcuts

If you have used a computer for more than a few years, you probably long for keyboard shortcuts for popular commands. Ubuntu allows you to set your own keyboard shortcuts for a wide variety of system commands. The easy-to-use Keyboard Shortcuts option under System, Preferences lists a lot of different actions that you can program shortcuts for (see Figure 2.12).

If you have one of those multimedia keyboards with lots of extra keys, this is the place to configure their use. Just click on the shortcut next to the action you want to configure and press the key that you want to map to this action. Repeat this until you have exhausted all the combinations that you want to configure. Then click Close to finish.

Keyboard Layout

Getting the layout of your keyboard right can make a huge difference in how you work. When you installed Ubuntu (see Chapter 1), you would have specified the default keyboard layout to use. However, there may be times when you need to switch layouts, which you can do using the Keyboard tool in the System, Preferences menu (see Figure 2.13).

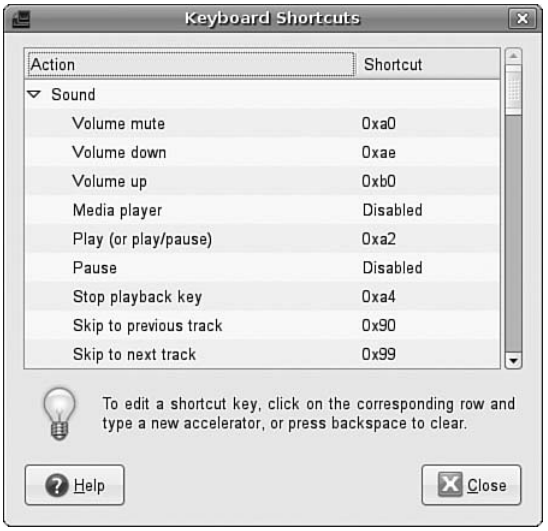


FIGURE 2.12 Make your life easier and your work go more quickly by configuring useful keyboard shortcuts.

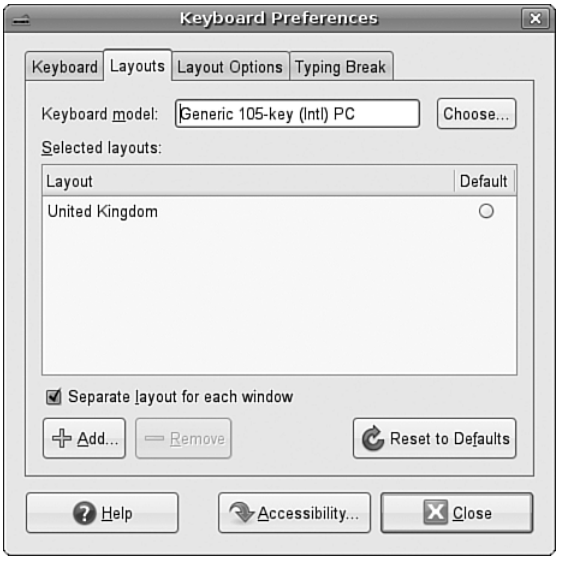


FIGURE 2.13 Use the Keyboard Layout tool to ensure that you have your keyboard settings configured correctly.

Certainly, I've had to configure this for my wife. You see, in the United Kingdom our @ key is located to the right of the colon/semicolon key, whereas in the United States it is located on the number 2 key. My wife spent quite a few years in the States and got used to the U.S. keyboard layout. So instead of her having to learn a new layout, I just configured her login

to use the U.S. layout and mine to use the U.K. layout, which has saved us from a number of arguments!

However, you can also use the Keyboard Layout tool to configure special key behaviors. For instance, some people prefer to swap the Caps-Lock and left Ctrl key around. You can set this option and others in the Layout Options tab. If you are not yet an l33t hacker, experiment with what's on offer; you may get some benefit from these customizations.

Finally, Ubuntu can also configure and enforce typing breaks, all in the name of good health. Simply set the length of working time, the duration of the break, and whether you want to be able to postpone. When this is activated at the end of the first length of working time, Ubuntu locks the computer and will not let you log back in until the break duration has passed. Of course, if you are in the middle of something important, setting the Postpone option may prove useful.

Mouse

There's not really much to configuring a mouse. Ubuntu does a great job of detecting most mice, except that it's not so good at configuring extra buttons over and above the left and right button and scroll wheel. The most useful option here is the Locate Pointer option, which highlights the mouse pointer when you press the Ctrl key. Of course, if you are left-handed, you can also swap the mouse buttons over, but this is a matter of personal preference.

Detecting and Configuring a Modem

More than 38 million users in the United States and another 85 million users around the world now connect to the Internet with cable or *digital subscriber line (DSL)* service, but for many users a modem is the standard way to connect with an *Internet service provider (ISP)* using the *Point-to-Point Protocol (PPP)*. Other common tasks for modems include sending and receiving faxes. If you add or change your modem after the initial installation, you must configure Ubuntu to use the new modem to perform all these tasks.

Ubuntu includes several tools you can use to configure and use an internal or external modem in your notebook or PC. Chapter 14, "Networking," contains the details about configuring Ubuntu to connect to the Internet using a modem. This section covers how to configure and use modems using serial ports (using a standard formerly known as RS232, but now called *EIA232*) or USB.

Configuring a Serial-Port Modem

Linux uses `/dev/ttySX`, `/dev/ttyUSBX`, or `/dev/usb/ttyUSBX` for serial ports, where *X* can range from `0` to `15`. You can add many additional ports to a system using multiport cards or chained USB devices. A PC's integral serial ports are generally recognized at boot time. To see a list of recognized ports for your system, pipe the `dmesg` command output through the `fgrep` command, as follows:

```
$ sudo dmesg ¦ grep tty
ttyS00 at 0x03f8 (irq = 4) is a 16550A
ttyS01 at 0x02f8 (irq = 3) is a 16550A
```

In this example, the `grep` command reports that two serial ports have been recognized in the `dmesg` output. Note that the device matching `ttyS00` is `/dev/ttyS0`, despite the kernel output. The PC's external modem can be attached (most likely using a male DB9 adapter) to either port. Under Linux, nearly all modem-dependent software clients look for a symbolic link named `/dev/modem` that points to the desired device. This link is not created by default; as root, however, you can create this device manually using the `ln` command like this:

```
$ sudo ln -s /dev/ttyS0 /dev/modem
```

In this example, `/dev/modem` will point to the first serial port.

Ubuntu's `network-admin` (shown in Figure 2.14 and found under System, Administration, Networking) will always detect the presence of a modem on the system. However, it does not activate the interface unless specifically told to do so. You can use the Auto-Detect button to find the correct modem port.

FIGURE 2.14 Ubuntu's `network-admin` will help you set up an appropriate modem.

Configuring WinModems for Laptops

Other issues regarding modems focus on Linux notebook users with laptops using *controllerless* modems. These modems use proprietary software to emulate a hardware modem and are commonly referred to as *WinModems* due to the software being available only on Windows. Despite the release of binary-only drivers to enable use of some of these modems, these devices remain the bane of Linux notebook and some desktop users.

You might find some support for Lucent (but not Lucent AMR), Motorola SM56-type, the IBM Mwave, and Conexant HSF (not HCF) controllers. At the time of this writing, there

was no support for any 3Com or U.S. Robotics controllerless modems. For links to drivers and more information, browse to the Linux WinModem web page at http://www.linmodems.org.

Configuring Power Management in Ubuntu

Advanced Configuration and Power Interface (ACPI) enables workstations and servers to automatically turn off when instructed to shut down. Most often used by Linux mobile users, ACPI can help extend battery sessions through the use of intelligent storage-cell circuitry, CPU throttling (similar to, but not the same as safety thermal throttling incorporated by Intel in Pentium III and IV CPUs), and control of displays and hard drives.

Most PCs support ACPI via the BIOS and hardware. ACPI support is configured, enabled, and then incorporated in the Linux kernel.

ACPI information is constantly available through the `acpi` command, which looks like this:

```
$ acpi -V
    Battery 1: charged, 100%
    Thermal 1: ok, 47.0 degrees C
    AC Adapter  1: on-line
```

This example provides information such as battery charge status, percentage of charge, as well as current system temperature and status of the AC adapter.

Alternatively, you can use the GUI tool that Ubuntu provides, which can be found under the System, Preferences menu as the Power Management option (shown in Figure 2.15).

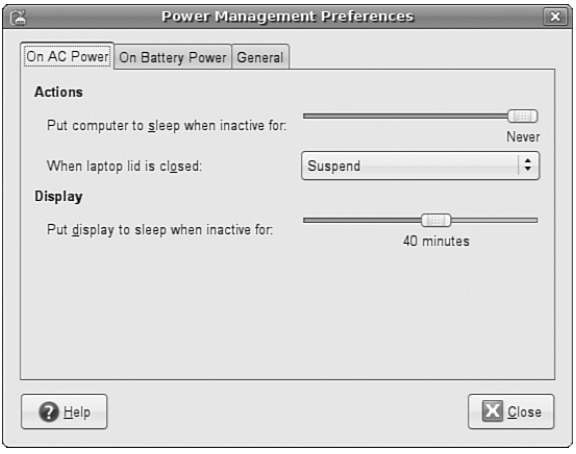


FIGURE 2.15 Gnome Power Management allows you to monitor battery status and configure specific power-related actions, such as closing the lid of your laptop or pressing the power button.

Fortunately, Ubuntu provides good support for suspend and hibernate. Suspend means that your computer writes its current state to memory and goes into a low power mode, while Hibernate writes the current state of the system to disk and powers off the computer. Either way, your computer will start much faster the next time you go to use it.

Resetting the Date and Time

The Ubuntu installer queries during installation for default time zone settings, and whether your computer's hardware clock is set to G*reenwich mean time (GMT)*—more properly known as *UTC* or *coordinated universal time*.

Linux provides a system date and time; your computer hardware provides a hardware clock-based time. In many cases, it is possible for the two times to drift apart. Linux system time is based on the number of seconds elapsed since January 1, 1970. Your computer's hardware time depends on the type of clock chips installed on your PC's motherboard, and many motherboard chipsets are notoriously subject to drift.

Keeping accurate time is not only important on a single workstation, but also critically important in a network environment. Backups, scheduled downtimes, and other network-wide actions need to be accurately coordinated.

Ubuntu provides several date and time utilities you can use at the command line or during an X session, including these:

`date`—Used to display, set, or adjust the system date and time from the command line

`hwclock`—A root command to display, set, adjust, and synchronize hardware and system clocks

`time-admin`—Ubuntu's graphical date, time, and network time configuration tool

Using the `date` Command

Use the `date` command to display or set your Linux system time. This command requires you to use a specific sequence of numbers to represent the desired date and time. To see your Linux system's idea of the current date and time, use the `date` command like this:

```
$ sudo date
Wed Jan 10 14:17:01 EDT 2005
```

To adjust your system's time (say, to January 27, 2006 at 8 a.m.), use a command line with the month, day, hour, minute, and year, like so:

```
$ sudo date 012606002003
Fri Jan 27 08:00:00 EDT 2006
```

Using the `hwclock` Command

Use the `hwclock` command to display or set your Linux system time, display or set your PC's hardware clock, or to synchronize the system and hardware times. To see your hardware date and time, use `hwclock` with its `--show` option like so:

```
$ sudo hwclock --show
Fri 27 Jan 2006 02:17:53 PM GMT  -0.193809 seconds
```

Use `hwclock` with its `--set` and `--date` options to manually set the hardware clock like so:

```
$ sudo hwclock --set --date "01/27/06 08:00:00"
$ sudo hwclock --show
Tue 27 Jan 2006 08:00:08 AM GMT -0.151718 seconds
```

In these examples, the hardware clock has been set using `hwclock`, which is then used again to verify the new hardware date and time. You can also `hwclock` to set the Linux system date and time date using your hardware clock's values with the Linux system date and time.

For example, to set the system time from your PC's hardware clock, use the `--hctosys` option like so:

```
$ sudo hwclock --hctosys
```

To set your hardware clock using the system time, use the `--systohc` option like so:

```
$ sudo hwclock --systohc
```

Changing the Time and Date

Ubuntu's graphical X tool named `time-admin` can be used to set your system date and time. The client is found in System, Administration, Time & Date.

After you press Enter, you are asked to enter your password. Type in your password and click the OK button. You will then see a window, as shown in Figure 2.16.

Set the date and time by using the Calendar and Time fields. You can also have your workstation obtain updated date and time information via the Internet by choosing Remote Time Servers under the Select Servers button. To do this, you need to have `ntpd` time daemon support installed. If it is not present, Ubuntu asks whether you want to retrieve it, and then installs it so that you can proceed. After it has been installed, restart the `time-admin` client and you will now be able to select specific Internet time servers for you to synchronize with.

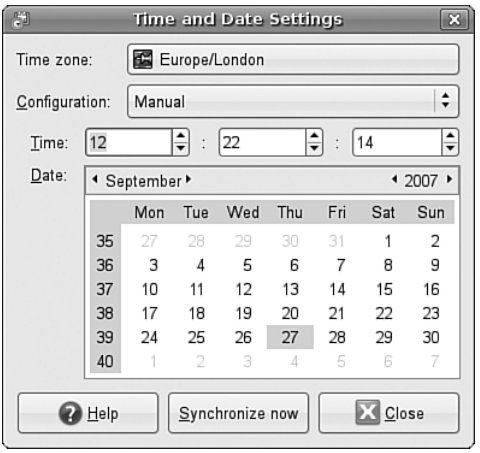


FIGURE 2.16 Use Ubuntu's `time-admin` client to set your system date and time.

Configuring and Using CD, DVD, and CD-RW Drives

Linux provides support for using a variety of CD and DVD devices and media. This section shows how to determine what device has been assigned to your CD drive and how to get additional drive information if the drive supports recording on optical media.

AT Attachment Packet Interface, or ATAPI, IDE-based CD drives are recognized during installation and work through the `ide-cd` kernel module. A symbolic link named `/dev/cdrom` is created and the device is mounted under `/media/cdrom`, both of which point to your CD's device (perhaps `/dev/hdb` or `/dev/hdc`). You can use many different types of CD drives with Linux, and you can easily replace, add, or upgrade your system to use a new drive. Part of a successful configuration involves the proper installation of the hardware and being able to determine the drive's device when using Linux.

Checking Drive Assignment

Linux recognizes CD and DVD drives upon booting if they are attached to your computer's motherboard with proper cabling and if they are assigned as either a master or slave on an IDE channel. Look through your kernel boot message for the drive device assignment, such as the following:

```
hdd: DVDROM 10X, ATAPI CD/DVD-ROM drive
```

If you have a DVD-capable drive, you generally should also have a symbolic link named `/dev/dvd` and an entry under `/media` that point to your drive's device because many DVD clients, such as `xine` or `vlc`, look for `/dev/dvd` by default. If you have a CD-RW drive, the Ubuntu installer inserts a kernel argument into your boot loader's configuration file that specifies use of the `ide-scsi` kernel, such as the following:

```
append="hdb=ide-scsi"
```

A similar entry in the grub boot loader's /etc/grub.conf file would look like this:

```
kernel boot/vmlinuz-6.5-1.358 ro root=/dev/hda2 hdb=ide-scsi
```

The first CD-RW drive is assigned to the device /dev/scd0 (although it might still be initially recognized while booting as an IDE device), with subsequent drives assigned to /dev/scd1, and so on. To initialize your drive for use, the following modules should be loaded:

```
Module          Size Used by  Not tainted
sg             30244  0 (autoclean)
sr_mod          15192  0 (autoclean)
cdrom          27872  0 (autoclean) [sr_mod]
ide-scsi         8128  0
scsi_mod         96572  2 [sr_mod ide-scsi]
```

Look for kernel message output regarding the device such as this:

```
Attached scsi CD-ROM sr0 at scsi0, channel 0, id 0, lun 0
sr0: scsi3-mmc drive: 0x/32x writer cd/rw xa/form2 cdda tray
Uniform CD-ROM driver Revision: 3.12
```

Your ATAPI-based CD-RW drive will then work as a SCSI device under emulation, and the symbolic link /dev/cdrom should point to /dev/scd0. You can also use the cdrecord command (included with Ubuntu's multimedia software packages) to acquire SCSI device information about your drive for later use during the burning operation, as follows:

```
# cdrecord -scanbus
Cdrecord 1.10 (i686-pc-linux-gnu) Copyright (C) 1995-2001 Jörg Schilling
Linux sg driver version: 3.1.22
Using libscg version 'schily-0.5'
scsibus0:
    0,0,0   0) 'HL-DT-ST' 'RW/DVD GCC-4120B' '2.01' Removable CD-ROM
    0,1,0   1) *
    0,2,0   2) *
    0,3,0   3) *
    0,4,0   4) *
    0,5,0   5) *
    0,6,0   6) *
    0,7,0   7) *
```

The pertinent information—0,0,0 in the example (SCSI bus, device ID, and logical unit number, or LUN)—can then be used during a burn operation like this:

```
# cdrecord -v speed=8 dev=0,0,0 -data -eject file_name.img
```

In this example, a CD-ROM data image named `file_name.img` is created on a CD-R or CD-RW media at a speed of `8`, and the new disk will be ejected after the write operation has completed.

NOTE

Ubuntu also includes the `dvdrecord`, `dvd+rw-format`, and `growisofs` commands, which can be used with DVD-R and DVD-RW drives.

Configuring Wireless Networks

Wireless networking used to be a pig to configure for Linux, requiring a lot of complicated steps to connect to a wireless network. However, Ubuntu includes a great utility called Network Manager that makes connecting to and managing wireless networks extremely easy. Thanks to the inclusion of several wireless chipset drivers in the Ubuntu Linux kernel, it is now easy to connect to WEP and WPA encrypted wireless networks.

When you log in to Ubuntu, you should see the Network Manager applet appear in the top panel (see Figure 2.17). This is the applet that handles and monitors network connections.

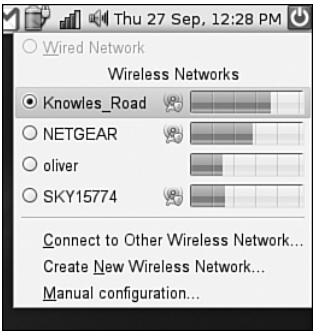


FIGURE 2.17 The Network Manager notification applet, seen here already connected to a wireless network.

Click the applet icon in the toolbar to connect to a wireless network. If your wireless access point broadcasts its SSID, it should appear in the list under wireless networks (similar to Figure 2.17). Simply click on the required network and Network Manager will detect what encryption (if any) is in use and ask you for the passkey. Enter this and Network Manager will start the wireless connection. The passkey is then stored in the default keyring, so if you have not yet used the keyring, you are asked to create a password. From now on, whenever you log in to Ubuntu, you are asked for the key to unlock the keyring.

If for some reason your wireless network does not appear (you might have your SSID hidden), you must use the Connect to Other Wireless Network option, which brings up the screen shown in Figure 2.18.

FIGURE 2.18 Configure your wireless network connection settings using Network Manager.

Network Manager can handle WEP and WPA Personal encryption. You are advised to use WPA encryption because it is the stronger of the two.

Network Manager can also connect to Cisco VPN connections, using the `vpnc` software. Install this using `synaptic` and you will be able to specify connection settings as appropriate, or if you have access to a predefined configuration (`.pcf` file) you can import it directly into Network Manager.

Configuring Firestarter

Ubuntu is unique among distros in that as default it does not listen on any network ports. This means that it is pretty much secure. However, you may have a requirement to open up ports for access by other local computers.

To configure the firewall, you need to download and install the `firestarter` package using either `synaptic` or `apt-get`. When installed, you can use it to open specific ports or even allow access to specific computers. It's a handy application, and you can find more details at http://www.fs-security.com

Related Ubuntu and Linux Commands

You will use these commands when performing post-installation configuration tasks:

- `acpi`—Views or uses power management settings and commands
- `cdrecord`—Gets SCSI device information and burns CD-ROMs
- `dmesg`—Views information reported by the Linux kernel

Reference

- The Linux Keyboard and Console HOWTO—Andries Brouwer's tome on keyboard and console issues; includes many troubleshooting tips.
- http://www.x.org/—The X.Org Foundation, home of X11R7.
- http://www.alsa-project.org—Home page for the Advanced Linux Sound Architecture project, an alternative set of sound drivers for Linux.
- http://www.opensound.com—Commercial sound drivers for Linux.
- /usr/src/linux-2.6/Documentation/power/pci.txt—Patrick Mochel's document regarding PCI power-management routes for Linux kernel and PCI hardware support programmers.
- http://tldp.org/HOWTO/Modem-HOWTO.html—One of the newest HOWTOs on using modems with Linux.
- http://tldp.org/HOWTO/Serial-HOWTO.html—David S. Lawyer's Serial HOWTO, with additional information about Linux and serial port use.
- http://www.camiresearch.com/Data_Com_Basics/RS232_standard.html—A description and tutorial on the EIA232 (formerly RS232) standard.
- http://www.qbik.ch/usb/devices/—The place to check for compatibility of USB devices for Linux.
- http://www.linmodems.org—This site provides links to several drivers for controllerless modem use under Linux.
- http://www.linux1394.org/—Home page for the Linux IEEE 1394 project with new information, links to updated drivers, and lists of compatible chipsets and devices.
- http://groups.google.com/—Search Usenet groups through Google; another alternative to http://marc.theaimsgroup.com/.
- http://www.linuxquestions.org/—The Linux Questions site, a useful set of community forums that can help you find answers to your more frustrating problems.
- http://cdrecord.berlios.de—Home page for the `cdrecord` command and related utilities.

CHAPTER 3

Working with Gnome

IN THIS CHAPTER

- The Gnome Desktop Environment
- Eye Candy for the Masses
- Basic X Concepts
- Using X
- Starting X
- KDE—The Other Environment
- XFce
- Reference

Imagine a world of black screens with white text, or for those of you who remember, green screens with green text. That used to be the primary interface for users to access computers with. Thankfully computing has moved on significantly and has adopted the graphical user interface or GUI as standard on most desktop and workstation platforms.

Ubuntu is no different and its primary window manager is called Gnome (the Gnu Network Object Model Environment). Based upon the ethos of simplicity by design, Gnome offers a rich and full interface that you can easily use to be productive. The principle design objectives include an intuitive system, meaning that it should be easy to pick up and use as well as good localization/internationalization support and accessibility.

Gnome is founded upon the X Window System, the graphical networking interface found on many Linux distributions which provides the basis for a wide range of graphical tools and window managers. More commonly known as just *X*, it can also be referred to as X11R7 and X11 (such as that found on Mac OS X). Coming from the world-renowned Massachusetts Institute of Technology, X has gone through several versions, each of which has extended and enhanced the technology. The open source implementation is managed by the X.Org foundation, the board of which is made up of several key figures from the open source world.

The best way to think about how X works is to see it as a client/server system. The X server provides services to programs that have been developed to make the most of the graphical and networking capabilities that are available under the server and in the supported libraries. X.Org provides versions for many different platforms, including

Linux and Mac OS X. Originally implemented as XFree86, X.Org was forked when a row broke out over certain restrictions that were going to be included in the XFree86 license. Taking a snapshot of code that was licensed under the previous version of the license, X.Org drove forward with its own implementation based on the code. Almost in unison, most Linux distributions turned their back on XFree86 and switched their development and efforts to X.Org.

In this chapter, you will learn how to work with Gnome and also the version of X that is included with Ubuntu. We will look at the fundamentals of X, as well as how to get X to work with any upgrades that might affect it, such as a new graphics card or that new flat panel display you just bought. We will also take a look at some of the other Window Managers that are included with Ubuntu, including KDE and Xfce.

The Ubuntu Family

When people talk about Ubuntu, they generally mean the original distribution launched in 2004 that uses the Gnome window manager. However, there are a number of derivatives of Ubuntu that use other window managers, and they are freely available for you to download and use. For instance, Kubuntu uses the KDE window manager, while Xubuntu uses Xfce instead. Despite their very visual differences, all three rely on the same core system, and it is also easy for you to add that window manager to your existing system by using one of the *-desktop meta-packages. These packages are designed to install all the base applications that are linked to that version of Ubuntu, so installing the kubuntu-desktop package would automatically install all the software needed for Kubuntu, while also retaining your existing desktop environment.

The Gnome Desktop Environment

A desktop environment for X provides one or more window managers and a suite of clients that conform to a standard graphical interface based on a common set of software libraries. When they are used to develop associated clients, these libraries provide graphical consistency for the client windows, menus, buttons, and other onscreen components, along with some common keyboard controls and client dialogs. The following sections discuss the primary desktop environment that is included with Ubuntu: Gnome.

Gnome: The GNU Network Object Model Environment

The Gnome project, which was started in 1997, is the brainchild of programmer whiz Miguel de Icaza. Gnome provides a complete set of software libraries and clients. Gnome depends on a window manager that is Gnome-aware. This means that to provide a graphical desktop with Gnome elements, the window manager must be written to recognize

and use Gnome. Some compliant window managers that are Gnome-aware include `Compiz` (the default Gnome window manager), Enlightenment, Metacity, Window Maker, IceWM, and `beryl`.

Ubuntu uses Gnome's user-friendly suite of clients to provide a consistent and user-friendly desktop. Gnome clients are found under the `/usr/bin` directory, and Gnome configuration files are stored under the `/etc/gnome` and `/usr/share/gnome` directories, with user settings stored in the home directory under `.gnome` and `gnome2`.

A representative GNOME desktop, running the removable media preferences tool used for setting actions to events, is shown in Figure 3.1.

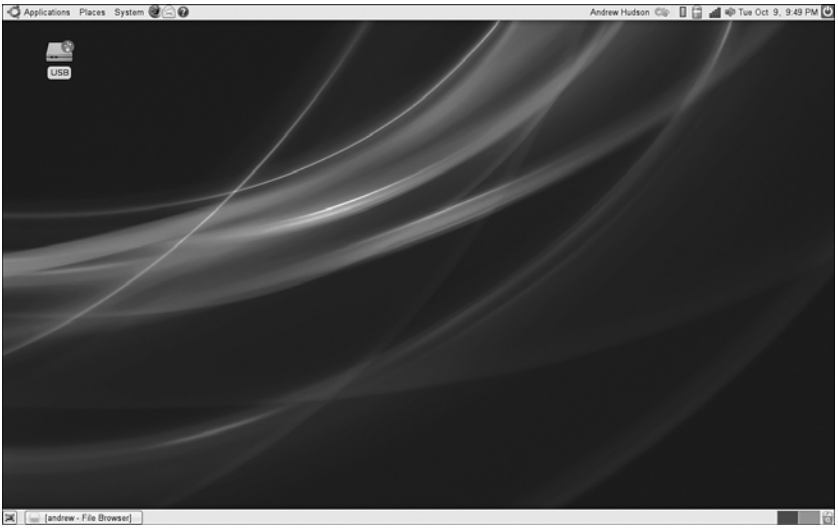


FIGURE 3.1 Ubuntu's Gnome desktop uses the Compiz window manager and offers a selection of Gnome themes.

You can configure your desktop in various ways and by using different menu items under the Preferences menu, which can be found as part of the main Desktop menu. The myriad of configurations options allow you to tailor every aspect of your system's look and feel. In Figure 3.2, you can see a selection of the preferences options available to you.

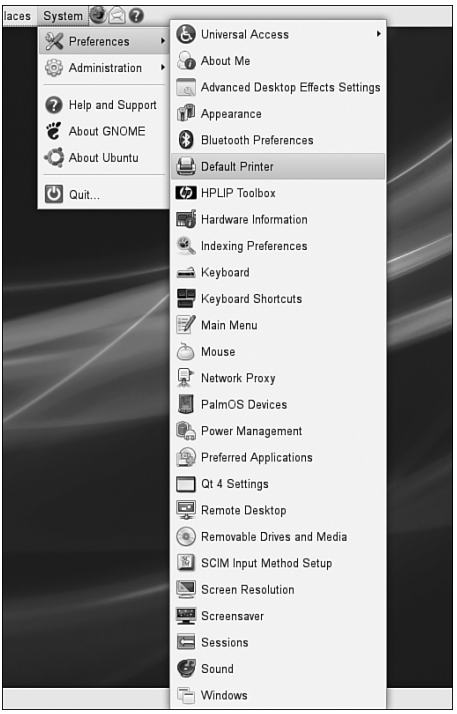


FIGURE 3.2 You can customize your Ubuntu desktop by using the Preference settings that are available in the System, Preferences menu.

Eye Candy for the Masses

Recent development work carried out on X has allowed the introduction of a number of hardware-accelerated effects within Ubuntu and its window managers. No longer do you have to drool at your Mac OS X-using colleagues when they work; now Ubuntu has a whole load of "wow" effects designed to add that professional touch to Linux.

Up until now, enabling these desktop effects has required a lot of work involving downloading specific packages and also configuring some of them using the console. However, with Ubuntu 7.10, this has been done away with and desktop effects are available out of the box, depending on whether your graphics card is powerful enough.

Ubuntu relies upon the Compiz window manager, which to most end users will not appear any differently to Metacity, the standard window manager in use by Ubuntu. You should already have the latest graphics card driver for your system as Ubuntu automatically gives you access to the proprietary driver through the Restricted Driver Manager. Check out Chapter 2 for more information on this tool.

Once you have verified your graphic driver situation, you will find a menu option under System, Preferences called Appearance (see Figure 3.3). Open it up and select the tab

called Visual Effects. By default this is set to Normal, but try setting it to Extra and see what happens. After a couple of seconds you may see your window decorations (title bar, minimize and maximize buttons) disappear and then reappear. It may seem that nothing has happened but grab hold of the window title bar and move it around. If everything has gone according to plan, then it should wobble! Click Close to save the settings and welcome to a world of fancy effects.

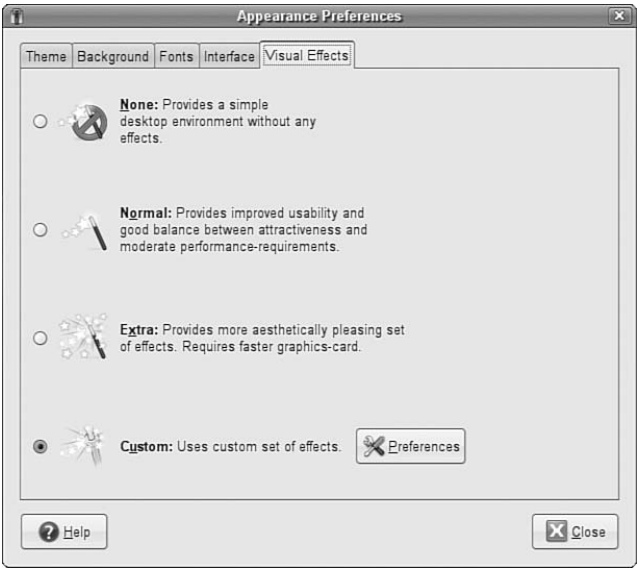


FIGURE 3.3 Use the Visual Effects tool to set the scene for some snazzy 3D effects.

The most obvious effect is that of "wobbly windows," which provide a fluid effect when you move your windows around the desktop area. Other effects include a smooth wipe from Desktop 1 to Desktop 2, activated by pressing Ctrl-Alt and either the left or right cursor key.

Basic X Concepts

The underlying engine of X11 is the X protocol, which provides a system of managing displays on local and remote desktops. The protocol uses a client/server model that allows an abstraction of the drawing of client windows and other decorations locally and over a network. An X server draws client windows, dialog boxes, and buttons that are specific to the local hardware and in response to client requests. The client, however, does not have to be specific to the local hardware. This means that system administrators can set up a network with a large server and clients and enable users to view and use those clients on workstations with totally different CPUs and graphics displays.

NOTE

What better way to demonstrate the capability of X to handle remote clients than by using its capabilities to produce this chapter. Although the OpenOffice.org file for this chapter resided on a Mac mini (running Ubuntu), the display and keyboard used were actually part of an Acer Ferrari notebook running Ubuntu 7.04 LTS, via an ethernet connection. Revisions were done using a Logitech keyboard and mouse of a desktop machine running Ubuntu 6.06.1, again connected to the Mac mini via X, but this time using a wireless connection.

Because X offers users a form of distributed processing, this means that Ubuntu can be used as a very cheap desktop platform for clients that connect to a powerful X server. The more powerful the X server, the larger the number of X-based clients that can be accommodated. This functionality can breathe new life into older hardware, pushing most of the graphical processing on to the server. A fast network is a must if you intend to run many X clients because X can become bandwidth-hungry.

X is hugely popular in the UNIX and Linux world for a variety of reasons. The fact that it supports nearly every hardware graphics system is a strong point, as well as strong multiplatform programming standards give it a solid foundation of developers committed to X. Another key benefit of X is its networking capability, which plays a central point in administration of many desktops and can also assist in the deployment of a thin-client computing environment. Being able to launch applications on remote desktops and also standardize installations serve to highlight the versatility of this powerful application.

More recent versions of X have also included support for shaped windows (that is, non-rectangular), graphical login managers (also known as *display managers*), and compressed fonts. Each release of X brings more features designed to enhance the user experience, including being able to customize how X client applications appear, right down to buttons and windows. Most office and home environments run Linux and X on their local machines. The more-enlightened companies and users harness the power of the networking features of X, enabling thin-client environments and allowing the use of customized desktops designed specifically for that company. Having applications launch from a single location makes the lives of system administrators a lot easier because they have to work on only one machine, rather than several.

Using X

X.Org 7.3 is the X server that is used with Ubuntu. The base `Xorg` distribution consists of 30 RPM packages (almost 120MB), which contain the server, along with support and development libraries, fonts, various clients, and documentation. An additional 1,000 or more X clients, fonts, and documentation are also included with Ubuntu.

NOTE

A full installation of X and related X.Org 7.3 files can consume more—usually much more—than 170MB of hard drive space. This is because additional clients, configuration files, and graphics (such as icons) are under the `/usr/bin` and `/usr/share` directory trees. You can pare excessive disk requirements by judiciously choosing which X-related packages (such as games) to install on workstations. However, with the increased capacity of most desktop PC hard drives today, the size requirements are rarely a problem, except in configuring thin-client desktops or embedded systems.

The `/usr` directory and its subdirectories contain the majority of `Xorg`'s software. Some important subdirectories are

- `/usr/bin`—This is the location of the X server and various X clients. (Note that not all X clients require active X sessions.)
- `/usr/include`—This is the path to the files necessary for developing X clients and graphics such as icons.
- `/usr/lib`—This directory contains required software libraries to support the X server and clients.
- `/usr/lib/X11`—This directory contains fonts, default client resources, system resources, documentation, and other files that are used during X sessions and for various X clients. You will also find a symbolic link to this directory, named `X11`, under the `/usr/lib` directory.
- `/usr/lib/modules`—This path to drivers and the X server modules used by the X server enables use of various graphics cards.

The main components required for an active local X session are installed on your system if you choose to use a graphical desktop. These components are the X server, miscellaneous fonts, a *terminal client* (that is, a program that provides access to a shell prompt), and a client known as a *window manager*. Window managers, which are discussed later in this chapter, administer onscreen displays, including overlapping and tiling windows, command buttons, title bars, and other onscreen decorations and features.

Elements of the `xorg.conf` File

The most important file for `Xorg` is the `xorg.conf` configuration file, which can be located in the `/etc/X11` directory. This file contains configuration information that is vital for X to function correctly, and is usually created during the installation of Ubuntu. Should you need to change anything post-install, you should use the `system-config-display` application, which we will cover later in this chapter. Information relating to hardware, monitors, graphics cards, and input devices is stored in the `xorg.conf` file, so be careful if you decide to tinker with it in a text editor!

Bullet Proof X

With Ubuntu 7.10 comes a new feature called Bullet Proof X. In short, it is designed to work no matter what may happen, so in the event of some cataclysmic event that destroys your main X system, you will still have some graphical way of getting yourself back into a fully functional X-based system. Thanks to Bullet Proof X you shouldn't need to edit your xorg.conf file, but it can be a good idea to understand what it is made up of, in case you ever need to troubleshoot an X problem. Let us take a look at the contents of the file so that you can get an idea of what X is looking for. The components, or sections, of the `xorg.conf` file specify the X session or *server layout,* along with pathnames for files that are used by the server, any options relating directly to the server, any optional support modules needed, information relating to the mouse and keyboard attached to the system, the graphics card installed, the monitor in use, and of course the resolution and color depth that Ubuntu uses. Of the 12 sections of the file, these are the essential components:

- `ServerLayout`—Defines the display, defines one or more screen layouts, and names input devices.
- `Files`—Defines the location of colors, fonts, or port number of the font server.
- `Module`—Tells the X server what graphics display support code modules to load.
- `InputDevice`—Defines the input devices, such as the keyboard and mouse; multiple devices can be used.
- `Monitor`—Defines the capabilities of any attached display; multiple monitors can be used.
- `Device`—Defines one or more graphics cards and specifies what optional features (if any) to enable or disable.
- `Screen`—Defines one or more resolutions, color depths, perhaps a default color depth, and other settings.

The following sections provide short descriptions of these elements; the `xorg.conf` man page contains full documentation of all the options and other keywords you can use to customize your desktop settings.

The `ServerLayout` Section

As noted previously, the `ServerLayout` section of the `xorg.conf` file defines the display and screen layouts, and it names the input devices. A typical `ServerLayout` section from an automatically configured `xorg.conf` file might look like this:

```
Section "ServerLayout"
        Identifier     "single head configuration"
        Screen      0  "Screen0" 0 0
        InputDevice    "Mouse0" "CorePointer"
        InputDevice    "Keyboard0" "CoreKeyboard"
        InputDevice    "DevInputMice" "AlwaysCore"
EndSection
```

In this example, a single display is used (the numbers designate the position of a screen), and two default input devices, `Mouse0` and `Keyboard0`, are used for the session.

The `Files` Section

The `Files` section of the `xorg.conf` file might look like this:

```
Section "Files"
    RgbPath     "/usr/lib/X11/rgb"
    FontPath   "unix/:7100"
EndSection
```

This section lists available session colors (by name, in the text file `rgb.txt`) and the port number to the X font server. The font server, `xfs`, is started at boot time and does not require an active X session. If a font server is not used, the `FontPath` entry could instead list each font directory under the `/usr/lib/X11/fonts` directory, as in this example:

```
FontPath "/usr/lib/X11/fonts/100dpi"
FontPath "/usr/lib/X11/fonts/misc"
FontPath "/usr/lib/X11/fonts/75dpi"
FontPath "/usr/lib/X11/fonts/type1"
FontPath "/usr/lib/X11/fonts/Speedo"
...
```

These directories contain the default compressed fonts that are available for use during the X session. The font server is configured by using the file named `config` under the `/etc/X11/fs` directory. This file contains a listing, or catalog, of fonts for use by the font server. By adding an `alternate-server` entry in this file and restarting the font server, you can specify remote font servers for use during X sessions. This can help centralize font support and reduce local storage requirements (even though only 25MB is required for the almost 5,000 fonts installed with Ubuntu and X).

The `Module` Section

The `Module` section of the `xorg.conf` file specifies loadable modules or drivers to load for the X session. This section might look like this:

```
Section "Module"
        Load  "dbe"
        Load  "extmod"
        Load  "fbdevhw"
        Load "glx"
        Load "record"
        Load  "freetype"
        Load  "type1"
        Load  "dri"

EndSection
```

These modules can range from special video card support to font rasterizers. The modules are located in subdirectories under the `/usr/lib/modules` directory.

The `InputDevice` Section

The `InputDevice` section configures a specific device, such as a keyboard or mouse, as in this example:

```
Section "InputDevice"
        Identifier  "Keyboard0"
        Driver      "kbd"

        Option  "XkbModel"      "pc105"
        Option  "XkbLayout"     "us"
EndSection
Section "InputDevice"
        Identifier  "Mouse0"
        Driver      "mouse"
        Option      "Protocol" "IMPS/2"
        Option      "Device" "/dev/input/mice"
        Option      "ZAxisMapping" "4 5"
        Option      "Emulate3Buttons" "yes"
EndSection
```

You can configure multiple devices, and there might be multiple `InputDevice` sections. The preceding example specifies a basic keyboard and a two-button PS/2 mouse (actually, a Dell touchpad pointer). An `InputDevice` section that specifies use of a USB device could be used at the same time (to enable mousing with PS/2 and USB pointers) and might look like this:

```
Section "InputDevice"
        Identifier  "Mouse0"
        Driver      "mouse"
        Option      "Device" "/dev/input/mice"
        Option      "Protocol" "IMPS/2"
        Option      "Emulate3Buttons" "off"
        Option      "ZAxisMapping" "4 5"
EndSection
```

The `Monitor` Section

The `Monitor` section configures the designated display device as declared in the `ServerLayout` section, as shown in this example:

```
Section "Monitor"
        Identifier   "Monitor0"
        VendorName   "Monitor Vendor"
        ModelName    "Monitor Model"
        DisplaySize  300     220
```

```
        HorizSync   31.5-48.5
        VertRefresh 50-70
        Option "dpms"
EndSection
```

Note that the X server automatically determines the best video timings according to the horizontal and vertical sync and refresh values in this section. If required, old-style modeline entries (used by distributions and servers prior to XFree86 4.0) might still be used. If the monitor is automatically detected when you configure X (see the "Configuring X" section, later in this chapter), its definition and capabilities are inserted in your `xorg.conf` file from the `MonitorsDB` database. This database contains more than 600 monitors and is located in the `/usr/share/hwdata` directory.

The `Device` Section

The `Device` section provides details about the video graphics chipset used by the computer, as in this example:

```
Section "Device"
        Identifier  "Videocard0"
        Driver      "radeon"
        VendorName  "Videocard vendor"
        BoardName   "ATI Radeon Mobility M6"

EndSection
```

This example identifies an installed video card as using an ATI Mobility M6 graphics chipset. The `Driver` entry tells the `Xorg` server to load the `radeon_drv.o` module from the `/usr/lib/modules/drivers` directory. Different chipsets have different options. For example, here's the entry for a NeoMagic video chipset:

```
Section "Device"
        Identifier   "NeoMagic (laptop/notebook)"
        Driver       "neomagic"
        VendorName   "NeoMagic (laptop/notebook)"
        BoardName     "NeoMagic (laptop/notebook)"
    Option     "externDisp"
    Option     "internDisp"
EndSection
```

In this example, the `Device` section specifies the driver for the graphics card (`neomagic_drv.o`) and enables two chipset options (`externDisp` and `internDisp`) to allow display on the laptop's LCD screen and an attached monitor.

The `Xorg` server supports hundreds of different video chipsets. If you configure X11 but subsequently change the installed video card, you need to edit the existing `Device` section or generate a new `xorg.conf` file, using one of the X configuration tools discussed in this chapter, to reflect the new card's capabilities. You can find details about options for some

chipsets in a companion man page. You should look at these sources for hints about optimizations and troubleshooting.

The `Screen` Section

The `Screen` section ties together the information from the previous sections (using the `Screen0`, `Device`, and `Monitor Identifier` entries). It can also specify one or more color depths and resolutions for the session. Here's an example:

```
Section "Screen"
        Identifier "Screen0"
        Device     "Videocard0"
        Monitor    "Monitor0"
        DefaultDepth     24
        SubSection "Display"
                Viewport   0 0
                Depth     16
                Modes    "1024x768" "800x600" "640x480"
        EndSubSection
EndSection
```

In this example, a color depth of thousands of colors and a resolution of 1024×768 is the default, with optional resolutions of 800×600 and 640×480. Multiple `Display` subsection entries with different color depths and resolutions (with settings such as `Depth 24` for millions of colors) can be used if supported by the graphics card and monitor combination. You can also use a `DefaultDepth` entry (which is `24`, or thousands of colors, in the example), along with a specific color depth to standardize display depths in installations.

You can also specify a desktop resolution larger than that supported by the hardware in your monitor or notebook display. This setting is known as a *virtual* resolution in the `Display` subsection. This allows, for example, an 800×600 display to pan (that is, slide around inside) a virtual window of 1024×768.

NOTE

If your monitor and graphics card support multiple resolutions and the settings are properly configured, you can use the key combination of Ctrl+Alt+Keypad+ or Ctrl+Alt+Keypad- to change resolutions on the fly during your X session.

Configuring X

Although the Ubuntu installer can be used to configure X during installation, problems can arise if the PC's video card is not recognized. If you are unable to configure X during installation (refer to Chapter 1, "Installing Ubuntu"), do not specify booting to a graphical configuration and skip the X configuration portion of the installation. Note that some installs, such as for servers, don't require that X be configured for use to support active X

sessions, but might require installation of X and related software to support remote users and clients.

You can use the following configuration tools, among others, to create a working `xorg.conf` file:

- `displayconfig-gtk`—This is Ubuntu's graphical configuration tool, which launches a handy GUI tool to help create an `xorg.conf` file.
- `Xorg`—The X server itself can create a skeletal working configuration.

The following sections discuss how to use each of these software tools to create a working `xorg.conf` file.

Configuring X with the displayconfig-gtk Client

You can use the displayconfig-gtk client to create or update an `xorg.conf` file. You can start by clicking the Screens and Graphics menu item found under System, Administration if you are already running X. You will be asked to enter your password in order to open the application; once you've done this, you'll see a window similar to Figure 3.4.

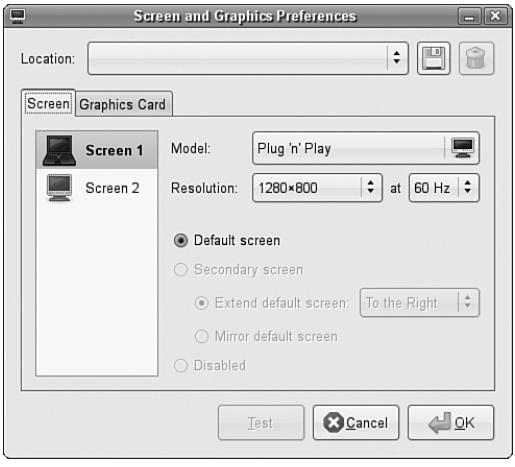


FIGURE 3.4 The *`displayconfig-gtk`* client provides a graphical configuration interface for creating or updating a system's *`xorg.conf`* file. Here you see the Screen Settings main screen, offering resolution and color-depth settings.

The Screen Settings window shows the current selected screen, along with details of the model, resolutions available, the refresh rate, and orientation. In the event that you are using two screens (dual- or multi-head displays), then you have the ability to select whether it is the primary or secondary screen, and the physical location of it.

If you click the Graphics Card tab, other configuration options become available, as shown in Figure 3.5.

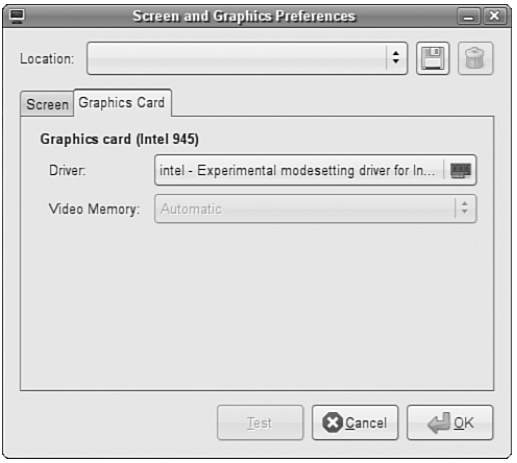


FIGURE 3.5 *displayconfig-gtk*'s hardware settings are used to configure a video card for X11R7.

Click the name of the graphics card to open up a list of alternative cards (shown in Figure 3.6). Simply select a manufacturer to see a list of cards associated with that manufacturer and to choose the correct one for your X sessions.

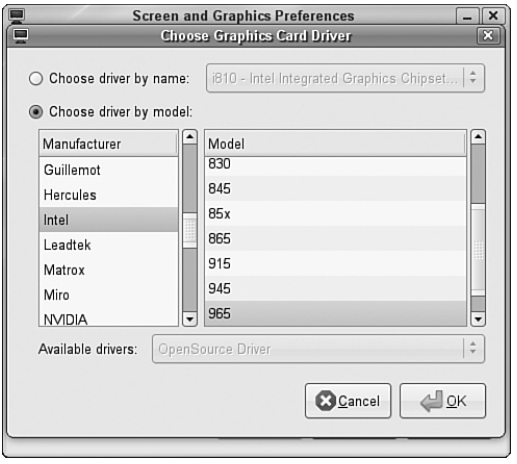


FIGURE 3.6 You can scroll to select the correct graphics card to use for your X sessions.

When you have finished your changes to your X server settings, you can quickly test them to see if they will work by clicking the Test button. Ubuntu will then attempt to use your chosen settings, allowing you to either choose to keep them or revert back to the original settings. If you are happy with the new mode, then click Keep Configuration. Finally, click the OK button to close the dialog box. You may then see a dialog advising that you have to log out and then log back in (or exit your X session and restart X) to use the new settings.

The new settings will be stored in a new `xorg.conf` file under the `/etc/X11` directory. If you find that the new settings do not work, you can simply copy the backup `xorg.conf` file named `xorg.conf.backup` to `xorg-conf` in the same directory to revert to your original settings.

Using `Xorg` to Configure X

You can create the `xorg.conf` file manually by typing one from scratch using a text editor, but you can also create one automatically by using the Xorg server or configuration utilities (as discussed in the previous sections). As the root operator, you can use the following on the server to create a test configuration file:

```
# X -configure
```

After you press Enter, a file named `xorg.conf.new` is created in root's home directory, the `/root` directory. You can then use this file for a test session, like this:

```
# X –config /root/xorg.conf.new
```

Starting X

You can start X sessions in a variety of ways. The Ubuntu installer sets up the system to have Linux boot directly to an X session using a *display manager* (that is, an X client that provides a graphical login). After you log in, you use a local session (running on your computer) or, if the system is properly configured, an X session running on a remote computer on the network. Logging in via a display manager requires you to enter a username and password. You can also start X sessions from the command line. The following sections describe these two methods.

NOTE

If you have used the Server install your system will boot to a text login—see "Command Line Quickstart" in Chapter 4 for more details on what to do here.

Using a Display Manager

An X display manager presents a graphical login that requires a username and password to be entered before access is granted to the X desktop. It also allows you to choose a

different desktop for your X session. Whether or not an X display manager is presented after you boot Linux is controlled by a *runlevel*—a system state entry in `/etc/event.d/`. The following runlevels are defined in files:

```
# 0 - halt (Do NOT set initdefault to this)
# 1 - Multiuser text mode
# 2 - X – graphical multiuser mode
# 3 - Same as 2
# 4 – Same as 2
# 5 – Same as 2
# 6 - reboot (Do NOT set initdefault to this)
```

Runlevels 2–5 are used for multiuser mode with a graphical X login via a display manager; booting to runlevel 1 provides a console, or text-based, login. The `initdefault` setting in the `/etc/events.d/rc-default` file determines the default run level. Historically, Ubuntu used the `/etc/inittab` file to handle runlevels, but with the introduction of upstart, this file no longer exists. Instead, there are several files under the `/etc/events.d/` directory, including an individual file for each of the virtual consoles (accessible by pressing Ctrl-Alt-F1 to F7).

Configuring `gdm`

The `gdm` display manager is part of the Gnome library and client distribution included with Ubuntu and provides a graphical login when a system boots directly to X. Its login (which is actually displayed by the `gdmlogin` client) hosts pop-up menus of window managers, languages, and system options for shutting down (halting) or rebooting the workstation. Although you can edit (as root) `gdm.conf` under the `/etc/gdm` directory to configure `gdm`, a much better way to configure Gnome 's display manager is to use the `gdmsetup` client.

You can use the `gdmsetup` client to configure many aspects of gdm. You launch this client from the System Menu, under Administration, Login Window. You will be prompted for your password and after you press Enter, you see the GDM Setup window, as shown in Figure 3.7.

You can specify settings for security, remote network logins, the X server, and session and session chooser setup by clicking on the tabs in the GDM Setup dialog.

Configuring `kdm`

The `kdm` client, which is part of KDE (which we cover later), offers a graphical login similar to `gdm`. You configure `kdm` by running the System Settings option in the Kicker (K Menu). When the System Settings window opens, click the Advanced tab and click on the Login Manager icon to see the window shown in Figure 3.8.

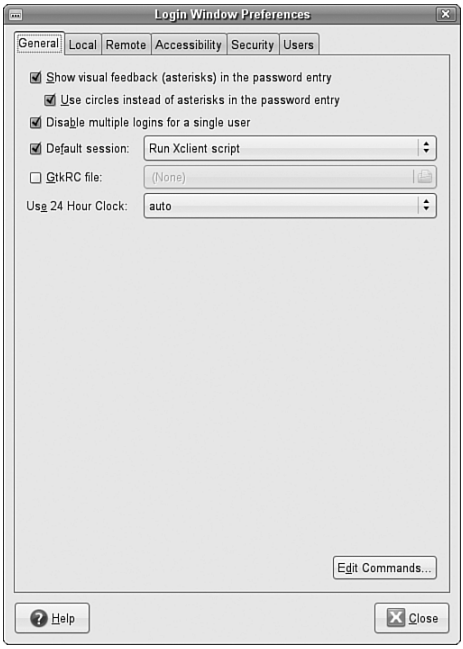


FIGURE 3.7 You use *gdmsetup* to configure the *gdmlogin* screen when using *gdm* as a display manager.

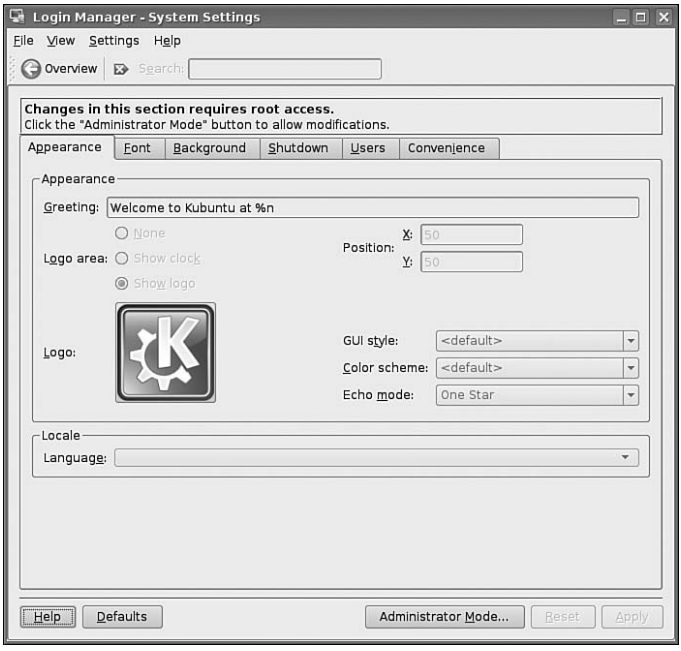


FIGURE 3.8 You configure *kdm* by choosing tabs and settings in the Login Manager dialog box.

To make any changes to the KDE display manager while logged in as a regular user, you must first click the Administrator Mode button, and then enter your password. You can click on a tab in the Control Center dialog to set configuration options. Options in these tabs allow you to control the login display, prompts, user icons, session management, and configuration of system options (for shutting down or rebooting). After you make your configuration choices in each tab, click the Apply button to apply the changes immediately; otherwise, the changes are applied when the X server restarts.

Using the `xdm` Display Manager

The `xdm` display manager is part of the `Xorg` distribution and offers a bare-bones login for using X. Although it is possible to configure `xdm` by editing various files under the `/etc/X11/xdm` directory, Gnome and KDE offer a greater variety of options in display manager settings. The default `xdm` login screen's display is handled by the `xsetroot` client, which is included with `Xorg`, and Owen Taylor's `xsri` client, as specified in the file `Xsetup_0` in the `xdm` directory under `/etc/X11`. The `xsri` client can be used to set the background color of the login display's desktop and to place an image in the initial display.

Changing Window Managers

Ubuntu makes it fairly painless to switch to another window manager or desktop environment. By *desktop environment*, we refer to not only the window manager but also the suite of related applications such as productivity or configuration tools.

First of all, you need to ensure that you have the relevant desktop environment installed on your system and the easiest way to do this is by installing the relevant *-desktop package (as described earlier in the sidebar "The Ubuntu Family"). You can do this either at the command line with a simple `apt-get install kubuntu-desktop` (in the case of a KDE desktop) or you can use synaptic to install the relevant packages; just search for desktop and look for Xubuntu or Kubuntu. Once the download and installation has completed (you might want to grab a coffee while you wait), you are all set to change environment.

The first thing you need to do is to log out of Ubuntu by going to System, Quit. Then select the Log Out option. After a couple of seconds, you will arrive at the graphical login prompt (as shown in Figure 3.9). At this point, press the F10 key to open the drop-down menu and click Select Session.

Here you can choose to switch your session to KDE (for example) and once you've clicked OK, you'll be asked whether you want to make this a permanent change or just for one session (i.e., to try it out). At this point, it's probably better to choose For This Session Only. Enter your user name and password to log in to your new desktop environment. Don't worry; because you selected For This Session Only you haven't yet committed yourself to switching permanently. Take the opportunity to try the new environment and make up your mind if you want to switch to it full time.

FIGURE 3.9 Use the Select Session option to select the desktop environment of your choice before you log in.

If you have made up your mind to switch to another environment full time, you can do it by taking one of two routes. The first one is to go to the Login Window (found under System, Administration, Login Window under Gnome and also found under the Kicker Menu, System, Login Window within KDE) and select the relevant window manager from the Default Session drop-down list, as shown in Figure 3.10.

Alternatively you can do it from the login window by pressing F10, clicking Select Session, choosing your desired window manager and selecting Make Default. This will set your default window manager to your specific choice.

Regardless of which route you take, it's easy to change back should you not like your new environment.

The next section gives you a brief introduction to the two other main window managers other than Gnome that are available either in Kubuntu or Xubuntu.

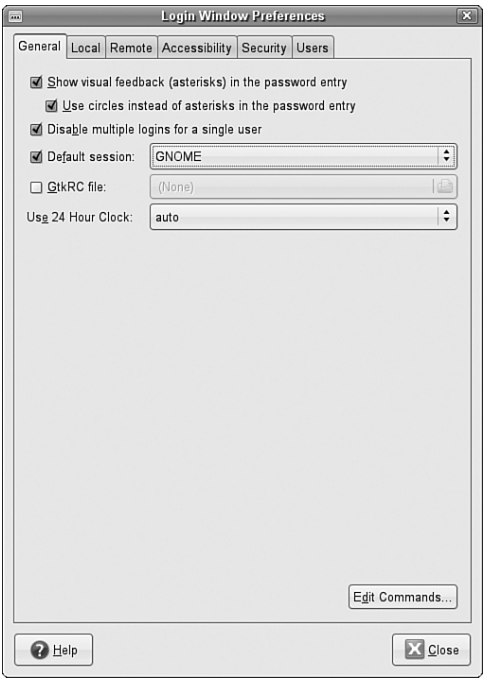


FIGURE 3.10 When you're ready to switch to another environment, use the Login Window to make the jump.

KDE—The Other Environment

One of the great things about Ubuntu is the choice it allows you. When Ubuntu was originally released, it favored the Gnome desktop, but fairly early on the decision was made to create a KDE alternative named Kubuntu. If you install the `kubuntu-desktop` package (either using `apt-get` or `synaptic`), you will have access to KDE and the associated Kubuntu applications.

KDE is somewhat different from Gnome in that it uses the QT libraries rather than GTK libraries, so the windows and other elements look different. Linus Torvalds himself has in the past expressed a distinct preference for KDE, and it also helps that KDE allows you to customize your working environment in pretty much any way imaginable.

A standard KDE desktop is shown in Figure 3.11.

FIGURE 3.11 Unlimited customization options abound within KDE; just be prepared to switch!

XFce

XFce is another desktop environment, suitable for computers with not much memory or processing power. It's based upon the same GTK libraries that are in use by Gnome, so it shares some of the look and feel of the Gnome desktop. That said, it comes bundled with a number of Xfce-specific applications to replace Gnome tools such as nautilus.

Some people just prefer the simplicity of Xfce, so we will leave it up to you if you want to use it. You can get access to it by installing the `xubuntu-desktop` package (either with `apt-get` or synaptic) and a sample desktop is shown in Figure 3.12.

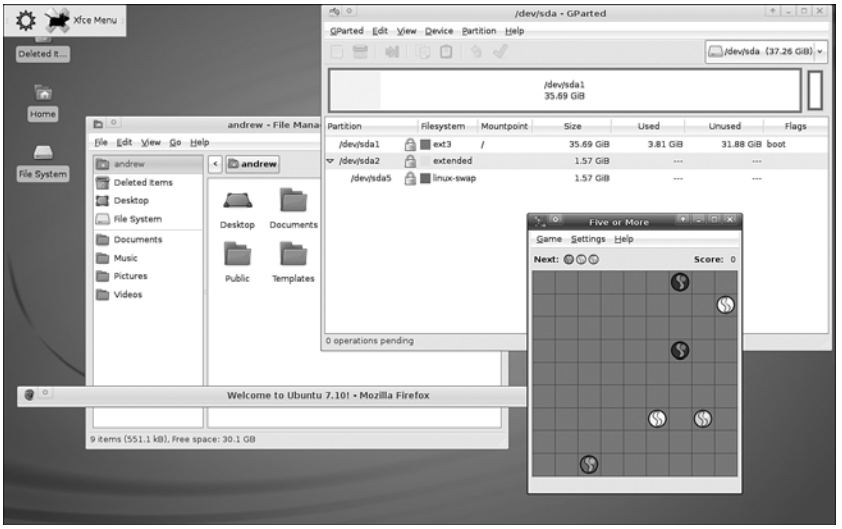


FIGURE 3.12 XFce—lightweight and simplicity, molded together in a great package.

Reference

- http://www.x.org/—Curators of the X Window System.
- http://www.x.org/Downloads_mirror.html—Want to download the source to the latest revision of X? Start at this list of mirror sites.
- http://www.xfree86.org/—Home of The XFree86 Project, Inc., which provided a graphical interface for Linux for nearly 10 years.
- http://www.kde.org/—The place to get started when learning about KDE and the latest developments.
- http://www.gnome.org/—The launch point for more information about Gnome, links to new clients, and Gnome development projects.
- http://www.compiz.org/—The official page for the Compiz project, including Compiz Fusion, a set of plugins for the Compiz window manager.
- http://www.xfce.org/—The official home page for the XFce project.

CHAPTER 4

Command Line Quickstart

IN THIS CHAPTER

- What Is the Command Line?
- Logging In to and Working with Linux
- Using the Text Editors
- Working with Permissions
- Working as Root
- Reading Documentation
- Reference

The command line is one of the most powerful tools available for use with Ubuntu, and indeed Linux. Knowledge of the commands associated with it and also how to string them together will make working with Ubuntu that much easier.

This chapter looks at some of the basic commands that you need to know to be productive at the command line. You will find out how to get to the command line, and also get to grips with some of the commands used to navigate around the file system. Later on in this book is the Command Line Masterclass (Chapter 30), which explores the subject in more depth.

What Is the Command Line?

Hang around Linux users for any length of time and it will not be long before you hear them speak in hushed tones about the command line or the terminal. Quite rightly too, as the command line offers a unique and powerful way to interact with Linux. However, for the most part, you may never need to access the command line because Ubuntu offers a slew of graphical tools that enable you to configure most things on your system.

But sometimes things go wrong and you may not have the luxury of a graphical interface to work with. It is in these situations that a fundamental understanding of the command line and its uses can be a real life saver.

It is tempting to think of the command line as the product of some sort of black and arcane art, and in some ways it can appear to be extremely difficult and complicated to use. However, perseverance is key and by the end of this

chapter you should at least be comfortable with using the command line and ready to move onto Chapter 30, "Command Line Masterclass."

More importantly, though, you will be able to make your way around a command line–based system, which you are likely to encounter if you work within a server environment.

This chapter introduces you to a number of commands, including commands that enable you to do the following tasks:

- **Perform routine tasks**—Logging in and out, using the text console, changing passwords, listing and navigating directories
- **Implement basic file management**—Creating files and folders, copying or moving them around the file system, renaming and ultimately deleting them (if necessary)
- **Execute basic system management**—Shutting down or rebooting, reading man pages, and using text-based tools to edit system configuration files

The information in this chapter is valuable for individual users or system administrators who are new to Linux and are learning to use the command line for the first time.

TIP

Those of you who have used a computer for many years will probably have come into contact with MS-DOS, in which case being presented with a black screen will fill you with a sense of nostalgia. Don't get too comfy; the command line in Linux is far superior to its distant MS-DOS cousin. Whereas MS-DOS skills are transferable only to other MS-DOS environments, the skills that you learn at the Linux command line can be transferred easily to other Unix-like operating systems, such as Solaris, OpenBSD, FreeBSD, and even Mac OS X, which allows you access to the terminal.

User Accounts

One concept you will have to get used to is that of user-based security. By and large, only two types of users will access the system as actual users. The first type is the regular user, of which you created one when you started Ubuntu for the first time (see Chapter 1 "Installing Ubuntu"). These users can change anything that is specific to them, such as the wallpaper on the desktop, their personal preferences, and so on. Note that the emphasis should be on anything that is specific to them, as it prevents regular users from making system-wide changes that could affect other users.

To make system-wide changes, you need to use super-user privileges, which you should have if your account was the first one specified (that is, when you specified a user during the installation). With super-user privileges you basically have access to the entire system and can carry out any task, even destructive ones! To use your super-user privileges you need to prefix the command you wish to execute with the command `sudo`. When you press Enter (after typing the remaining command), you will be prompted for your password, which you should type in followed by the Enter key. Ubuntu will then carry out the command, but with super-user privileges.

An example of the destructive nature of working as the super-user can be found in the age-old example of sudo `rm -rf /`, which erases all the data on your hard drive. You need to be especially careful when using your super-user privileges; otherwise, you may make irreparable damage to your system.

Don't let this worry you, though, as the ability to work as the super-user is fundamental to a healthy Linux system. Without it you would not be able to install new software, edit system configuration files, or do any number of administration tasks. By the end of this chapter, you will feel comfortable working with your super-user privileges and be able to adequately administer your system.

Ubuntu works slightly differently to other Linux distributions by giving users super-user privileges by default. If you work with any other Linux distro, you will quickly come across the `root` user, which is a super-user account. So rather than having to type in `sudo` before every command, the root account can simply issue the command and not have to worry about entering a password. You can tell when you are working at a root prompt because you will see the pound sign (#). Within Ubuntu, the `root` account is disabled by default in preference to giving super-user privileges to users. If you wish to enable the root account, then issue the command `sudo passwd`. When prompted, enter your user password. You will then be asked for a new UNIX password; this will be the password for the root account, so make sure and remember it. You will also be prompted to repeat the password, in case you've made any mistakes. Once you've typed it in and pressed Enter, the `root` account will now be active. You'll find out how to switch to `root` later on.

An alternative way of getting a root prompt, without having to enable the root account, is to issue the command `sudo -i`. After entering your password, you will find yourself at a root prompt (#). Do what you need to do and when you are finished, type `exit` and press Enter to return to your usual prompt.

As with most things, Ubuntu offers you a number of ways to access the command line. You can use the Terminal entry in Applications, Accessories, but by far the simplest way is to press Ctrl + Alt + F1. Ubuntu switches to a black screen and a traditional login prompt that resembles the following:

```
Ubuntu 7.10 (Gutsy) hostname tty1
Login:
```

TIP

This is actually one of six virtual consoles that Ubuntu provides for your use. After you have accessed a virtual console, you can use the Alt key and F1 through F6 to switch to a different console. If you want to get back to the graphical interface, press Alt + F7. You can also switch between consoles by holding the Alt key and pressing either the left or the right cursor key to move down or up a console, such as tty1 to tty2.

Ubuntu is waiting for you to log in as a user, so go ahead and enter your username and press the return key. Ubuntu then prompts you for your password, which you should

enter. Note that Ubuntu does not show any characters while you are typing your password in. This is a good thing because it prevents any shoulder surfers from seeing what you've typed or the length of the password.

Hitting the Return key drops you to a shell prompt, signified by the dollar sign:

```
andrew@ubuntu ~]$
```

This particular prompt tells me that I am logged in as the user `andrew` on the system `ubuntu` and I am currently in my home directory (Linux uses the tilde as shorthand for the home directory).

TIP

Navigating through the system at the command line can get confusing at times, especially when a directory name occurs in several different places. Fortunately, Linux includes a simple command that tells you exactly where you are in the file system. It's easy to remember because the command is just an abbreviation of present working directory, so type **`pwd`** at any point to get the full path of your location. For example, typing `pwd` after following these instructions shows `/home/yourusername`, meaning that you are currently in your home directory.

Using the `pwd` command can save you a lot of frustration when you have changed directory half a dozen times and have lost track.

Another way to quickly access the terminal is to go to Applications, Accessories and choose the Terminal entry. Ubuntu opens up `gnome-terminal`, which allows you to access the terminal while remaining in Gnome. This time, the terminal appears as black text on a white background. Accessing the terminal this way, or by using the Ctrl + Alt + F1 method makes no difference because you are interacting directly with the terminal itself.

Navigating Through the File System

Use the `cd` command to navigate through the Ubuntu file system. This command is generally used with a specific directory location or pathname, like this:

```
$ cd /etc/apt/
```

Under Ubuntu, the cd command can also be used with several shortcuts. For example, to quickly move up to the *parent* (higher-level) directory, use the cd command like this:

```
$ cd ..
```

To return to one's home directory from anywhere in the Linux file system, use the `cd` command like this:

```
$ cd
```

You can also use the `$HOME` shell environment variable to accomplish the same thing. Type this command and press Enter to return to your home directory:

```
$ cd $HOME
```

You can accomplish the same thing by using the tilde (~) like this:

```
$ cd ~
```

Don't forget the `pwd` command to remind you where you are within the file system!

Another important command to use is the `ls` command, which lists the contents of the current directory. It's commonly used by itself, but a number of options (or switches) available for `ls` give you more information. For instance, the following command returns a listing of all the files and directories within the current directory, including any hidden files (denoted by a `.`prefix) as well as a full listing, so it will include details such as the permissions, owner and group, size, and last modified time and date:

```
$ ls -al
```

You can also issue the command

```
$ ls -R
```

which scans and lists all the contents of the subdirectories of the current directory. This might be a lot of information, so you may want to redirect the output to a text file so you can browse through it at your leisure by using the following:

```
$ ls -alR > listing.txt
```

We've included a table showing some of the top-level directories that are part of a standard Linux distro in Table 4.1.

TABLE 4.1 Basic Linux Directories

Name	Description
`/`	The root directory
`/bin`	Essential commands
`/boot`	Boot loader files, Linux kernel
`/dev`	Device files
`/etc`	System configuration files
`/home`	User home directories
`/initrd`	Initial RAM disk boot support (used during boot time)
`/lib`	Shared libraries, kernel modules
`/lost+found`	Directory for recovered files (if found after a file system check)
`/media`	Mount point for removable media, such as DVDs and floppy disks

TABLE 4.1 Continued

Name	Description
`/mnt`	Usual mount point for local, remote file systems
`/opt`	Add-on software packages
`/proc`	Kernel information, process control
`/root`	Super-user (root home)
`/sbin`	System commands (mostly root only)
`/srv`	Holds information relating to services that run on your system
`/sys`	Real-time information on devices used by the kernel
`/tmp`	Temporary files
`/usr`	Secondary software file hierarchy
`/var`	Variable data (such as logs); spooled files

Some of the important directories in Table 4.1, such as those containing user and root commands or system configuration files, are discussed in the following sections. You use and edit files under these directories when you use Ubuntu.

Linux also includes a number of GNU commands you can use to search the file system. These include the following:

- `whereis` *`command`*—Returns the location of the command and its man page.
- `whatis` *`command`*—Returns a one-line synopsis from the command's man page.
- `locate` *`file`*—Returns locations of all matching file(s); an extremely fast method of searching your system because `locate` searches a database containing an index of all files on your system. However, this database (about 4MB in size and named `slocate.db`, under the `/var/lib/slocate` directory) is built daily at 4:20 a.m. by default, and does not contain pathnames to files created during the workday or in the evening. If you do not keep your machine on constantly, you can run the `updatedb` command either using `sudo` or by using the `root` account to manually start the building of the database.
- `apropos` *`subject`*—Returns a list of commands related to subject.

Managing Files with the Shell

Managing files in your home directory involves using one or more easily remembered commands. If you have any familiarity with the now-ancient DOS, you recognize some of these commands (although their names are different from those you remember). Basic file management operations include paging (reading), moving, renaming, copying, searching, and deleting files and directories. These commands include the following:

- `cat` *`filename`*—Outputs contents of filename to display
- `less` *`filename`*—Allows scrolling while reading contents of filename

- `mv file1 file2`—Renames *file1* to *file2*
- `mv file dir`—Moves file to specified directory
- `cp file1 file2`—Copies *file1* and creates *file2*
- `rm file`—Deletes file
- `rmdir dir`—Deletes directory (if empty)
- `grep string file(s)`—Searches through files(s) and displays lines containing matching string

Note that each of these commands can be used with pattern-matching strings known as *wildcards* or *expressions*. For example, to delete all files in the current directory beginning with the letters `abc`, you can use an expression beginning with the first three letters of the desired filenames. An asterisk (`*`) is then appended to match all these files. Use a command line with the `rm` command like this:

```
$ rm abc*
```

Linux shells recognize many types of filenaming wildcards, but this is different from the capabilities of Linux commands supporting the use of more complex expressions. You learn more about using wildcards in Chapter 11.

NOTE

Learn more about using expressions by reading the `grep` manual pages (`man grep`).

Working with Compressed Files

Another file management operation is compression and decompression of files, or the creation, listing, and expansion of file and directory archives. Linux distributions usually include several compression utilities you can use to create, compress, expand, or list the contents of compressed files and archives. These commands include the following:

- `bunzip2`—Expands a compressed file
- `bzip2`—Compresses or expands files and directories
- `gunzip`—Expands a compressed file
- `gzip`—Compresses or expands files and directories
- `tar`—Creates, expands, or lists the contents of compressed or uncompressed file or directory archives known as *tape archives* or *tarballs*

Most of these commands are easy to use. The `tar` command, however, has a somewhat complex (although capable) set of command-line options and syntax. Even so, you can

quickly learn to use `tar` by remembering a few simple invocations on the command line. For example, to create a compressed archive of a directory, use `tar`'s `czf` options like this:

```
$ tar czf dirname.tgz dirname
```

The result is a compressed archive (a file ending in `.tgz`) of the specified directory (and all files and directories under it). Add the letter `v` to the preceding options to view the list of files added during compression and archiving. To list the contents of the compressed archive, substitute the `c` option with the letter `t`, like this:

```
$ tar tzf archive
```

Of course, if many files are in the archive, a better invocation (to easily read or scroll through the output) is

```
$ tar tzf archive ¦ less
```

To expand the contents of a compressed archive, use `tar`'s `zxf` options, like so:

```
$ tar zxf archive
```

The `tar` utility decompresses the specified archive and extracts the contents in the current directory.

Use Essential Commands from the `/bin` and `/sbin` Directories

The `/bin` directory (about 5MB if you do a full install) contains essential commands used by the system for running and booting Linux. In general, only the root operator uses the commands in the `/sbin` directory. Many (though not all) these commands are *statically* linked which means that such commands do not depend on software libraries residing under the `/lib` or `/usr/lib` directories. Nearly all the other applications on your system are *dynamically* linked—meaning that they require external software libraries (also known as *shared* libraries) to run.

Use and Edit Files in the `/etc` Directory

More than 65MB of system configuration files and directories reside under the `/etc` directory if you install all the software included with this book. Some major software packages, such as Apache, OpenSSH, and `xinetd`, have directories of configuration files under `/etc`. Other important system-related configuration files in `/etc` are

- `fstab`—The file system table is a text file listing each hard drive, CD-ROM, floppy, or other storage device attached to your PC. The table indexes each device's partition information with a place in your Linux file system (directory layout) and lists other options for each device when used with Linux (see Chapter 32, "Kernel and Module Management"). Nearly all entries in `fstab` can be manipulated by root using the `mount` command.

- `modprobe.d/`—This folder holds all the instructions to load kernel modules that are required as part of the system startup, and replaces the historic modprobe.conf file.
- `passwd`—The list of users for the system, along with user account information. The contents of this file can be changed by various programs, such as `useradd` or `chsh`.
- `printcap`—The system's printer capabilities database (discussed in the section "Overview of Ubuntu Printing" in Chapter 8, "Printing with Ubuntu").
- `shells`—A list of approved shells (command-line interfaces).

Protect the Contents of User Directories—`/home`

The most important data on a Linux system resides in the user's directories, found under the `/home` directory. Segregating the system and user data can be helpful in preventing data loss and making the process of backing up easier. For example, having user data reside on a separate file system or mounted from a remote computer on the network might help shield users from data loss in the event of a system hardware failure.

Use the Contents of the `/proc` Directory to Interact with the Kernel

The content of the `/proc` directory is created from memory and exists only while Linux is running. This directory contains special "files" that either extract information from or send information to the kernel. Many Linux utilities extract information from dynamically created directories and files under this directory, also known as a *virtual file system*. For example, the `free` command obtains its information from a file named `meminfo`:

```
$ free
             total       used       free     shared    buffers     cached
Mem:       1026320     822112     204208          0      41232     481412
-/+ buffers/cache:     299468     726852
Swap:      2031608          0    2031608
```

This information constantly changes as the system is used. You can get the same information by using the `cat` command to see the contents of the `meminfo` file:

```
$ cat /proc/meminfo
MemTotal:      1026320 kB
MemFree:        204200 kB
Buffers:         41252 kB
Cached:         481412 kB
SwapCached:          0 kB
Active:         307232 kB
Inactive:       418224 kB
HighTotal:      122692 kB
HighFree:          244 kB
LowTotal:       903628 kB
```

```
LowFree:         203956 kB
SwapTotal:      2031608 kB
SwapFree:       2031608 kB
Dirty:                0 kB
Writeback:            0 kB
AnonPages:       202804 kB
Mapped:           87864 kB
Slab:             21736 kB
SReclaimable:     12484 kB
SUnreclaim:        9252 kB
PageTables:        5060 kB
NFS_Unstable:         0 kB
Bounce:               0 kB
CommitLimit:    2544768 kB
Committed_AS:    712024 kB
VmallocTotal:    114680 kB
VmallocUsed:       6016 kB
VmallocChunk:    108148 kB
HugePages_Total:      0
HugePages_Free:       0
HugePages_Rsvd:       0
Hugepagesize:      4096 kB
```

The `/proc` directory can also be used to dynamically alter the behavior of a running Linux kernel by "echoing" numerical values to specific files under the `/proc/sys` directory. For example, to "turn on" kernel protection against one type of denial of service (DOS) attack known as *SYN flooding*, use the `echo` command to send the number `1` (one) to the following `/proc` path:

```
$ sudo echo 1 >/proc/sys/net/ipv4/tcp_syncookies
```

Other ways to use the `/proc` directory include

- Getting CPU information, such as the family, type, and speed from `/proc/cpuinfo`.
- Viewing important networking information under `/proc/net`, such as active interfaces information under `/proc/net/dev`, routing information in `/proc/net/route`, and network statistics in `/proc/net/netstat`.
- Retrieving file system information.
- Reporting media mount point information via USB; for example, the Linux kernel reports what device to use to access files (such as `/dev/sda`) if a USB camera or hard drive is detected on the system. You can use the `dmesg` command to see this information.

- Getting the kernel version in `/proc/version`, performance information such as uptime in `/proc/uptime`, or other statistics such as CPU load, swap file usage, and processes in `/proc/stat`.

Work with Shared Data in the `/usr` Directory

The `/usr` directory contains software applications, libraries, and other types of shared data for use by anyone on the system. Many Linux system administrators give `/usr` its own partition. A number of subdirectories under `/usr` contain manual pages (`/usr/share/man`), software package shared files (`/usr/share/`*`name_of_package`*, such as `/usr/share/emacs`), additional application or software package documentation (`/usr/share/doc`), and an entire subdirectory tree of locally built and installed software, `/usr/local`.

Temporary File Storage in the `/tmp` Directory

As its name implies, the `/tmp` directory is used for temporary file storage; as you use Linux, various programs create files in this directory.

Access Variable Data Files in the `/var` Directory

The `/var` directory contains subdirectories used by various system services for spooling and logging. Many of these variable data files, such as print spooler queues, are temporary, whereas others, such as system and kernel logs, are renamed and rotated in use. Incoming electronic mail is usually directed to files under `/var/spool/mail`.

Linux also uses `/var` for other important system services. These include the top-most File Transfer Protocol (FTP) directory under `/var/ftp` (see Chapter 18, "Remote File Serving with FTP"), and the Apache web server's initial home page directory for the system, `/var/www/html`. (See Chapter 17, "Apache Web Server Management," for more information on using Apache.)

Logging In to and Working with Linux

You can access and use a Linux system in a number of ways. One way is at the console with a monitor, keyboard, and mouse attached to the PC. Another way is via a serial console, either by dial-up via a modem or a PC running a terminal emulator and connected to the Linux PC via a null modem cable. You can also connect to your system through a wired or wireless network, using the `telnet` or `ssh` commands. The information in this section shows you how to access and use the Linux system, using physical and remote text-based logins.

NOTE

This chapter focuses on text-based logins and use of Linux. Graphical logins and using a graphical desktop are described in Chapter 3, "Working with Gnome."

Text-Based Console Login

If you sit down at your PC and log in to a Linux system that has not been booted to a graphical login, you see a prompt similar to this one:

```
Ubuntu 7.10 (gutsy) ubuntu tty1
login:
```

Your prompt might vary, depending on the version of Ubuntu you are using. In any event, at this prompt, type in your username and press Enter. When you are prompted for your password, type it in and press Enter.

NOTE

Note that your password is not echoed back to you, which is a good idea. Why is it a good idea? Well, people are prevented from looking over your shoulder and seeing your screen input. It is not difficult to guess that a five-letter password might correspond to the user's spouse's first name!

Logging Out

Use the `exit` or `logout` commands to exit your session. Type the command and press Enter. You are then returned to the login prompt. If you use virtual consoles, remember to exit each console before leaving your PC. (Otherwise, someone could easily sit down and use your account.)

Logging In and Out from a Remote Computer

Although you can happily log in on your computer, an act known as a *local* login, you can also log in to your computer via a network connection from a remote computer. Linux-based operating systems provide a number of remote access commands you can use to log in to other computers on your local area network (LAN), wide area network (WAN), or the Internet. Note that not only must you have an account on the remote computer, but the remote computer must be configured to support remote logins—otherwise, you won't be able to log in.

NOTE

See Chapter 14 to see how to set up network interfaces with Linux to support remote network logins and Chapter 11 to see how to start remote access services (such as `sshd`).

The best and most secure way (barring future exploits) to log in to a remote Linux computer is to use the `ssh` or Secure Shell client. Your login and session are encrypted while you work on the remote computer. The `ssh` client features many different

command-line options, but can be simply used with the name or IP address of the remote computer, like this:

```
[andrew@laptop ~]$ ssh 192.168.0.41
The authenticity of host '192.168.0.41 (192.168.0.41)' can't be established.
RSA key fingerprint is e1:db:6c:da:3f:fc:56:1b:52:f9:94:e0:d1:1d:31:50.
Are you sure you want to continue connecting (yes/no)?

 yes
```

The first time you connect with a remote computer using `ssh`, Linux displays the remote computer's encrypted identity key and asks you to verify the connection. After you type **`yes`** and press Enter, you are warned that the remote computer's identity (key) has been entered in a file named `known_hosts` under the `.ssh` directory in your home directory. You are also prompted to enter your password:

```
Warning: Permanently added '192.168.0.41' (RSA) \
to the list of known hosts.
andrew@192.168.0.41's password:
andrew@optimus:~$
```

After entering your password, you can then work on the remote computer. Again, everything you enter on the keyboard in communication with the remote computer is encrypted. Use the `exit` or `logout` commands to exit your session and return to the shell on your computer.

Using Environment Variables

A number of in-memory variables are assigned and loaded by default when the user logs in. These variables are known as shell *environment variables*, which can be used by various commands to get information about your environment, such as the type of system you are running, your home directory, and the shell in use. Environment variables are used by Linux operating systems to help tailor the computing environment of your system, and include helpful specifications and setup, such as default locations of executable files and software libraries. If you begin writing shell scripts, you might use environment variables in your scripts. Until then, you only need to be aware of what environment variables are and do.

The following list includes a number of environment variables, along with descriptions of how the shell uses them:

- `PWD`—To provide the name of the current working directory, used by the `pwd` command (such as `/home/andrew/foo`)
- `USER`—To declare the user's name, such as `andrew`
- `LANG`—To set language defaults, such as English

- `SHELL`—To declare the name and location of the current shell, such as `/bin/bash`
- `PATH`—To set the default location of executable files, such as `/bin`, `/usr/bin`, and so on
- `TERM`—To set the type of terminal in use, such as `vt100`, which can be important when using screen-oriented programs, such as text editors
- `MACHINE`—To declare system type, system architecture, and so on

NOTE

Each shell can have its own feature set and language syntax, as well as a unique set of default environment variables. See Chapter 15 for more information about using the different shells included with Ubuntu.

At the command line, you can use the `env` or `printenv` commands to display these environment variables, like so:

```
$ env
SSH_AGENT_PID=5761

SHELL=/bin/bash

DESKTOP_STARTUP_ID=

TERM=xterm

GTK_RC_FILES=/etc/gtk/gtkrc:/home/andrew/.gtkrc-1.2-gnome2

WINDOWID=56623199

USER=andrew

...
USERNAME=andrew

PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games

DESKTOP_SESSION=default

GDM_XSERVER_LOCATION=local

PWD=/usr/local

LANG=en_GB.UTF-8
```

```
GNOME_KEYRING_PID=5714

GDM_LANG=en_GB.UTF-8

SHLVL=1

HOME=/home/andrew

LOGNAME=andrew

XDG_DATA_DIRS=/usr/local/share/:/usr/share/:/usr/share/gdm/

...
LESSOPEN=¦ /usr/bin/lesspipe %s

WINDOWPATH=7

DISPLAY=:0.0

LESSCLOSE=/usr/bin/lesspipe %s %s

COLORTERM=gnome-terminal

XAUTHORITY=/home/andrew/.Xauthority

_=/usr/bin/env

OLDPWD=/usr/share/locale
```

This abbreviated list shows a few common variables. These variables are set by configuration or *resource* files contained in the `/etc`, `/etc/skel`, or user `/home` directory. You can find default settings for `bash`, for example, in `/etc/profile`, `/etc/bashrc`, `.bashrc`, or `.bash_profile` files installed in your home directory. Read the man page for `bash` for details about using these configuration files.

One of the most important environment variables is `$PATH`, which defines the location of executable files. For example, if, as a regular user, you try to use a command that is not located in your `$PATH` (such as the imaginary `command` command), you will see something like this:

```
$ command
-bash: command: command not found
```

However, you might know that `command` is definitely installed on your system, and you can verify this by using the `whereis` command, like so:

```
$  whereis command
command: /sbin/command
```

You can also run the command by typing its full pathname, or complete directory specification like this:

```
$ /sbin/command
```

As you can see in this example, the `command` command is indeed installed. What happened is that by default, the `/sbin` directory is not in your `$PATH`. One of the reasons for this is that commands under the `/sbin` directory are normally intended to be run only by root. You can add `/sbin` to your `$PATH` by editing the file `.bash_profile` in your home directory (if you use the `bash` shell by default, like most Linux users). Look for the following line:

```
PATH=$PATH:$HOME/bin
```

You can then edit this file, perhaps using the `vi` editor (discussed in this chapter), to add the `/sbin` directory like so:

```
PATH=$PATH:/sbin:$HOME/bin
```

Save the file. The next time you log in, the `/sbin` directory is in your `$PATH`. One way to use this change right away is to read in the new settings in `.bash_profile` by using the `bash` shell's `source` command like so:

```
$ source .bash_profile
```

You can now run commands located in the `/sbin?` directory without the need to explicitly type the full pathname.

Some Linux commands also use environment variables, for example, to acquire configuration information (such as a communications program looking for a variable such as `BAUD_RATE`, which might denote a default modem speed).

To experiment with the environment variables, you can modify the `PS1` variable to manipulate the appearance of your shell prompt. If you are working with `bash`, you can use its built-in `export` command to change the shell prompt. For example, if your default shell prompt looks like

```
[andrew@laptop ~]$
```

You can change its appearance by using the `PS1` variable like this:

```
$  PS1='$OSTYPE r00lz ->'
```

After you press Enter, you see

```
linux-gnu r00lz ->
```

NOTE

See the `bash` man page for other variables you can use for prompt settings.

Using the Text Editors

Linux distributions include a number of applications known as *text editors* that you can use to create text files or edit system configuration files. Text editors are similar to word processing programs, but generally have fewer features, work only with text files, and might or might not support spell checking or formatting. The text editors range in features and ease of use, but are found on nearly every Linux distribution. The number of editors installed on your system depends on what software packages you've installed on the system.

Some of the console-based text editors are

- `emacs`—The comprehensive GNU `emacs` editing environment, which is much more than an editor; see the section "Working with `emacs`" later in this chapter
- `joe`—Joe's Own Editor, a text editor, which can be used to emulate other editors
- `nano`—A simple text editor similar to the `pico` text editor included with the `pine` email program
- `vim`—An improved, compatible version of the `vi` text editor (which we call `vi` in the rest of this chapter because it has a symbolic link named `vi` and a symbolically linked manual page)

Note that not all text editors described here are *screen oriented*. Some of the text editors for the X Window System, which provide a graphical interface, such as menu bars, buttons, scrollbars and so on, are

- `gedit`—A GUI text editor for GNOME
- `kate`—A simple KDE text editor
- `kedit`—Another simple KDE text editor

A good reason to learn how to use a text-based editor, such as `vi`, is that system maintenance and recovery operations generally never take place during X Window sessions (negating the use of a GUI editor). Many larger, more complex, and capable editors do not work when Linux is booted to its single-user or maintenance mode. If anything does go wrong with your system, you probably won't be able to get into the X Window system, making knowledge and experience of using both the command line and text

editors such as vi important. Make a point of opening some of the editors and playing around with them; you never know—you might just thank me someday!

Another reason to learn how to use a text-based editor under the Linux console mode is so that you can edit text files through dial-up or network shell sessions because many servers do not host graphical desktops.

Working with vi

The editor found on nearly every Unix and Linux system is, without a doubt, the vi editor, originally written by Bill Joy. This simple-to-use but incredibly capable editor features a somewhat cryptic command set, but you can put it to use with only a few commands. Although more experienced Unix and Linux users continue to use vi extensively during computing sessions, many newer users might prefer learning an easier-to-use text editor such as pico or GNU nano. Die-hard GNU fans and programmers definitely use emacs.

That said, learning how to use vi is a good idea. You might need to edit files on a Linux system with a minimal install, or a remote server without a more extensive offering of installed text editors. Chances are better than good that vi will be available.

You can start an editing session by using the vi command like this:

```
$ vi file.txt
```

The vi command works by using an insert (or editing) mode, and a viewing (or command) mode.

When you first start editing, you are in the viewing mode. You can use your cursor or other navigation keys (as shown later) to scroll through the text. To start editing, press the i key to insert text or the a key to append text. When finished, use the Esc key to toggle out of the insert or append modes and into the viewing (or command) mode. To enter a command, type a colon (:), followed by the command, such as w to write the file, and press Enter.

Although vi supports many complex editing operations and numerous commands, you can accomplish work by using a few basic commands. These basic vi commands are

- **Cursor movement**—h, j, k, l (left, down, up, and right)
- **Delete character**—x
- **Delete line**—dd
- **Mode toggle**—Esc, Insert (or i)
- **Quit**—:q
- **Quit without saving**—:q!
- **Run a shell command**—:sh (use 'exit' to return)

- **Save file—**`:w`
- **Text search—**`/`

NOTE

Use the `vimtutor` command to quickly learn how to use `vi`'s keyboard commands. The tutorial takes less than 30 minutes, and it teaches new users how to start or stop the editor, navigate files, insert and delete text, and perform search, replace, and insert operations.

Working with `emacs`

Richard M. Stallman's GNU `emacs` editor, like `vi`, is included with Linux and nearly every other Linux distribution. Unlike other Unix and Linux text editors, `emacs` is much more than a simple text editor—it is an editing environment and can be used to compile and build programs, act as an electronic diary, appointment book and calendar, compose and send electronic mail, read Usenet news, and even play games. The reason for this capability is that `emacs` contains a built-in language interpreter that uses the Elisp (`emacs` LISP) programming language. `emacs` is not installed in Ubuntu by default; instead you'll need to install it using `apt-get` or `synaptic`. The package you need is simply `emacs`.

You can start an `emacs` editing session like this:

```
$ emacs file.txt
```

TIP

If you start `emacs` when using `X11`, the editor launches in its own floating window. To force `emacs` to display inside a terminal window instead of its own window (which can be useful if the window is a login at a remote computer), use the `-nw` command-line option like this: `emacs -nw file.txt`.

The `emacs` editor uses an extensive set of keystroke and named commands, but you can work with it by using a basic command subset. Many of these basic commands require you to hold down the Ctrl key, or to first press a *meta* key (generally mapped to the Alt key). The basic commands are listed in Table 4.2.

TABLE 4.2 Emacs Editing Commands

Action	Command
Abort	Ctrl+g
Cursor left	Ctrl+b
Cursor down	Ctrl+n
Cursor right	Ctrl+f
Cursor up	Ctrl+p

TABLE 4.2 Continued

Action	Command
Delete character	Ctrl+d
Delete line	Ctrl+k
Go to start of line	Ctrl+a
Go to end of line	Ctrl+e
Help	Ctrl+h
Quit	Ctrl+x, Ctrl+c
Save As	Ctrl+x, Ctrl+w
Save file	Ctrl+x, Ctrl+s
Search backward	Ctrl+r
Search forward	Ctrl+s
Start tutorial	Ctrl+h, t
Undo	Ctrl+x, u

TIP

One of the best reasons to learn how to use `emacs` is that you can use nearly all the same keystrokes to edit commands on the `bash` shell command line. Another reason is that like `vi`, `emacs` is universally available on nearly every Unix and Linux system, including Apple's Mac OS X.

Working with Permissions

Under Linux (and Unix), everything in the file system, including directories and devices, is a file. And every file on your system has an accompanying set of permissions based on ownership. These permissions form the basis for security under Linux, and designate each file's read, write, and execute permission for you, members of your group, and all others on the system.

You can examine the default permissions for a file you create by using the `umask` command, or as a practical example, by using the `touch` command and then the `ls` command's long-format listing like this:

```
$ touch file
$ ls -l file
-rw-rw-r--    1 andrew    andrew           0 Nov 11 12:28 file
```

In this example, the `touch` command is used to quickly create a file. The `ls` command then reports on the file, displaying information (from left to right) in the first field of output (such as `-rw-rw-r--` previously):

- **The type of file created**—Common indicators of the type of file are a leading letter in the output. A blank (which is represented by a dash in the preceding example) designates a plain file, `d` designates a directory, `c` designates a character device (such as `/dev/ttyS0`), and `b` is used for a block device (such as `/dev/sda`).
- **Permissions**—Read, write, and execute permissions for the owner, group, and all others on the system. (You learn more about these permissions later in this section.)
- **Number of links to the file**—The number one (`1`) designates that there is only one file, whereas any other number indicates that there might be one or more hard-linked files. Links are created with the `ln` command. A hard-linked file is an exact copy of the file, but it might be located elsewhere on the system. Symbolic links of directories can also be created, but only the root operator can create a hard link of a directory.
- **The owner**—The account that created or owns the file; you can change this designation by using the `chown` command.
- **The group**—The group of users allowed to access the file; you can change this designation by using the `chgrp` command.
- **File size and creation/modification date**—The last two elements indicate the size of the file in bytes and the date the file was created or last modified.

Assigning Permissions

Under Linux, permissions are grouped by owner, group, and others, with read, write, and execute permission assigned to each, like so:

```
Owner    Group    Others
rwx      rwx      rxw
```

Permissions can be indicated by mnemonic or octal characters. Mnemonic characters are

- `r` indicates permission for an owner, member of the owner's group, or others to open and read the file.
- `w` indicates permission for an owner, member of the owner's group, or others to open and write to the file.
- `x` indicates permission for an owner, member of the owner's group, or others to execute the file (or read a directory).

In the previous example for the file named `file`, the owner, `andrew`, has read and write permission, as does any member of the group named `andrew`. All other users may only read the file. Also note that default permissions for files created by the root operator will be different because of `umask` settings assigned by the shell.

Many users prefer to use numeric codes, based on octal (base 8) values, to represent permissions. Here's what these values mean:

- 4 indicates read permission.
- 2 indicates write permission.
- 1 indicates execute permission.

In octal notation, the previous example file has a permission setting of 664 (read + write or 4 + 2, read + write or 4 + 2, read-only or 4). Although you can use either form of permissions notation, octal is easy to use quickly after you visualize and understand how permissions are numbered.

NOTE

In Linux, you can create groups to assign a number of users access to common directories and files, based on permissions. You might assign everyone in accounting to a group named `accounting`, for example, and allow that group access to accounts payable files while disallowing access by other departments. Defined groups are maintained by the root operator, but you can use the `newgrp` command to temporarily join other groups to access files (as long as the root operator has added you to the other groups). You can also allow or deny other groups' access to your files by modifying the group permissions of your files.

Directory Permissions

Directories are also files under Linux. For example, again use the `ls` command to show permissions like this:

```
$ mkdir foo
$ ls -ld foo
drwxrwxr-x    2 andrew    andrew        4096 Jan 23 12:37 foo
```

In this example, the `mkdir` command is used to create a directory. The `ls` command and its `-ld` option is used to show the permissions and other information about the directory (not its contents). Here you can see that the directory has permission values of 775 (read + write + execute or 4 + 2 + 1, read + write + execute or 4 + 2 + 1, and read + execute or 4 + 1).

This shows that the owner and group members can read and write to the directory and, because of execute permission, also list the directory's contents. All other users can only list the directory contents. Note that directories require execute permission for anyone to be able to view their contents.

You should also notice that the `ls` command's output shows a leading `d` in the permissions field. This letter specifies that this file is a directory; normal files have a blank field

in its place. Other files, such as those specifying a block or character device, have a different letter.

For example, if you examine the device file for a Linux serial port, you will see

```
$ ls -l /dev/ttyS0
crw-rw---- 1 root dialout 4, 64 Sep 28 21:44 /dev/ttyS0
```

Here, `/dev/ttyS0` is a character device (such as a serial communications port and designated by a `c`) owned by root and available to anyone in the `dialout` group. The device has permissions of `660` (read + write, read + write, no permission).

On the other hand, if you examine the device file for an IDE hard drive, you see

```
$ ls -l /dev/sda
brw-rw---- 1 root disk 8, 0 Sep 28 21:44 /dev/sda
```

In this example, `b` designates a block device (a device that transfers and caches data in blocks) with similar permissions. Other device entries you will run across on your Linux system include symbolic links, designated by `s`.

You can use the `chmod` command to alter a file's permissions. This command uses various forms of command syntax, including octal or a mnemonic form (such as `u`, `g`, `o`, or `a` and `rwx`, and so on) to specify a desired change. The `chmod` command can be used to add, remove, or modify file or directory permissions to protect, hide, or open up access to a file by other users (except for root, which can access any file or directory on a Linux system).

The mnemonic forms of `chmod`'s options (when used with a plus character, `+`, to add, or a minus sign, `-`, to take away) designate the following:

- `u`—Adds or removes user (owner) read, write, or execute permission
- `g`—Adds or removes group read, write, or execute permission
- `o`—Adds or removes read, write, or execute permission for others not in a file's group
- `a`—Adds or removes read, write, or execute permission for all users
- `r`—Adds or removes read permission
- `w`—Adds or removes write permission
- `x`—Adds or removes execution permission

For example, if you create a file, such as a `readme.txt`, the file will have default permissions (set by the `umask` setting in `/etc/bashrc`) of

```
-rw-rw-r--    1 andrew    andrew          12 Jan  2 16:48 readme.txt
```

As you can see, you and members of your group can read and write the file. Anyone else can only read the file (and only if it is outside your home directory, which will have read, write, and execute permission set only for you, the owner). You can remove all write permission for anyone by using `chmod`, the minus sign, and `aw` like so:

```
$ chmod a-w readme.txt
$ ls -l readme.txt
-r--r--r--    1 andrew    andrew          12 Jan  2 16:48 readme.txt
```

Now, no one can write to the file (except you, if the file is in your home or `/tmp` directory because of directory permissions). To restore read and write permission for only you as the owner, use the plus sign and the `u` and `rw` options like so:

```
$ chmod u+rw readme.txt
$ ls -l readme.txt
-rw-------    1 andrew    andrew          12 Jan  2 16:48 readme.txt
```

You can also use the octal form of the `chmod` command, for example, to modify a file's permissions so that only you, the owner, can read and write a file. Use the `chmod` command and a file permission of `600`, like this:

```
$ chmod 600 readme.txt
```

If you take away execution permission for a directory, files might be hidden inside and may not be listed or accessed by anyone else (except the root operator, of course, who has access to any file on your system). By using various combinations of permission settings, you can quickly and easily set up a more secure environment, even as a normal user in your home directory.

Understanding Set User ID and Set Group ID Permissions

Another type of permission is "set user ID", known as *suid*, and "set group ID" (*sgid*) permissions. These settings, when used in a program, enable any user running that program to have program owner or group owner permissions for that program. These settings enable the program to be run effectively by anyone, without requiring that each user's permissions be altered to include specific permissions for that program.

One commonly used program with suid permissions is the `passwd` command:

```
$ ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 29104 May 18 10:59 /usr/bin/passwd
```

This setting allows normal users to execute the command (as root) to make changes to a root-only accessible file, `/etc/passwd`.

You also can assign similar permission with the `chfn` command. This command allows users to update or change `finger` information in `/etc/passwd`. You accomplish this permission modification by using a leading `4` (or the mnemonic `s`) in front of the three octal values.

NOTE

Other files that might have suid or guid permissions include `at`, `rcp`, `rlogin`, `rsh`, `chage`, `chsh`, `ssh`, `crontab`, `sudo`, `sendmail`, `ping`, `mount`, and several Unix-to-Unix Copy (UUCP) utilities. Many programs (such as games) might also have this type of permission to access a sound device.

Files or programs that have suid or guid permissions can sometimes present security holes because they bypass normal permissions. This problem is compounded if the permission extends to an executable binary (a command) with an inherent security flaw because it could lead to any system user or intruder gaining root access. In past exploits, this typically happened when a user fed a vulnerable command with unexpected input (such as a long pathname or option); the command would fail, and the user would be presented a root prompt. Although Linux developers are constantly on the lookout for poor programming practices, new exploits are found all the time, and can crop up unexpectedly, especially in newer software packages that haven't had the benefit of peer developer review.

Savvy Linux system administrators keep the number of suid or guid files present on a system to a minimum. The `find` command can be used to display all such files on your system:

```
# find / -type f -perm +6000 -exec ls -l {} \;
```

NOTE

The `find` command is quite helpful and can be used for many purposes, such as before or during backup operations. See the section "Using Backup Software" in Chapter 13, "Backing Up."

Note that the programs do not necessarily have to be removed from your system. If your users really do not need to use the program, you can remove the program's execute permission for anyone. You have to decide, as the root operator, whether your users are allowed to, for example, mount and unmount CD-ROMs or other media on your system. Although Linux-based operating systems can be set up to accommodate ease of use and convenience, allowing programs such as `mount` to be suid might not be the best security policy. Other candidates for suid permission change could include the `chsh`, `at`, or `chage` commands.

Working as Root

The root, or super-user account, is a special account and user on Unix and Linux systems. Super-user permissions are required in part because of the restrictive file permissions assigned to important system configuration files. You must have root permission to edit these files or to access or modify certain devices (such as hard drives). When logged in as root, you have total control over your system, which can be dangerous.

When you work in root, you can destroy a running system with a simple invocation of the `rm` command like this:

```
# rm -fr /
```

This command line not only deletes files and directories, but also could wipe out file systems on other partitions and even remote computers. This alone is reason enough to take precautions when using root access.

The only time you should run Linux as the super-user is when you are configuring the file system, for example, or to repair or maintain the system. Logging in and using Linux as the root operator isn't a good idea because it defeats the entire concept of file permissions.

NOTE

The next couple of paragraphs assume that you have enabled the root account, as described at the start of this chapter.

Knowing how to run commands as root without logging in as root can help avoid serious missteps when configuring your system. Linux comes with a command named `su` that enables you to run one or more commands as root and then quickly returns you to normal user status. For example, if you would like to edit your system's file system table (a simple text file that describes local or remote storage devices, their type, and location), you can use the `su` command like this:

```
$ su -c "nano -w /etc/fstab"
Password:
```

After you press Enter, you are prompted for a password that gives you access to root. This extra step can also help you "think before you leap" into the command. Enter the root password, and you are then editing `/etc/fstab`, using the `nano` editor with line wrapping disabled.

CAUTION

Before editing any important system or software service configuration file, make a backup copy. Then make sure to launch your text editor with line wrapping disabled. If you edit a configuration file without disabling line wrapping, you could insert spurious carriage returns and line feeds into its contents, causing the configured service to fail when restarting. By convention, nearly all configuration files are formatted for 80-character text width, but this is not always the case. By default, the `vi` and `emacs` editors don't use line wrap.

You can use `sudo` in the same way to allow you to execute one-off commands. The preceding example would look like this, using `sudo`:

```
$ sudo nano -w /etc/fstab
```


Creating Users

When a Linux system administrator creates a user, an entry in `/etc/passwd` for the user is created. The system also creates a directory, labeled with the user's username, in the `/home` directory. For example, if you create a user named `bernice`, the user's home directory is `/home/bernice`.

NOTE

In this chapter, you learn how to manage users from the command line. See Chapter 10 for more information on user administration with Ubuntu using graphical administration utilities, such as the `system-config-users` client.

Use the `useradd` command, along with a user's name, to quickly create a user:

```
$ sudo useradd andrew
```

After creating the user, you must also create the user's initial password with the `passwd` command:

```
$ sudo passwd andrew
Changing password for user andrew.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

Enter the new password twice. If you do not create an initial password for a new user, the user cannot log in.

You can view useradd's default new user settings by using the command and its -D option, like this:

```
$ useradd -D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
```

These options display the default group ID, home directory, account and password policy (active forever with no password expiration), the default shell, and the directory containing defaults for the shell.

The useradd command has many different command-line options. The command can be used to set policies and dates for the new user's password, assign a login shell, assign group membership, and other aspects of a user's account.

Deleting Users

Use the userdel command to delete users from your system. This command removes a user's entry in the system's /etc/passwd file. You should also use the command's -r option to remove all the user's files and directories (such as the user's mail spool file under /var/spool/mail):

```
$ sudo userdel -r andrew
```

If you do not use the -r option, you have to manually delete the user's directory under /home, along with the user's /var/spool/mail queue.

Shutting Down the System

Use the shutdown command to shut down your system. The shutdown command has a number of different command-line options (such as shutting down at a predetermined time), but the fastest way to cleanly shut down Linux is to use the -h or halt option, followed by the word now or the numeral zero (0), like this:

```
$ sudo shutdown -h now
```

or

```
$ sudo shutdown -h 0
```

To incorporate a timed shutdown and a pertinent message to all active users, use shutdown's time and message options, like so:

```
$ sudo shutdown -h 18:30 "System is going down for maintenance this evening"
```

This example shuts down your system and provides a warning to all active users 15 minutes before the shutdown (or reboot). Shutting down a running server can be considered drastic, especially if there are active users or exchanges of important data occurring (such as a backup in progress). One good approach is to warn users ahead of time. This can be done by editing the system Message of the Day (MOTD) `motd` file, which displays a message to users after login. To create your custom MOTD, use a text editor and change the contents of `/etc/motd`. You can also make downtimes part of a regular schedule, perhaps to coincide with security audits, software updates, or hardware maintenance.

You should shut down Ubuntu for only a few very specific reasons:

- You are not using the computer and want to conserve electrical power.
- You need to perform system maintenance that requires any or all system services to be stopped.
- You want to replace integral hardware.

TIP

Do not shut down your computer if you suspect that one or more intruders has infiltrated your system; instead, disconnect the machine from any or all networks and make a backup copy of your hard drives. You might want to also keep the machine running to examine the contents of memory and to examine system logs.

Rebooting the System

You should also use the `shutdown` command to reboot your system. The fastest way to cleanly reboot Linux is to use the `-r` option, and the word `now` or the numeral zero (`0`):

```
$ sudo shutdown -r now
```

or

```
$ sudo shutdown -r 0
```

Both rebooting and shutting down can have dire consequences if performed at the wrong time (such as during backups or critical file transfers, which arouses the ire of your system's users). However, Linux-based operating systems are designed to properly stop active system services in an orderly fashion. Other commands you can use to shut down and reboot Linux are the `halt` and `reboot` commands, but the `shutdown` command is more flexible.

Reading Documentation

Although you learn the basics of using Ubuntu in this book, you need time and practice to master and troubleshoot more complex aspects of the Linux operating system and your

distribution. As with any operating system, you can expect to encounter some problems or perplexing questions as you continue to work with Linux. The first place to turn for help with these issues is the documentation included with your system; if you cannot find the information you need there, check Ubuntu's website.

Linux, like Unix, is a self-documenting system, with man pages accessible through the `man` command. Linux offers many other helpful commands for accessing its documentation. You can use the `apropos` command—for example, with a keyword such as `partition`—to find commands related to partitioning, like this:

```
$ apropos partition
diskdumpfmt          (8)  - format a dump device or a partition
fdisk                (8)  - Partition table manipulator for Linux
GNU Parted [parted]  (8)  - a partition manipulation program
mpartition           (1)  - partition an MSDOS hard disk
MPI_Cart_sub         (3)  - Partitions a communicator into subgroups which form
                            lower-dimensional cartesian subgrids
partprobe            (8)  - inform the OS of partition table changes
pvcreate             (8)  - initialize a disk or partition for use by LVM
sfdisk               (8)  - Partition table manipulator for Linux
```

To find a command and its documentation, you can use the `whereis` command. For example, if you are looking for the `fdisk` command, you can do this:

```
$ whereis fdisk
fdisk: /sbin/fdisk /usr/share/man/man8/fdisk.8.gz
```

Using Man Pages

To learn more about a command or program, use the `man` command, followed by the name of the command. Man pages for Linux and X Window commands are within the `/usr/share/man`, `/usr/local/share/man`, and `/usr/X11R6/man` directories; so, for example, to read the `rm` command's man page, use the `man` command like this:

```
$ man rm
```

After you press Enter, the `less` command (a Linux command known as a *pager*) displays the man page. The `less` command is a text browser you can use to scroll forward and backward (even sideways) through the document to learn more about the command. Type the letter **`h`** to get help, use the forward slash to enter a search string, or press **`q`** to quit.

NOTE

Although nearly all the hundreds of GNU commands included with Linux each have a man page, you must use the `info` command to read detailed information about using a GNU command. For example, to learn even more about `bash` (which has a rather extensive manual page), use the `info` command like this:

```
$ info bash
```

Press the **n** and **p** keys to navigate through the document, or scroll down to a menu item on the screen and press Enter to read about a specific feature. Press **q** to quit reading.

Related Ubuntu and Linux Commands

The following programs and built-in shell commands are commonly used when working at the command line. These commands are organized by category to help you understand the command's purpose. If you need to find full information for using the command, you can find that information under the command's man page.

Managing users and groups—`chage`, `chfn`, `chsh`, `edquota`, `gpasswd`, `groupadd`, `groupdel`, `groupmod`, `groups`, `mkpasswd`, `newgrp`, `newusers`, `passwd`, `umask`, `useradd`, `userdel`, `usermod`

Managing files and file systems—`cat`, `cd`, `chattr`, `chmod`, `chown`, `compress`, `cp`, `dd`, `fdisk`, `find`, `gzip`, `ln`, `mkdir`, `mksfs`, `mount`, `mv`, `rm`, `rmdir`, `rpm`, `sort`, `swapon`, `swapoff`, `tar`, `touch`, `umount`, `uncompress`, `uniq`, `unzip`, `zip`

Managing running programs—`bg`, `fg`, `kill`, `killall`, `nice`, `ps`, `pstree`, `renice`, `top`, `watch`

Getting information—`apropos`, `cal`, `cat`, `cmp`, `date`, `diff`, `df`, `dir`, `dmesg`, `du`, `env`, `file`, `free`, `grep`, `head`, `info`, `last`, `less`, `locate`, `ls`, `lsattr`, `man`, `more`, `pinfo`, `ps`, `pwd`, `stat`, `strings`, `tac`, `tail`, `top`, `uname`, `uptime`, `vdir`, `vmstat`, `w`, `wc`, `whatis`, `whereis`, `which`, `who`, `whoami`

Console text editors—`ed`, `jed`, `joe`, `mcedit`, `nano`, `red`, `sed`, `vim`

Console Internet and network commands—`bing`, `elm`, `ftp`, `host`, `hostname`, `ifconfig`, `links`, `lynx`, `mail`, `mutt`, `ncftp`, `netconfig`, `netstat`, `pine`, `ping`, `pump`, `rdate`, `route`, `scp`, `sftp`, `ssh`, `tcpdump`, `traceroute`, `whois`, `wire-test`

Reference

- http://www.winntmag.com/Articles/Index.cfm?ArticleID=7420—An article by a Windows NT user who, when experimenting with Linux, blithely confesses to rebooting the system after not knowing how to read a text file at the Linux console.
- http://standards.ieee.org/regauth/posix/—IEEE's POSIX information page.
- http://www.itworld.com/Comp/2362/lw-01-government/#sidebar—Discussion of Linux and POSIX compliance.
- http://www.pathname.com/fhs/—Home page for the Linux FHS, Linux Filesystem Hierarchy Standard.
- http://www.tldp.org/—Browse the HOWTO section to find and read The Linux Keyboard and Console HOWTO—Andries Brouwer's somewhat dated but eminently useful guide to using the Linux keyboard and console.
- http://www.gnu.org/software/emacs/emacs.html—Home page for the FSF's GNU `emacs` editing environment; you can find additional documentation and links to the source code for the latest version here.
- http://www.vim.org/—Home page for the `vim` (`vi` clone) editor included with Linux distributions. Check here for updates, bug fixes, and news about this editor.
- http://www.courtesan.com/sudo/—Home page for the `sudo` command. Check here for the latest updates, security features, and bug fixes.

PART II

Desktop Ubuntu

IN THIS PART

This page intentionally left blank

CHAPTER 5

On the Internet

In the modern world, the Internet is everywhere. From cell phones to offices, from games consoles to iPods, we are surrounded by multiple access routes to online information and communication. Ubuntu is no outsider when it comes to accessing information through the Internet and comes equipped with web browsers, email clients, and other tools that you can use to connect to other people across the globe.

In this chapter, we'll take a look at some of the popular Internet applications that are available with Ubuntu. You'll find out about Firefox, and its KDE alternative, Konqueror. We'll also investigate some of the email clients that you can install while using Ubuntu. Other topics include RSS feed readers, Instant Messaging (through IRC and other networks), and reading newsgroups. Finally we'll take a quick look at Ekiga, which is a videoconferencing program for Ubuntu.

A Brief Introduction to the Internet

The Internet itself was first brought to life by the U.S. Department of Defense in 1969. It was called ARPANet after the Department of Defense's Advanced Research Projects Agency. Designed to build a network that would withstand major catastrophe (this was the peak of the Cold War), it soon grew to encompass more and more networks to build the Internet. Then, in 1991, Tim Berners-Lee of CERN developed the idea of the World Wide Web, including Hypertext Transfer Protocol (HTTP) and Hypertext Markup Language (HTML). This gave us what we now know to be the Internet.

IN THIS CHAPTER

Getting Started with Firefox

One of the most popular web browsers, and in fact the default web browser in Ubuntu, is Mozilla Firefox (see Figure 5.1). Built on a solid code base that is derived from the Mozilla Suite, Firefox offers an alternative to surfing the Internet using Internet Explorer. There have been more than 265 million downloads of Firefox since its release in late 2004, and it has grabbed significant market share from Internet Explorer.

In Ubuntu, you can find Firefox under the Applications, Internet menu at the top of your screen. An even simpler way to start Firefox is to click the small world icon next to the Actions menu. Either way, Firefox opens.

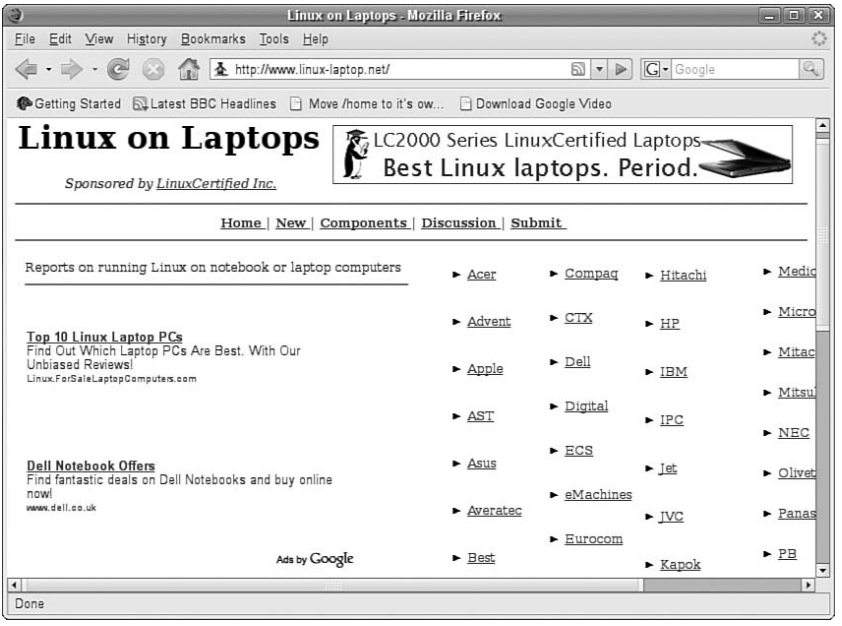


FIGURE 5.1 Mozilla Firefox—rediscover the Web. Firefox enables you to add on numerous upgrades, further enhancing your experience.

Beyond the basic program is a wealth of plug-ins and extensions that can increase the capabilities of Firefox beyond simple web browsing. Plug-ins such as Shockwave Flash and Java are available instantly, as are multimedia codecs for viewing video content, whereas extensions provide useful additions to the browsing experience. For example, ForecastFox is an extension that gives you your local weather conditions, and Bandwidth Tester is a tool that calculates your current bandwidth. As Firefox grows, there will be more and more extensions and plug-ins that you can use to enhance your browsing pleasure.

Finding and obtaining these plug-ins and extensions is made very easy as Mozilla developers have helpfully created a site dedicated to helping you get more from Firefox. Particular favorites are the Adblock Plus and the StumbleUpon plug-ins. Adblock Plus

allows you to nuke all those annoying banners and animations that take up so much bandwidth while you are browsing. StumbleUpon is a neat plug-in that takes you to web pages based on your preferences. Be warned, though, that StumbleUpon can be quite addictive and you will end up wasting away many hours clicking the stumble button!

Another plug-in that we make a lot of use of is Google BrowserSync. If, like us, you work across multiple computers then you will no doubt have had to re-create bookmarks at every different computer and try to keep them the same. Google makes this whole process much easier by allowing you to synchronize not only your bookmarks, but also your cookies, browser history, and finally any saved passwords across multiple browsers. Bear in mind that you can choose what you want to synchronize, making it easy just to replicate your bookmarks.

Konqueror

KDE users have the option to use Konqueror, which is the default browser for KDE (see Figure 5.2). As well as handling file system navigation, Konqueror can also be used to surf the web. It, too, is based on the Gecko rendering engine as found in Firefox.

FIGURE 5.2 Konqueror, the standard KDE web browser.

Choosing an Email Client

Back in the days of Unix, there were various text-based email clients such as `elm` and `pine` (Pine Is Not Elm). Although they looked basic, they allowed the average user to interact with his email, both for composing and reading correspondence. With the advent of mainstream computing and the realization that people needed friendly GUI interfaces to

be productive came a plethora of email clients, with some of them being cross-platform and compatible among Linux, Windows, and Mac OS X, not to mention Unix.

Evolution

Evolution is the standard email client that comes with Ubuntu, and to call it an email client would be to sincerely underestimate its usefulness as an application. Not only does it handle email, but it can also look after contacts and calendaring, as well as managing your tasks (see Figure 5.3). The next section demonstrates how to configure Evolution to handle email.

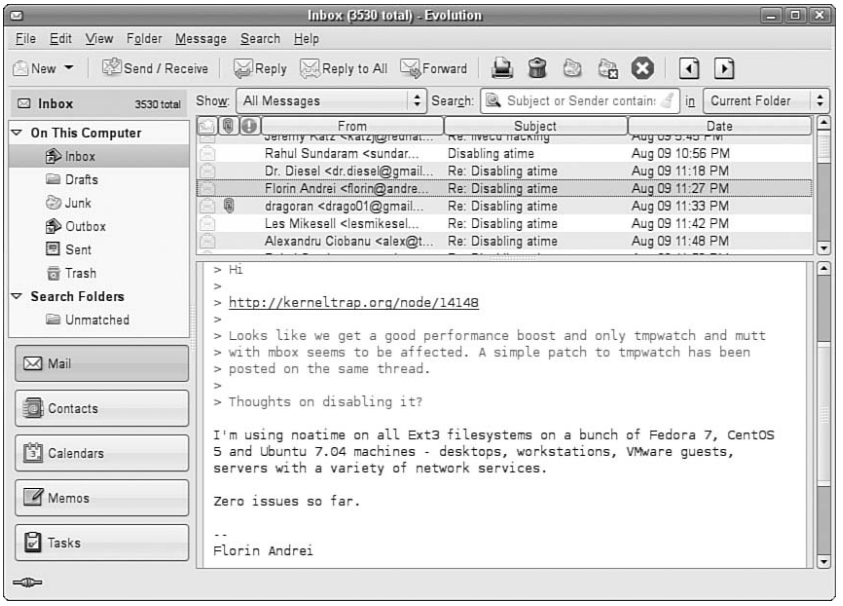


FIGURE 5.3 With Evolution, you can handle all your email and contacts, as well as make appointments and track tasks.

You need to have the following information to successfully configure Evolution:

- Your email address
- Your incoming email server name and type (that is, pop.email.com, POP, and IMAP)
- Your username and password for the incoming server
- Your outgoing email server name (that is, smtp.email.com)

After you have all the information, you can start Evolution. The first screen you are presented with is the Account Setup Assistance screen (see Figure 5.4).

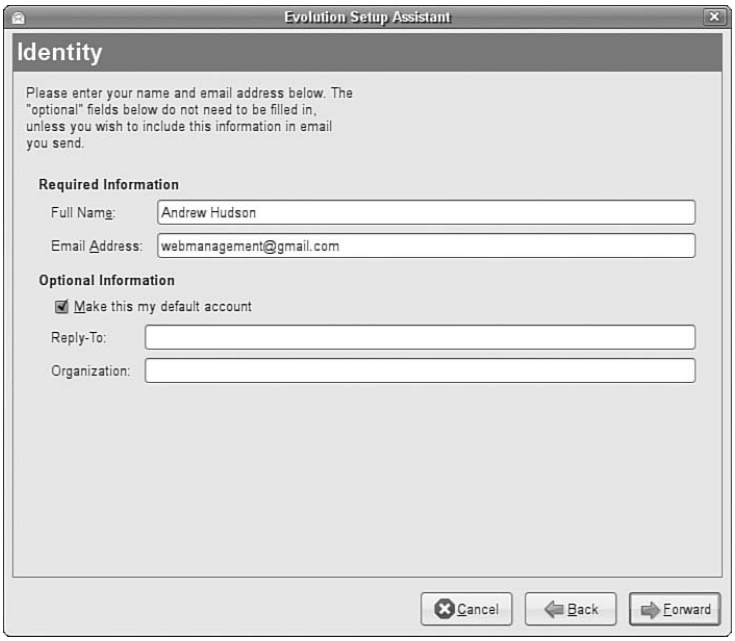


FIGURE 5.4 You can launch and configure Evolution with just a few simple commands. The Identity screen, the first of several screens, asks you to enter your information. Click Forward to proceed.

The next screen permits you to configure Evolution to use your Mail Transfer Agent. You can choose POP, IMAP, the local spools found in `/var/mail` in either `mbox` or `maildir` format, a local MTA, or None if you simply want to use the other features of Evolution. As shown in Figure 5.5, you can also set your password.

You must also choose between SMTP or Sendmail for sending your mail; enter your email address, and choose a time zone (very important for your calendar). Finally, you will see the opening Evolution window in Figure 5.6.

Each icon in the left pane of the main Evolution window opens a different window when selected. Each view has options that can be configured to suit your needs; you'll find access to the preferences dialog box under the Edit menu, which is shown in Figure 5.7.

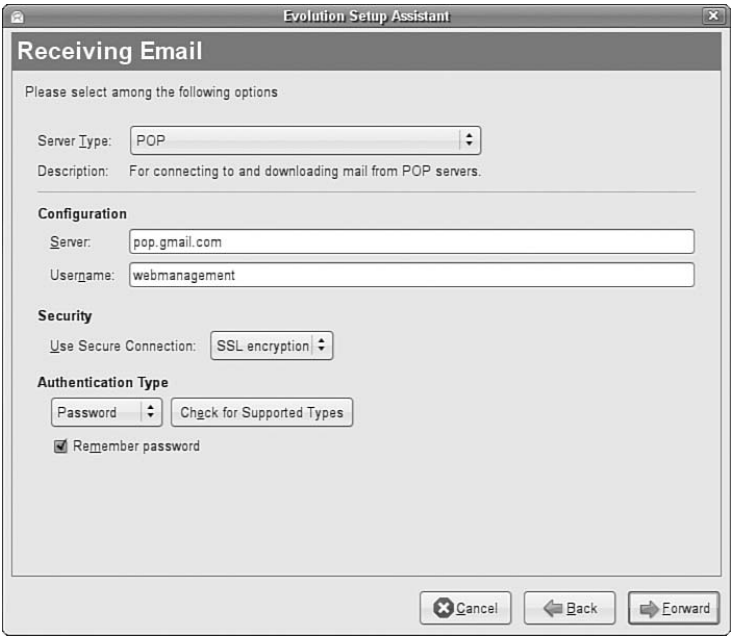


FIGURE 5.5 The Receiving Mail screen requires information from your ISP or system administrator.

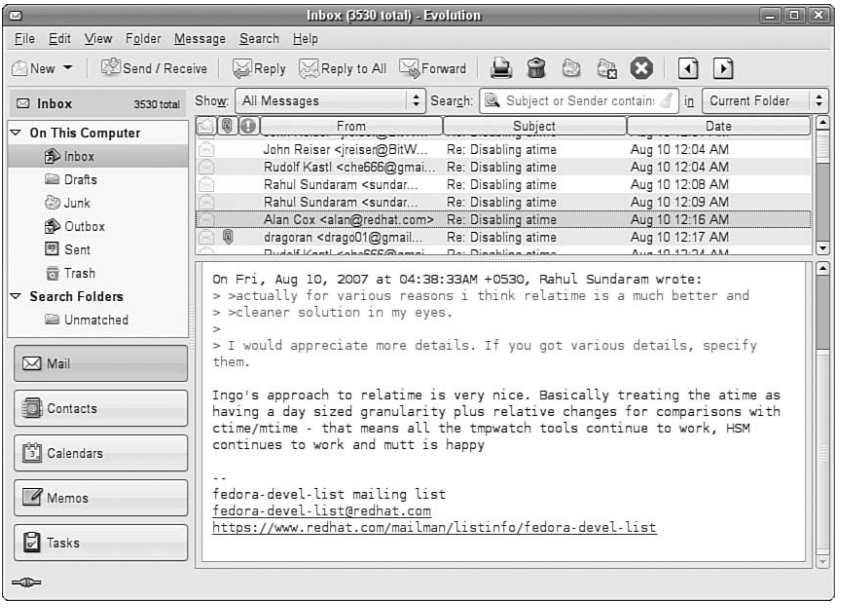


FIGURE 5.6 The standard Evolution display. On the left, you can see buttons to choose Mail, Contacts, Calendars, and Tasks windows.

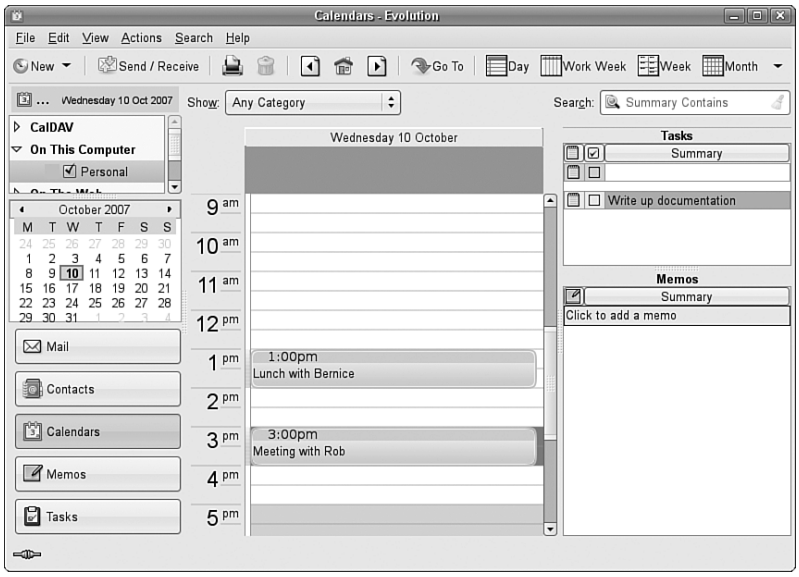


FIGURE 5.7 The calendar application Tools screen is where the information can be shared with others. Here, the times and dates can be configured.

Mozilla Thunderbird

Mozilla Thunderbird (see Figure 5.8) is the sister program to Firefox. Whereas Firefox is designed to browse the web, Thunderbird's specialty is communication. It can handle email, network news (see later in this chapter), and RSS feeds.

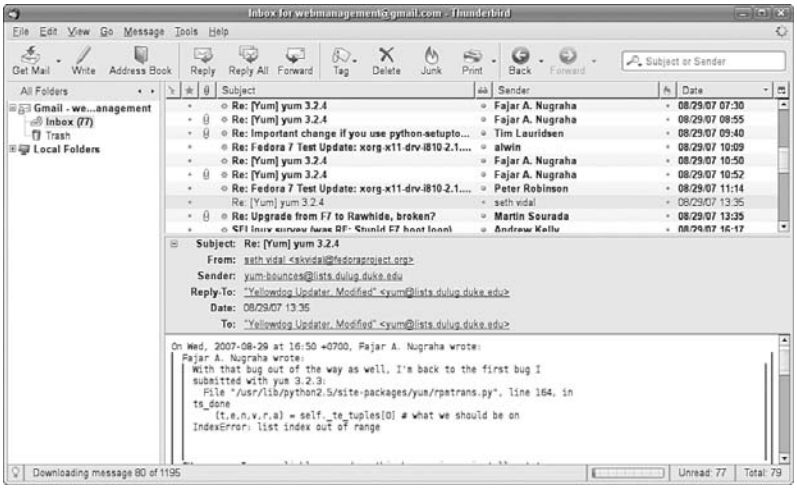


FIGURE 5.8 The natural companion to Firefox, Mozilla's lightweight email client Thunderbird can be found in use all over the world.

Thunderbird is not installed by default with Ubuntu, so you will have to use either `apt-get` or `synaptic` to install it. As with Firefox, there are many plug-ins and extensions to enhance your email and newsreading.

KMail

If you are using the KDE Desktop Environment rather than the Ubuntu default GNOME desktop, you will also have KMail installed. As with Balsa, it will not take users of Outlook Express or Mozilla Mail very long to get used to the KMail interface. Some useful features found in KMail are the choice of `mbox` or `maildir` formats, improved filter creation, the capability to sort mail into threads, and the capability to apply filters at the MTA. Figure 5.9 shows the KMail email program. KMail offers IMAP access, extensive filtering, `mbox` and `maildir` formats, and the capability to easily integrate MTAs such as Procmail, Spamassassin, or custom processing scripts.

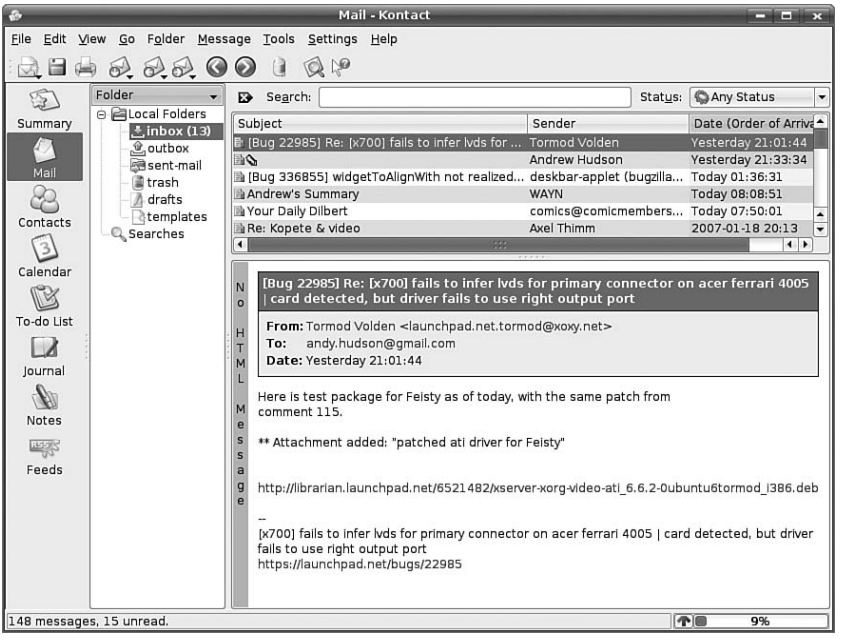


FIGURE 5.9 The KMail email client, part of the KDE Desktop Environment.

Other Mail Clients

The mail clients included by Ubuntu are only a few of those available. Claws is very popular because it offers spell-checking while typing and is well suited for use in large network environments in which network overhead and RAM usage are important considerations.

RSS Readers

RSS is one of the protocols of Web 2.0, the next generation of Internet content. Although RSS has been in use for a couple of years now, it has only recently started to really take off, thanks to adoption across a large number of websites and portals.

The key advantage of RSS is that you can quickly read news from your specific choice of websites at a time that suits you. Some services offer just the articles' headlines, whereas others offer full articles for you to view. RSS feeds can be accessed in various ways, even through your web browser!

Firefox

Firefox implements RSS feeds as what it calls Live Bookmarks (shown in Figure 5.10), which are essentially bookmarks with sub-bookmarks, each linking to a new page from your chosen website. I like to have several news sites grouped together under a folder on my toolbar called News, allowing me to quickly browse through my collection of sites and pick out articles that really interest me.

FIGURE 5.10 Live Bookmarks for Firefox, making all your news fixes just a mouse click away.

Liferea

Of course, not everyone wants to read RSS feeds with the browser. The main problem with reading RSS feeds with Firefox is that you get to see only the headline, rather than any actual text. This is where a dedicated RSS reader comes in handy, and Liferea (see Figure 5.11) is one of the best.

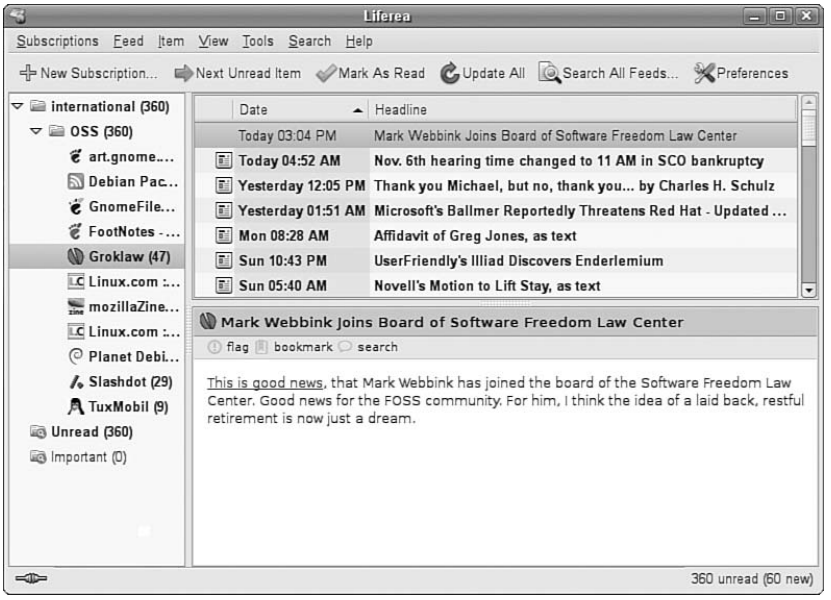


FIGURE 5.11 Read your daily news feeds with Liferea, a fantastic and easy-to-use RSS feed reader.

It is not installed by default, so you have to retrieve it by going to Applications, Add/Remove Software. After it is installed, you can find it under the Applications, Internet menu labeled simply Liferea.

By default, Liferea offers a number of RSS feeds, including Planet Debian, Groklaw, and Slashdot. Adding a new feed is very straightforward. All you need to do is select New Subscription under the Feeds menu and paste the URL of the RSS feed into the box. Liferea then retrieves all the current items available through that field, and displays the feed name on the left side for you to select and start reading.

Instant Messaging with Pidgin

Instant Messaging is one of the biggest ways for people to interact over the web. AOL was the primary force behind this, especially in America, but other networks and systems soon came onto the market providing users with a wealth of choice.

No longer just a consumer tool, instant messaging is now a part of the corporate world, with many different companies deploying internal instant messaging software for collaboration.

Pidgin was created as a multi-protocol instant messaging client enabling you to connect to several different networks that use differing protocols, such as AIM, MSN, Jabber, and others.

You can find Pidgin under Applications, Internet, listed as Pidgin Internet Messenger as is shown in Figure 5.12.

FIGURE 5.12 Pidgin is the Swiss army knife of instant messaging applications; it can work across many different IM networks.

Getting started with Pidgin is simple; when you start Pidgin, you are asked to add a new account. You select the network protocol for the account (i.e., AIM, MSN, etc.) and fill out your screen name and password, finally clicking Save to create the account within Pidgin and automatically log in. If any of your contacts are online, then they will appear in the buddy list window.

Internet Relay Chat

As documented in RFC 2812 and RFC 2813, the IRC protocol is used for text conferencing. Like mail and news, IRC uses a client/server model. Although it is rare for an individual to establish an IRC server, it can be done. Most people use public IRC servers and access them with IRC clients.

Ubuntu provides a number of graphical IRC clients, including X-Chat, licq, and Seamonkeychat, but there is no default chat client for Ubuntu. Ubuntu also provides the console clients `epic` and `licq` for those who eschew X. If you don't already have a favorite, you should try them all.

CAUTION

You should never use an IRC client while you are the root user. It is better to create a special user just for IRC because of potential security problems. To use X-Chat in this manner, you open a terminal window, use `su` to change to your IRC user, and start the xchat client.

X-Chat is a popular IRC client, and it is the client that is used in this chapter's example. The HTML documents for X-Chat are available in `/usr/share/docs/xchat`. It is a good idea to read them before you begin because they include an introduction to and cover some of the basics of IRC. You need to download and install X-Chat to launch the X-Chat client, select the IRC Client item from the Internet menu found under the Extras menu, or you can launch it from the command line, like this:

```
$ xchat &
```

The X-Chat application enables you to assign yourself up to three nicknames. You can also specify your real name and your username. Because many people choose not to use their real names in IRC chat, you are free to enter any names you desire in any of the spaces provided. You can select multiple nicknames; you might be banned from an IRC channel under one name, and you could then rejoin using another. If this seems slightly juvenile to you, you are beginning to get an idea of the type of behavior on many IRC channels.

When you open the main X-Chat screen, a list of IRC servers appears, as shown in Figure 5.13. After you choose a server by double-clicking it, you can view a list of channels available on that server by choosing Window, List Window. The X-Chat Channel List window appears. In that window, you can choose to join channels featuring topics that interest you. To join a channel, you double-click it.

The Wild Side of IRC

Do not be surprised at the number of lewd topics and the use of crude language on public IRC servers. For a humorous look at the topic of IRC cursing, see http://www.irc.org/fun_docs/nocuss.html. This site also offers some tips for maintaining IRC etiquette, which is essential if you do not want to be the object of any of that profanity! Here are some of the most important IRC etiquette rules:

- Do not use colored text, all-capitalized text, blinking text, or "bells" (beeps caused by sending `^G` to a terminal).
- Show respect for others.
- Ignore people who act inappropriately.

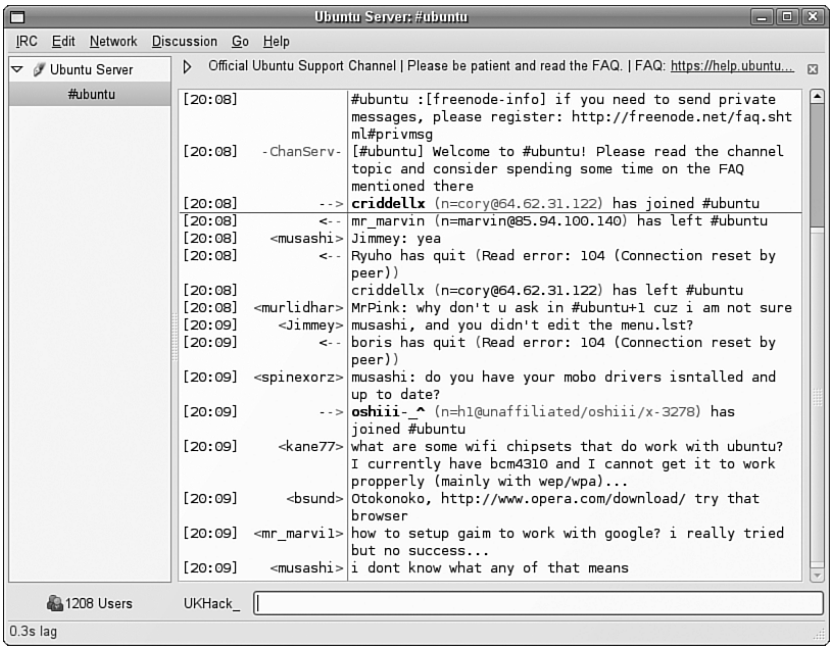


FIGURE 5.13 The main X-Chat screen presents a list of available public servers from which to select.

After you select a channel, you can join in the conversation, which appears as onscreen text. The messages scroll down the screen as new messages appear.

TIP

You can establish your own IRC server even though Ubuntu does not provide one. Setting up a server is not a task for anyone who is not well versed in Linux or IRC.

A popular server is IRCd, which you can obtain from ftp://ftp.irc.org/irc/server/. Before you download IRCd, you should look at the README file to determine what files you need to download and read the information at http://www.irchelp.org/irchelp/ircd/.

Usenet Network Newsgroups

The concept of newsgroups revolutionized the way information was exchanged between people across a network. The Usenet network news system created a method for people to electronically communicate with large groups of people with similar interests. As you will see, many of the concepts of Usenet news are embodied in other forms of collaborative communication.

Usenet newsgroups act as a form of public bulletin board system. Any user can subscribe to individual newsgroups and send (or *post*) messages (called *articles*) to the newsgroup so that all the other subscribers of the newsgroup can read them. Some newsgroups include an administrator, who must approve each message before it is posted. These are called *moderated* newsgroups. Other newsgroups are *open*, allowing any subscribed member to post a message. When an article is posted to the newsgroup, it is transferred to all the other hosts in the news network.

Usenet newsgroups are divided into a hierarchy to make it easier to find individual newsgroups. The hierarchy levels are based on topics, such as computers, science, recreation, and social issues. Each newsgroup is named as a subset of the higher-level topic. For example, the newsgroup `comp` relates to all computer topics. The newsgroup `comp.laptops` relates to laptop computer issues. Often the hierarchy goes several layers deep. For example, the newsgroup `comp.databases.oracle.server` relates to Oracle server database issues.

NOTE

The format of newsgroup articles follows the strict guidelines defined in the Internet standards document Request for Comments (RFC) 1036. Each article must contain two distinct parts: header lines and a message body.

The header lines identify information about when and by whom the article was posted. The body of the message should contain only standard ASCII text characters. No binary characters or files should be posted within news articles. To get around this restriction, binary files are converted to text data, through use of either the standard Unix uuencode program or the newer Multipurpose Internet Mail Extensions (MIME) protocol. The resulting text file is then posted to the newsgroup. Newsgroup readers can then decode the posted text file back into its original binary form.

A collection of articles posted in response to a common topic is called a *thread*. A thread can contain many articles as users post messages in response to other posted messages. Some newsreader programs allow the user to track articles based on the threads to which they belong. This helps simplify the organization of articles in the newsgroup.

TIP

The free news server `news.gmane.org` makes the Red Hat and Ubuntu mail lists available via newsgroups. The beta list is available as `gmane.linux.redhat.rhl.beta`. It is a handy way to read threaded discussions and easier than using the Ubuntu mail list archives.

The protocol used to transfer newsgroup articles from one host to another is Network News Transfer Protocol (NNTP), defined in RFC 975. (You can search RFCs at ftp://metalab.unc.edu/pub/docs/rfc/; look at the file `rfc-index.txt`.) NNTP was designed as a simple client/server protocol that enables two hosts to exchange newsgroup articles in an efficient manner.

The Pan News Client Newsreader

Whether or not your Ubuntu server is set up as a news server, you can use a newsreader program to read newsgroup articles. The newsreader programs require just a connection to a news server. It does not matter whether the news server is on the same machine or is a remote news server on the other side of the world.

Several programs are available for Unix systems to connect to news servers to read and post articles in newsgroups. Here we'll discuss the Pan news client.

Pan is a graphical newsreader client that works with GNOME and is the default newsreader for Ubuntu. If you have the GNOME libraries installed (and they usually are installed by default), you can also use Pan with the K Desktop Environment (KDE). Pan has the capability to download and display all the newsgroups and display posted news articles. You can launch it by using the GNOME or KDE desktop panel or from the command line of an X terminal window with the command `pan &`. Pan supports combining multipart messages and the `yenc` encoding/decoding protocol. Figure 5.14 shows a sample Pan display.

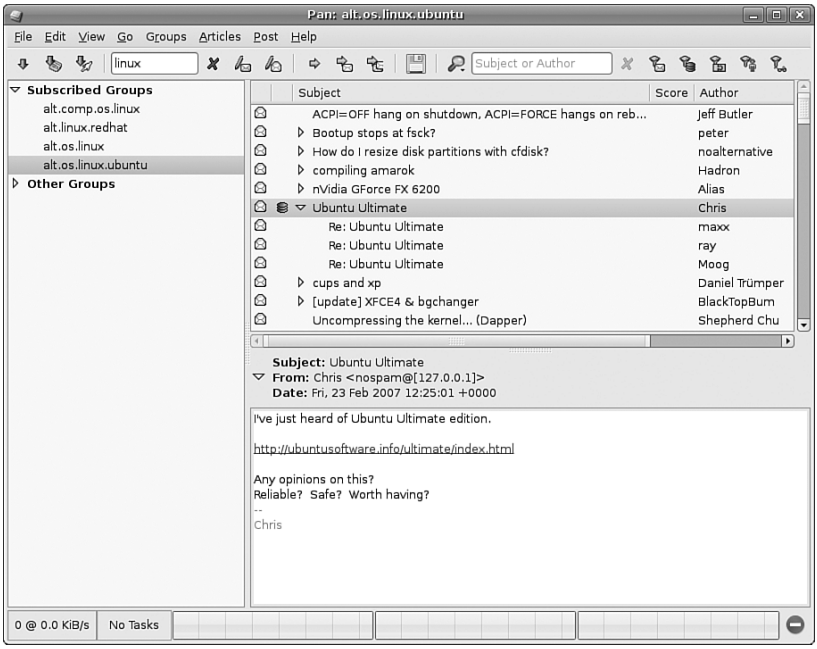


FIGURE 5.14 The Pan graphical newsreader is one of the nicest available for Linux, shown here displaying an image attached to a news article.

The first time you run Pan, a configuration wizard appears and prompts you for your name, the SMTP server name, the NNTP server name, and the name you want to use to identify the connection (in the example shown in Figure 5.15, we use a custom news server). After the wizard is finished, you are prompted to download a list of the newsgroups the server provides; this might take a while. If you need to change the news server or add an additional server, you can access the Preferences item under the Edit menu to bring up the list of servers. Then, you highlight the appropriate one and click Edit to change it or just click the New button to add a new news server.

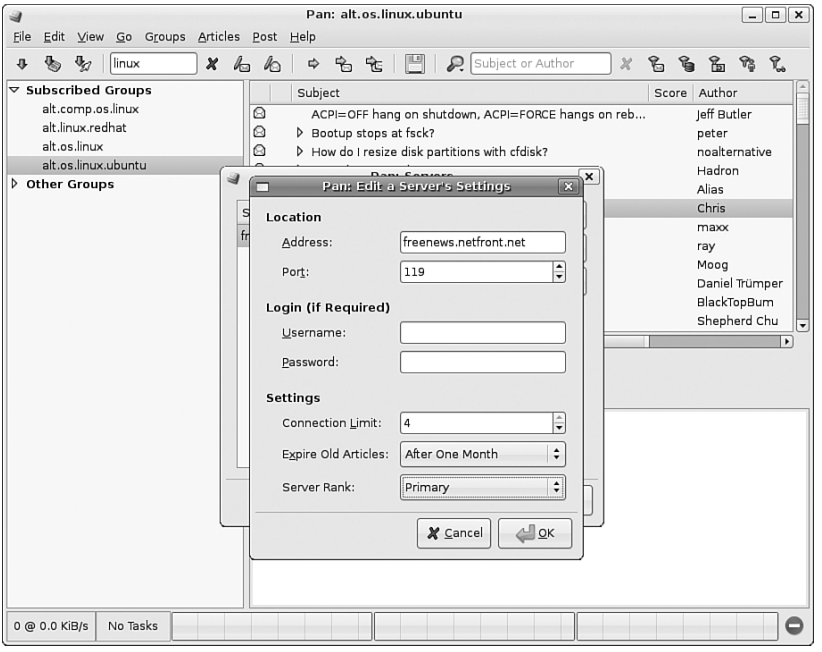


FIGURE 5.15 The Pan news server configuration window.

Videoconferencing with Ekiga

Ekiga is an Internet videoconferencing application that provides two-way voice and picture transmission over the Internet by using the H.323 protocol for IP telephony (also known as *Voice over IP* [VoIP]). It is an application similar to Microsoft NetMeeting and is provided with Ubuntu as the default videoconferencing client.

Before you can take full advantage of the phone and videoconferencing capabilities of Ekiga, you must configure a full-duplex–capable sound card and video device (see Chapter 7, "Multimedia Applications") as well as a camera.

Ekiga is found in the Internet menu as Videoconferencing; you click on the icon to launch it. When you start the Ekiga application for the first time, a configuration wizard

(called a "druid") runs and you are greeted by the first of four configuration screens. You simply enter your name, email address, and location and select your connection type. The settings for your audio and video devices are automatically detected; you can view them by selecting the Preferences item from the Edit menu. Figure 5.16 shows Ekiga in action, ready to dial another user.

FIGURE 5.16 *Ekiga* is surprisingly simple to use. A video source is not necessary; a static picture can be used as well.

When you have Ekiga running, you must register (from within Ekiga) with the server at http://ekiga.net/ to enable conferencing; Ekiga does this automatically for you if you told it to do so during the initial configuration.

You can find an informative FAQ at the Ekiga home page at http://www.Ekiga.org/ that you should read in full before using Ekiga. Also, an excellent article about VoIP is at http://freshmeat.net/articles/view/430/.

NOTE

If you frequently use VoIP applications such as Ekiga, you will tire of repetitively typing in long IP addresses to make connections. To avoid this hassle, you can use a "gatekeeper"—similar in purpose to a DNS server—to translate names into IP addresses. OpenH323 Gatekeeper is one such popular gatekeeper application. It is not provided with Ubuntu, but you can obtain it from http://www.gnugk.org/.

Reference

- http://www.novell.com/—The home of Ximian Evolution, the standard email client for Ubuntu.
- http://www.mozilla.com/—The home page for Mozilla Firefox, Thunderbird, and the Mozilla Suite.
- http://www.spreadfirefox.com/—The Firefox advocacy home page is useful for converting those Internet Explorer types.
- http://www.konqueror.org/—The homepage for Konqueror.
- http://www.claws-mail.org/— The homepage for Claws, the email client.
- http://ekiga.net/—Sign up here for a free SIP account for use with Ekiga.

CHAPTER 6

Productivity Applications

IN THIS CHAPTER

- Introducing OpenOffice.org
- Office Suites for Ubuntu
- Productivity Applications Written for Microsoft Windows
- Reference

With the rapid growth of open source software, businesses have directly benefited from developments in office productivity suites. Many businesses already use OpenOffice.org and its commercial counterpart, StarOffice, and they are already enjoying the cost benefits of not having to pay license fees or support costs. Of course, more suites are available than just OpenOffice.org, and in this chapter, we will explore the options available.

NOTE

OpenOffice.org is not 100% compatible with Microsoft Office. Why is this? Well, Microsoft is notoriously secretive about its proprietary file formats, and the only way that OpenOffice.org could ensure compatibility would be to reverse-engineer each file format, an exercise akin to taking apart a telephone to see how it works. This reverse-engineering could be classed as illegal under U.S. law, which would make OpenOffice.org somewhat of a potential hot-potato if they chose this path. However, OpenOffice.org manages to maintain a very high standard of importing and exporting so you should not experience too many problems.

A productivity suite could be classed as containing two or more applications that could be used for creating documents, presentations, spreadsheets, and databases. Other applications could include email clients, calculators/formula editors, and even illustration packages. Commonly they are all tied together by a default look and feel, which makes sticking to one particular suite much easier. Because Ubuntu uses OpenOffice.org as its standard office suite, we

will introduce you to Writer and Calc, the two most popular OpenOffice.org components. We will also take a brief look at some of the other Linux-based office suites that are available.

Working with OpenOffice.org

For the majority of users of productivity suites, OpenOffice.org should fulfill most, if not all, of your requirements. However, the first hurdle you need to get over is not whether it can do what you require of it, but rather whether it can successfully import and export to proprietary Microsoft formats. In the main, OpenOffice.org should import and export with minimal hassle, perhaps getting a bit stuck with some of the more esoteric Microsoft Office formatting. Given that most users do not go much beyond tabs, columns, and tables, this level of compatibility should suffice.

However, you are strongly advised to round up a selection of documents that could potentially fall foul of the import/export filter and test them thoroughly (of course, keeping a backup of the originals!). There is nothing worse than for a system administrator who has deployed a new productivity suite than to suddenly get users complaining that they cannot read their files. This would quickly destroy any benefits felt from the other useful functions within OpenOffice.org, and could even spell the return of proprietary formats and expensive office suites. Many users do not mind switching to OpenOffice.org, largely because the user interface closely resembles that of similar Microsoft applications. This helps to settle users into their environment and should dispel any fears they have over switching. Such similarity makes the transition to OpenOffice.org a lot easier.

Of course, just looking similar to Microsoft applications is not the only direct benefit. OpenOffice.org supports a huge array of file formats, and is capable of exporting to nearly 70 different types of documents. Such a wide variety of file formats means that you should be able to successfully use OpenOffice.org in nearly any environment.

Introducing OpenOffice.org

OpenOffice.org contains a number of productivity applications for use in creating text documents, preparing spreadsheets, organizing presentations, managing projects, and more. The following components of the OpenOffice.org package are included with Ubuntu:

- **Writer**—This word processing program enables you to compose, format, and organize text documents. If you are accustomed to using Microsoft Word, the functionality of OpenOffice.org Writer will be familiar to you. You will learn how to get up and running with Writer later on in this chapter.
- **Calc**—This spreadsheet program enables you to manipulate numbers in a spreadsheet format. Support for all but the most esoteric Microsoft Excel functions means that trading spreadsheets with Excel users should be successful. Calc offers some limited compatibility with Excel macros, but those macros generally have to be rewritten. We walk through setting up a basic spreadsheet with some formulas as well as showing you how to build a basic Data Pilot later on in this chapter.

- **Impress**—This presentation program is similar to Microsoft PowerPoint and enables you to create slide show presentations that include graphs, diagrams, and other graphics. Impress also works well with PowerPoint files.
- **Math**—This math formula editor enables you to write mathematical formulas with a number of math fonts and symbols for inclusion in a word processing document. Such symbols are highly specialized and not easily included in the basic functionality of a word processor. This is of interest primarily to math and science writers, but Math can be useful to anyone who needs to include a complex formula in text.
- **Base**—This database was introduced with the OpenOffice.org 2.0 suite, which is provided with Ubuntu. It provides a fully functional database application.
- **Draw**—This graphics application allows you to create images for inclusion in the documents produced with OpenOffice.org. It saves files only in OpenOffice.org format, but it can import most common image formats.
- **Dia**—This technical drawing editor from the GNOME Office suite enables you to create measured drawings, such as those used by architects and engineers. Its functionality is similar to that of Microsoft Visio.
- **Planner**—You can use this project management application for project planning, scheduling, and tracking; this application is similar to, but not compatible with, Microsoft Project. It is found in the Office menu as the Project Planner item.

A Brief History of OpenOffice.org

The OpenOffice.org office suite is based on a commercial suite called StarOffice. Originally developed by a German company, StarOffice was purchased by Sun Microsystems in the United States. One of the biggest complaints about the old StarOffice was that all the component applications were integrated under a StarOffice "desktop" that looked very much like a Microsoft Windows desktop, including a Start button and menus. This meant that to edit a simple document, unneeded applications had to be loaded, making the office suite slow to load, slow to run, and quite demanding on system resources.

After the purchase of StarOffice, Sun Microsystems released a large part of the StarOffice code under the GNU Public License, and development began on what has become OpenOffice.org, which is freely available under the GPL. Sun continued development on StarOffice and released a commercial version as StarOffice 6.0. The significant differences between the free and commercial versions of the software are that StarOffice provides more fonts and even more import/export file filters than OpenOffice.org (these filters cannot be provided in the GPL version because of licensing restrictions) and StarOffice provides its own relational database, Software AG's Adabas D database. The StarOffice counterpart to OpenOffice.org 2.3 is StarOffice 8.

Configuring OpenOffice.org

The installation of OpenOffice.org is done on a systemwide basis, meaning that all users have access to it. However, individual users have to go into OpenOffice.org to configure it for their individual needs. This initial configuration happens transparently the first time you load any of the OpenOffice.org components, and might mean the application takes a little longer to load as a result. Be patient, and your desired application will appear.

TIP

OpenOffice.org is constantly improving its productivity applications. You can check the OpenOffice.org website (http://www.openoffice.org/) for the latest version. The website provides a link to download the source or a pre-compiled version of the most current working installation files. A more current version might offer file format support that you need. Should you need a Windows-compatible version, you will also find it at the website.

Shown in Figure 6.1 is the Office menu, which is found under Applications. You can see the entries for Database (Base), Presentation (Impress), Spreadsheet (Calc), and Word Processor (Writer).

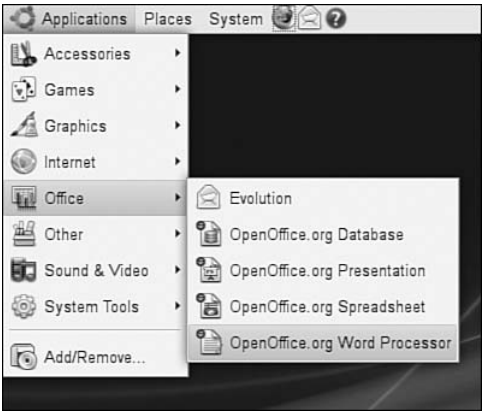


FIGURE 6.1 The OpenOffice.org suite provided by Ubuntu is simple to configure and use.

As is the case with many Linux applications, you may be somewhat overwhelmed by the sheer number of configuration options available to you in OpenOffice.org. Mercifully, a lot of thought has gone into organizing these options, which are available if you click the Tools menu and select Options. It does not matter which program you use to get to this dialog box; it appears the same if summoned from Writer, Impress, or Calc. It acts as a central configuration management tool for all OpenOffice.org applications. You can use it to set global options for all OpenOffice.org applications, or specific options for each individual component. For instance, in Figure 6.2, you can change the user details and information, and this is reflected across all OpenOffice.org applications.

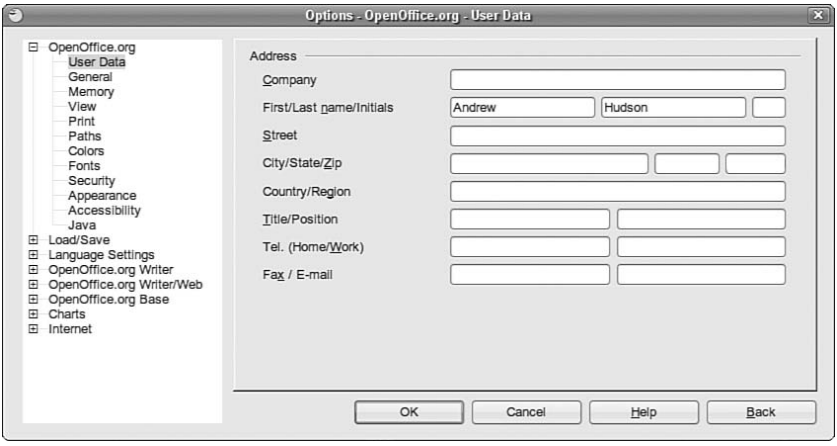


FIGURE 6.2 You can set user details for all OpenOffice.org applications from this dialog.

TIP

Two websites provide additional information on the functionality of OpenOffice.org:

http://lingucomponent.openoffice.org/download_dictionary.html—This site provides instructions and files for installing spelling and hyphenation dictionaries, which are not included with OpenOffice.org.

http://sourceforge.net/projects/ooextras/—This site provides templates, macros, and clip art, which are not provided with OpenOffice.org.

OpenOffice.org is a constant work in progress, but the current release is on par with the Sun version of StarOffice 8. You can browse to the OpenOffice.org website to get documentation and answers to frequently asked questions and to offer feedback.

Working with OpenOffice.org Writer

Out of all the applications that make up OpenOffice.org, the one that you are most likely to use on a regular basis is Writer, the OpenOffice.org word processor. With a visual style very similar to Microsoft's Word, Writer has a number of strengths over its commercial and vastly more expensive rival. In this section, you will learn how to get started with Writer and make use of some of its powerful formatting and layout tools.

NOTE

You may be interested to know that Writer was the primary word processor chosen to write and edit this book.

Getting Started with Writer

You can access Writer either through its shortcut on the panel or by going to the Applications, Office menu and selecting Word Processor. After a few seconds, Writer opens up with a blank document and a blinking cursor awaiting your command. It can be tempting to just dive in and start typing your document, but it can be worthwhile to do some initial configuration before you start work.

First of all, make sure that the options are set to your requirements. Click the Tools menu and select Options to bring up the Options dialog box, as seen in Figure 6.2. The initial screen allows you to personalize OpenOffice.org with your name, address, and contact details, but there are options to configure features that you might also want to alter. First of all, check that your default paths are correct by clicking the Paths option. You might want to alter the My Documents path, as shown in Figure 6.3, to something a little more specific than just your home directory.

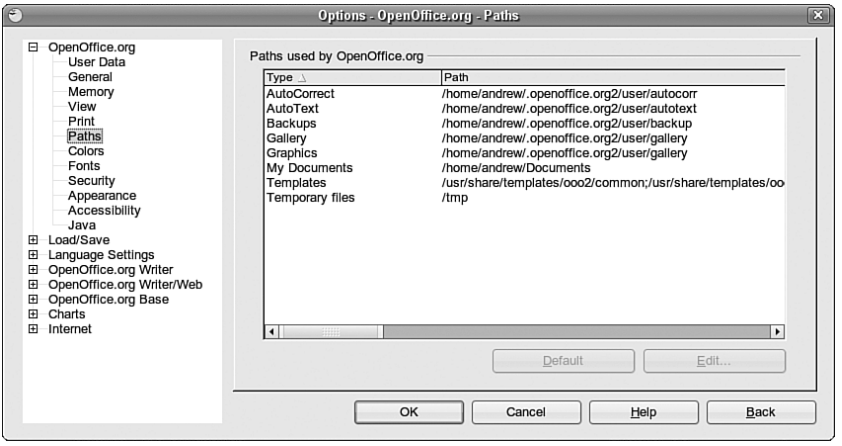


FIGURE 6.3 Click the Edit button to choose your default documents directory.

You might also want to change OpenOffice.org so that it saves in Microsoft Word format by default, should you so require. This can be done under the Load/Save General options shown in Figure 6.4, and it is a good idea if you value your work to change the Autorecovery settings so that it saves every couple of minutes.

Also shown in Figure 6.4 are a set of options that are specific to Writer. From top to bottom, they are

- **General**—Specify options that affect the general use of Writer.
- **View**—Specify what you want Writer to display.
- **Formatting Aids**—Specify whether you want to see non-printing characters.
- **Grid**—Create a grid that you can use to snap frames and images in place.
- **Basic Fonts**—Select your default fonts for your document here.

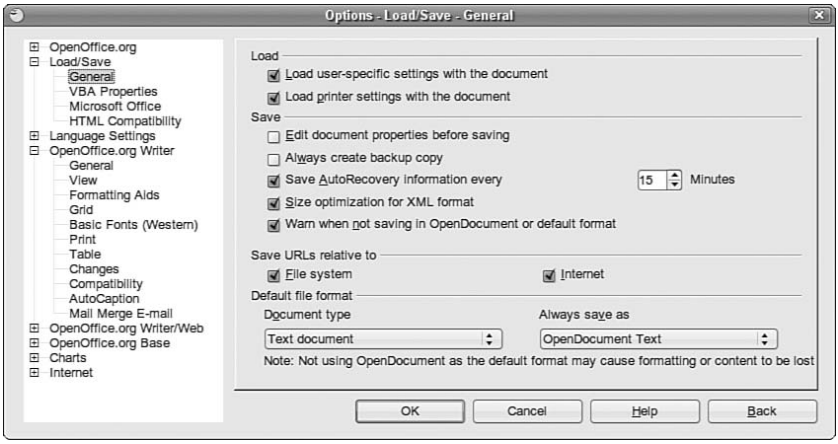


FIGURE 6.4 Make sure that you are working with most appropriate file formats for you.

- **Print**—Specify exactly what you want Writer to output when you print your document.
- **Table**—Set options for drawing tables within Writer.
- **Changes**—Define how Writer handles changes to documents.
- **Compatibility**—A set of rules that Writer uses to ensure close compatibility with earlier versions of Writer.
- **AutoCaption**—Create automatic captions for images, charts, and other objects.

A little bit of time working through these options can give you a highly personalized and extremely productive environment.

Working with Styles and Formatting

One of the significant benefits of using Writer is the ability you have to easily apply formatting and styles to extremely complex documents. Depending on the types of documents you work with, you might want to consider creating your own styles beyond the 20 included by default. You can access styles through either the Style drop-down box in the toolbar or the Styles and Formatting window shown in Figure 6.5. If you cannot see the window, press the F11 key to display it.

The easiest way to work with the Styles and Formatting tool is to highlight the text you want to style and double-click the required style in the window. There are quite a few to choose from, but you might find them restrictive if you have more specialized needs. To start defining your own styles, press Ctrl+F11 to bring up the Style Catalog, shown in Figure 6.6, where you add, modify, and delete styles for pages, paragraphs, lists, characters, and frames.

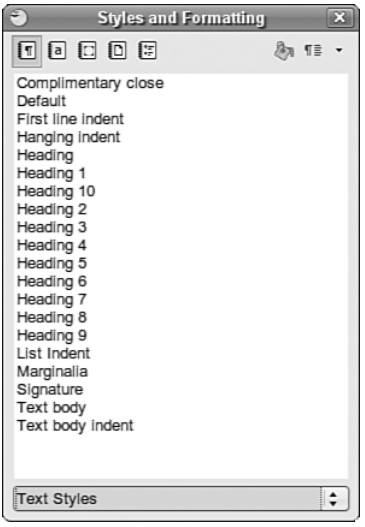


FIGURE 6.5 Writer's quick and easy-to-use Styles and Formatting tool.

FIGURE 6.6 Writer's powerful Style Catalog gives you control over every aspect of styling.

Working with OpenOffice.org Calc

The spreadsheet component of OpenOffice.org is named Calc, and is a very capable Excel alternative.

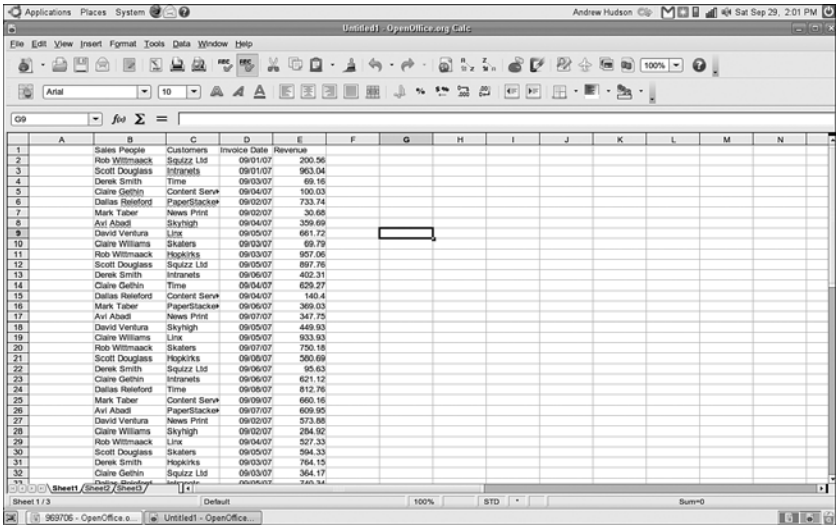
Calc is used for storing numerical information that you need to analyze in some way. So, for instance, you could use it to help you budget month by month. It can take care of the calculations for you, as long as you tell Calc what you want it to do. Anyone with experience in Excel will feel right at home with Calc.

In this section, we will show you how to get started with Calc, including entering formulas and formatting. We will also take a look at some of the more advanced features of Calc, including the Data Pilot feature, which allows you to easily summarize information.

Getting Started with Calc

You can either click the shortcut icon that is located on the top GNOME panel, or select Spreadsheet from the Office menu under the Applications main menu. Whichever route you take, the result is the same and Calc starts to load.

By default, Calc loads with a blank spreadsheet just waiting for you to enter information into it. In Figure 6.7, you can see that we have already started to enter some basic information into Calc.



	A	B	C	D	E
1		Sales People	Customers	Invoice Date	Revenue
2		Rob Wittmaack	Squizz Ltd	09/01/07	200.56
3		Scott Douglass	Intranets	09/01/07	963.04
4		Derek Smith	Time	09/03/07	69.16
5		Claire Gelhin	Content Serv	09/04/07	100.03
6		Dallas Releford	PaperStacke	09/02/07	733.74
7		Mark Taber	News Print	09/02/07	30.68
8		Avi Abadi	Skyhigh	09/04/07	359.69
9		David Ventura	Linx	09/05/07	661.72
10		Claire Williams	Skaters	09/03/07	69.79
11		Rob Wittmaack	Hopkirks	09/03/07	957.06
12		Scott Douglass	Squizz Ltd	09/05/07	897.76
13		Derek Smith	Intranets	09/06/07	402.31
14		Claire Gelhin	Time	09/04/07	629.27
15		Dallas Releford	Content Serv	09/04/07	140.4
16		Mark Taber	PaperStacke	09/06/07	369.03
17		Avi Abadi	News Print	09/07/07	347.75
18		David Ventura	Skyhigh	09/05/07	449.93
19		Claire Williams	Linx	09/05/07	933.93
20		Rob Wittmaack	Skaters	09/07/07	750.18
21		Scott Douglass	Hopkirks	09/05/07	580.69
22		Derek Smith	Squizz Ltd	09/06/07	95.63
23		Claire Gelhin	Intranets	09/06/07	621.12
24		Dallas Releford	Time	09/05/07	812.76
25		Mark Taber	Content Serv	09/09/07	660.16
26		Avi Abadi	PaperStacke	09/07/07	609.95
27		David Ventura	News Print	09/02/07	573.88
28		Claire Williams	Skyhigh	09/02/07	284.92
29		Rob Wittmaack	Linx	09/04/07	527.33
30		Scott Douglass	Skaters	09/05/07	594.33
31		Derek Smith	Hopkirks	09/03/07	764.15
32		Claire Gelhin	Squizz Ltd	09/03/07	364.17

FIGURE 6.7 Use Calc to store numerical and statistical information.

Calc's layout makes it easy to organize information into rows and columns. As you can see in the example, we have sales people listed in the left column, customers in the second column, Invoice Date in the third column, and finally Revenue in the fourth column. At the moment, there are no formulas entered to help you interpret the data. Clicking the E43 cell selects it and enables you to enter in a formula in the top formula bar. If you enter in the equal sign, Calc knows that you are entering a formula and works accordingly.

In this example, we want to know the total revenue brought in up to now, so the formula to enter is `=sum(E3:E42)`, followed by Return. Calc automatically enters the result into cell E43 for you to see. Now you want to see what the average order value was. To do this, you have to obtain the number orders made. For this you can use the `counta` function to count the number of entries in a given list. This is usually used when you need to find out how many entries there are in a text list. So, in cell B43, enter **`=counta(B3:B42)`** and press Enter. Calc now counts the number of entries in the range and returns the total in

B43. All that remains for you to do is divide the total revenue by the number of orders to find the average order value. So, in cell E44, enter the formula **=E43/B43** to get the average order value.

TIP

Calc offers some nifty little features that you can use quickly if you need to. The handiest one in our opinion is the capability to select multiple cells and see straight away the total and average of the range. You will find these figures in the bottom-right status bar. This has saved us numerous times when we have needed to get this information quickly!

Formatting Your Spreadsheets

Getting back to our example, it looks a little basic at the moment as there is no formatting involved. For instance, what's the billing currency? You can also see that some of the cells have text that does not fit, which is highlighted by a small right arrow in the cell. We should also add some labels and titles to our spreadsheet to make it a bit more visually appealing.

To start off, all the revenue figures can be changed into currency figures. To do this, select all the cells containing revenue information and click on the small icon shown in Figure 6.8. This immediately formats the cells so that they display the dollar sign and also puts in a thousands separator to make the numbers easier to read.

FIGURE 6.8 Make numbers more meaningful with the currency and percentage icons.

Now you need to space all the cells so that you can read all the information. A quick and easy way to do this is to click the area immediately to the left of column A and immediately above row 1 to select the entire spreadsheet. Now all you have to do is double-click the dividing lines and each column resizes according to its longest entry.

Next you can add a little color to the worksheet by using the paint can icon in the toolbar. Select the range B2 to E2 with the mouse cursor and click the paint can icon to bring up the color window shown in Figure 6.9. Now select the color you want to use and

Calc fills the cells with that color. You can also change the font color by using the icon immediately to the right in the same way.

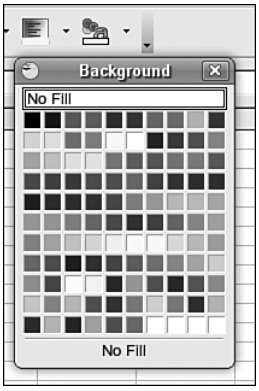


FIGURE 6.9 Add a touch of color to an otherwise dull spreadsheet with the fill background icon.

Finally you need a couple more finishing touches. The first one is to enlarge the font for the column headers. Select the range B2 to E2 again and click the font size in the toolbar to change it to something a little larger. You might also want to use the bold and italic options to emphasize the headers and also the totals some more.

If you have followed the steps as described, you should end up with a spreadsheet similar to the one in Figure 6.10.

Summarizing Data with Calc

Calc includes a powerful tool that lets you summarize large groups of data to help you when you need to carry out any analysis. This tool is called a *data pilot*, and you can use it to quickly summarize data that might normally take a long time if you did the calculations manually. Using the sample spreadsheet from earlier, we will take you through how to build a simple data pilot, showing you how to analyze and manipulate long lists of data.

The previous section featured a spreadsheet that showed sales people, customers, date of invoice, and revenue. At the foot of the spreadsheet were a couple of formulas that enabled you to quickly see the total revenue earned and the average order value.

Now you want to find out how much sales people have earned individually. Of course you could add this up manually with a calculator, but that would defeat the point of using Calc. So, you need to create a data pilot to summarize the information.

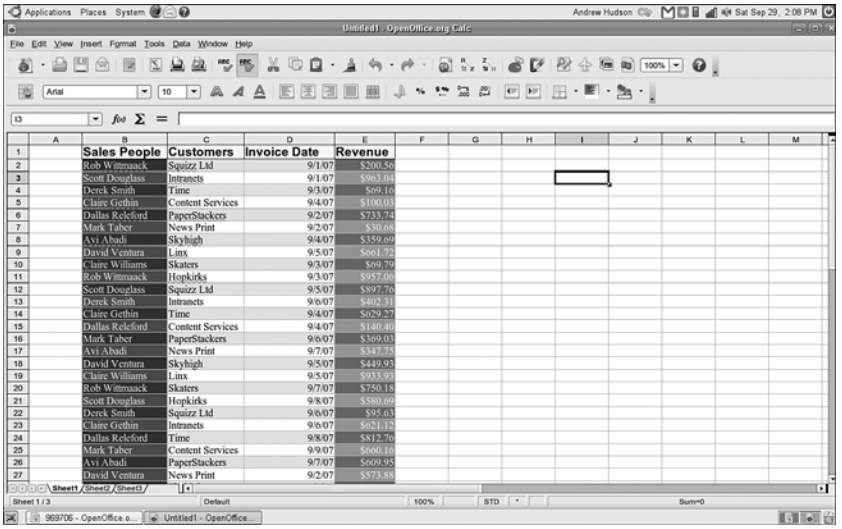


FIGURE 6.10 The finished article, looking a lot better than before!

First, you need to select all the cells from B2 to E42 as they contain the data you want to analyze. After these are selected, click on the Data menu and select Data Pilot, Start to open the Data Pilot Wizard. The first screen is shown in Figure 6.11 and is defaulted to current selection. Make sure that you choose this one to use the data in the selected range and click OK to continue.

FIGURE 6.11 Use either the current selection or an external data source to provide the data pilot with information.

The next screen enables you to lay out your data pilot as you want it. In this example, you want to have Sales Person in the left column marked Row Fields, so click and drag the Sales Person option from the list on the right and drop it onto the Row Fields area. You also want to see the revenue broken down by Date, so drag the Invoice Date box to the Column Field area. Finally, drag Customers to the Page Field area (so you can filter by Customer) and drag the Revenue option to the Data Field area. You should end up with something like Figure 6.12 and you are almost ready to display your data pilot.

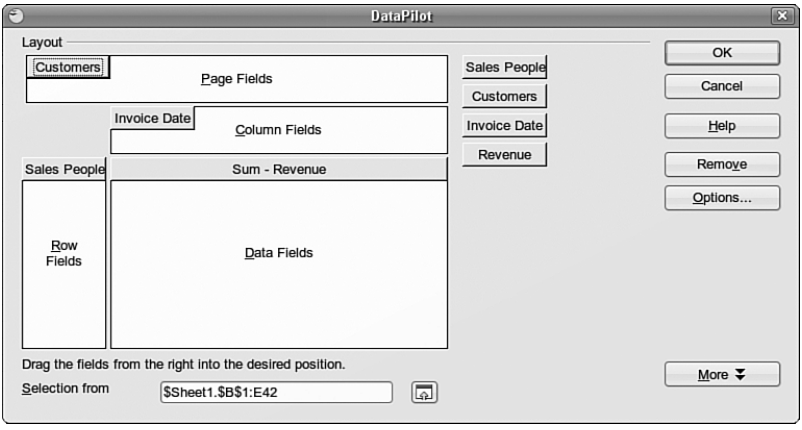


FIGURE 6.12 Lay out your data pilot as you want it.

The final piece in the puzzle is to tell Calc where you want it to place the finished data pilot. To do this, click the More button to drop down some extra options and select the box Send Results To to choose a new sheet. When you click OK now, Calc builds the data pilot and displays it on a new sheet in your workbook. The new data pilot can be seen in Figure 6.13.

	A	B	C	D	E	F	G	H	I	J	K
1	Filter										
2	Customers	- all -									
3											
4	Sum - Revenue	Invoice Date									
5	Sales People	09/01/07	09/02/07	09/03/07	09/04/07	09/05/07	09/06/07	09/07/07	09/08/07	09/09/07	Total Result
6	Avi Abadi				$392.96			$957.70			$1,350.66
7	Claire Gethin			$364.17	$729.30		$621.12	$960.88			$2,675.47
8	Claire Williams		$284.92	$89.79		$833.93	$763.96				$2,052.59
9	Dallas Releford		$733.74		$140.40	$740.34			$1,215.39		$2,829.87
10	David Ventura		$573.88		$281.51	$1,111.65					$1,967.05
11	Derek Smith			$833.31		$72.02	$497.94				$1,403.27
12	Mark Taber		$30.66				$606.55			$660.16	$1,297.38
13	Rob Willmaack	$200.56		$957.06	$327.33			$1,417.67			$3,102.62
14	Scott Douglass	$963.04				$1,819.88			$580.69		$3,363.61
15	Total Result	$1,163.60	$1,623.22	$2,224.34	$2,071.50	$4,677.82	$2,489.56	$3,336.26	$1,796.07	$660.16	$20,042.53
16											

FIGURE 6.13 Summarize large volumes of numerical data with ease, using Calc's Data Pilot function.

Office Suites for Ubuntu

As we have mentioned earlier, OpenOffice.org is the default application suite for Ubuntu. However, with all things open source, there are plenty of alternatives should you find that OpenOffice.org does not meet your specific requirements. These include the popular Gnome Office and also KOffice, the default KDE productivity suite. You are more likely to hear more about OpenOffice.org, especially as more and more people wake up to the fact that it is compatible with Microsoft Office file formats. In fact, the state of Massachusetts recently elected to standardize on two file formats for use in government: the Adobe Acrobat PDF format and the OASIS OpenDocument format, both of which are supported natively in OpenOffice.org.

NOTE

The decision by the state of Massachusetts to standardize on PDF and OpenDocument has huge ramifications for the open source world. It is the first time that OpenDocument, an already-agreed open standard, has been specified in this way. What it means is that anyone who wishes to do business with the state government must use OpenDocument-based file formats, and not the proprietary formats in use by Microsoft. Unfortunately for Microsoft, it does not have support for OpenDocument in any of its applications, making them useless to anyone wishing to work with the state government. This is despite Microsoft being a founding member of OASIS, who developed and ratified the OpenDocument standard!

Working with Gnome Office

The other office suite available for GNOME is Gnome Office, which is a collection of individual applications. Unlike OpenOffice.org, Gnome Office does not have a coherent suite of applications, meaning that you have to get used to using a word processor that offers no integration with a spreadsheet, and that cannot work directly with a presentation package. However, if you need only one or two components, it is worthwhile investigating Gnome Office.

The GTK Widget Set

Open Source developers are always trying to make it easier for people to build applications and help in development. To this end, there are a number of widgets or toolkits that other developers can use to rapidly create and deploy GUI applications. These widgets control things such as drop-down lists, Save As dialogs, window buttons, and general look and feel. Unfortunately, whereas Windows and Apple developers have to worry about only one set of widgets each, Linux has a plethora of different widgets, including GTK+, QT, and Motif. What is worse is that these widgets are incompatible with one another, making it difficult to easily move a finished application from one widget set to another.

GTK is an acronym for *GIMP Tool Kit*. The GIMP (The GNU Image Manipulation Program) is a graphics application very similar to Adobe Photoshop. By using the GTK-based jargon, we save ourselves several hundred words of typing and help move along our discussion of GNOME Office. You might also see similar references to QT and Motif, as well as other widget sets, in these chapters.

Here are some of the primary components of the Gnome Office suite that are available in Ubuntu:

- **AbiWord**—This word processing program enables you to compose, format, and organize text documents and has some compatibility with the Microsoft Word file format. It uses plug-ins (programs that add functionality such as language translation) to enhance its functionality.

- **Gnumeric**—This spreadsheet program enables you to manipulate numbers in a spreadsheet format. Support for all but the most esoteric Microsoft Excel functions means that users should have little trouble trading spreadsheets with Excel users.
- **The GIMP**—This graphics application allows you to create images for general use. It can import and export all common graphic file formats. The GIMP is analogous to Adobe's Photoshop application and is described in Chapter 7, "Multimedia Applications."
- **Evolution**—Evolution is a mail client with an interface similar to Microsoft Outlook, providing email, scheduling, and calendaring. It is described in Chapter 5, "On the Internet."

The loose association of applications known as Gnome Office includes several additional applications that duplicate the functionality of applications already provided by Ubuntu. Those extra GNOME applications are not included in a default installation of Ubuntu to eliminate redundancy. They are all available from the Gnome Office website, at http://www.gnome.org/gnome-office/. Both The GIMP and Evolution are available with Ubuntu by default. You have to use `yum` or `pirut` to retrieve the remaining components.

Ubuntu provides the AbiWord editor as part of its Extras, shown in Figure 6.14. AbiWord can import XML, Microsoft Word, RTF, UTF8, plain text, WordPerfect, KWord, and a few other formats. AbiWord is notable for its use of plug-ins, or integrated helper applications, that extend its capabilities. These plug-ins add language translation, HTML editing, a thesaurus, a Linux command shell, and an online dictionary, among other functions and features. If you just need a simple yet powerful word processing application, you should examine AbiWord.

AbiWord is not installed by default in Ubuntu, so you'll need to install it either using `apt-get` or `synaptic`. The package is simply called `abiword-gnome`, although you might want to install the meta-package `gnome-office`, which will include the other Gnome Office applications.

After you've installed Abiword, it becomes available in the Applications menu, under the Office submenu. Simply click the icon to launch the application.

If you are familiar with Microsoft Works, the AbiWord interface will be familiar to you because its designers based the interface upon Works.

You can use the Gnumeric spreadsheet application to perform financial calculations and to graph data, as shown in Figure 6.15. It can import comma- or tab-separated files, text, or files in the Gnumeric XML format, saving files only as XML or text. You need to install Gnumeric using either `apt-get` or `synaptic` in the same way as Abiword. If you have already installed the `gnome-office` package, Gnumeric will be available under Applications, Office as Gnumeric Spreadsheet.

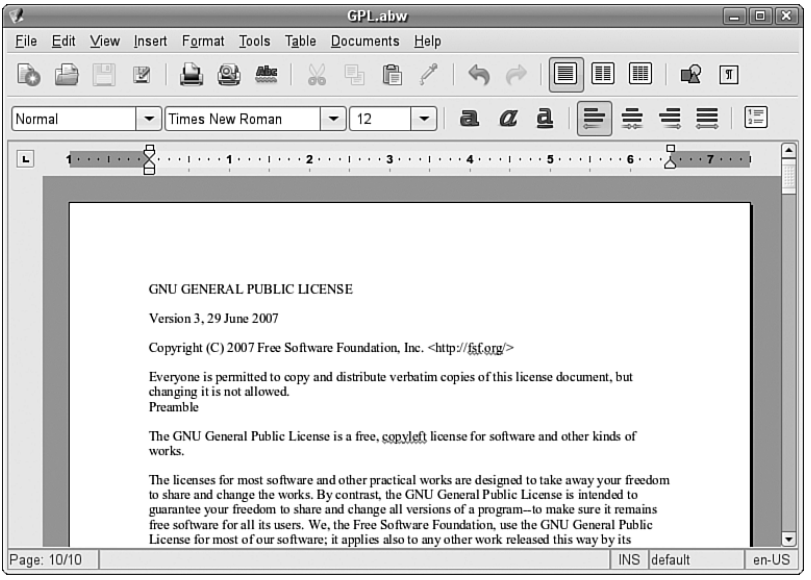


FIGURE 6.14 AbiWord is a word processing program for Ubuntu, GNOME, and X11. It handles some formats that OpenOffice.org cannot, but does not yet do well with Microsoft Word formats.

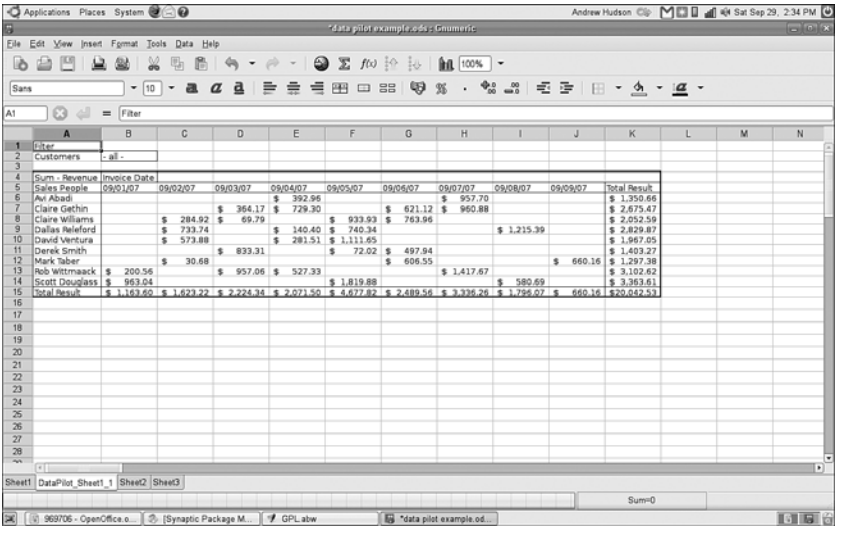


FIGURE 6.15 GNOME's Gnumeric is a capable financial data editor—here working with the same spreadsheet used earlier. OpenOffice.org also provides a spreadsheet application, as does KOffice.

After you press Enter, the main Gnumeric window appears. You enter data in the spreadsheet by clicking a cell and then typing in the text box. To create a graph, you click and drag over the spreadsheet cells to highlight the desired data, and then you click the Graph Wizard icon in Gnumeric's toolbar. Gnumeric's graphing component launches and you are guided through a series of dialogs to create a graph. When you are finished, you can click and drag a blank area of your spreadsheet, and the graph appears.

The Project Planner application is useful for tracking the progress of projects, much like its Windows counterpart, Microsoft Project. When the main window is displayed, you can start a new project or import an existing project. The application provides three views: Resources, Gantt Charts, and Tasks.

Working with KOffice

The KDE office suite KOffice was developed to provide tight integration with the KDE desktop. Integration enables objects in one application to be inserted in other applications via drag-and-drop, and all the applications can communicate with each other, so a change in an object is instantly communicated to other applications. The application integration provided by KDE is a significant enhancement to productivity. (Some GNOME desktop applications share a similar communication facility with each other.) If you use the KDE desktop instead of the default GNOME desktop, you can enjoy the benefits of this integration, along with the Konqueror web and file browser.

The word processor for KOffice is KWord. KWord is a frames-based word processor, meaning that document pages can be formatted in framesets that hold text, graphics, and objects in enclosed areas. Framesets can be used to format text on a page that includes text and images within columns that the text needs to flow around, making KWord an excellent choice for creating documents other than standard business letters, such as newsletters and brochures.

KWord and other components of KOffice are still under development and lack all the polished features of OpenOffice.org and AbiWord. However, it does have the ability to work with the OpenDocument format found in OpenOffice.org, as well as limited compatibility with Microsoft file formats.

You can access the KOffice components from the Office menu.

KWord asks you to select a document for your session. The KWord client, shown in Figure 6.16, offers sophisticated editing capabilities, including desktop publishing.

The KOffice KSpread client is a functional spreadsheet program that offers graphing capabilities. Like KWord, KSpread can be accessed from the Office menu.

KDE includes other productivity clients in its collection of KOffice and related applications. These clients include an address book, time tracker, calculator, notepad, and scheduler. One popular client is Kontact, which provides daily, weekly, work week, and monthly views of tasks, to-do lists, and scheduled appointments with background alarms. A journal, or diary, function is also supported within it, and you can synchronize information with your Palm Pilot. You can launch this client from the Office menu.

A typical Kontact window is shown in Figure 6.17.

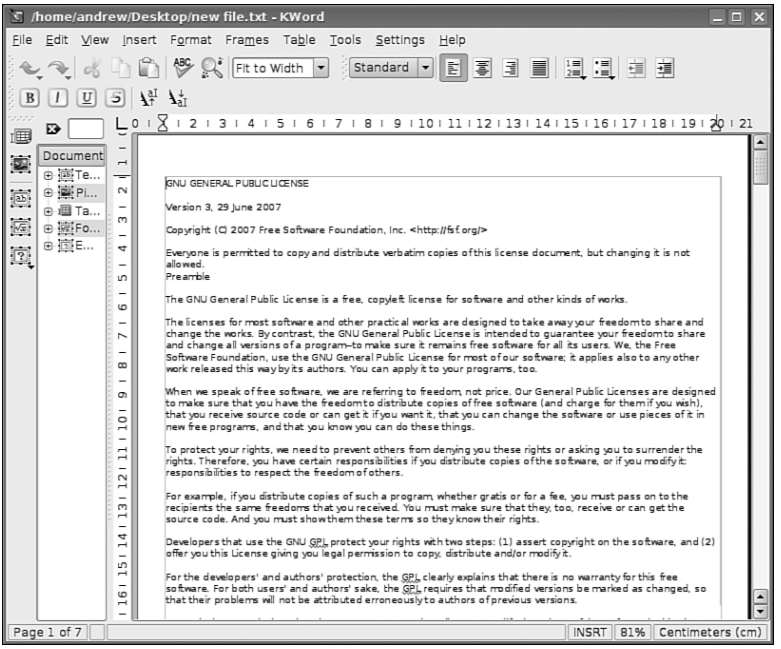


FIGURE 6.16 The KOffice KWord word processing component is a sophisticated frames-based WYSIWYG editor that is suitable for light desktop publishing, supporting several formats, including WordPerfect.

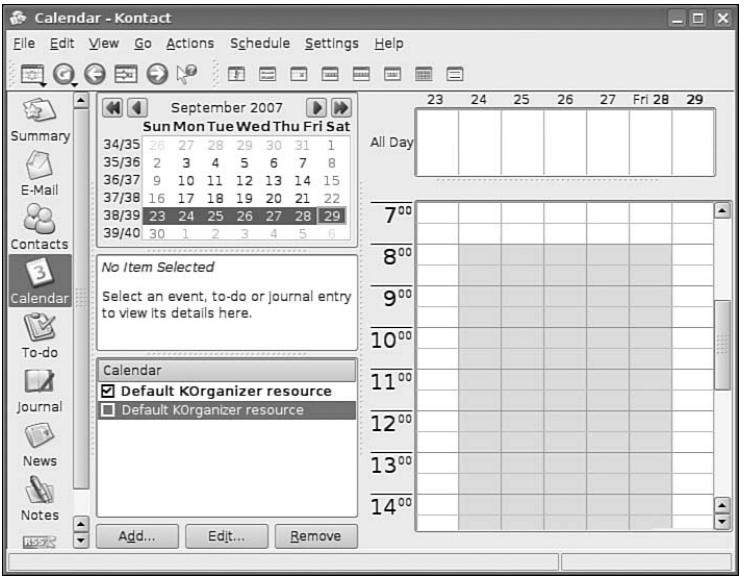


FIGURE 6.17 KDE's Kontact client supports editing of tasks and schedules that you can sync with your PDA. Shown here is the address book as well.

Productivity Applications Written for Microsoft Windows

Microsoft Windows is fundamentally different from Linux, yet you can install and run some Microsoft Windows applications in Linux by using an application named Wine. Wine enables you to use Microsoft Windows and DOS programs on Unix-based systems. Wine includes a program loader that you can use to execute a Windows binary, along with a `.dll` library that implements Windows command calls, translating them to the equivalent Unix and X11 command calls. Because of frequent updates to the Wine code base, Wine is not included with Ubuntu. Download a current version of Wine from http://www.winehq.org/. To see whether your favorite application is supported by Wine, you can look at the Wine application database at http://appdb.winehq.org/appbrowse.php.

As well, there are other solutions to enable use of Microsoft productivity applications, primarily CodeWeavers' CrossOver Office. If you are after a painless way of running not only Microsoft Office, but also Apple iTunes and other software, you should really pay CodeWeavers a visit. CrossOver Office is one of the simplest programs you can use to get Windows-based programs to work. Check out www.codeweavers.com to download a trial version of the latest software. It requires registration, but do not worry—the guys at CodeWeavers are great and will not misuse your details. The big plus is that you get a whole month to play around with the trial before you decide whether to buy it. Of course, you might get to the end of the 30 days and realize that Linux does what you want it to do and you don't want to go back to Windows. Do not be afraid; take the plunge!

Relevant Ubuntu Commands

The following commands give you access to productivity applications, tools, and processes in Ubuntu:

`oowriter`—OpenOffice.org's Writer

oocalc—OpenOffice.org's Calc

ooimpress—OpenOffice.org's Impress

`koshell`—KDE's KOffice office suite shell

`kspread`—KDE's KSpread spreadsheet

`gimp`—The GIMP (GNU Image Manipulation Package)

`gnumeric`—A spreadsheet editor for GNOME

`planner`—A project management client for GNOME

`abiword`—A graphical word processor for GNOME

Reference

- http://www.openoffice.org—The home page for the OpenOffice.org office suite.
- http://www.gnome.org/gnome-office/—The GNOME Office site.
- http://www.koffice.org/—The home page for the KOffice suite.
- http://www.codeweavers.com/—Website of the hugely popular CrossOver Office from CodeWeavers that allows you to run Windows programs under Linux.

CHAPTER 7

Multimedia Applications

IN THIS CHAPTER

The twenty-first century has become the century of the digital lifestyle, with millions of computer users around the world embracing new technologies, such as digital cameras, MP3 players, and other assorted multimedia gadgets. Whereas 10 years ago you might have had a collection of WAV files littering your Windows installation, nowadays you are more likely to have hundreds, if not thousands of MP3 files scattered across various computers. Along with video clips, animations, and other graphics, the demand for organizing and maintaining these vast libraries is driving development of applications. Popular proprietary applications such as iTunes and Google's Picasa are coveted by Linux users, but open source applications are starting to appear that provide real alternatives, and for some the final reasons they need to move to Linux full time.

This chapter provides an overview of some of the basic multimedia tools included with Ubuntu. You will see how to create your own CDs, watch TV, rip audio CDs into the open source Ogg audio format for playback, as well as manage your media library. You will also learn about how Ubuntu handles graphics and pictures.

Listening to Music

Perhaps the most basic multimedia application you will need is a CD Player. Pretty much everyone knows what a CD is, and the vast majority of people own CDs. Ubuntu can easily handle CD Audio through the default CD Player under Applications, Sound & Video, as shown in Figure 7.1.

FIGURE 7.1 Basic, but functional, is the order of the day for Ubuntu's CD Player.

If all you are after is a basic CD playing application, then you can't really go wrong with CD Player.

The default music player is Rhythmbox, which is designed to play music files in a selection of different formats, such as locally stored Ogg files, Internet Radio Stations, or CDs (as shown in Figure 7.2). It is found in Applications, Sound & Video as Rhythmbox Music Player. You can also use it to subscribe to podcasts available through the Internet.

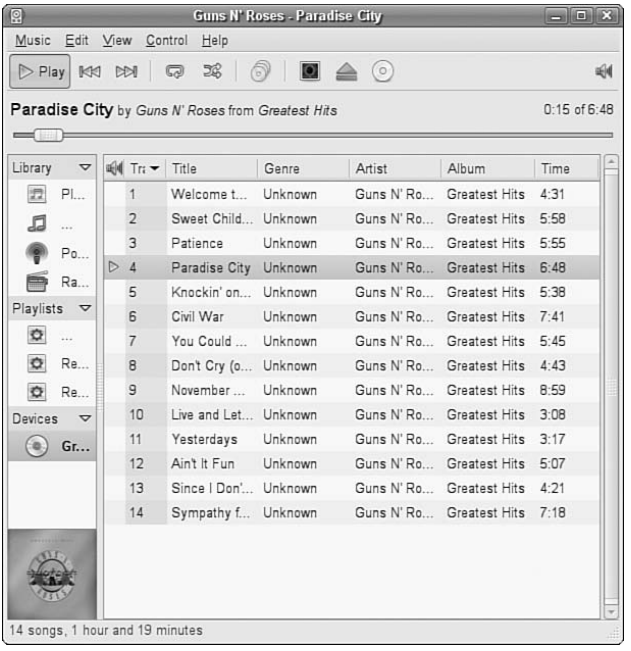


FIGURE 7.2 Rhythmbox can handle podcasts, Internet radio stations, CDs, and local sound files.

Another popular music player is Xmms, a Winamp clone, which in the full version can play not only music, but MPEG1/2/3 video as well. Xmms (see Figure 7.3) supports a number of plug-ins that can add dancing, lighted oscilloscope-like displays, redirect its

output to other devices, support unusual file formats, sync animations to the music, and otherwise increase its geek appeal exponentially. You will have to use `apt-get` to install it with the following command:

```
$ sudo apt-get install xmms
```

after which it will be located under Applications, Sound & Video, Xmms.

FIGURE 7.3 The very popular Xmms music player, seen here playing a local Ogg-Vorbis file.

Other music and sound-related applications can be found in the Sound & Video menu, and, of course, you are free to install your own selection of applications as well.

Getting Music into Ubuntu with Sound Juicer

A handy utility that is included with Ubuntu is Sound Juicer, found under Applications, Sound and Video. Sound Juicer automatically detects when you install a CD and attempt to retrieve the track details from the Internet. From there it will rip the CD tracks into Ogg files for storage on your filesystem. You can see Sound Juicer in action in Figure 7.4.

FIGURE 7.4 Create your own digital music collection with Sound Juicer.

Graphics Manipulation

Over a very short space of time, digital cameras and digital imagery have become extremely popular, to the point where some traditional film camera manufacturers are switching solely to digital. This meteoric rise has led to an increase in the number of applications that can handle digital imagery. Linux, thanks to its rapid pace of development, is now highly regarded as a multimedia platform of choice for editing digital images.

This section of the chapter discusses The GIMP, a powerful graphics manipulation tool. You also learn about graphic file formats supported by Ubuntu, as well as some tools you can use to convert them if the application you want to use requires a different format.

The GNU Image Manipulation Program

One of the best graphics clients available is The GIMP. The GIMP is a free, GPLed image editor with sophisticated capabilities that can import and export more than 30 different graphics formats, including files created with Adobe Photoshop. It is often compared with Photoshop, and The GIMP represents one of the GNU Projects' first significant successes. Many images in Linux were prepared with The GIMP.

The GIMP can be found under the Applications, Graphics menu as simply The GIMP.

You see an installation dialog box when The GIMP is started for the first time, and then a series of dialog boxes that display information regarding the creation and contents of a local GIMP directory. This directory can contain personal settings, preferences, external application resource files, temporary files, and symbolic links to external software tools used by the editor.

What Does Photoshop Have That Isn't in The GIMP?

Although The GIMP is powerful, it does lack two features Adobe Photoshop offers that are important to some graphics professionals.

The first of these is the capability to generate color separations for commercial press printers (CMYK for the colors cyan, magenta, yellow, and key [or black]). The GIMP uses RGB (red, green, and blue), which is great for video display, but not so great for printing presses. The second feature The GIMP lacks is the use of Pantone colors (a patented color specification) to ensure accurate color matching.

If these features are unimportant to you, The GIMP is an excellent tool. If you must use Adobe Photoshop, the current version of CodeWeavers' CrossOver Office will run Photoshop in Linux.

These deficiencies might not last long. A CMYK plug-in is in the works, and the Pantone issues are likely to be addressed in the near future as well.

After the initial configuration has finished, The GIMP's main windows and toolboxes appear. The GIMP's main window contains tools used for selecting, drawing, moving, view enlarging or reducing, airbrushing, painting, smudging, copying, filling, and

selecting color. Depending on the version installed on your system, the toolbox can host more than 25 different tools.

The toolbox's File, Xtns, and Help menus are used for file operations (including sending the current image by electronic mail), image acquisition or manipulation, and documentation, respectively. If you right-click an open image window, you see the wealth of The GIMP's menus, as shown in Figure 7.5.

FIGURE 7.5 Right-click on an image window to access The GIMP's cascading menus.

7

Using Scanners in Ubuntu

With the rise of digital photography, there has been an equal decline in the need for image scanners. However, there are still times that you want to use a scanner, and Ubuntu makes it easy.

You can also use many types of image scanners with The GIMP. In the past, the most capable scanners required a SCSI port. Today, however, most scanners work through a USB port. You must have scanner support enabled for Linux (usually through a loaded kernel module, `scanner.o`) before using a scanner with The GIMP.

Although some scanners can work via the command line, you will enjoy more productive scanning sessions if you use a graphical interface because GUI features, such as previewing and cropping, can save time before actually scanning an image. Most scanners in use with Linux use the Scanner Access Now Easy (*SANE*) package, which supports and enables graphical scanning sessions.

SANE consists of two software components. A low-level driver enables the hardware support and is specific to each scanner. Next, a graphical scanner interface X client

known as `xsane` is used as a plug-in or ancillary program (or script) that adds features to The GIMP.

NOTE

Although `xsane` is commonly used as a GIMP plug-in, it can also be used as a standalone program. Another useful program is Joerg Schulenburg's `gocr` client, used for optical character recognition (*OCR*). Although not a standalone application, it is included in the Kooka scanning application. This program works best with 300 dots per inch (*dpi*) scans in several different graphics formats. OCR is a resource-intensive task and can require hundreds of megabytes of disk storage!

A list of currently supported scanners can be found at http://www.sane-project.org/sane-supported-devices.html. Unfortunately, if your scanner doesn't appear on the list, you should not expect it to work with the SANE software. There is also a list on that same page for drivers not yet included, but you must be able to compile the application from source to use them.

Supported USB scanners are automatically detected and the appropriate driver is loaded automatically. The USB devices tell the USB system several pieces of information when they are connected—the most important of which are the vendor ID and the device ID. This identification is used to look up the device in a table and load the appropriate driver. You will find that Ubuntu successfully identifies and configures most modern USB-based scanners.

Many scanners are supported in Linux. If yours is not, it still might be possible to use it. The Kooka and Xsane scanner applications are included with Ubuntu and are fairly straightforward to use. They can both be found in the Graphics menu as the Scanner Tool.

Working with Graphics Formats

Image file formats are developed to serve a specific technical purpose (lossless compression, for example, where the file size is reduced without sacrificing image quality) or to meet a need for a proprietary format for competitive reasons. Many file formats are covered by one or more patents. For example, the GIF format had fallen into disfavor with the open-source crowd because the patent holder waited a while before deciding to enforce his patent rights rather than being upfront with requests for patent royalties.

If you want to view or manipulate an image, you need to identify the file format to choose the proper tool for working with the image. The file's extension is your first indicator of the file's format. The graphics image formats supported by the applications included with Ubuntu include

- `.bmp`—Bitmapped graphics, commonly used in Microsoft Windows
- `.gif`—CompuServe Graphics Interchange Format
- `.jpg`—Joint Photographic Experts Group

- `.pcx`—IBM Paintbrush
- `.png`—Portable Network Graphics
- `.svg`—Scalable Vector Graphics
- `.tif`—Tagged Image File format

An extensive list of image file extensions can be found in the `man` page for ImageMagick, an excellent application included with Ubuntu, which you learn more about in upcoming sections of this chapter.

TIP

Ubuntu includes dozens of graphics conversion programs that are accessible through the command line, and there are few, if any, graphics file formats that cannot be manipulated when using Linux. These programs can be called in Perl scripts, shell scripts, or command-line pipes to support many types of complex format conversion and image manipulation tasks. See the man pages for the `ppm`, `pbm`, `pnm`, and `pgm` families of commands. Also see the man page for the `convert` command, which is part of a suite of extremely capable programs included with the ImageMagick suite.

Often, a file you want to manipulate in some way is in a format that cannot be used by either your graphics application or the final application. The solution is to convert the image file—sometimes through several formats. The `convert` utility from ImageMagick is useful, as is the `netpbm` family of utilities. If it is not already installed, ImageMagick can be installed with the Add Remove Software GUI found in the System Settings menu; the `netpbm` tools are always installed by default.

The `convert` utility converts between image formats recognized by ImageMagick. Color depth and size also can be manipulated during the conversion process. You can use ImageMagick to append images, surround them with borders, add labels, rotate and shade them, and perform other manipulations well suited to scripting. Commands associated with ImageMagick include `display`, `animate`, `identify`, and `import`. The application supports more than 130 different image formats (all listed in the man page for ImageMagick).

The `netpbm` tools are installed by default because they compose the underpinnings of graphics format manipulation. The man page for each image format lists related conversion utilities; the number of those utilities gives you some indication of the way that format is used and shows how one is built on another:

- The man page for `ppm`, the portable pixmap file format, lists 47 conversion utilities related to `ppm`. This makes sense because `ppm`, or *portable pixmap*, is considered the lowest common denominator for color image files. It is therefore often used as an intermediate format.

- The man page for `pgm`, the portable graymap file format, lists 22 conversion utilities. This makes sense because `pgm` is the lowest common denominator for grayscale image files.
- The man page for `pnm`, the portable anymap file format, lists 31 conversion utilities related to it. However, there is no format associated with PNM because it operates in concert with `ppm`, `pgm`, and `pbm`.
- An examination of the man page for `pbm`, the portable bitmap file format, reveals no conversion utilities. It's a monochrome format and serves as the foundation of the other related formats.

Capturing Screen Images

You can use graphics manipulation tools to capture images that are displayed on your computer screen. Although this technique was used for the production of this book, it has broader uses; there is truth to the cliché that a picture is worth a thousand words. Sometimes it is easier to show an example than it is to describe it.

A captured screen image (also called a *screen grab* or a *screenshot*) can be used to illustrate an error in the display of an application (a font problem, for example) or an error dialog that is too complex to copy down by hand. You might just want to share an image of your beautifully crafted custom desktop configuration with your friends or illustrate your written documents.

When using the GNOME desktop, you can take advantage of the built-in screenshot mechanism (`gnome-panel-screenshot`). Access this tool by pressing the Print Screen key. (Alt+Print Screen takes a screenshot of only the window that has focus on a desktop.) Captured images are saved in `.png` format.

Using Digital Cameras with Ubuntu

Most digital cameras used in connection with Ubuntu fall into one of two categories: webcams (small, low-resolution cameras connected to the computer's interface) or hand-held digital cameras that record image data on disks or memory cards for downloading and viewing on a PC. Ubuntu supports both types. Other types of cameras, such as surveillance cameras that connect directly to a network via wired or wireless connections, need no special support (other than a network connection and viewing software) to be used with a Linux computer.

Ubuntu supports hundreds of different digital cameras, from early parallel-port (CPiA chipset-based) cameras to today's USB-based cameras. You can even use Intel's QX3 USB microscope with Ubuntu. If you prefer a standalone network-based webcam, explore the capabilities of Linux-based cameras from Axis (at http://www.axis.com/products/video/camera/productguide.htm). The following sections describe some of the more commonly used still camera hardware and software supported by Ubuntu.

Handheld Digital Cameras

Digital cameras are one of the major success stories of the last few years. Now you can take pictures and see previews of your pictures immediately. The pictures themselves are stored on discs or memory cards that can be easily plugged into Ubuntu for further manipulation, using The GIMP or other software. Unfortunately, most of the supplied software that comes with the cameras tend to be for Windows users only, making you reliant on the packages supplied with Ubuntu.

The good news, though, is that because of the good development carried out in Ubuntu and GNOME, you are now able to plug pretty much any camera into your computer through a USB interface and Ubuntu automatically recognizes the camera as a USB mass storage device. You can even set Ubuntu to recognize when a camera is plugged in so that it automatically imports your photographs for you.

To do this, you need to set up your settings for removable drives and media. You can find this in the System, Preferences menu. Click the Cameras tab and select the option to import digital photographs when connected (see Figure 7.6).

FIGURE 7.6 Use GNOME's intelligent handling of removable media by setting it to import your photographs automatically.

Now whenever you connect a digital camera to your computer GNOME automatically detects it (see Figure 7.7), and asks whether you want to import the photographs.

By default, GNOME uses the excellent gThumb package (see Figure 7.8), which, although basic-looking, offers an easy-to-use interface and powerful cataloging capabilities.

FIGURE 7.7 GNOME detects the presence of a digital camera and asks whether the photos should be imported.

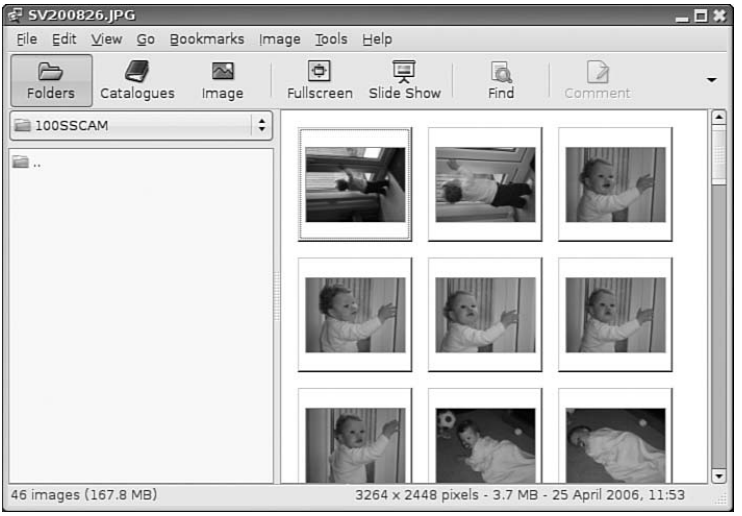


FIGURE 7.8 Make use of gThumb to manage your extensive photo collection.

Using F-Spot

Ubuntu has access to the superb F-Spot photo management application.

When F-Spot is installed, you can find it under the Applications, Graphics menu listed as F-Spot Photo Manager. If you have used the popular Google Picasa application, you will feel instantly at home with F-Spot because it is similar in many ways.

The first time you open F-Spot, you are asked to import your first batch of photographs, as shown in Figure 7.9. You can also assign a tag to them, if you want to track particular types of photographs. You might want to allow F-Spot to copy the photograph to a new directory, `Photos`—something that may help you organize your photos on the system.

When you are ready, click Import to let F-Spot import the photos into the library. The pictures appear in the F-Spot library, and are stored according to the date they were taken. This information is given to F-Spot by the EXIF information that your camera stores each time you take a picture. In Figure 7.10, you can see the standard F-Spot window.

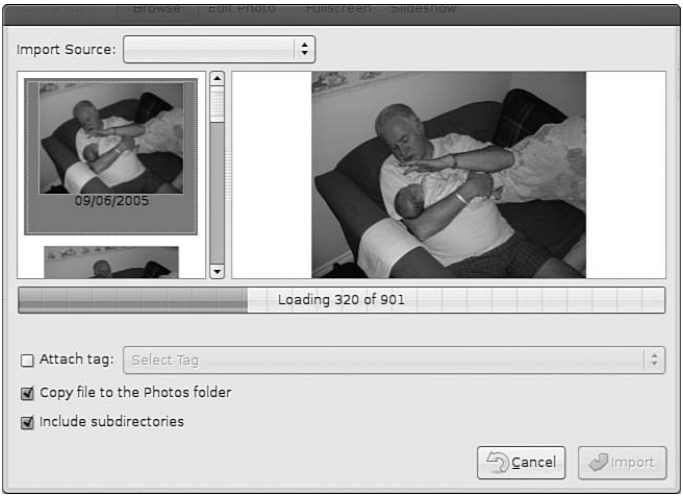


FIGURE 7.9 The versatile F-Spot makes adding photos to your library easy.

FIGURE 7.10 Browse through your extensive photo collection and correct minor problems using F-Spot.

Use the timeline across the top of the window to browse through your photographs, and you can do some minor editing by double-clicking on any photograph. F-Spot is still in its infancy, but development is ongoing, so keep an eye open for any major updates.

Burning CDs and DVDs in Ubuntu

Linux is distributed across the Internet through the use of ISOs that are waiting to be written to CDs or DVDs. Therefore, learning how to burn discs is essential if you have to download and install a Linux distribution. Not only that, but you are likely to want to use CDs and, more commonly, DVDs to back up your music, family pictures, or other important files. With DVD writers being so cheap, the format is now pervasive, and more and more people use cheap DVDs as way of archiving simply due to the larger storage size available. Of course, you can use blank CD media, but they don't have anywhere near the capacity offered by DVDs albeit being slightly cheaper. Today's high-resolution digital cameras can occupy upward of 3MB per shot, and music files can be anything from 1MB to 10MB+ in size. These file sizes make DVD the obvious choice, but there are still occasions when you need to write to a CD. You can use CDs and DVDs to

- Record and store multimedia data, such as backup files, graphics images, and music.
- Rip audio tracks from music CDs (*ripping* refers to extracting music tracks from a music CD) and compile your own music CDs for your personal use.

Linux audio clients and programs support the creation and use of many different types of audio formats. Later sections of this chapter discuss sound formats, sound cards, music players, and much more. Because CD burning is used for many other purposes in Ubuntu, we cover the essentials of that process first in this chapter. To record multimedia data on a CD, you must have installed a drive with CD writing capabilities on your system. To make certain that your CD writer is working, use `wodim -scanbus` to get the information for using the CD drive under SCSI (small computer system interface) emulation:

```
# wodim -scanbus
Cdrecord-Clone 2.01a32-dvd (i686-pc-linux-gnu) Copyright (C) 1995-2001 Jörg
Schilling
Linux sg driver version: 3.5.27
Using libscg version 'schily-0.8'
scsibus0:
        0,0,0     0) 'HL-DT-ST' 'RW/DVD GCC-4120B' '2.01' Removable CD-ROM
        0,1,0     1) *
        0,2,0     2) *
        0,3,0     3) *
        0,4,0     4) *
        0,5,0     5) *
        0,6,0     6) *
        0,7,0     7) *
```

Here, you can see that the CD writer (in this example, a CD writer/DVD reader) is present and is known by the system as device `0,0,0`. The numbers represent the `scsibus/target/lun` (logical unit number) of the device. You need to know this device number when you burn the CD, so write it down or remember it.

Creating CDs and DVDs with Ubuntu's Graphical Clients

Although adequate for quick burns and use in shell scripting, the command-line technique for burning CDs and DVDs is an awkward choice for many people until they become proficient at it and learn all the arcane commands. Fortunately, Ubuntu provides several graphical clients.

Nautilus

With Ubuntu, enhanced functionality has been included in the default file browser Nautilus. Under the Places menu item is a CD/DVD Creator selection. To use it, insert a blank CD or DVD into your CD-R/DVD-R drive. You must have two Nautilus windows open: one that shows the files you want to save to the CD, and a second one open to the CD/DVD Creator Folder (accessed in Nautilus by the Places menu, CD/DVD Creator) location. Click on the Write to Disc button as shown in Figure 7.11 to bring up the Write dialog; at the next dialog box, choose the format to which you want to write the disc. Nautilus CD/DVD Creator supports writing to a disc image file, commonly known as *ISO*. You can also give your new disc a label and tell Nautilus at what speed you want to write the disc. Finally, click the Write button to start the burning process—it is that simple!

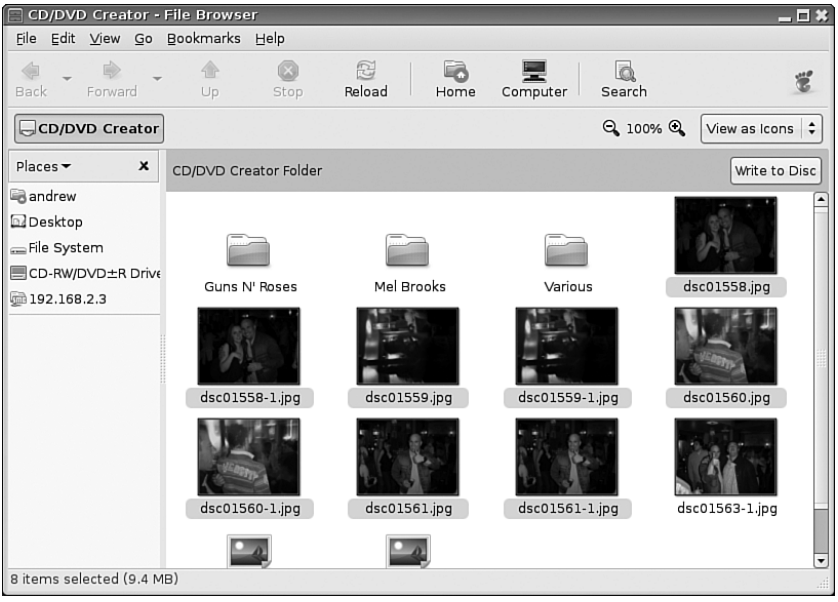


FIGURE 7.11 Creating a CD or DVD using the Nautilus browser is made easy with the drag-and-drop features it provides.

GnomeBaker

If you require a bit more flexibility than just dragging and dropping files into a CD folder, then you should consider GnomeBaker, which enables you to burn Data CDs and DVDs, and also master Audio CDs. It's not installed by default, so make sure you use the command

```
$ sudo apt-get install gnomebaker
```

to retrieve and install the application.

GnomeBaker itself is very easy to use, and can be found under the Applications, Sound and Video menu. When you start GnomeBaker, you are immediately prompted to choose what it is you want to create, as shown in Figure 7.12.

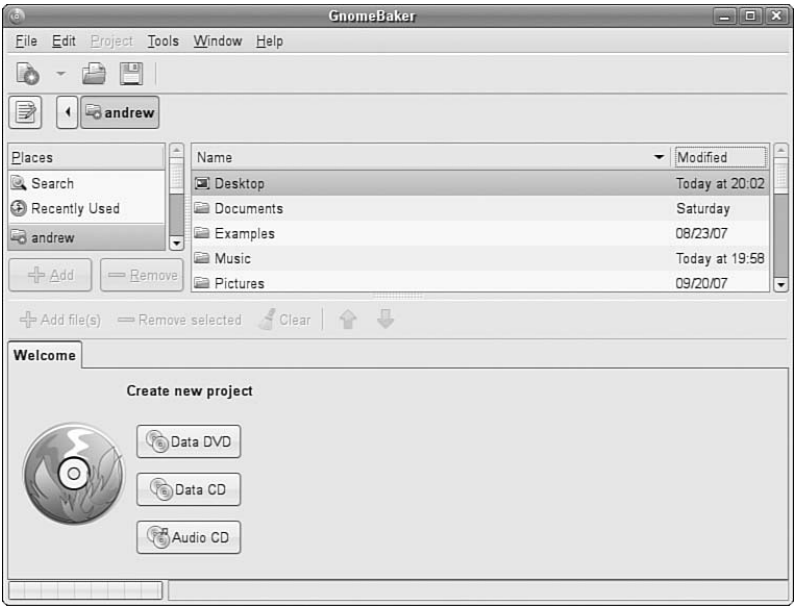


FIGURE 7.12 GnomeBaker offers a task-based approach to burning optical media.

After you've chosen what you want to create, you can then navigate to the files you want to burn in the top half of the GnomeBaker window and drag them to the bottom project area. In Figure 7.13, I am backing up an audio CD that was ripped to my hard drive by Sound Juicer.

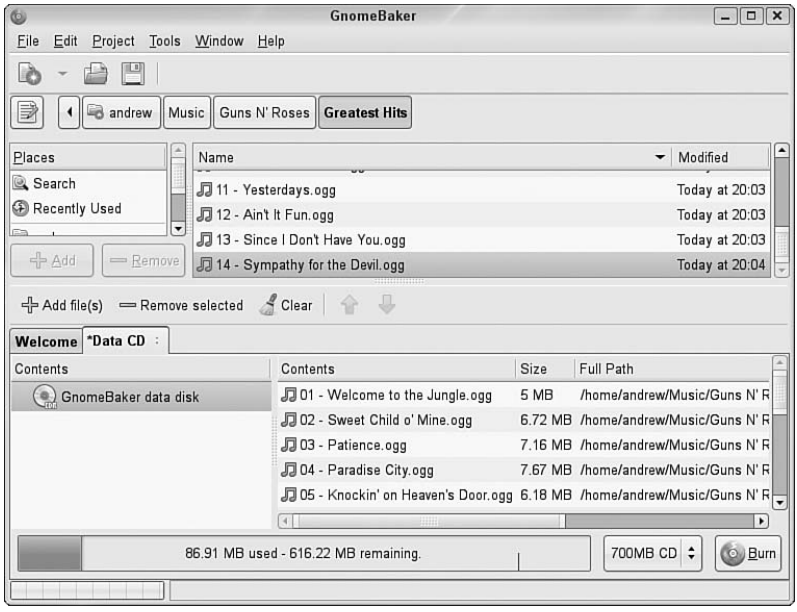


FIGURE 7.13 Use GnomeBaker to back up your CD or photo collection.

TIP

An excellent Internet site for CD-related information is http://www.cdmediaworld.com/.

The Gracenote CDDB Music Recognition Service licenses a database service to software developers so that they can include additional functionality in their applications by accessing the database and having their applications display information about the music CD, including the artist and song title, the CD's track list, and so on. The database server at `cddb.cddb.org`, when contacted by the appropriate software, identifies the appropriate CD and sends the information to be displayed locally. Many CD player applications provide this functionality. The service is interactive: If you have a CD that is not in the CDDB database, the website tells you how you can add the information to the database.

Creating CDs from the Command Line

In Linux, creating a CD at the command line is a two-step process. You first create the `iso9660`-formatted image, and you then burn or write the image onto the CD. The `iso9660` is the default file system for CD-ROMs.

Use the `mkisofs` command to create the ISO image. The `mkisofs` command has many options (see the man page for a full listing), but use the following for quick burns:

```
$ mkisofs -r -v -J -l -o /tmp/our_special_cd.iso /source_directory
```

The options used in this example are as follows:

- `-r`—Sets the permission of the files to more useful values. `UID` and `GID` (individual and group user ID requirements) are set to zero, all files are globally readable and searchable, and all files are set as executable (for Windows systems).
- `-v`—Displays verbose messages (rather than terse messages) so that you can see what is occurring during the process; these messages can help you resolve problems if they occur.
- `-J`—Uses the Joliet extensions to `ISO9660` so that your Windows-using buddies can more easily read the CD. The Joliet (for Windows), Rock Ridge (for Unix), and HSF (for Mac) extensions to the `iso9660` standard are used to accommodate long filenames rather than the eight-character DOS filenames that the `iso9660` standard supports.
- `-l`—Allows 31-character filenames; DOS does not like it, but everyone else does.
- `-o`—Defines the directory where the image will be written (that is, the output) and its name. The `/tmp` directory is convenient for this purpose, but the image could go anywhere you have write permissions.
- `/source_directory`—Indicates the path to the source directory; that is, the directory containing the files you want to include. There are ways to append additional paths and exclude directories (and files) under the specified path—it is all explained in the man page, if you need that level of complexity. The simple solution is to construct a new directory tree and populate it with the files you want to copy, and then make the image using that directory as the source.

Many more options are available, including options to make the CD bootable.

After you have created the ISO image, you can write it to the CD with the `cdrecord` command:

```
$ cdrecord -eject -v speed=12 dev=0,0,0 /tmp/our_special_cd.iso
```

The options used in this example are as follows:

- `-eject`—Ejects the CD when the write operation is finished.
- `-v`—Displays verbose messages.
- `speed=`—Sets the speed; the rate depends on the individual drive's capabilities. If the drive or the recordable medium is poor, you can use lower speeds to get a good burn.
- `dev=`—Specifies the device number of the CD writer (the number I told you to write down earlier).

NOTE

You can also use the `blank=` option with the `cdrecord` command to erase CD-RW disks. The `cdrecord` command has fewer options than `mkisofs` does, but it offers the `-multi` option, which enables you to make multisession CDs. A multisession CD enables you to write a data track, quit, and then add more data to the CD later. A single-session CD can be written to only once; any leftover CD capacity is wasted. Read about other options in the `cdrecord` man page.

Current capacity for CD media is 700MB of data or 80 minutes of music. (There are 800MB/90 minute CDs, but they are rare.) Some CDs can be overburned; that is, recorded to a capacity in excess of the standard. The `cdrecord` command is capable of overburning if your CD-RW drive supports it. You can learn more about overburning CDs at http://www.cdmediaworld.com/hardware/cdrom/cd_oversize.shtml/.

Creating DVDs from the Command Line

There are several competing formats for DVD, and with prices rapidly falling, it is more likely that DVD-writing drives will become commonplace. The formats are as follows:

- DVD+R
- DVD-R
- DVD+RW
- DVD-RW

Differences in the + and – formats have mostly to do with how the data is modulated onto the DVD itself, with the + format having an edge in buffer underrun recovery. How this is achieved impacts the playability of the newly created DVD on any DVD player. The DVD+ format also has some advantages in recording on scratched or dirty media. Most drives support the DVD+ format. As with any relatively new technology, your mileage may vary.

We focus on the DVD+RW drives because most drives support that standard. The software supplied with Ubuntu has support for writing to DVD-R/W (rewritable) media as well. It will be useful for you to review the DVD+RW/+R/-R[W] for Linux HOWTO at http://fy.chalmers.se/~appro/linux/DVD+RW/ before you attempt to use `dvd+rw-tools`, which you need to install to enable DVD creation (also known as *mastering*) as well as the `cdrtools` package. You can ignore the discussion in the HOWTO about kernel patches and compiling the tools.

TIP

The 4.7GB size of DVD media is measured as 1000 megabytes per gigabyte, instead of the more commonly used 1024 megabytes per gigabyte, so do not be surprised when the actual formatted capacity, about 4.4GB, is less than you anticipated. `dvd+rw-tools` does not allow you to exceed the capacity of the disk.

You need to have the `dvd+rw-tools` package installed (as well as the `cdrtools` package). The `dvd+rw-tools` package contains the `growisofs` application (that acts as a front end to `mkisofs`) as well as the DVD formatting utility.

You can use DVD media to record data in two ways. The first way is much the same as that used to record CDs in a session, and the second way is to record the data as a true file system, using packet writing.

Session Writing

To record data in a session, you use a two-phase process:

1. Format the disk with `dvd+rw-format /dev/scd0` (only necessary the first time you use a disk).
2. Write your data to the disk with `growisofs -Z /dev/scd0 -R -J /your_files`.

The `growisofs` command simply streams the data to the disk. For subsequent sessions, use the `-M` argument instead of `-Z`. The `-Z` argument is used only for the initial session recording; if you use the `-Z` argument on an already used disk, it erases the existing files.

CAUTION

Some DVDs come preformatted; formatting them again when you use them for the first time can make the DVD useless. Always be sure to carefully read the packaging your DVD comes in to ensure that you are not about to create another coaster!

TIP

Writing a first session of at least 1GB helps maintain compatibility of your recorded data with other optical drives. DVD players calibrate themselves by attempting to read from specific locations on the disk; you need data there for the drive to read it and calibrate itself.

Also, because of limitations to the ISO9660 file system in Linux, do not start new sessions of a multisession DVD that would create a directory past the 4GB boundary. If you do so, it causes the offsets used to point to the files to "wrap around" and point to the wrong files.

Packet Writing

Packet writing treats the CD or DVD disk like a hard drive in which you create a file system (like ext3) and format the disk, and then write to it randomly as you would to a conventional hard drive. This method, although commonly available on Windows-based computers, is still experimental for Linux and is not yet covered in detail here.

TIP

DVD+RW media are capable of only about 1,000 writes, so it is very useful to mount them with the `noatime` option to eliminate any writing to update their inodes or simply mount them as read-only when it's not necessary to write to them.

It is possible to pipe data to the `growisofs` command:

```
# your_application ¦ growisofs -Z /dev/scd0=/dev/fd/0
```

It is also possible to burn from an existing image (or file, named pipe, or device):

```
# growisofs -Z /dev/scd0=image
```

The `dvd+rw-tools` documentation, found at `/usr/share/doc/dvd+rw-tools-*/index.html`, is required reading before your first use of the program. We also suggest that you experiment with DVD-RW (rewritable) media first because if you make mistakes, you will still be able to reuse the disk rather than create several new coasters for your coffee mug.

Sound and Music

Linux historically had a reputation of lacking good support for sound and multimedia applications in general. However, great strides have been made in recent years to correct this, and support is now a lot better than it used to be. (It might make you smile to know that Microsoft no longer supports the Microsoft Sound Card, but Linux users still enjoy support for it, no doubt just to annoy the folks in Redmond.) Unix, however, has always had good multimedia support as David Taylor, Unix author and guru, points out:

> "The original graphics work for computers was done by Evans & Sutherland on Unix systems. The innovations at MIT's Media Lab were done on Unix workstations. In 1985, we at HP Labs were creating sophisticated multimedia immersive work environments on Unix workstations, so maybe Unix is more multimedia than suggested. Limitations in Linux support doesn't mean Unix had the same limitations. I think it was more a matter of logistics, with hundreds of sound cards and thousands of different possible PC configurations."

That last sentence sums it up quite well. Unix had a limited range of hardware to support; Linux has hundreds of sound cards. Sound card device driver support has been long lacking from manufacturers, and there is still no single standard for the sound subsystem in Linux.

In this section, you learn about sound cards, sound file formats, and the sound applications provided with Ubuntu.

Sound Cards

Ubuntu supports a wide variety of sound hardware and software. Two models of sound card drivers compete for prominence in today's market:

- ALSA, the Advanced Linux Sound Architecture, which is entirely open source
- OSS, the Open Sound System, which offers free and commercial drivers

Ubuntu uses ALSA because ALSA is the sound architecture for the 2.6 series kernels.

ALSA supports a long list of sound cards. You can review the list at http://www.alsa-project.org/alsa-doc/. If your sound card is not supported, it might be supported in the commercial version of OSS. You can download a trial version of commercial software and test your sound card at http://www.opensound.com/download.cgi.

Ubuntu detects most sound cards during the original installation. If you add or replace a sound card after the initial install, the Kudzu New Hardware Configuration utility automatically detects and configures it at the next reboot. To configure the sound card at any other time, use the `system-config-soundcard` graphical tool. The graphical tool can be found under the System, Administration menu as the Soundcard Detection menu item.

Adjusting Volume

Ubuntu offers a handy utility that you can use to control the volumes for various outputs from your computer. For a simple master volume control, just click on the speaker icon in the top-right corner of the screen and move the slider up or down, as seen in Figure 7.14.

FIGURE 7.14 Control the master volume level with the volume slider.

Alternatively you can control all the output volumes for the system to make sure that you have set everything to your taste, as shown in Figure 7.15. To access the volume control, right-click on the speaker icon and select Open Volume Control.

Sound Formats

A number of formats exist for storing sound recordings. Some of these formats are associated with specific technologies, and others are used strictly for proprietary reasons. Ubuntu supports several of the most popular sound formats, including

- **raw (`.raw`)**—More properly known as *headerless format*, audio files using this format contain an amorphous variety of specific settings and encodings. All other sound files contain a short section of code at the beginning—a header—that identifies the format type.

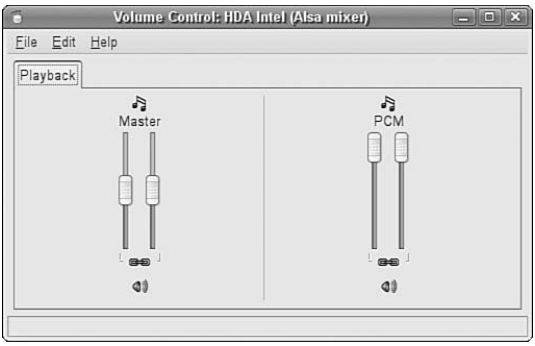


FIGURE 7.15 Use the volume control to manage volume settings for all your sound output devices.

- **MP3 (`.mp3`)**—A popular, but commercially licensed, format for the digital encoding used by many Linux and Windows applications. MP3 is not supported by any software included with Ubuntu (which advises you to use the open source Ogg-Vorbis format instead).
- **WAV (`.wav`)**—The popular uncompressed Windows audio-visual sound format. It is often used as an intermediate file format when encoding audio.
- **Ogg-Vorbis (`.ogg`)**—Ubuntu's preferred audio encoding format. You enjoy better compression and audio playback, and freedom from lawsuits when you use this open-source encoding format for your audio files.

NOTE

Because of patent and licensing issues, Ubuntu has removed support for the MPEG, MPEG2, and MPEG3 (MP3) file formats in Ubuntu Linux. Although we cannot offer any legal advice, it appears that individuals using MP3 software are okay; it is just that Ubuntu cannot distribute the code because it sells its distribution. It seems—at this point—perfectly all right for you to obtain an MP3-capable version of Xmms (for example), which is a Winamp clone that plays MPEG1/2/3 files. You can get Xmms directly from http://www.xmms.org/ because that group has permission to distribute the MP3 code.

You can also enable the MP3 codec within Ubuntu by using the livna.org `yum` repository. You do this by installing the `gstreamer-plugins-mp3` package, which enables the MP3 codec in all the GNOME applications.

Another alternative is to use the Ogg-Vorbis format; it is completely free of restrictions. A ripper for CD music is available from http://www.thekompany.com/projects/tkcog-gripper/ and an MP3-to-Ogg converter is available from http://faceprint.com/code/. Or, you could download and install the non-crippled versions of multimedia applications from FreshRPMs at http://www.freshrpms.net/.

Ubuntu includes software (such as the `sox` command used to convert between sound formats) so that you can more easily listen to audio files provided in a wide variety of formats, such as AU (from NeXT and Sun), AIFF (from Apple and SGI), IFF (originally from Commodore's Amiga), RA (from Real Audio), and VOC (from Creative Labs).

TIP

To learn more about the technical details of audio formats, read Chris Bagwell's Audio Format FAQ at http://www.cnpbagwell.com/audio.html.

Ubuntu also offers utilities for converting sound files from one format to another. Conversion utilities come in handy when you want to use a sound in a format not accepted by your current application of choice. A repository of conversion utilities resides at http://ibiblio.org/pub/linux/apps/sound/convert/!INDEX.html and includes MP3 and music CD–oriented utilities not found in Ubuntu. You have to know how to compile and install from source, however. If you see something useful, have a look at http://www.rpmfind.net/ to locate a binary RPM if you don't feel up to the task.

Ubuntu does provide `sox`, a self-described sound translator that converts music among the AIFF, AU, VAR, DAT, Ogg, WAV, and other formats. It also can be used to change many other parameters of the sound files.

Timidity is a MIDI-to-WAV converter and player. If you are interested in MIDI and musical instruments, Timidity is a handy application; it handles karaoke files as well, displaying the words to accompany your efforts at singing.

Viewing Video

You can use Ubuntu tools and applications to view movies and other video presentations on your PC. This section presents some TV and motion picture video software tools included with the Ubuntu distribution you received with this book.

TV and Video Hardware

To watch TV and video content on your PC, you must install a supported TV card or have a video/TV combo card installed. A complete list of TV and video cards supported by Ubuntu is at http://www.exploits.org/v4l/.

Freely available Linux support for TV display from video cards that have a TV-out jack is poor. That support must come from the X driver, not from a video device that Video4Linux supports with a device driver. Some of the combo TV-tuner/video display cards have support, including the Matrox Marvel, the Matrox Rainbow Runner G-Series, and the RivaTV cards. Many other combo cards lack support, although an independent developer might have hacked something together to support his own card. Your best course of action is to perform a thorough Internet search with Google.

Many of the TV-only PCI cards are supported. In Linux, however, they are supported by the video chipset they use, and not by the name some manufacturer has slapped on a

generic board (the same board is typically sold by different manufacturers under different names). The most common chipset is the Brooktree `Bt***` series of chips; they are supported by the `bttv` device driver.

If you have a supported card in your computer, it should be detected during installation. If you add it later, the Kudzu hardware detection utility should detect it and configure it. You can always configure it by hand.

To determine what chipset your card has, use the `lspci` command to list the PCI device information, find the TV card listing, and look for the chipset that the card uses. For example, the `lspci` output for my computer shows

```
$ lspci
00:00.0 Host bridge: Advanced Micro Devices [AMD] AMD-760 [IGD4-1P]
➥System Controller (rev 13)
00:01.0 PCI bridge: Advanced Micro Devices [AMD] AMD-760 [IGD4-1P]
➥ AGP Bridge
00:07.0 ISA bridge: VIA Technologies, Inc. VT82C686 [Apollo Super South]
➥ (rev 40)
00:07.1 IDE interface: VIA Technologies, Inc. VT82C586B PIPC Bus Master IDE
➥ (rev 06)
00:07.2 USB Controller: VIA Technologies, Inc. USB (rev 1a)
00:07.3 USB Controller: VIA Technologies, Inc. USB (rev 1a)
00:07.4 SMBus: VIA Technologies, Inc. VT82C686 [Apollo Super ACPI]
➥ (rev 40)
00:09.0 Multimedia audio controller: Ensoniq 5880 AudioPCI (rev 02)
00:0b.0 Multimedia video controller: Brooktree Corporation Bt878 Video Capture
➥ (rev 02)
00:0b.1 Multimedia controller: Brooktree Corporation Bt878 Audio Capture
➥ (rev 02)
00:0d.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL-8029(AS)
00:0f.0 FireWire (IEEE 1394): Texas Instruments TSB12LV23 IEEE-1394
➥ Controller
00:11.0 Network controller: Standard Microsystems Corp [SMC]
➥ SMC2602W EZConnect
01:05.0 VGA compatible controller: nVidia Corporation NV15 [GeForce2 Ti]
➥ (rev a4)
```

Here, the lines listing the multimedia video controller and multimedia controller say that this TV board uses a Brooktree `Bt878` Video Capture chip and a Brooktree `Bt878` Audio Capture chip. This card uses the `Bt878` chipset. Your results will be different, depending on what card and chipset your computer has. This card happened to be an ATI All-in-Wonder VE (also known as ATI TV-Wonder). (The VE means *Value Edition*; hence, there is no TV-out connector and no radio chip on the card; what a value!) The name of the chipset says that the card uses the `bttv` driver.

In the documentation directory is a file named `CARDLIST`, and in that file is the following entry, among others:

```
card=64 - ATI TV-Wonder VE
```

There are 105 cards listed, as well as 41 radio cards, including

```
card=0 -  *** UNKNOWN/GENERIC ***
```

which is what you could have used had you not known the manufacturer's name for the card.

The file named `Modules.conf`, located in the same directory, offers the following example of information to place in the `/etc/modules.conf` file:

```
$ i2c
alias char-major-89    i2c-dev
options i2c-core       i2c_debug=1
options i2c-algo-bit   bit_test=1

$ bttv
alias char-major-81    videodev
alias char-major-81-0  bttv
options bttv           card=2 radio=1
options tuner          debug=1
```

All you need do is enter this information into `/etc/modules.conf` and change the value for `card=2` to `card=64` to match your hardware. You can delete the reference to the radio card (`radio=2`) because there isn't one and leave the other values alone. Then you must execute

```
$ sudo depmod -a
```

to rebuild the modules dependency list so that all the modules are loaded automatically. When finished, all you need do is execute

```
$ sudo modprobe bttv
```

and your TV card should be fully functional. All the correct modules will be automatically loaded every time you reboot. Ubuntu is clever enough to detect and configure a supported TV card that is present during installation.

TIP

Other useful documentation can be found in `/usr/src/linux-2.6/Documentation/_video4linux`. After you have identified a driver for a device, it does not hurt to look at the source code for it because so little formal documentation exists for many drivers; much of it is in the source code comments.

The development of support for TV cards in Linux has coalesced under the Video4Linux project. The Video4Linux software provides support for video capture, radio, and teletext devices in Ubuntu.

Video Formats

Ubuntu recognizes a variety of video formats. The formats created by the MPEG group, Apple, and Microsoft dominate, however. At the heart of video formats are the *codecs*—the encoders and decoders of the video and audio information. These codecs are typically proprietary, but free codecs do exist. Here is a list of the most common video formats and their associated file extensions:

- `.mpeg`—The MPEG video format; also known as `.mpg`
- `.qt`—The QuickTime video format from Apple
- `.mov`—Another QuickTime video format
- `.avi`—The Windows audio visual format

TIP

An RPM that provides a Divx codec for Linux can be found at http://www.freshrpms.net/. Divx is a patented MPEG-4 video codec that is the most widely used codec of its type. It allows for compression of MPEG-2 video by a factor of 8. See http://www.divx.com/ for more information.

The GetCodecs application is a Python script with a GUI interface that downloads, installs, and configures your Ubuntu system with multimedia codecs not provided by Ubuntu, such as MP3, Divx, and DVD codecs. The script can be obtained from http://sourceforge.net/projects/getcodecs/.

If you need to convert video from one format to another, you use encoder applications called *grabbers*. These applications take raw video data from a video device such as a camera or TV card, and convert it to one of the standard MPEG formats or to a still image format, such as JPEG or GIF. Ubuntu does not supply any encoder applications (other than `ppmtompeg`, which encodes MPEG-1 video), but you can find them at http://www.freshrpms.net/ or another online source (see the "Reference" section at the end of this chapter).

Viewing Video in Linux

Out of the box, Ubuntu does not support any of the proprietary video codecs due to licensing restrictions. However this functionality can be restored if you install the full version of the applications described in this section from the Universe repository. There you can find multimedia applications such as Ogle, Xine, AlsaPlayer, Gstreamer, Grip, Mplayer, VCDImager, VideoLAN-client, Xmms, and Zapping.

You can use Linux software to watch TV, save individual images (take snapshots) from a televised broadcast, save a series of snapshots to build animation sequences, or capture video, audio, or both. The following sections describe some of the ways in which you can put Linux multimedia software to work for you.

You can watch MPEG and DVD video with Xine. Xine is a versatile and popular media player that is not included with Ubuntu. Xine is used to watch AVI, QuickTime, Ogg, and MP3 files (the latter is disabled in Ubuntu).

Adobe Flash

The Adobe Flash plug-in for the Firefox browser is a commercial multimedia application that isn't provided with Ubuntu out of the box, but many people find it useful. Adobe Flash enables you to view Flash content at websites that support it. The easiest way of getting hold of the official version of Flash is to install the `flashplugin-nonfree` package either using `apt-get` or `synaptic`. Once you've done this, any flash animations will play quite happily within any Firefox-based browsers.

Another interesting video viewer application is MPlayer (not provided by Ubuntu), a movie player for Linux. MPlayer can use Win32 codecs and it supports a wider range of video formats than Xine, including Divx and some RealMedia files. MPlayer also uses some special display drivers that support Matrox, 3Dfx, and Radeon cards and can make use of some hardware MPEG decoder boards for better MPEG decoding. Look for Ubuntu packages at http://www.mplayerhq.hu; a Win32 codec package is also available, as well as other codec packages and a GUI interface.

Personal Video Recorders

The best reason to attach a television antenna to your computer, however, is to use the video card and the computer as a personal video recorder.

The commercial personal video recorder, TiVo, uses Linux running on a PowerPC processor to record television programming with a variety of customizations. TiVo has a clever interface and wonderful features, including a record/playback buffer, programmed recording and pause, slow motion, and reverse effects. Ubuntu does not provide any of the many applications that attempt to mimic the TiVo functionality on a desktop PC running Linux. However, several such applications, including DVR, The Linux TV Project, and OpenPVR, are listed at http://www.exploits.org/v4l/. These projects are in development and do not provide `.rpm` files, so you have to know how to download from CVS and compile your own binaries. For something a little easier, check out MythTV at http://www.mythtv.org/; a Ubuntu `.rpm` file should be available from ATRpms.

Linux, TiVo, and PVRs

Some TiVo users say that using this Linux-based device has changed their lives. Indeed, the convenience of using a personal video recorder (*PVR*) can make life a lot easier for inveterate channel surfers. Although PVR applications are not included with Ubuntu, open source developers are working on newer and better versions of easy-to-install and easy-to-use PVR software for Linux. For more information about TiVo, which requires a monthly charge and a phone line (or broadband connection with a newer TiVo2), browse to http://www.tivo.com/. Unrepentant Linux hardware hackers aiming to disembowel or upgrade a TiVo can browse to http://www.9thtee.com/tivoupgrades.htm or read the TiVo Hack FAQ at http://www.tivofaq.com/. A PVR makes viewing television a lot more fun!

A number of Linux sites are devoted to PVR software development. Browse to the DVR project page at http://www.pierrox.net/dvr/.

DVD and Video Players

You can now easily play DVDs with Ubuntu as long as you install the appropriate software. (Ubuntu doesn't provide any.) Browse to http://www.videolan.org/, and then download, build, and install the `vlc` client.

You must have a CPU of at least 450MHz and a working sound card to use a DVD player. The default Ubuntu kernel supports the DVD CD-ROM file system. As mentioned earlier, Xine and MPlayer do a great job of playing DVD files.

NOTE

The VideoLAN HOWTO found at http://videolan.org/ discusses the construction of a network for streaming video. Although you might not want to create a network, a great deal of useful information about the software and hardware involved in the enterprise can be generalized for use elsewhere, so it is worth a look. The site also contains a link to a HOWTO about cross-compiling on Linux to produce a Windows binary.

Reference

- http://www.cdcopyworld.com/—A resource for technical information about CD media and CD writers.
- http://hardware.redhat.com/hcl/—A database of supported hardware.
- http://www.opensound.com/download.cgi—The commercial OSS sound driver trial version download.
- http:/www.xmms.org/—Home to the Xmms audio player.
- http://www.thekompany.com/projects/tkcoggripper/—A free (but not GPL) Ogg CD ripper.

- http://faceprint.com/code/—An MP3 to Ogg converter named mp32ogg.
- http://www.ibiblio.org/pub/linux/apps/sound/convert/!INDEX.html—Home to several sound conversion utilities.
- http://linux-sound.org/—An excellent resource for Linux music and sound.
- http://www.cnpbagwell.com/audio.html—The Audio Format FAQ.
- http://www.icecast.org/—A streaming audio server.
- http://www.linuxnetmag.com/en/issue4/m4icecast1.html—An Icecast tutorial.
- http://linuxselfhelp.com/HOWTO/MP3-HOWTO-7.html—The MP3 HOWTO contains brief descriptions of many audio applications and, although it focuses on the MP3 format, the information is easily generalized to other music formats.
- http://www.exploits.org/v4l/—Video for Linux resources.
- http://fame.sourceforge.net/—Video encoding tools.
- http://teletext.mb21.co.uk/faq.shtml—The Teletext FAQ.
- http://xine.sourceforge.net/—Home of the Xine DVD/video player.
- http://www.MPlayerHQ.hu/homepage/—Home to the MPlayer video player.
- http://www.videolan.org/—A VideoLAN project with good documentation.
- http://fy.chalmers.se/~appro/linux/DVD+RW/—The DVD+RW/+R/-R[W] for Linux, a HOWTO for creating DVDs under Linux.
- http://www.gimp.org—Home page of The GIMP (Gnu Image Manipulation Program).
- http://f-spot.org—Home page of the F-Spot project.
- http://www.linuxformat.co.uk—Website of Linux Format, home of a long-running GIMP tutorial by Michael J Hammel.
- http://www.exif.org—More information on EXIF and how it is used in digital cameras.
- http://www.sane-project.org—Home page of the SANE (*Scanner Access Now Easy*) project.
- http://www.imagemagick.org—Home page for ImageMagick.
- http://www.codeweavers.com—Home of the popular crossover office; required if you want to try to run Photoshop under Linux.
- http://gimp.net/tutorials/—Official tutorials for The GIMP.

CHAPTER 8

Printing with Ubuntu

IN THIS CHAPTER

- Overview of Ubuntu Printing
- Configuring and Managing Print Services
- Creating and Configuring Local Printers
- Reference

From the word *go*, Ubuntu provides support for a huge range of printers from many different manufacturers. This chapter looks at how to get your printer connected and talking to Ubuntu, as well as at the software that Ubuntu uses to manage printers and print jobs.

In keeping with most of the other Linux distributions, Ubuntu uses CUPS (*Common Unix Printing System*) to handle printers. Other systems are supported, such as LPRng, but you do not have access to some of the graphical management tools from within Ubuntu.

The Internet Printing Protocol

CUPS supports the Internet Printing Protocol, known as *IPP*, and offers a number of unique features, such as network printer directory (printer browsing) services, support for encryption, and support for PostScript Printer Description (`.ppd`) files.

According to the Internet Engineering Task Force (*IETF*), IPP grew out of a 1996 proposal by Novell to create a printing protocol for use over the Internet. Since then, the system has been developed and has matured into a stable print system for use on a variety of Linux and Unix-like operating platforms.

Overview of Ubuntu Printing

Ubuntu's print filter system is the main engine that enables the printing of many types of documents. The heart of that engine is the GNU GPL version of Aladdin's Ghostscript interpreter, the `gs` client. The system administrator's printer configuration tool is the `system-config-printer` client.

NOTE

Ubuntu's print system can be used to print to local (attached) or remote (network) printers. If you use a local printer, it is represented by a printer device, such as `/dev/lp0` or `/dev/usb/lp0` (if you have a USB printer). Local and remote printers use print *queue*s defined in your system's printer capabilities database, `/etc/printcap`. A document being printed is known as a print *job*, and you can view and control your list, or queue, of current print jobs in the spool directory, which is `/var/spool/cups`. Note that you may control only your print jobs; only the root operator can control print jobs of any user on the system.

To add a printer to your system, you use the `system-config-printer` client to create, configure, and save the printer's definition. The client saves the definition as an entry in your system's printer capabilities database, `/etc/printcap`. Each definition contains a text field with the name of the printer, its host, and name of the print queue. Printed documents are spooled to the `/var/spool/cups` directory. A sample `printcap` definition might look like

```
# This file was automatically generated by cupsd(8) from the
# /etc/cups/printers.conf file.  All changes to this file
# will be lost.
lp¦lp:rm=officejet:rp=lp:¨
```

CUPS maintains its own database of defined printers under the `/etc/cups` directory in a file named `printers.conf`. For example, an associated printer defined in `/etc/printcap` previously might have the following entry in `/etc/cups/printers.conf`:

```
<DefaultPrinter lp>
Info Created by system-config-printer 0.7.x
DeviceURI parallel:/dev/lp0
Location HP P135 local printer
State Idle
Accepting Yes
JobSheets none none
QuotaPeriod 0
PageLimit 0
KLimit 0
</Printer>
```

This example shows the definition for the printer named `lp`, along with its associated device, description, state, and other information. The various possible fields and entries in this file are documented in the `printer.conf` man page.

CUPS uses a print server (daemon) named `cupsd`, also called a *scheduler* in the CUPS documentation. The server can be controlled, like other Ubuntu services, by the `/etc/init.d/cupsys` command or `system-config-services` client. How the server works on a system is determined by settings in its configuration file, `cupsd.conf`, found under the `/etc/cups` directory. CUPS executables are found under the `/usr/lib/cups` directory.

The `cupsd.conf` man page documents more than 80 different settings for the server, which you can configure to match your system, network, or printing environment. Default CUPS-related files and directories are stored under the `/usr/share/cups` directory. Logging can be set to seven different levels, with information about access and errors stored in log files under the `/var/log/cups` directory.

Resource requirements can be tailored through other settings, such as `MaxCopies` to set the maximum number of copies of a print job by a user, `MaxJobs` to set a limit on the number of active print jobs, and `MaxJobsPerUser` to set a limit on the number of active jobs per user. The `RIPCache` setting (8MB by default) controls the amount of memory used for graphics cache during printing.

For example, if you want to limit printing to 20 copies of a document or page at a time and only 10 simultaneous print jobs per user, use settings such as

```
MaxCopies 20
MaxJobsPerUser 10
```

TIP

Do not forget to restart the CUPS server after making any changes to its configuration file. Changes are activated only when the service is restarted (when the daemon rereads its configuration file). See the "GUI-Based Printer Configuration Quickstart" section later in this chapter.

Because CUPS does not use the traditional Berkeley-style print spooling system, `lpd`, you can change the name of the printer capabilities database from the default `/etc/printcap`. Encryption can be used for printing, with secure access behavior determined by settings in `/etc/cups/client.conf`. Network access settings include port, connection, IP address, domains, and limits to the number and size of client requests.

Configuring and Managing Print Services

Your task as a system administrator (or root operator of your workstation) is to properly define local or remote printers and to ensure that printing services are enabled and running properly. Fortunately, Ubuntu includes a graphical print service configuration tool that makes this job easy. You should use these tools to configure printing, as you learn in this section of the chapter. But first, take a moment to read through a quick overview of the configuration process.

CAUTION

Do not manually edit your `/etc/printcap`. Any changes will be lost when the printing service is restarted or if your system is rebooted. If you need to create customized printer entries, save the entries in `/etc/printcap.local` and then restart the printing service.

You can configure printing services using the `system-config-printer-gui` graphical interface. Most of the detailed information in this chapter refers to the use of the GUI. The overview sections that follow, however, give you a solid foundation in both configuration approaches. You learn the details of these processes in later sections of the chapter.

GUI-Based Printer Configuration Quickstart

Configuring a printer for Ubuntu is easy but you must use root permission to do it. Make sure that the `cupsd` daemon is installed and running. If you elect to use printing support when you install Ubuntu, the daemon and related software will be installed. If you're not sure whether `cupsd` is running, you can quickly drop to a terminal and use the `/etc/init.d/cupsys` command with the `status` keyword like so:

```
$ sudo /etc/init.d/cupsys status
```

You will see either

```
Status of Common Unix Printing System: cupsd is not running.
```

or, if `cupsd` is running, an acknowledgement such as

```
Status of Common Unix Printing System: cupsd is running.
```

If `cupsd` is installed but not running, start the daemon like so:

```
$ sudo /etc/rc.d/init.d/cups start
```

To access the GUI select System, Administration, Printing.

You then simply follow the prompts to define your printer and add local or remote printing services. You should print a test page before saving your changes. Use the printer configuration client or the File menu's Print menu item from a GNOME or KDE client.

NOTE

The `system-config-printer` utility is a replacement for gnome-cups-manager, which was previously the default printer configuration tool for Ubuntu. This new utility was actually originally developed for Fedora and has been adopted by Ubuntu.

Managing Printing Services

After defining a printer, you can use the command line to view and control your print jobs, or if root, all print jobs and printers on your system. Table 8.1 contains a partial list of CUPS and related printing commands and drivers included with Ubuntu.

TABLE 8.1 Print-Related Commands and Drivers

Name	Description
a2ps	Formats text files for PostScript printing
accept	Controls CUPS print job destinations
cancel	Cancels a CUPS print job
disable	Controls CUPS printers
dvi[lj, lj4l, lj2p, lj4]	Converts TeX DVI files to specific PCL format
enable	Controls CUPS printers
encscript	Converts text files to PostScript
escputil	Epson Stylus inkjet printer utility
grolbp	groff driver for Canon LBP-4 and LBP-8 laser printers
gs	The Ghostscript interpreter
gsbj[dj500, lp]	Ghostscript BubbleJet printer drivers
gsdj[dj500, lj, lp]	Ghostscript DeskJet printer drivers
lpadmin	CUPS command-line–based printer utility
lp	Starts a CUPS print job
lpc	A Berkeley-subset CUPS printer control client
lpf	General printer filter
lprm	A Berkeley-compatible CUPS job queue utility
lpstat	Displays CUPS print jobs and printer status
mpage	PostScript text formatting utility
pbm[2ppa, page, to10x, toepson, toppa, toptx]	Portable bitmap conversion utilities
pr	Text formatting command
psmandup	Duplex printing utility for nonduplex printers
reject	Controls CUPS print job destinations
setup	Launches printer configuration tool
smbclient	SMB print spooler
smbprint	SMB print shell script
smbspool	SMB printer spooler
thinkjettopbm	Portable bitmap to ThinkJet printer conversion utility

Most Linux systems use PostScript as the default document format for printing. Ubuntu uses the gs command along with CUPS to manage local and remote print jobs and the type of data transferred during a print job. The gs command is used to translate the document stream into a format accepted by the destination printer (which most likely uses HPCL).

You can use the Ghostscript interpreter gs to display its built-in printer devices by using the gs interpreter with its --help command-line option like this:

```
# gs --help
```

NOTE

Ubuntu includes graphical clients you can use to view many different types of documents. For example, to display PostScript documents (including compressed PostScript documents) or PostScript images, use the `gv` client. To display Portable Document Format (PDF) documents, you can use `gv` or the `xpdf` client.

The `gs` command outputs many lines of help text on command-line usage and then lists built-in printer and graphics devices. Another way to get this information is to start `gs` and then use the `devicenames ==` command like this:

```
# gs
GPL Ghostscript SVN PRE-RELEASE 8.61 (2007-08-02)

Copyright (C) 2007 Artifex Software, Inc.  All rights reserved.

This software comes with NO WARRANTY: see the file PUBLIC for details.

GS>devicenames ==

[/epson /pnggray /lp3000c /epl2050 /pgnm /ljet4d\
 /cljet5pr /pbmraw /lips4v /cdj550 /mag16 /laserjet\
 /bj200 /dfaxhigh /ibmpro /alc8500 /bmpgray\
 /hpdj600 /tiffgray /hpdj310 /pswrite\
...
```

Not all the devices are listed in this example.

Aladdin or GNU?

At least two versions of Ghostscript are available for Linux. One version is named AFPL Ghostscript, which formerly went by the name Aladdin Ghostscript. This version is licensed under the Aladdin Free Public License, which disallows commercial distribution. The other version is called GNU Ghostscript, which is distributed under the GNU General Public License. For details about the different versions or for answers to questions regarding licensing, see the Ghostscript home page at http://www.cs.wisc.edu/~ghost/.

Creating and Configuring Local Printers

Creating a local printer for your Ubuntu system can be accomplished in a few easy steps. The cupsd daemon should also be running before you begin (start the daemon manually as shown earlier in this chapter).

To launch `system-config-printer`, go to System, Administration and choose the Printing menu option.

Creating the Print Queue

The Ubuntu `system-config-printer` tool walks you through a process to create a new printer. To begin configuration of a local (attached) printer, click the New Printer toolbar button in `system-config-printer`'s main window. An Add a New Printer configuration dialog appears, as shown in Figure 8.1.

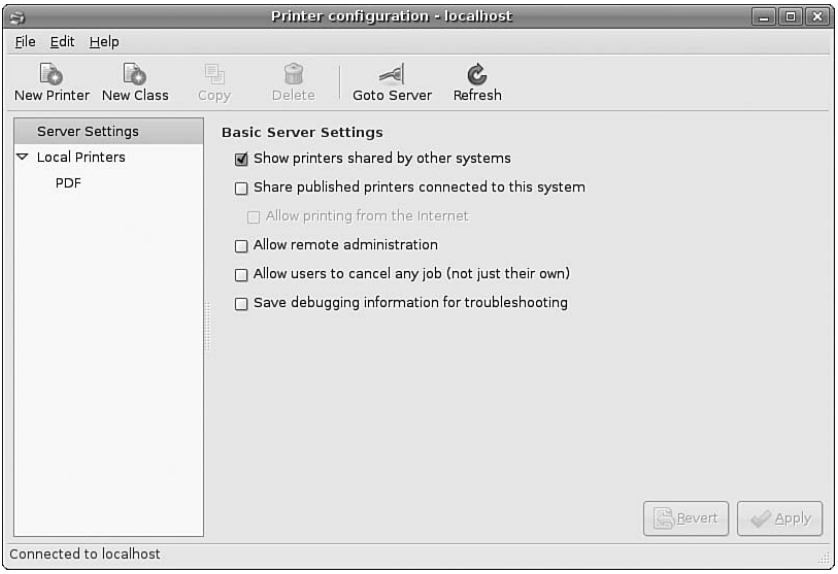


FIGURE 8.1 Click the New Printer toolbar button to start the configuration of a new printer for your system.

The Printer Name dialog appears. Type a desired name for the new printer (such as `lp`), enter a short description and optional location information and then click the Forward button. The Connection Type dialog appears, as shown in Figure 8.2. Select the connection type that is appropriate for you. You can select a number of different connection types, depending on your specific requirements. Normally you will use the LPT#1 option if your printer is connected by a standard Parallel (or what used to be called Centronics) cable. Alternatively, if you are connecting to a printer that has a JetDirect port (most HP network-capable printers fit in this category), select the appropriate option and enter the network address for the printer.

Next up you need to select the make/manufacturer of the printer that you are setting up, shown in Figure 8.3.

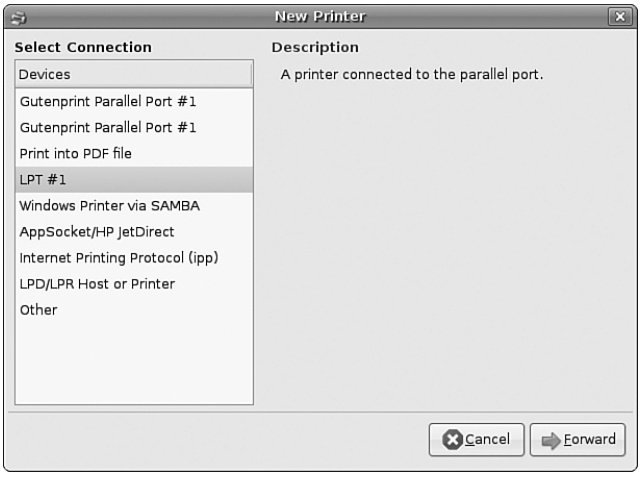


FIGURE 8.2 Select the appropriate connection method for your printer and enter the relevant details.

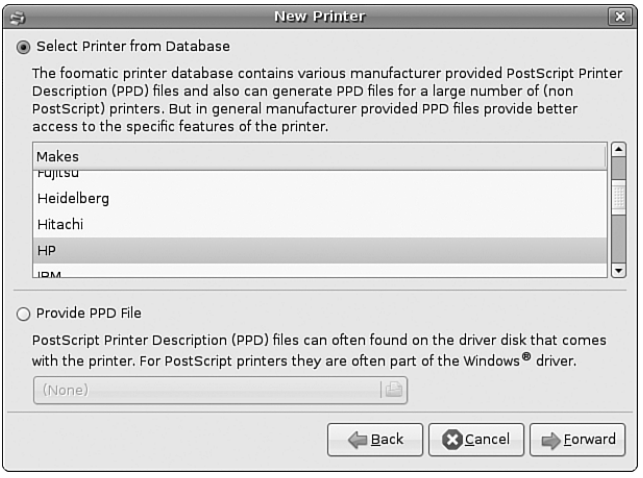


FIGURE 8.3 Select the make or manufacturer of your printer from this dialog box to help Ubuntu narrow down the driver options.

Note that you can configure a printer for Ubuntu even if it is not attached to your computer. After you select your printer's manufacturer, a list of printers from that manufacturer (such as HP, as shown in Figure 8.4) appears. Select your printer from the list, and then click the Forward button.

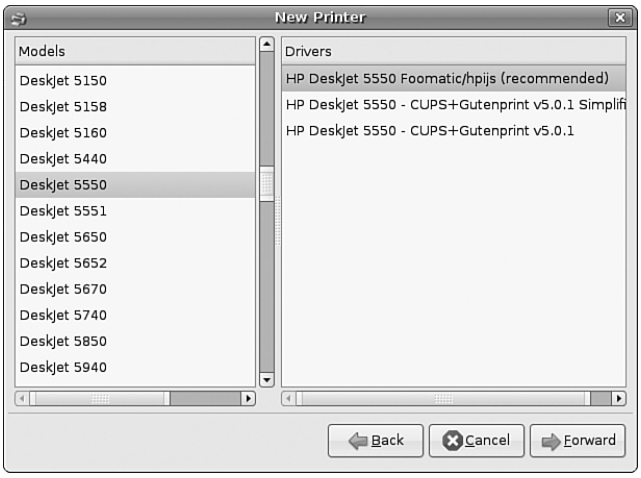


FIGURE 8.4 Select your printer from the list and click the Forward button to finish the configuration of a locally connected printer.

Do not worry if you do not see your printer listed in the selection; it is possible to select a related, although different, printer model and still be able to print to your printer. For example, many HP printers can be used by selecting the DeskJet 500 for monochrome or 500C model for color printing.

NOTE

You can also browse to http://www.linuxprinting.org/ to find out what drivers to use with your printer or to see a cross-referenced listing of printers supported by each driver. You might also find new and improved drivers for the latest printers on the market.

You can experiment to see which printer selection works best for your printer if its model is not listed. You might not be able to use all the features of your printer, but you will be able to set up printing service. Click Next when you have made your choice.

Now you can name the printer and give it a description and location as shown in Figure 8.5. The name is important, as this will be what the users see when they print. The description and location are optional, but Ubuntu autofills your location with the hostname of your computer.

If you are happy with the details, click the Apply button to commit your changes to the system.

You can see the new printer defined in the `system-config-printer` main window, as shown in Figure 8.6.

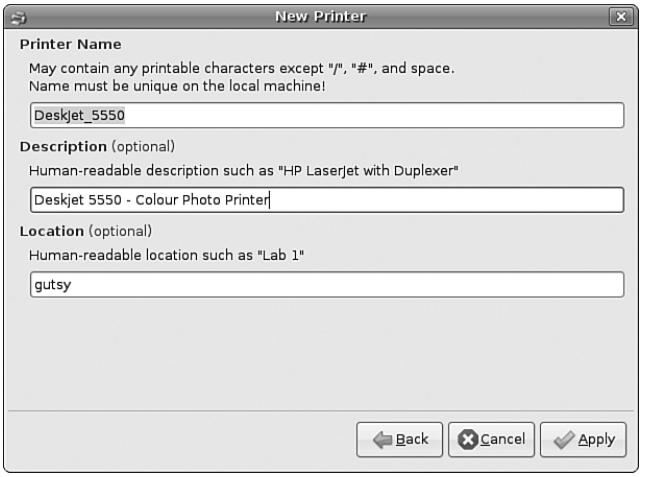


FIGURE 8.5 Give your printer a meaningful name, and if you want, a description and location.

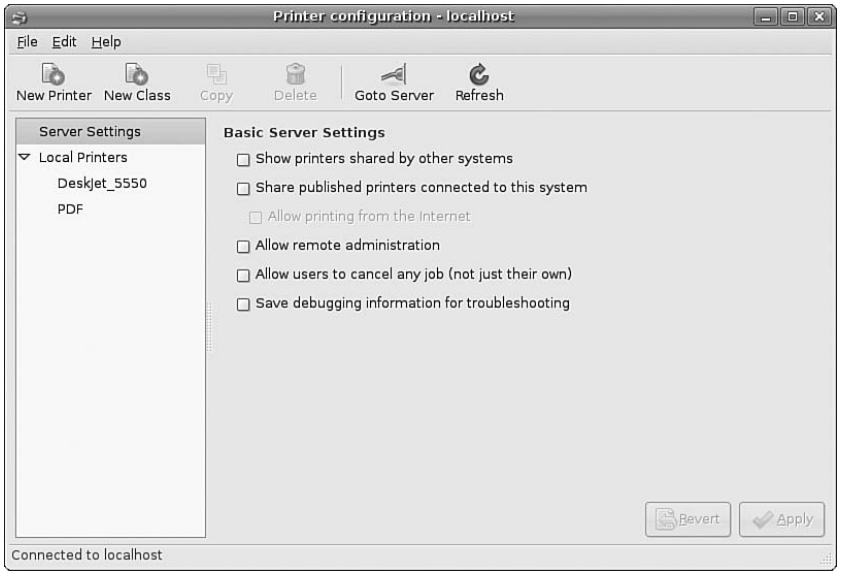


FIGURE 8.6 New printer entries created in */etc/printcap* displayed in *system-config-printer*'s main window.

Editing Printer Settings

You also use the `system-config-printer` tool to edit the newly defined printers. To edit the printer settings, highlight the printer's listing in the printer browser window. You can then select specific settings related to that printer by using the tabs that appear in the right side of the dialog box. The Settings dialog is shown in Figure 8.7.

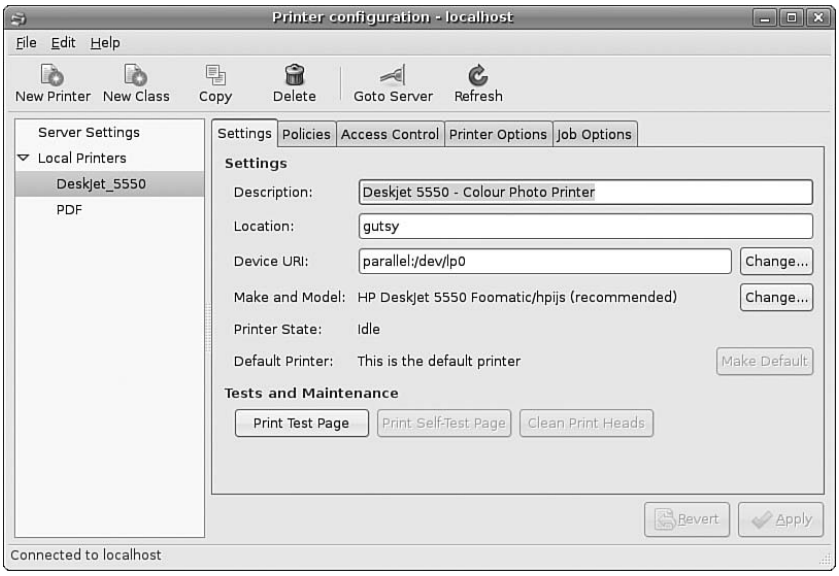


FIGURE 8.7 Edit a printer's settings by using tabs in *system-config-printer*.

The first tab in this dialog enables you to assign a new name for the printer. In this example, the printer has the name laserjet. Other tabs in this dialog enable you to change the queue type or queue options (such as whether to print a banner page or set the image area of a page), to select or update the driver, or to choose available print options for the printer (shown in Figure 8.8).

When you finish editing your printer, click the Apply button to save your changes and automatically restart the `cupsd` daemon. This step is extremely important; you have to update the printer settings and restart the `cupsd` daemon to force it to reread your new settings. Click Quit from the File menu when finished.

Related Ubuntu and Linux Commands

The following commands help you manage printing services:

- `accept`—Controls print job access to the CUPS server via the command line
- `cancel`—Cancels a `print` job from the command line
- `cancel`—Command-`line` control of print queues
- `disable`—Controls `printing` from the command line
- `enable`—Command-`line` control CUPS printers
- `lp`—Command-line `control` of printers and print service
- `lpc`—Displays status of printers and print service at the console
- `lpq`—Views print `queues` (pending print jobs) at the console
- `lprm`—Removes `print` jobs from the print queue via the command line
- `lpstat`—Displays `printer` and server status
- `system-config-printer`—Ubuntu's graphical printer configuration tool

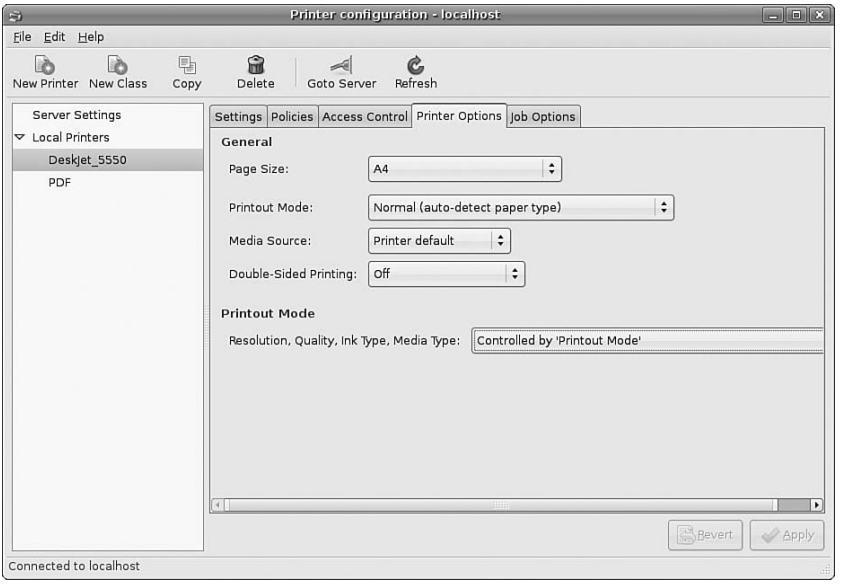


FIGURE 8.8 A printer's general options can be changed on the Settings tab of *system-config-printer*.

Reference

- http://www.linuxprinting.org/—Browse here for specific drivers and information about USB and other types of printers.
- http://www.hp.com/wwsolutions/linux/products/printing_imaging/index.html—Short but definitive information from HP regarding printing product support under Linux.
- http://www.cups.org/—A comprehensive repository of CUPS software, including versions for Ubuntu.
- http://www.pwg.org/ipp/—Home page for the Internet Printing Protocol standards.
- http://www.linuxprinting.org/cups-doc.html—Information about the Common UNIX Printing System (CUPS).
- http://www.cs.wisc.edu/~ghost/—Home page for the Ghostscript interpreter.

CHAPTER 9

Games

IN THIS CHAPTER

- Linux Gaming
- Installing Games for Ubuntu
- Playing Windows Games with Cedega
- Reference

For any operating system to have mass-market appeal, it has to have a number of compatible games. Let's face it, no one wants to use computers just for word processing or databases—they want to be able to use them as a source of relaxation and even fun! In this chapter, we look at the state of Linux gaming and tell you how to get some of the current blockbusters up and running in a Linux environment. We even show you how to run Windows-based games under Linux.

Linux Gaming

A number of games come as part of the Ubuntu distribution, and they are divided into three distinct camps: KDE games, GNOME games, and X games. Our favorites are Planet Penguin Racer and Torc (see Figure 9.1), but there are a few others for you to choose from. The best part, of course, is trying each one and seeing what you think. Many other free games are available across the Web, so go to Google and see what you come up with.

However, games for Linux do not stop there—a few versions of popular Windows-based games have been across to the Linux platform, including DOOM 3, Unreal Tournament 2004, and Quake 4. These three popular games have native Linux support and in some cases can run at similar, if not better, speeds than their Windows counterparts. There's even an emulator available that enables you to play classic adventure games, such as the Secret of Monkey Island, natively under Linux.

Finally, an implementation of the Wine code called Cedega is optimized especially for games. This uses application interfaces to make Windows games believe they are running on a Windows platform and not a Linux platform.

Bear in mind that Wine stands for *wine is not an emulator*, so do not start thinking of it as such—the community can get quite touchy about it!

FIGURE 9.1 Sliding around corners in the high-speed Torc.

A major gripe of Linux users has been the difficulty involved in getting modern 3D graphics cards to work. Thankfully, both ATI and Nvidia support Linux, albeit by using closed-source drivers. This means that Ubuntu does not ship with native 3D drivers for either graphics card. It is fairly easy to get a hold of these drivers and install them using either `synaptic` or `apt`.

Installing Proprietary Video Drivers

Unfortunately, both Nvidia and ATI still produce proprietary drivers, meaning that the source code is not open and available for developers to look at. This means that it is hard for some Linux distros to include them as part of their standard package manifest. However, Ubuntu have taken the pragmatic approach of including both Nvidia and ATI drivers within the main Ubuntu distro, albeit disabled by default. That way the end user can, if they so wish, activate those drivers to take advantage of them.

NOTE

Don't think that proprietary drivers are the only way on Linux, as there is a lot of development going into providing totally free and open source drivers for slightly older graphics cards. Ubuntu will automatically select the best "free" driver for your system and allow you to switch the proprietary driver should you want to. Although the open source drivers will provide 3D acceleration, this support doesn't always extend to the more recent graphics cards.

It's very easy to activate the proprietary driver if you need to; all you have to do is use the Restricted Drivers Manager found under System, Administration and which is shown in Figure 9.2.

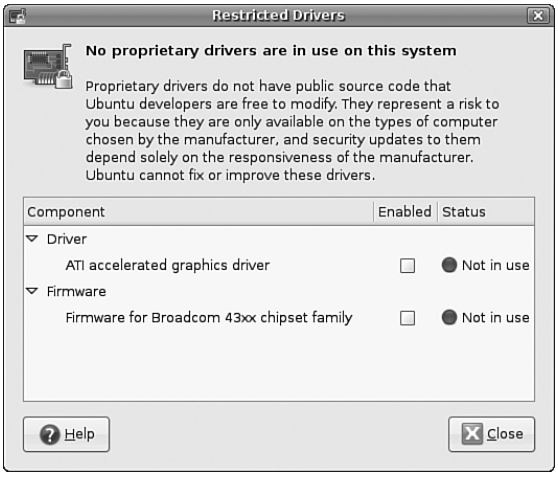


FIGURE 9.2 Use the Restricted Drivers Manager to activate or deactivate the appropriate proprietary graphics driver for your graphics card.

You will be prompted for your password before you proceed, as you will be setting a system-wide option. Once the Restricted Drivers Manager has opened up, look for the entry that says ATI (or Nvidia) Accelerated Graphics Driver and check the box. Ubuntu will confirm that you want to use the proprietary driver and, if you agree, will automatically download and configure the relevant driver. To activate the driver, you will need to log out of GNOME by going to System, Log Off. The next time you log in, Ubuntu will automatically switch to the proprietary driver.

Installing Games in Ubuntu

In this section, we'll take a look at how to install some of the more popular games for Ubuntu. Alongside the usual shoot-em-up games, you'll also find one or two strategy focused titles.

DOOM 3

The follow-up to the infamous DOOM and DOOM II was released in the second half of 2004 (see Figure 9.3), and it provides a way to run it under Linux. You still have to purchase the Windows version because you need some of the files that are on the CDs. The rest of the files are available from id Software at http://zerowing.idsoftware.com/linux/doom. You can find other information about graphics cards there, too.

FIGURE 9.3 Descending into the pits of hell. DOOM 3 is one of the most graphic computer games available.

You can download the file `doom3-linux-*.run` from the id Software FTP server or by using BitTorrent. When that's finished, open a terminal and change to the directory in which you saved the file. Type the following command:

```
# sh doom3-linux-*.run
```

This begins the installation of the demo. As with other commercial games, you must agree to an EULA before you can install. Follow the installation procedure. When it finishes, you need to get the Windows CDs ready.

The files you need to copy across are the following:

- `pak000.pk4`
- `pak001.pk4`
- `pak002.pk4`
- `pak003.pk4`
- `pak004.pk4`

They must be saved in the `/usr/local/games/doom3/base/` directory. After you copy the files, you can start the game by typing **`doom3`** or start the dedicated server for multiplayer games by typing **`doom3-dedicated`**.

Unreal Tournament 2004

Unreal Tournament 2004 (or *UT2004*, as it is affectionately known) from Epic natively supports Linux in both its 32-bit and 64-bit incarnations (see Figure 9.4). Be aware that if you run the 64-bit version, you need to ensure that your graphics drivers are supported under 64-bit mode.

FIGURE 9.4 Unreal Tournament 2004 builds on the classic deathmatch scenario with more enemies and more combatants!

Installation is easy, and there are two ways to do it. You can insert the DVD and mount it, or you can open the DVD in GNOME and double-click the `linux-installer.sh` icon. When you are asked whether you want to run it or display its contents, click Run in Terminal to launch the graphical installer. As with DOOM 3, you must read and accept the terms of the EULA before you are allowed to install UT2004. You are given the option of where you want to install the software; the default is in your home directory. After you select the destination directory, click Begin Install; UT2004 does the rest.

The alternative way to access the graphical installer is via the command line. Change the directory to `/media/cdrom/` and type the following:

```
$ sh linux-install.sh
```

This brings up the graphical installer. Continue through this and, when finished, you should find Unreal Tournament 2004 in `/home/username/ut2004`.

If you want to uninstall UT2004, you can use the uninstall script in the `ut2004` directory. Type this:

```
$ sh uninstall.sh
```

After confirmation, Unreal Tournament removes itself from your system.

Quake 4

Being based on the DOOM 3 engine, you could almost expect Quake 4 (seen in Figure 9.5) to ship with a good deal of support for Linux. To get started, you must have the Windows version of the software because you need several files as well as the CD key to be able to play the game. First things first, though. Head on over to http://zerowing.idsoftware.com/linux/quake4/ to download the required Linux installer (`quake4-linux-1.0*.run`) by either direct FTP or the more bandwidth-friendly BitTorrent.

After you download the file, drop down to a command line and type the following:

```
$ sudo sh quake4-linux-1.0*.run
```

Then press Enter. The installer starts up and asks you a couple questions. After you answer these, the installer creates the necessary files and folders. All you need to do is to copy several files from the `/quake4/qbase` directory on the DVD to `/usr/local/bin/quake4/qbase`. You can start the game by typing **`quake4`** at a command prompt.

Wolfenstein: Enemy Territory

Whereas the earlier Return to Castle Wolfenstein was both single- and multiplayer, the freely available Wolfenstein: Enemy Territory is multiplayer only (see Figure 9.6). Available in Win32 and Linux native versions, you can download it from http://www.SplashDamage.com/. After you download the 260MB file named `et-linux-2.55.x86.run`, install the game by typing the following:

```
$sudo sh et-linux-2.55.x86.run
```

Then accept the defaults. A symlink exists in `/usr/local/bin` to the script that loads the game.

FIGURE 9.5 Based on the popular DOOM 3 engine, Quake 4 pits you against the evil Strogg. Get out there and frag 'em!

FIGURE 9.6 Teamwork is the key to victory in this lush but hostile graphical environment.

Battle for Wesnoth

Of course, there is more to Linux gaming than just first-person shooters. One of the most popular games currently available for Linux is Battle for Wesnoth (see Figure 9.7), a strategy game much in the vein of Age of Empires. Based in a fantasy land, you are responsible for building armies to wage war against your foes.

Battle for Wesnoth is easy to install with Ubuntu. You need to select the `wesnoth` package using either `synaptic` or `aptitude`. When installed, you launch it from the command line by entering the following:

```
$ wesnoth
```

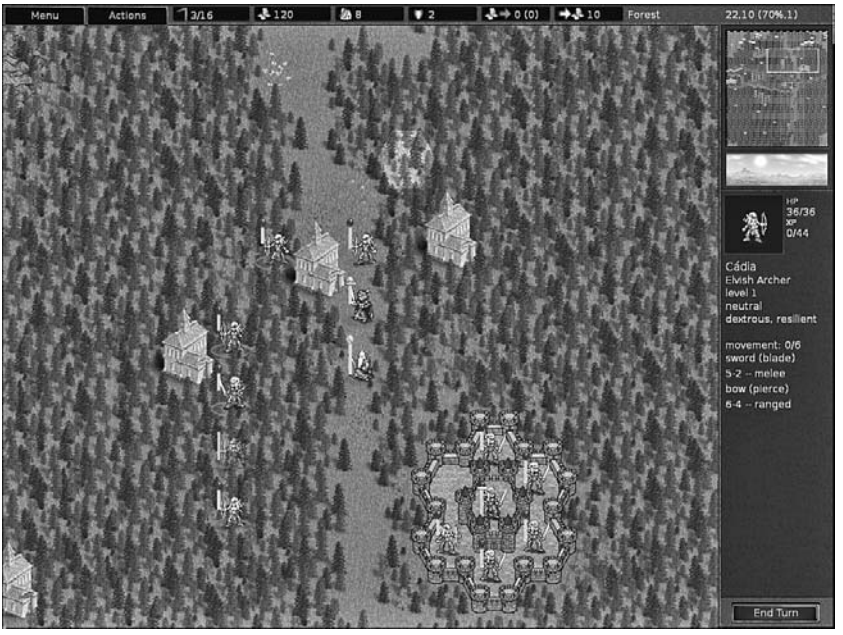


FIGURE 9.7 Flex your strategic brain by playing Battle for Wesnoth—a rich and full land of fantasy of adventure.

Playing Windows Games with Cedega

As mentioned earlier, the key to mass-market appeal of an operating system is in the applications available for it. A group of developers saw that the vast majority of the computing world was using Windows-based productivity and gaming software and decided to develop a way to run this software on Linux, thereby giving Linux users access to this large application base. The developers came up with a program called Wine, which has been updated regularly and forms the basis of the gaming variant called Cedega. This is a commercial product available from developers TransGaming Technologies (http://www.transgaming.com/), so you cannot retrieve it using `aptitude`.

However, Cedega is a popular and up-to-date product with support for recent releases such as Elder Scrolls IV: Oblivion and World of Warcraft. Because the state of Cedega is constantly changing, TransGaming Technologies has a subscription service, which means that you get updates for the code when they are released—ensuring that you are able to enjoy not only the games of today, but also those of tomorrow.

So, if you cannot wait for Linux to become more popular with game developers, use Cedega as a stop-gap until they can be persuaded to support Linux directly.

TIP

The keys to successful gaming in Linux are to always read the documentation thoroughly, always investigate the Internet resources thoroughly, and always understand your system. Installing games is a great way to learn about your system because the reward of success is so much fun.

Reference

- http://www.transgaming.com/—The official TransGaming Technologies website provides details of games that are directly supported under Cedega.
- http://www.linuxgames.com/—A good source of up-to-date information about the state of Linux gaming.
- http://zerowing.idsoftware.com/linux/doom/—Includes a complete how-to and troubleshooting guide for running DOOM 3 under Linux.
- http://www.unrealtournament.com/—The official site of Unreal Tournament.
- http://www.nvnews.net/vbulletin/forumdisplay.php?f=14—The Official Nvidia Linux driver support forum.
- http://www.nvidia.com/object/linux.html—Home page for the Nvidia Linux drivers.
- http://tinyurl.com/3pm2v—Home page for the ATI Linux drivers (courtesy of tinyurl.com).

This page intentionally left blank

PART III

System Administration

IN THIS PART

This page intentionally left blank

CHAPTER 10

Managing Users

IN THIS CHAPTER

If it weren't for users, system administrators would have a quiet life! However, it's impossible for you have a system with absolutely no users, so it is important that you learn how to effectively manage and administer your users as they work with your system. Whether you are creating a single user account or modifying a group that holds hundreds of user accounts, the fundamentals of user administration are the same.

User management and administration includes looking after allocating and managing home directories, putting in place good password policies, and applying an effective security policy including disk quotas and file and directory access permissions. We will take a look at all of these areas within this chapter, as well as the different types of user that you are likely to find on an average Linux system.

User Accounts

You can normally find three types of users on all Linux systems: the super user, the day-to-day user, and the system user. Each type of user is essential to the smooth running of your system and learning the differences between the three is essential if you are to work efficiently and safely within your Linux environment.

All users who access your system must have accounts on the system itself. Ubuntu uses the /etc/passwd file to store information on the user accounts that are present on the system. All users, regardless of their type, have a one-line entry in this file that contains their username (typically used for logging in to the system), a password field (which contains an x to denote that a password is present), a User ID (commonly referred to as the UID), and a Group ID

(commonly referred to as the GID). The last two fields show the location of the home directory (usually /home/username) and the default shell for the user (`/bin/bash` is the default for new users).

NOTE

Just because the password field contains an x doesn't mean that this is your password! All passwords are actually encrypted and stored in `/etc/shadow` for safe keeping. Ubuntu automatically refers to this file whenever a password is required.

In keeping with other UNIX-based operating systems, Linux and Ubuntu make use of the established UNIX file ownership and permission system. All files (which can include directories and devices) can be assigned one or more of read, write, and/or execute permissions. These three "flags" can also be assigned to the owner of the file, a member of a group, or anyone on the system. The security for a file is drawn from these permissions and also file ownership. As the system administrator (more commonly referred to as the super user), it is your responsibility to manage these settings effectively and ensure that the users have proper UIDs and GIDS. Perhaps most importantly, you will need to lock away sensitive files from users who should not have access to them, through file permissions.

The Super User/Root User

No matter how many system administrators there are for a system, there can ever only be one super user account. The super user account, more commonly referred to as the root user, has total and complete control over all aspects of the system. They can go anywhere in the filesystem, grant and revoke access to files and directories, and can carry out any operation on the system, including destroying it if they so wish. The root user is unique in that it has a UID of 0 and GID of 0.

As you can probably guess by now, the root user has supreme power over your system. With this in mind, it's important that you do not work as root all the time as you may inadvertently cause serious damage to your system, perhaps even making it totally unusable. Instead, you should rely on root only when you need to make specific changes to your system that require root privileges. As soon as you've finished your work, you can switch back to your normal user account to carry on working.

Within Ubuntu, you execute a command with root privileges by way of the `sudo` command like so:

```
$sudo apt-get update
```

You will be prompted for your password so you should enter this. Ubuntu will then carry out the command (in this case, updating information about available software) as if you were running it as root.

The Root User

If you've used other Linux distros, you are likely to be a little puzzled by the use of the `sudo` command. In short, Ubuntu allows the first user on the system access to full root privileges through the `sudo` command. It also disables the root account so no one can actually login with the username root.

In other Linux distros, you would change to the root user by issuing the command `su -` followed by the root password. This will land you at the root prompt, which is shown as a pound sign (#). From here you are able to execute any command you wish. To get to a root prompt in Ubuntu, you need to execute the command `sudo -i` which, after you enter your password, will give you the prompt so familiar to other Linux distros. When you've finished working as root, just type `exit` and hit Enter to get back to a normal user prompt (`$`).

A regular user is someone who logs on to the system to make use of it for nonadministrative tasks such as word processing or email. These users do not need to make systemwide changes or manage other users. However, they might want to be able to change settings specific to them (for instance, a desktop background). Of course, depending on how draconian the root user is, regular users might not even be able to do that!

The super user grants privileges to regular users by means of file and directory permissions (as covered in Chapter 4, "Command Line Quickstart"). For example, if the super user does not want you to change your settings in `~/.profile` (the ~ is a shell shortcut representing your home directory), root can alter the permissions so that you may read from, but not write to, that file.

CAUTION

Because of the potential for making a catastrophic error as the super user (using the command `rm -rf /*` is the classic example, but do not ever try it!), always use your system as a regular user and become root only temporarily to do sysadmin duties. While you are on a multiuser system, consider this advice an absolute rule; if root were to delete the wrong file or kill the wrong process, the results could be disastrous for the business. On your home system, you can do as you please, and running as root makes many things easier, but less safe. In any setting, however, the risks of running as root are significant and we cannot stress how important it is to be careful when working as root.

The third type of user is the system user. The system user is not a person, but rather an administrative account that the system uses during day-to-day running of various services. For example, the system user named `apache` owns the apache web server and all the associated files. Only itself and root can have access to these files—no one else can access or make changes to these files. System users do not have a home directory or password, nor do they permit access to the system through a login prompt.

You will find a list of all the users on a system in the `/etc/passwd` file. Ubuntu refers to these users as the *standard users* because they are found on every Ubuntu computer as the default set of system (or logical) users provided during the initial installation. This "standard" set differs among Linux distributions.

User IDs and Group IDs

A computer is, by its very nature, a number-oriented machine. It identifies users and groups by numbers known as the *user ID (UID)* and *group ID (GID)*. The alphabetic names display on your screen just for the your ease of use.

As previously mentioned, the root user is UID 0. Numbers from 1 through 499 and 65,534 are the system, or logical, users. Regular users have UIDs beginning with 1,000; Ubuntu assigns them sequentially beginning with this number.

With only a few exceptions, the GID is the same as the UID.

Ubuntu creates a private GID for every UID of 1,000 and greater. The system administrator can add other users to a GID or create a totally new group and add users to it. Unlike Windows NT and some UNIX variants, a group cannot be a member of another group in Linux.

File Permissions

As you learned in Chapter 4, permissions are of three types: read, write, and execute (`r`, `w`, `x`). For any file or directory, permissions can be established in three categories: `user`, `group`, and `global`. In this section, we focus on group permissions, but there is a highlight of the commands used to change the `group`, `user`, or `access` permissions of a file or directory: `Seth`

- `chgrp`—Changes the group ownership of a file or directory
- `chown`—Changes the owner of a file or directory
- `chmod`—Changes the access permissions of a file or directory

These commands, which modify file ownerships and permissions, can be used to model organizational structures and permissions in the real world onto your Ubuntu system (see the next section, "Managing Groups"). For example, a human resources department can share health-benefit memos to all company employees by making the files readable (but not writable) by anyone in an accessible directory. On the other hand, programmers in the company's research and development section, although able to access each other's source code files, would not have read or write access to HR pay-scale or personnel files (and certainly would not want HR or marketing poking around R&D).

These commands are used to easily manage group and file ownerships and permissions from the command line. It is essential that you know these commands because sometimes you might have only a command-line interface to work with; perhaps some idiot system administrator set incorrect permissions on X11, for example, rendering the system incapable of working with a graphical interface.

User Stereotypes

As is the case in many professions, exaggerated characterizations (**stereotypes** or **caricatures**) have emerged for users and system administrators. Many stereotypes contain elements of truth mixed with generous amounts of hyperbole and humor and serve to assist us in understanding the characteristics of and differences in the stereotyped subjects. The stereotypes of the "luser" and the "BOFH" (users and administrators, respectively) also serve as cautionary tales describing what behavior is acceptable and unacceptable in the computing community.

Understanding these stereotypes allows you to better define the appropriate and inappropriate roles of system administrators, users, and others. You can find a good reference for both at Wikipedia: http://http://en.wikipedia.org/wiki/BOFH and http://en.wikipedia.org/wiki/Luser.

Managing Groups

Groups can make managing users a lot easier. Instead of having to assign individual permissions to every user, you can use groups to grant or revoke permissions to a large number of users quickly and easily. Setting group permissions allows you to set up workspaces for collaborative working and to control what devices can be used, such as external drives or DVD writers. This approach also represents a secure method of limiting access to system resources to only those users who need them. As an example, the sysadmin could put the users `andrew`, `paul`, `michael`, `bernice`, `mark`, and `john` in a new group named `unleashed`. Those users could each create files intended for their group work and `chgrp` those files to `unleashed`.

Now, everyone in the `unleashed` group—but no one else except root—can work with those files. The sysadmin would probably create a directory owned by that group so that its members could have an easily accessed place to store those files. The sysadmin could also add other users such as `bernice` and `ildiko` to the group and remove existing users when their part of the work is done. The sysadmin could make the user `andrew` the group administrator so that `andrew` could decide how group membership should be changed. You could also put restrictions on the DVD writer so that only `andrew` could burn DVDs, thus protecting sensitive material from falling into the wrong hands.

Different UNIX operating systems implement the group concept in various ways. Ubuntu uses a scheme called *UPG*, the *user private group*, in which all users are assigned to a group with their own name by default. (The user's username and group name are identical.) All the groups are listed in `/etc/group` file.

Here is a partial list of a sample `/etc/group` file:

```
$ sudo cat /etc/group
root:x:0:
daemon:x:1:
bin:x:2:
```

```
sys:x:3:

mail:x:8:
news:x:9:
...
gdm:x:111:
andrew:x:1000:
admin:x:112:andrew
```

This example contains a number of groups, mostly for services (`mail`, `news`, and so on) and devices (floppy, disk, and so on). As previously mentioned, the system services groups enable those services to have ownership and control of their files. For example, adding `postfix` to the `mail` group, as shown previously, enables the `postfix` application to access `mail`'s files in the manner that `mail` would decide for group access to its file. Adding a regular user to a device's group permits the regular user to use the device with permissions granted by the group owner. Adding user `andrew` to the group `cdrom`, for example, would allow `andrew` to use the optical drive device. You learn how to add and remove users from groups in the next section.

Group Management Tools

Ubuntu provides several command-line tools for managing groups, but also provides graphical tools for such. Many experienced sysadmins prefer the command-line tools because they are quick and easy to use and can be included in scripts if the sysadmin desires to script a repetitive task.

Here are the most commonly used group management command-line tools:

`groupadd`—This command creates and adds a new group.

`groupdel`—This command removes an existing group.

`groupmod`—This command creates a group name or GIDs but doesn't add or delete members from a group.

`gpasswd`—This command creates a group password. Every group can have a group password and an administrator. Use the `-A` argument to assign a user as group administrator.

`useradd -G`—The `-G` argument adds a user to a group during the initial user creation. (More arguments are used to create a user.)

`usermod -G`—This command allows you to add a user to a group so long as the user is not logged in at the time.

`grpck`—A command for checking the `/etc/group` file for typos.

As an example, there is a DVD-RW device (`/dev/scd0`) on our computer that the sysadmin wants a regular user named `john` to have access to. To grant `john` that access, we would follow these steps:

1. Add a new group with the `groupadd` command:

```
$ sudo groupadd dvdrw
```

2. Change the group ownership of the device to the new group with the `chgrp` command:

```
$ sudo chgrp dvdrw /dev/scd0
```

3. Add the approved user to the group with the `usermod` command:

```
$ sudo usermod -G dvdrw  john
```

4. Make user `john` the group administrator with the `gpasswd` command so that she can add new users to the group:

```
$ sudo gpasswd -A john
```

Now, the user `john` has permission to use the DVD-RW drive, as would anyone else added to the group by the super user or `john` because he is now also the group administrator and can add users to the group.

The sysadmin can also use the graphical interface that Ubuntu provides, as shown in Figure 10.1. It is accessed under System, Administration as the Users and Groups entry.

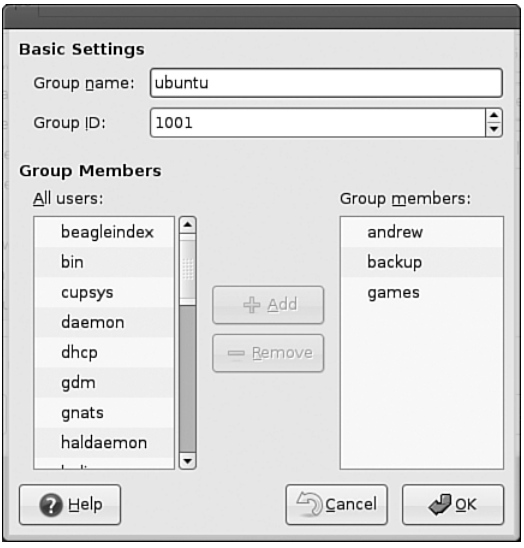


FIGURE 10.1 Use the manage groups option to allow you to assign users to groups.

Note that the full set of group commands and options are not available from the graphical interface, limiting the usefulness of the GUI to a subset of the most frequently used

commands. You learn more about using the Ubuntu User Manager GUI in the next section of this chapter.

Managing Users

We have mentioned users previously, but in this section we examinelook at how the sysadmin can actually manage the users. Users must be created, assigned a UID, provided a home directory, provided an initial set of files for their home directory, and assigned to groups so that they can use the system resources securely and efficiently. The system administrator might elect to restrict a user's access not only to files, but to the amount of disk space they use, too.

User Management Tools

Ubuntu provides several command-line tools for managing users, but also provides graphical tools too. Many experienced sysadmins prefer the command-line tools because they are quick and easy to use and they can be included in scripts if the sysadmin prefers to script a repetitive task. Here are the most frequently used commands used to manage users:

`useradd`—This command is used to add a new user account to the system. Its options permit the sysadmin to specify the user's home directory and initial group or to create the user with the default home directory and group assignments.

`useradd -D`—This command sets the system defaults for creating the user's home directory, account expiration date, default group, and command shell. See the specific options in `man useradd`. Used without any arguments, it displays the defaults for the system. The default set of files for a user are found in `/etc/skel`.

NOTE

The set of files initially used to populate a new user's home directory are kept in `/etc/skel`. This is convenient for the system administrator because any special files, links, or directories that need to be universally applied can be placed in `/etc/skel` and will be duplicated automatically with appropriate permissions for each new user.

```
$ ls -al /etc/skel
total 20

drwxr-xr-x   2 root root 4096 2007-08-09 00:59 .

drwxr-xr-x 111 root root 4096 2007-08-20 09:54 ..

-rw-r--r--   1 root root  220 2007-05-17 12:59 .bash_logout

-rw-r--r--   1 root root 2298 2007-05-17 12:59 .bashrc

lrwxrwxrwx   1 root root   26 2007-08-13 19:42 Examples \
```

```
-> /usr/share/example-content

-rw-r--r--   1 root root  566 2007-05-17 12:59 .profile
```

Each line provides the file permissions, the number of files housed under that file or directory name, the file owner, the file group, the file size, the creation date, and the filename.

As you can see, root owns every file here, but the `adduser` command (a symbolic link to the actual command named `useradd`) copies everything in `/etc/skel` to the new home directory and resets file ownership and permissions to the new user.

Certain user files might exist that the system administrator doesn't want the user to change; the permissions for those files in `/home/username` can be reset so that the user can read them but can't write to them.

`userdel`—This command completely removes a user's account (thereby eliminating that user's home directory and all files it contains).

`passwd`—This command updates the "authentication tokens" used by the password management system.

TIP

To lock a user out of his account, use the following command:

```
$ sudo passwd -l username
```

This prepends an `!` (exclamation point, also called a *bang*) to the user's encrypted password; the command to reverse the process uses the `-u` option. This is a more elegant and preferred solution to the problem than the traditional UNIX way of manually editing the file.

`usermod`—This command changes several user attributes. The most commonly used arguments are `-s` to change the shell and `-u` to change the UID. No changes can be made while the user is logged in or running a process.

`chsh`—This command changes the user's default shell. For Ubuntu, the default shell is `/bin/bash`, known as the *Bash,* or *Bourne Again Shell.*

Adding New Users

The command-line approach to adding this user is actually quite simple and can be accomplished on a single line. In the example shown here, the sysadmin uses the `useradd` command to add the new user `bernice`. The command `adduser` (a variant found on some UNIX systems) is a symbolic link to `useradd`, so both commands work the same. In this example, we use the `-p` option to set the password the user requested; we use the `-s` to set

his special shell, and the `-u` option to specify her UID. (If we create a user with the default settings, we do not need to use these options.) All we want to do can be accomplished on one line:

```
$ sudo useradd bernice -p sTitcher -s /bin/zsh -u 1002
```

The sysadmin can also use the graphical interface that Ubuntu provides, as shown in Figure 10.2. It is accessed as the Users and Groups item from the Administration menu. Here, the sysadmin is adding a new user to the system where user `bernice` uses the `bash` command shell.

These are the steps we used to add the same account as shown in the preceding command, but using the graphical User Manager interface:

1. Launch the Ubuntu User Manager graphical interface by clicking on the Users and Groups menu item found in the System Settings menu.
2. Click the Add User button to bring up the Add User dialog window.
3. Fill in the form with the appropriate information as requested, ensuring you create a good password.
4. Click the Advanced tab and open the drop-down Shell menu to select the `bash` shell.
5. Using the arrows found in the UID dialog, increment the UID to 1413.
6. Click OK to save the settings.

Note that the user is being manually assigned the UID of `1413` because that is her UID on another system machine that will be connected to this machine. Because the system only knows her as `1001` and not as `bernice`, the two machines would not recognize `bernice` as the same user if two different UIDs were assigned.

NOTE

A Linux username can be any alphanumeric combination that does not begin with a special character reserved for shell script use (see Chapter 11 for disallowed characters, mostly punctuation characters). In Chapter 4, we told you that usernames are typically the user's first name plus the first initial of her last name. That is a common practice on larger systems with many users because it makes life simpler for the sysadmin, but is neither a rule nor a requirement.

Monitoring User Activity on the System

Monitoring user activity is part of the sysadmin's duties and an essential task in tracking how system resources are being used. The `w` command tells the sysadmin who is logged in, where he is logged in, and what he is doing. No one can hide from the super user. The `w` command can be followed by a specific user's name to show only that user.

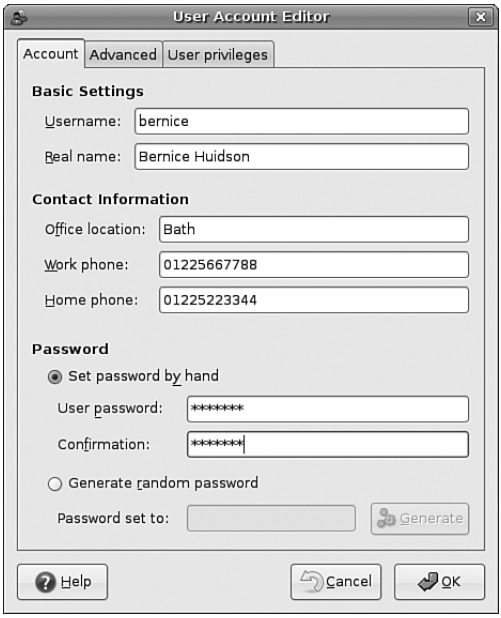


FIGURE 10.2 Adding a new user is simple. The GUI provides a more complete set of commands for user management than for group management.

The `ac` command provides information about the total connect time of a user measured in hours. It accesses the `/var/log/wtmp` file for the source of its information. The `ac` command proves most useful in shell scripts to generate reports on operating system usage for management review. Note that to use the `ac` command, you will have to install the `acct` package using either `synaptic` or `apt-get`.

TIP

Interestingly, a phenomenon known as **timewarp** can occur in which an entry in the `wtmp` files jumps back into the past and `ac` shows unusual amounts of time accounted for users. Although this can be attributed to some innocuous factors having to do with the system clock, it is worthy of investigation by the sysadmin because it can also be the result of a security breach.

The `last` command searches through the `/var/log/wtmp` file and lists all the users logged in and out since that file was first created. The user `reboot` exists so that you might know who has logged in since the last reboot. A companion to `last` is the command `lastb`, which shows all failed, or bad, logins. It is useful for determining whether a legitimate user is having trouble or a hacker is attempting access.

NOTE

The accounting system on your computer keeps track of usage user statistics and is kept in the current `/var/log/wtmp` file. That file is managed by the `init` and `login` processes. If you want to explore the depths of the accounting system, use the GNU info system: `info accounting`.

Managing Passwords

Passwords are an integral part of Linux security, and they are the most visible part to the user. In this section, you learn how to establish a minimal password policy for your system, where the passwords are stored, and how to manage passwords for your users.

System Password Policy

An effective password policy is a fundamental part of a good system administration plan. The policy should cover the following:

- Allowed and forbidden passwords
- Frequency of mandated password changes
- Retrieval or replacement of lost or forgotten passwords
- Password handling by users

The Password File

The password file is `/etc/passwd`, and it is the database file for all users on the system. The format of each line is as follows:

```
username:password:uid:gid:gecos:homedir:shell
```

The fields are self-explanatory except for the *gecos* field. This field is for miscellaneous information about the user, such as the users' full name, his office location, office and home phone numbers, and possibly a brief text message. For security and privacy reasons, this field is little used nowadays, but the system administrator should be aware of its existence because the *gecos* field is used by traditional UNIX programs such as `finger` and `mail`. For that reason, it is commonly referred to as the *finger information field*. The data in this field will be comma delimited; the *gecos* field can be changed with the `cgfn` (change `finger`) command.

Note that a colon separates all fields in the `/etc/passwd` file. If no information is available for a field, that field is empty, but all the colons remain.

If an asterisk appears in the password field, that user will not be permitted to log on. Why does this feature exist? So that a user can be easily disabled and (possibly) reinstated later without having to be created all over again. The system administrator manually edits this field, which is the traditional UNIX way of accomplishing this task. Ubuntu provides improved functionality with the `passwd -l` command mentioned earlier.

Several services run as pseudo-users, usually with `root` permissions. These are the system, or logical, users mentioned previously. You would not want these accounts available for general login for security reasons, so they are assigned `/sbin/nologin` as their shell, which prohibits any logins from those "users."

A list of `/etc/passwd` reveals the following:

```
$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
...
hplip:x:106:7:HPLIP system user,,,:/var/run/hplip:/bin/false
andrew:x:1000:1000:Andrew Hudson,17,01225112233,01225445566:\
/home/andrew:/bin/bash
beagleindex:x:107:65534::/var/cache/beagle:/bin/false
```

Note that all the password fields do not show a password, but contain an `x` because they are *shadow passwords*, a useful security enhancement to Linux, discussed in the following section.

Shadow Passwords

It is considered a security risk to keep any password in `/etc/passwd` because anyone with read access can run a cracking program on the file and obtain the passwords with little trouble. To avoid this risk, *shadow passwords* are used so that only an `x` appears in the password field of `/etc/passwd`; the real passwords are kept in `/etc/shadow`, a file that can only be read by the sysadmin (and *PAM*, the *Pluggable Authentication Modules* authentication manager; see the "PAM Explained" sidebar for an explanation of PAM).

Special versions of the traditional `password` and `login` programs must be used to enable shadow passwords. Shadow passwords are automatically enabled during the installation phase of the operating system on Ubuntu systems.

Let's examine a listing of the shadow companion to `/etc/passwd`, the `/etc/shadow` file:

```
$ sudo cat /etc/shadow
root:*:13299:0:99999:7:::
daemon:*:13299:0:99999:7:::
bin:*:13299:0:99999:7:::
...
haldaemon:!:13299:0:99999:7:::
```

```
gdm:!:13299:0:99999:7:::
hplip:!:13299:0:99999:7:::
andrew:$1$6LT/qkWL$sPJPp.2QkpC8JPtpRk906/:13299:0:99999:7:::
beagleindex:!:13299:0:99999:7:::
```

The fields are separated by colons and are, in order:

- The user's login name.
- The encrypted password for the user.

  The day of which the last password change occurred, measured in the number of days since January 1, 1970. This date is known in UNIX circles as the *epoch*. Just so you know, the billionth second since the epoch occurred was in September 2001; that was the UNIX version of Y2K—as with the real Y2K, nothing much happened.

  The number of days before the password can be changed (prevents changing a password and then changing it back to the old password right away—a dangerous security practice).

  The number of days after which the password must be changed. This can be set to force the change of a newly issued password known to the system administrator.

  The number of days before the password expiration that the user is warned it will expire.

  The number of days after the password expires that the account is disabled (for security).

  Similar to the password change date, although this is the number of days since January 1, 1970 that the account has been disabled.

The final field is a "reserved" field and is not currently allocated for any use.

Note that password expiration dates and warnings are disabled by default in Ubuntu. These features are not used on home systems and usually not used for small offices. It is the sysadmin's responsibility to establish and enforce password expiration policies.

The permissions on the `/etc/shadow` file should be set so that it is not writable or readable by regular users: The permissions should be `600`.

PAM Explained

Pluggable Authentication Modules (PAM) is a system of libraries that handle the tasks of authentication on your computer. It uses four management groups: account management, authentication management, password management, and session management. This allows the system administrator to choose how individual applications will authenticate users. Ubuntu has preinstalled and preconfigured all the necessary PAM files for you.

The configuration files in Ubuntu are found in `/etc/pam.d`. These files are named for the service they control, and the format is as follows:

```
type control module-path module-arguments
```

The `type` field is the management group that the rule corresponds to. The `control` field tells PAM what to do if authentication fails. The final two items deal with the PAM module used and any arguments it needs. Programs that use PAM typically come packaged with appropriate entries for the `/etc/pam.d` directory. To achieve greater security, the system administrator can modify the default entries. Misconfiguration can have unpredictable results, so back up the configuration files before you modify them. The defaults provided by Ubuntu are adequate for home and small office users.

An example of a PAM configuration file with the formatted entries as described previously is shown next. Here are the contents of `/etc/pam.d/gdm`:

```
#%PAM-1.0
auth    requisite       pam_nologin.so
auth    required        pam_env.so readenv=1
auth    required        pam_env.so readenv=1 envfile=/etc/default/locale
@include common-account
session required        pam_limits.so
@include common-session
@include common-password
```

Amusingly, even the PAM documents state that you do not really need (or want) to know a lot about PAM to use it effectively.

You will likely need only the PAM system administrator's guide. You can find it at http://www.kernel.org/pub/linux/libs/pam/Linux-PAM-html/Linux-PAM_SAG.html.

Managing Password Security for Users

Selecting appropriate user passwords is always an exercise in trade-offs. A password such as *password* (do not laugh, it has been used too often before in the real world with devastating consequences) is just too easy for an intruder to guess, as are simple words or number combinations (a street address, for example). A security auditor for one of my former employers used to take the cover sheet from an employee's personnel file (which contained the usual personal information of name, address, birth date, and so on) and then attempt to log on to a terminal with passwords constructed from that information—and often succeeded in logging on.

On the other hand, a password such as `2a56u'"F($84u&#^Hiu44Ik%$([#EJD` is sure to present great difficulty to an intruder (or an auditor). However, that password is so difficult to remember that it would be likely that the password owner would write that password down and tape it next to her keyboard. I worked for a company in which the entry code to one of the buildings was etched into the cigarette bin outside the door; we never found out who did this, but quickly changed the security number. This is but one of many examples of poor security in the field.

The sysadmin has control, with settings in the `/etc/shadow` file, over how often the password must be changed. The settings can be changed using a text editor, the `change` command, or a configuration tool such as Ubuntu's User Manager, as shown previously in Figure 10.1. Click on the Password Info tab under that particular user's Properties to set individual password policies.

Changing Passwords in a Batch

On a large system, there might be times when a large number of users and their passwords need some attention. The super user can change passwords in a batch by using the `chpasswd` command, which accepts input as a name/password pair per line in the following form:

```
$ sudo chpasswd username:password
```

Passwords can be changed *en masse* by redirecting a list of name and password pairs to the command. An appropriate shell script can be constructed with the information gleaned from Chapter 11.

However, Ubuntu also provides the `newusers` command to add users in a batch from a text file. This command also allows a user to be added to a group, and a new directory can be added for the user, too.

Granting System Administrator Privileges to Regular Users

On occasion, it is necessary for regular users to run a command as if they were the root user. They usually do not need these powers, but they might on occasion—for example, to temporarily access certain devices or run a command for testing purposes.

There are two ways to run commands with root privileges: The first is useful if you are the super user and the user; the second if you are not the regular user (as on a large, multiuser network).

Temporarily Changing User Identity with the `su` Command

What if you are also root but are logged on as a regular user because you are performing nonadministrative tasks and you need to do something that only the super user can do? The `su` command is available for this purpose.

NOTE

A popular misconception is that the `su` command is short for *super user*; it just means *substitute user*. An important but often overlooked distinction is that between `su` and `su -`. In the former instance, you become that user but keep your own environmental variables (such as paths). In the latter, you inherit the environment of that user. This is most noticeable when you use `su` to become the super user, root. Without appending the `-`, you do not inherit the path variable that includes `/bin` or `/sbin`, so you must always enter the full path to those commands when you just `su` to `root`.

Don't forget that on a standard Ubuntu system, the first created user is classed as root, while the true root account is disabled. To enable the root account, enter the command `sudo passwd` at the command line and enter your password and a new root password. Once this has been completed, you can `su` to root.

Because almost all Linux file system security revolves around file permissions, it can be useful to occasionally become a different user with permission to access files belonging to other users or groups or to access special files (such as the communications port `/dev/ttyS0` when using a modem or the sound device `/dev/audio` when playing a game). You can use the `su` command to temporarily switch to another user identity, and then switch back.

TIP

It is never a good idea to use an **Internet Relay Chat (IRC)** client as the root user, and you might not want to run it using your regular user account. Simply create a special new user just for IRC and `su` to that user in a terminal widow to launch your IRC client.

The `su` command spawns a new shell, changing both the UID and GID of the existing user and automatically changes the environmental variables associated with that user, known as *inheriting the environment*. Refer to Chapter 4 for more information on environmental variables.

The syntax for the `su` command is as follows:

```
$ su option  username  arguments
```

The man page for `su` gives more details, but some highlights of the `su` command are here:

```
-c, --command COMMAND
      pass a single COMMAND to the shell with -c

-m, --preserve-environment
      do not reset environment variables

-l   a full login simulation for the substituted user,
     the same as specifying the dash alone
```

You can invoke the `su` command in different ways that yield diverse results. By using `su` alone, you can become root, but you keep your regular user environment. This can be verified by using the `printenv` command before and after the change. Note that the working directory (you can execute `pwd` as a command line to print the current working directory) has not changed. By executing the following, you become root and inherit root's environment:

```
$ su -
```

By executing the following, you become that user and inherit the super user's environment—a pretty handy tool. (Remember: Inheriting the environment comes from using

the dash in the command; omit that, and you keep your "old" environment.) To become another user, specify a different user's name on the command line:

```
$ su - other_user
```

When leaving an identity to return to your usual user identity, use the `exit` command. For example, while logged on as a regular user,

```
$ su - root
```

the system prompts for a password:

```
Password:
```

When the password is entered correctly, the root user's prompt appears:

```
#
```

To return to the regular user's identity, just type

```
# exit
```

This takes you to the regular user's prompt:

```
$
```

If you need to allow other users access to certain commands with root privileges, you must give them the root password so that they can use `su`—that definitely is not a secure solution. The next section describes a more flexible and secure method of allowing normal users to perform selected root tasks.

Granting Root Privileges on Occasion—The `sudo` Command

It is often necessary to delegate some of the authority that root wields on a system. For a large system, this makes sense because no single individual will always be available to perform super user functions. The problem is that UNIX permissions come with an all-or-nothing authority. Enter `sudo`, an application that permits the assignment of one, several, or all of the root-only system commands.

NOTE

As mentioned earlier, the `sudo` command is pervasive in Ubuntu, because it is used by default. If you want to get to a root shell, and thereby removing the need to type `sudo` for every command, just enter `sudo -i` to get the root prompt. To return to a normal user prompt, enter `exit` and press Return.

After it is configured, using `sudo` is simple. An authorized user merely precedes the super user authority-needed command with the `sudo` command, like so:

```
$ sudo command
```

After getting the user's password, sudo checks the /etc/sudoers file to see whether that user is authorized to execute that particular command; if so, sudo generates a "ticket" for a specific length of time that authorizes the use of that command. The user is then prompted for his password (to preserve accountability and provide some measure of security), and then the command is run as if root had issued it. During the life of the ticket, the command can be used again without a password prompt. If an unauthorized user attempts to execute a sudo command, a record of the unauthorized attempt is kept in the system log and a mail message is sent to the super user.

Three man pages are associated with sudo: sudo, sudoers, and visudo. The first covers the command itself, the second the format of the /etc/sudoers file, and the third the use of the special editor for /etc/sudoers. You should use the special editing command because it checks the file for parse errors and locks the file to prevent others from editing it at the same time. The visudo command uses the vi editor, so you might need a quick review of the vi editing commands found in Chapter 4 in the section "Working with vi." You begin the editing by executing the visudo command with this:

```
$ sudo visudo
```

The default /etc/sudoers file looks like this:

```
# /etc/sudoers
#
# This file MUST be edited with the 'visudo' command as root.
#
# See the man page for details on how to write a sudoers file.
# Host alias specification

# User alias specification

# Cmnd alias specification

# Defaults

Defaults        !lecture,tty_tickets,!fqdn

# User privilege specification
root    ALL=(ALL) ALL

# Members of the admin group may gain root privileges
%admin  ALL=(ALL) ALL
```

The basic format of a sudoers line in the file is as follows:

```
user host_computer=command
```

The user can be an individual user or a group (prepended by a % to identify the name as a group). The host_computer is normally ALL for all hosts on the network and localhost for the local machine, but the host computer can be referenced as a subnet or any specific host. The command in the sudoers line can be ALL, a list of specific commands, or a restriction on specific commands (formed by prepending a ! to the command). A number of options are available for use with the sudoers line, and aliases can be used to simplify the assignment of privileges. Again, the sudoers man page will give the details, but here are a few examples:

If we uncomment the line

```
# %wheel        ALL=(ALL)       NOPASSWD: ALL
```

any user we add to the wheel group can execute any command without a password.

Suppose that we want to give a user john permission across the network to be able to add users with the graphical interface. We would add the line

```
john ALL=/users-admin
```

or perhaps grant permission only on her local computer:

```
john 192.168.1.87=/usr/bin/users-admin
```

If we want to give the editor group systemwide permission with no password required to delete files:

```
%editors ALL=NOPASSWD: /bin/rm
```

If we want to give every user permission with no password required to mount the CD drive on the localhost:

```
ALL localhost=NOPASSWD:/sbin/mount /dev/scd0 /mnt/cdrom /sbin/umount /mnt/cdrom
```

It is also possible to use wildcards in the construction of the sudoers file. Aliases can be used, too, to make it easier to define users and groups. Although the man page for sudoers contains some examples, http://www.komar.org/pres/sudo/toc.html provides illustrative notes and comments of sudo use at a large aerospace company. The sudo home page at http://www.sudo.ws/ is also a useful resource for additional explanations and examples.

The following command presents users with a list of the commands they are entitled to use:

```
$ sudo -l
```

Adding Extra Sudo Users

As mentioned earlier, by default Ubuntu grants the first created user full root access through the `sudo` command. If you need to add this capability for other users, then you can do this easily by adding each user to the `admin` group or by using the User Manager tool to allow them to Administer the System, which can be found in the User Privileges tab when you edit the properties for a user.

Disk Quotas

On large systems with many users, you need to control the amount of disk space a user has access to. Disk quotas are designed specifically for this purpose. Quotas, managed per each partition, can be set for both individual users as well as groups; quotas for the group need not be as large as the aggregate quotas for the individuals in the groups.

When files are created, both a user and a group own them; ownership of the files is always part of the metadata about the files. This makes quotas based on both users and groups easy to manage.

NOTE

Disk quota management is never done on a home system and rarely, if ever, done on a small office system.

To manage disk quotas, you must have the `quota` and `quotatool` packages installed on your system. Quota management with Ubuntu is not enabled by default and has traditionally been enabled and configured manually by system administrators. Sysadmins use the family of quota commands, such as `quotacheck` to initialize the quota database files, `edquota` to set and edit user quotas, `setquota` to configure disk quotas, and `quotaon` or `quotaoff` to control the service. (Other utilities include `warnquota` for automatically sending mail to users over their disk space usage limit.)

Implementing Quotas

To reiterate, quotas might not be enabled by default, even if the quota software package is installed on your system. When quotas are installed and enabled, you can see which partitions have either user quotas, group quotas, or both by looking at the fourth field in the `/etc/fstab` file. For example, one line in `/etc/fstab` shows that quotas are enabled for the `/home` partition:

```
/dev/hda5      /home     ext3         defaults,usrquota,grpquota 1 1
```

The root of the partition with quotas enabled will have the files `quota.user` or `quota.group` in them (or both files, if both types of quotas are enabled), and the files will contain the actual quotas. The permissions of these files should be `600` so that users cannot read or write to them. (Otherwise, users would change them to allow ample space

for their music files and Internet art collections.) To initialize disk quotas, the partitions must be remounted. This is easily accomplished with the following:

```
$ sudo mount -o ro,remount partition_to_be_remounted mount_point
```

The underlying console tools (complete with man pages) are

- `quotaon`, `quotaoff`—Toggles quotas on a partition.
- `repquota`—A summary status report on users and groups.
- `quotacheck`—Updates the status of quotas (compares new and old tables of disk usage); it is run after `fsck`.
- `edquota`—A basic quota management command.

Manually Configuring Quotas

Manual configuration of quotas involves changing entries in your system's file system table, `/etc/fstab`, to add the `usrquota` mount option to the desired portion of your file system. As an example in a simple file system, quota management can be enabled like this:

```
LABEL=/           /               ext3    defaults,usrquota        1 1
```

Group-level quotas can also be enabled by using the `grpquota` option. As the root operator, you must then create a file (using our example of creating user quotas) named `quota.user` in the designated portion of the file system, like so:

```
$ sudo touch /quota.user
```

You should then turn on the use of quotas using the `quotaon` command:

```
$ sudo quotaon -av
```

You can then edit user quotas with the `edquota` command to set hard and soft limits on file system use. The default system editor (`vi` unless you change your `EDITOR` environment variable) will be launched when editing a user's quota.

Any user can find out what their quotas are with

```
$ quota -v
```

NOTE

Ubuntu does not support any graphical tools that enable you to configure disk quotas. A Quota mini-HOWTO is maintained at http://www.tldp.org/HOWTO/mini/Quota.html.

Related Ubuntu Commands

You will use these commands to manage user accounts in Ubuntu:

`ac`—A user account-statistics command

`change`—Sets or modifies user password expiration policies

`chfn`—Creates or modifies user finger information in `/etc/passwd`

`chgrp`—Modifies group memberships

`chmod`—Changes file permissions

`chown`—Changes file ownerships

`chpasswd`—Batch command to modify user passwords

`chsh`—Modifies a user's shell

`groups`—Displays existing group memberships

`logname`—Displays a user's login name

`newusers`—Batches user management command

`passwd`—Creates or modifies user passwords

`su`—Executes shell or command as another user

`sudo`—Manages selected user execution permissions

`useradd`—Creates, modifies, or manages users

`usermod`—Edits a user's login profile

Reference

- http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/_User-Authentication-HOWTO.html—The User-Authentication HOWTO describes how user and group information is stored and used for authentication.
- http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/Shadow-Password-HOWTO.html—The Shadow-Password HOWTO delves into the murky depths of shadow passwords and even discusses why you might not want to use them.
- http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/_Security-HOWTO.html—A must-read HOWTO, the Security HOWTO is a good overview of security issues. Especially applicable to this chapter are sections on creating accounts, file permissions, and password security.
- http://www.secinf.net/unix_security/Linux_Administrators_Security_Guide/—A general guide, the Linux System Administrator's Security Guide has interesting sections on limiting and monitoring users.
- http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/Config-HOWTO.html—How can you customize some user-specific settings? The Config HOWTO Software Configuration gives some advice.

- http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/Path.html—How can one know the true path? The Path HOWTO sheds light on this issue. You need to understand paths if you want to guide the users to their data and applications.
- http://www.courtesan.com/sudo/—The Superuser `DO` command is a powerful and elegant way to delegate authority to regular users for specific commands.
- http://www.kernel.org/pub/linux/libs/pam/index.html—The Pluggable Authentication Modules suite contains complex and highly useful applications that provide additional security and logging for passwords. PAM is installed by default in Ubuntu. It isn't necessary to understand the intricacies of PAM to use it effectively.

CHAPTER 11

Automating Tasks

In this chapter you learn about the five ways to automate tasks on your system: making them services that run as your system starts, making them services you start and stop by hand, scheduling them to run at specific times, connecting multiple commands together on the shell, and writing custom scripts that group everything together under one command.

After you turn on the power switch, the boot process begins with the computer executing code stored in a chip called the BIOS; this process occurs no matter what operating system you have installed. The Linux boot process begins when the code known as the boot loader starts loading the Linux kernel and ends only when the login prompt appears.

As a system administrator, you will use the skills you learn in this chapter to control your system's services and manage runlevels on your computer. Understanding the management of the system services and states is essential to understanding how Linux works (especially in a multi-user environment) and will help untangle the mysteries of a few of your Ubuntu system's configuration files. Furthermore, a good knowledge of the `cron` daemon that handles task scheduling is essential for administrators at all skill levels.

This chapter is also an introduction to the basics of creating *shell scripts*, or executable text files written to conform to shell syntax. Shell scripts run like any other command under Linux and can contain complex logic or a simple series of Linux command-line instructions. You can also run other shell scripts from within a shell program. The features and functions for several Linux shells are discussed in this chapter after a short introduction to working from

IN THIS CHAPTER

the shell command line. You learn how to write and execute a simple shell program using `bash`, one of the most popular Linux shells.

Running Services at Bootup

Although most people consider a computer to be either on or off, in Ubuntu there are a number of states in between. Known as *runlevels*, they control what system services are started upon bootup. These services are simply applications running in the background that provide some needed function to your system, such as getting information from your mouse and sending it to the display; or a service could monitor the partitions to see whether they have enough free space left on them. Services are typically loaded and run (also referred to as being started) during the boot process, in the same way as Microsoft Windows services are. Internally, Ubuntu uses a system known as Upstart for fast booting, but this has a special backward compatibility layer that uses runlevels in the way that Linux veterans are accustomed.

You can manage nearly every aspect of your computer and how it behaves after booting via configuring and ordering boot scripts, as well as by using various system administration utilities included with Ubuntu. In this chapter, you learn how to work with these boot scripts and system administration utilities. This chapter also offers advice for troubleshooting and fixing problems that might arise with software configuration or the introduction or removal of various types of hardware from your system.

Beginning the Boot Loading Process

Although the actual boot loading mechanism for Linux varies on different hardware platforms (such as the SPARC, Alpha, or PowerPC systems), Intel-based PCs running Ubuntu most often use the same mechanism throughout product lines. This process is accomplished through a Basic Input Output System, or BIOS. The BIOS is an application stored in a chip on the motherboard that initializes the hardware on the motherboard (and often the hardware that's attached to the motherboard). The BIOS gets the system ready to load and run the software that we recognize as the operating system.

As a last step, the BIOS code looks for a special program known as the boot loader or boot code. The instructions in this little bit of code tell the BIOS where the Linux kernel is located, how it should be loaded into memory, and how it should be started.

If all goes well, the BIOS looks for a bootable volume such as a floppy disk, CD-ROM, hard drive, RAM disk, or other media. The bootable volume contains a special hexadecimal value written to the volume by the boot loader application (likely either GRUB or LILO, although LILO is not provided with Ubuntu) when the boot loader code was first installed in the system's drives. The BIOS searches volumes in the order established by the BIOS settings (for example, the floppy first, followed by a CD-ROM, and then a hard drive) and then boots from the first bootable volume it finds. Modern BIOS's allow considerable flexibility in choosing the device used for booting the system.

NOTE

If the BIOS detects a hardware problem, the boot process will fail and the BIOS will generate a few beeps from the system speaker. These "beep codes" indicate the nature of the problem the BIOS has encountered. The codes vary among manufacturers, and the diagnosis of problems occurring during this phase of the boot process is beyond the scope of this book and does not involve Linux. If you encounter a problem, you should consult the motherboard manual or contact the manufacturer of the motherboard.

Next, the BIOS looks on the bootable volume for boot code in the partition boot sector also known as the Master Boot Record (MBR) of the first hard disk. The MBR contains the boot loader code and the partition table—think of it as an index for a book, plus a few comments on how to start reading the book. If the BIOS finds a boot loader, it loads the boot loader code into memory. At that point, the BIOS's job is completed, and it passes control of the system to the boot loader.

The boot loader locates the Linux kernel on the disk and loads it into memory. After that task is completed, the boot loader passes control of the system to the Linux kernel. You can see how one process builds on another in an approach that enables many different operating systems to work with the same hardware.

Ubuntu can use a variety of boot loaders, including GRUB (the default for Ubuntu), LILO (a long-time standard but not available with Ubuntu), BootMagic (a commercial program), and others.

NOTE

Linux is very flexible and can be booted from multiple images on a CD-ROM, over a network using PXE (pronounced "pixie") or NetBoot, or on a headless server with the console display sent over a serial or network connection. Work is even underway to create a special Linux BIOS at http://www.linuxbios.org/ that will expedite the boot process because Linux does not need many of the services offered by the typical BIOS.

This kind of flexibility enables Linux to be used in a variety of ways, such as remote servers or diskless workstations, which are not generally seen in personal home use.

Loading the Linux Kernel

In a general sense, the kernel manages the system resources. As the user, you do not often interact with the kernel, but instead just the applications that you are using. Linux refers to each application as a process, and the kernel assigns each process a number called a *process ID (PID)*. First, the Linux kernel loads and runs a process named `init`, which is also known as the "father of all processes" because it starts every subsequent process.

NOTE

Details about the sequence of events that occur when the Linux kernel is loaded can be found in the file `/usr/src/linux-2.6/init/main.c` if you have installed the Linux kernel documentation.

This next step of the boot process begins with a message that the Linux kernel is loading, and a series of messages will be printed to the screen, giving you the status of each command. A failure should display an error message. The `-quiet` option may be passed to the kernel at boot time to suppress many of these messages.

If the boot process were halted at this point, the system would just sit idle and the screen would be blank. In order to make the system useful for users, we need to start the system services. Those services are some of the applications that allow us to interact with the system.

System Services and Runlevels

The `init` command boots Ubuntu to a specific system state, commonly referred to as its runlevel.

Runlevels determine which of the many available system services are started, as well as in which order they start. A special runlevel is used to stop the system, and a special runlevel is used for system maintenance. As you will see, there are other runlevels for special purposes.

You will use runlevels to manage the system services running on your computer. All these special files and scripts are set up during your installation of Ubuntu Linux, and they receive their initial values based on your choices during the installation—as described in Chapter 3, "Installing Ubuntu," and Chapter 4, "Post-Installation Configuration." You can change and control them manually, as you learn later in this chapter using tools of varying sophistication.

Runlevel Definitions

The Ubuntu runlevels are defined for the Ubuntu system in `/etc/init.d`.

Each runlevel tells the `init` command what services to start or stop. Although runlevels might all have custom definitions, Ubuntu has adopted some standards for runlevels:

- **Runlevel 0**—Known as "halt," this runlevel is used to shut down the system.
- **Runlevel 1**—This is a special runlevel, defined as "single," which boots Ubuntu to a root access shell prompt where only the root user may log in. It has networking, X, and multi-user access turned off. This is the maintenance or rescue mode. It allows the system administrator to perform work on the system, make backups, or repair configuration or other files.
- **Runlevel 2**—This is the default runlevel for Ubuntu.

- **Runlevels 3–5**—These runlevels aren't used in Ubuntu but are often used in other Linux distributions.
- **Runlevel 6**—This runlevel is used to reboot the system.

Runlevel 1 (also known as single-user mode or maintenance mode) is most commonly used to repair file systems and change the root password on a system when the password has been forgotten. Trespassers with physical access to the machine can also use runlevel 1 to access your system.

CAUTION

Never forget that uncontrolled physical access is virtually a guarantee of access to your data by an intruder.

Booting into the Default Runlevel

Ubuntu boots into runlevel 2 by default, which means it starts the system as normal and leaves you inside the X Window System looking at the Gnome login prompt. It knows what runlevel 2 needs to load by looking in the `rc*.d` directories in `/etc`. Ubuntu contains directories for rc0.d through to rc5.d and rcS.d.

Assuming that the value is `1`, the `rc` script then executes all the scripts under the `/etc/rc.1` directory and then launches the graphical login.

If Ubuntu is booted to runlevel 1, for example, scripts beginning with the letter K followed by scripts beginning with the letter S under the `/etc/rc1.d` directory are then executed:

```
# ls /etc/rc1.d/
K01gdm          K19hplip       K20laptop-mode    K20vsftpd    K80slapd
K01usplash          K20acpi-support       K20makedev    k21acpid    K86ppp
...etc...
K19cupsys    K20inetutils-inetd    K20ssh    K74-bluez-utils    S20single
```

These scripts, as with all scripts in the `rc*.d` directories, are actually symbolic links to system service scripts under the `/etc/init.d` directory.

The `rc1.d` links are prefaced with a letter and number, such as K15 or S10. The (K) or (S) in these prefixes indicate whether or not a particular service should be killed (K) or started (S) and pass a value of stop or start to the appropriate `/etc/init.d` script. The number in the prefix executes the specific `/etc/init.d` script in a particular order. The symlinks have numbers to delineate the order in which they are started. Nothing is sacred about a specific number, but some services need to be running before others are started. You would not want your Ubuntu system to attempt, for example, to mount a remote Network File System (NFS) volume without first starting networking and NFS services.

Booting to a Non-Default Runlevel with GRUB

There might come a time when you do not want to boot into the default runlevel, such as when you want to repair the `X` server or install a new graphics driver. You'll need to follow several specific steps to boot to a non-default runlevel if you use the default boot loader for Ubuntu, GRUB.

NOTE

If you have enabled a GRUB password, you must first press **p**, type your password, and then press Enter before using this boot method.

The GRUB boot loader passes arguments, or commands, to the kernel at boot time. These arguments are used, among other things, to tell GRUB where the kernel is located and also to pass specific parameters to the kernel, such as how much memory is available or how special hardware should be configured.

To override the default runlevel, you can add an additional kernel argument to GRUB as follows:

1. At the graphical boot screen, press **e** (for edit), scroll down to select the kernel, and press **e** again.
2. Press the spacebar, type **single** or **1** (Ubuntu allows **S** and **s** as well), and press Enter.
3. Finally, press **b** to boot, and you'll boot into runlevel `1` instead of the default runlevel listed in `/etc/inittab`.

Understanding `init` Scripts and the Final Stage of Initialization

Each `/etc/init.d` script, or `init` script, contains logic that determines what to do when receiving a start or stop value. The logic might be a simple switch statement for execution, as in this example:

```
case "$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;
  restart)
        restart
        ;;
  reload)
        reload
        ;;
```

```
  status)
        rhstatus
        ;;
  condrestart)
        [ -f /var/lock/subsys/smb ] && restart || :
        ;;
  *)
        echo $"Usage: $0 {start|stop|restart|status|condrestart}"
        exit 1
esac
```

Although the scripts can be used to customize the way that the system runs from power-on, absent the replacement of the kernel, this script approach also means that the system does not have to be halted in total to start, stop, upgrade, or install new services.

Note that not all scripts will use this approach, and that other messages might be passed to the service script, such as restart, reload, or status. Also, not all scripts will respond to the same set of messages (with the exception of start and stop, which they all have to accept by convention) because each service might require special commands.

After all the system scripts have been run, your system is configured and all the necessary system services have been started. If you are using a runlevel other than 5, the final act of the `init` process is to launch the user shell—`bash`, `tcsh`, `zsh`, or any of the many command shells available. The shell launches and you see a login prompt on the screen.

Controlling Services at Boot with Administrative Tools

In the Services dialog (shown in Figure 11.1) Ubuntu lists all the services that you can have automatically start at boot time. They are usually all enabled by default, but you can simply uncheck the ones you don't want and click OK. It is not recommended that you disable services randomly "to make things go faster." Some services might be vital for the continuing operation of your computer, such as the graphical login manager and the system communication bus.

Changing Runlevels

After making changes to system services and runlevels, you can use the `telinit` command to change runlevels on-the-fly on a running Ubuntu system. Changing runlevels this way allows system administrators to alter selected parts of a running system in order to make changes to the services or to put changes into effect that have already been made (such as reassignment of network addresses for a networking interface).

For example, a system administrator can quickly change the system to maintenance or single-user mode by using the `telinit` command with its **`s`** option like this:

```
# telinit S
```

FIGURE 11.1 You can enable and disable Ubuntu's boot-up services by toggling the check-boxes in the Services dialog.

The `telinit` command uses the `init` command to change runlevels and shut down currently running services.

After booting to single-user mode, you can then return to multi-user mode, like this:

```
# telinit 2
```

TIP

Linux is full of shortcuts: If you exit the single-user shell by typing **`exit`** at the prompt, you will go back to the default runlevel without worrying about using `telinit`.

Troubleshooting Runlevel Problems

Reordering or changing system services during a particular runlevel is rarely necessary when using Ubuntu unless some disaster occurs. But system administrators should have a basic understanding of how Linux boots and how services are controlled in order to perform troubleshooting or to diagnose problems. By using additional utilities such as the **`dmesg ¦ less`** command to read kernel output after booting or by examining system logging with **`cat /var/log/messages ¦ less`**, it is possible to gain a bit more detail about what is going on when faced with troublesome drivers or service failure.

To better understand how to troubleshoot service problems in Ubuntu, look at the diagnosis and resolution of a typical service-related issue.

In this example, X will not start: You don't see a desktop displayed, nor does the computer seem to respond to keyboard input. The X server might either be hung in a loop, repeatedly failing, or might exit to a shell prompt with or without an error message.

The X server only attempts to restart itself in runlevel 2, so to determine whether the X server is hung in a loop, try switching to runlevel 1.

TIP

If you are working on a multi-user system and might inadvertently interrupt the work of other users, ask them to save their current work; then change to a safer runlevel, such as single user mode.

Change to runlevel 1 by running the command **`telinit 1`**. This switch to runlevel 1 will stop the X server from attempting to restart itself.

Now you can easily examine the error and attempt to fix it.

First, try to start the X server "naked" (without also launching the window manager). If you are successful, you will get a gray screen with a large X in the middle. If so, kill X with the Ctrl+Alt+Backspace key combination, and look at your window manager configuration. (This configuration varies according to which window manager you have chosen.)

Let us assume that X won't run "naked." If we look at the log file for Xorg (it's clearly identified in the `/var/log` directory), we'll pay attention to any line that begins with `(EE)`, the special error code. We can also examine the error log file, `.xsessions-error`, in our home directory if such a file exists.

If we find an error line, the cause of the error might or might not be apparent to us. The nice thing about the Linux community is that it is very unlikely that you are the first person to experience that error. Enter the error message (or better, a unique part of it) into http://www.google.com/linux and discover what others have had to say about the problem. You might need to adjust your search to yield usable results, but that level of detail is beyond the scope of this chapter. Make adjustments and retest as before until you achieve success.

Fix the X configuration and start X with **`startx`**. Repeat as necessary.

CAUTION

Before making any changes to any configuration file, always make a backup copy of the original, unmodified file. Our practice is to append the extension `.original` to the copy because that is a unique and unambiguous identifier.

If you need to restore the original configuration file, do not rename it, but copy it back to its original name.

Starting and Stopping Services Manually

If you change a configuration file for a system service, it is usually necessary to stop and restart the service to make it read the new configuration. If you are reconfiguring the `X` server, it is often convenient to change from runlevel 2 to runlevel 1 to make testing easier and then switch back to runlevel 2 to re-enable the graphical login. If a service is improperly configured, it is easier to stop and restart it until you have it configured correctly than it is to reboot the entire machine.

The traditional way to manage a service (as root) is to call the service's `/etc/init.d` name on the command line with an appropriate keyword, such as `start`, `status`, or `stop`. For example, to start the Apache web server, call the `/etc/init.d/apache2` script like this:

```
sudo /etc/init.d/apache2 start
Starting apache 2.2 web server                                    [  OK  ]
```

The script will execute the proper program(s) and report the status of it. Stopping services is equally easy, using the `stop` keyword.

Scheduling Tasks

There are three ways to schedule commands in Ubuntu, all of which work in different ways. The first is the `at` command, which specifies a command to run at a specific time and date relative to today. The second is the `batch` command, which is actually a script that redirects you to the `at` command with some extra options set so your command runs when the system is quiet. The last option is the `cron` daemon, which is the Linux way of executing tasks at a given time.

Using `at` and `batch` to Schedule Tasks for Later

If there is a time-intensive task you want to run, but you do not want to do it while you are still logged in, you can tell Ubuntu to run it later with the `at` command. To use `at`, you need to tell it the time at which you want to run and then press Enter. You will then see a new prompt that starts with `at>`, and everything you type there until you press `Ctrl+D` will be the commands you want `at` to run.

When the designated time arrives, `at` will perform each action individually and in order, which means later commands can rely on the results of earlier commands. In this next example, run at just after 5 p.m., `at` is used to download and extract the latest Linux kernel at a time when the network should be quiet:

```
[paul@caitlin ~]$ at now + 7 hours
at> wget http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.10.tar.bz2
at> tar xvfjp linux-2.6.10.tar.bz2
at> <EOT>
job 2 at 2005-01-09 17:01
```

Specifying `now + 7 hours` as the time does what you would expect: `at` was run at 5 p.m., so the command will run just after midnight that night. When your job has finished, `at` will send you an email with a full log of your job's output; type `mail` at the console to bring up your mailbox and then press the relevant number to read `at`'s mail.

If you have a more complex job, you can use the `-f` parameter to have `at` read its commands from a file, like this:

```
echo wget http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.10.tar.bz2\;
tar xvfjp linux-2.6.10.tar.bz2 > myjob.job
at -f myjob.job tomorrow
```

As you can see, `at` is flexible about the time format it takes; you can specify it in three ways:

- Using the `now` parameter, you can specify how many minutes, hours, days, or weeks relative to the current time—for example, `now + 4 weeks` would run the command one month from today.
- You can also specify several special times, including `tomorrow`, `midnight`, `noon`, or `teatime` (4 p.m.). If you do not specify a time with `tomorrow`, your job is set for precisely 24 hours from the current time.
- You can specify an exact date and time using `HH:MM MM/DD/YY` format—for example, 16:40 22/12/05 for 4:40 p.m. on the 22nd of December 2005.

When your job is submitted, `at` will report the job number, date, and time that the job will be executed; the queue identifier; plus the job owner (you). It will also capture all your environment variables and store them along with the job so that, when your job runs, it can restore the variables, preserving your execution environment.

The job number and job queue identifier are both important. When you schedule a job using `at`, it is placed into queue "a" by default, which means it runs at your specified time and takes up a normal amount of resources.

There is an alternative command, `batch`, which is really just a shell script that calls `at` with a few extra options. These options (`-q b -m now`, if you were interested) set `at` to run on queue b (`-q b`), mailing the user on completion (`-m`), and running immediately (`now`). The queue part is what is important: Jobs scheduled on queue b will only be executed when system load falls below 0.8—that is, when the system is not running at full load. Furthermore, it will run with a lower niceness, meaning a queue jobs usually have a niceness of 2, whereas b queue jobs have a niceness of 4.

Because `batch` always specifies `now` as its time, you need not specify your own time; it will simply run as soon as the system is quiet. Having a default niceness of 4 means that batched commands will get less system resources than a queue job's (`at`'s default) and less system resources than most other programs. You can optionally specify other queues using `at`. Queue c runs at niceness 6, queue d runs at niceness 8, and so on. However, it is important to note that the system load is only checked before the command is run. If the

load is lower than 0.8, your batch job will be run. If the system load subsequently rises beyond 0.8, your batch job will continue to run, albeit in the background, thanks to its niceness value.

When you submit a job for execution, you will also be returned a job number. If you forget this or just want to see a list of other jobs you have scheduled to run later, use the `atq` command with no parameters. If you run this as a normal user, it will print only your jobs; running it as a superuser will print everyone's jobs. The output is in the same format as when you submit a job, so you get the ID number, execution time, queue ID, and owner of each job.

If you want to delete a job, use the `atrm` command followed by the ID number of the job you want to delete. This next example shows `atq` and `atrm` being used to list jobs and delete one:

```
[paul@caitlin ~]$ atq
14      2005-01-20 23:33 a paul
16      2005-02-03 22:34 a paul
17      2005-01-25 22:34 a paul
15      2005-01-22 04:34 a paul
18      2005-01-22 01:35 b paul
[paul@caitlin ~]$ atrm 16
[paul@caitlin ~]$ atq
14      2005-01-20 23:33 a paul
17      2005-01-25 22:34 a paul
15      2005-01-22 04:34 a paul
18      2005-01-22 01:35 b paul
```

In that example, job 16 is deleted using `atrm`, and so it does not show up in the second call to `atq`.

The default configuration for `at` and `batch` is to allow everyone to use it, which is not always the desired behavior. Access is controlled through two files: `/etc/at.allow` and `/etc/at.deny`. By default, `at.deny` exists but is empty, which allows everyone to use `at` and `batch`. You can enter usernames into `at.deny`, one per line, to stop those users scheduling jobs.

Alternatively, you can use the `at.allow` file; this does not exist by default. If you have a blank `at.allow` file, no one except root is allowed to schedule jobs. As with `at.deny`, you can add usernames to `at.allow` one per line, and those users will be able to schedule jobs. You should use either `at.deny` or `at.allow`: When someone tries to run `at` or `batch`, Ubuntu checks for her username in `at.allow`. If it is in there, or if `at.allow` does not exist, Ubuntu checks for her username in `at.deny`. If her username is in `at.deny` or `at.deny` does not exist, she is not allowed to schedule jobs.

Using `cron` to Run Jobs Repeatedly

The `at` and `batch` commands work well if you just want to execute a single task at a later date, but they are less useful if you want to run a task frequently. Instead, there is the `crond` daemon for running tasks repeatedly based upon system—and user—requests. Cron has a similar permissions system to `at`: Users listed in the `cron.deny` file are not allowed to use Cron, and users listed in the `cron.allow` file are. An empty `cron.deny` file—the default—means everyone can set jobs. An empty `cron.allow` file means that no one (except root) can set jobs.

There are two types of jobs: system jobs and user jobs. Only root can edit *system* jobs, whereas any user whose name appears in `cron.allow` or does not appear in `cron.deny` can run *user* jobs. System jobs are controlled through the `/etc/crontab` file, which by default looks like this:

```
SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user command
17 * * * * root run-parts --report /etc/cron.hourly
25 6 * * * root test -x /usr/sbin/anacron ¦¦ run-parts --report /etc/cron.daily
47 6 * * 7 root test -x /usr/sbin/anacron ¦¦ run-parts --report /etc/cron.weekly

52 6 1 * * root test -x /usr/sbin/anacron ¦¦ run-parts --report /etc/cron.monthly
```

The first two lines specify which shell should be used to execute the job (defaults to the shell of the user who owns the `crontab` file, usually `/bin/bash`), and the search path for executables that will be used. It's important that you avoid using environment variables in this path statement, as they may not be set when the job runs.

The next line starts with a pound sign (`#`) and so is treated as a comment and ignored. The next four lines are the important parts: They are the jobs themselves.

Each job is specified in seven fields that define the time to run, owner, and command. The first five commands specify the execution time in quite a quirky order: minute (0–59), hour (0–23), day of the month (1–31), month of the year (1–12), and day of the week (0–7). For day of the week, both 0 and 7 are Sunday, which means that 1 is Monday, 3 is Wednesday, and so on. If you want to specify "all values" (that is, every minute, every hour, every day, and so on), use an asterisk, `*`.

The next field specifies the username of the owner of the job. When a job is executed, it uses the username specified here. The last field is the command to execute.

So, the first job runs at minute 17, every hour of every day of every month and executes the command `run-parts /etc/cron.hourly`. The `run-parts` command is a simple script that runs all programs inside a given directory—in this case, `/etc/cron.hourly`. So, in this case, the job will execute at 00:17 (17 minutes past midnight), 01:17, 02:17, 03:17, and so on, and will use all the programs listed in the `cron.hourly` directory.

The next job runs at minute 25 and hour 6 of every day of every month, running `run-parts /etc/cron.daily`. Because of the hour limitation, this script will run only once per day, at 6:25 a.m. Note that it uses minute 25 rather than minute 17 so that daily jobs do not clash with hourly jobs. You should be able to guess what the next two jobs do, simply by looking at the commands they run!

Inside each of those four directories (`cron.hourly`, `cron.daily`, `cron.weekly`, and `cron.monthly`) are a collection of shell scripts that will be run by `run-parts`. For example, in `cron.daily` you will have scripts like `logrotate`, which handles backing up of log files, and `makewhatis`, which updates the `whatis` database. You can add other system tasks to these directories if you want to, but you should be careful to ensure your scripts are correct.

CAUTION

The `cron` daemon reads all the system `crontab` files and all user `crontab` files once a minute (on the minute, i.e. at 6:00:00, 6:01:00, and so on) to check for changes. However, any new jobs it finds will not be executed until at least 1 minute has passed.

For example, if it is 6:01:49 (that is, 49 seconds past 1 minute past 6 a.m.) and you set a `cron` job to run at 6:02, it will not execute. At 6:02, the `cron` daemon will reread its configuration files and see the new job, but it will not be able to execute it. If you set the job to run at 6:02 a.m. every day, it will be executed the following morning and every subsequent morning.

This same situation exists when deleting jobs. If it is 6:01:49 and you have a job scheduled to run at 6:02, deleting it will make no difference: `cron` will run it before it rereads the `crontab` files for changes. However, after it has reread the `crontab` file and noticed the job is no longer there, it will not be executed in subsequent days.

There are alternative ways of specifying dates. For example, you can use sets of dates and times by using hyphens of commas, such as hours 9–15 would execute at 9, 10, 11, 12, 13, 14, and 15 (from 9 a.m. to 3 p.m.), whereas 9,11,13,15 would miss out at the even hours. Note that it is important you do not put spaces into these sets because the `cron` daemon will interpret them as the next field. You can define a step value with a slash (`/`) to show time division: `*/4` for hours means "every four hours all day", and `0-12/3` means "every three hours from midnight to noon." You can also specify day and month names rather than numbers, using three-character abbreviations: Sun, Mon, Tue, Fri, Sat for days, or Jan, Feb, Mar, Oct, Nov, Dec for months.

As well as system jobs, there are also user jobs for those users who have the correct permissions. User jobs are stored in the `/var/spool/cron` directory, with each user having his own file in his named after his username—for instance, `/var/spool/cron/paul` or `/var/spool/cron/root`. The contents of these files contain the jobs the user wants to run and take roughly the same format as the `/etc/crontab file`, with the exception that the owner of the job should not be specified because it will always be the same as the filename.

To edit your own `crontab` file, type **`crontab -e`**. This brings up a text editor (`vim` by default, but you can set the `EDITOR` environment variable to change that) where you can enter your entries. The format of this file is a little different from the format for the main `crontab` because this time there is no need to specify the owner of the job—it is always you.

So, this time each line is made up of six fields: minute (0–59), hour (0–23), day of the month (1–31), month of the year (1–12), day of the week (0–7), and then the command to run. If you are using `vim` and are new to it, press `i` to enter insert mode to edit your text; then press Esc to exit insert mode. To save and quit, type a colon followed by `wq` and press Enter.

When programming, we tend to use a `sandbox` subdirectory in our home directory where we keep all sorts of temporary files that we were just playing around with. We can use a personal job to empty that directory every morning at 6 a.m. so that we get a fresh start each morning. Here is how that would like in our `crontab` file:

```
0 6 * * * rm -rf /home/paul/sandbox/*
```

If you are not allowed to schedule jobs, you will be stopped from editing your `crontab` file.

Once your jobs are placed, you can use the command `crontab -l` to list your jobs. This just prints the contents of your `crontab` file, so its output will be the same as the line you just entered.

If you want to remove just one job, the easiest thing to do is type `crontab -e` to edit your `crontab` file in `vim`; then, after having moved the cursor to the job you want to delete, type `dd` (two *d*s) to delete that line. If you want to delete all your jobs, you can use `crontab -r` to delete your `crontab` file.

Basic Shell Control

Ubuntu includes a rich assortment of capable, flexible, and powerful shells. Each shell is different but has numerous built-in commands and configurable command-line prompts and might include features such as command-line history, the ability to recall and use a previous command line, and command-line editing. As an example, the `bash` shell is so powerful that it is possible to write a minimal web server entirely in `bash`'s language using 114 lines of script (see the link at the end of this chapter).

Although there are many shells to choose from, most people stick with the default, `bash`. This is because `bash` does everything most people need to do, and more. Only change your shell if you really need to.

Table 11.1 lists each shell, along with its description and location, in your Ubuntu file system.

TABLE 11.1 Shells with Ubuntu

Name	Description	Location
`bash`	The Bourne Again SHell	`/bin/bash`
`ksh`	The Korn shell	`/bin/ksh, /usr/bin/ksh`
`pdksh`	A symbolic link to `ksh`	`/usr/bin/pdksh`
`rsh`	The restricted shell (for network operation)	`/usr/bin/rsh`
`sh`	A symbolic link to `bash`	`/bin/sh`
`tcsh`	A `csh`-compatible shell	`/bin/tcsh`
`zsh`	A compatible `csh`, `ksh`, and `sh` shell	`/bin/zsh`

Learning More About Your Shell

All the shells listed in Table 11.1 have accompanying `man` pages, along with other documentation under the `/usr/share/doc` directory. Some of the documentation can be quite lengthy, but it is generally much better to have too much documentation than too little! The `bash` shell includes more than 100 pages in its manual, and the `zsh` shell documentation is so extensive that it includes the `zshall` meta `man` page (use `man zshall` to read this overview)!

The Shell Command Line

Having a basic understanding of the capabilities of the shell command line can help you write better shell scripts. If, once you have finished reading this short introduction, you want to learn more about the command line, check out Chapter 33, "Command Line Masterclass." You can use the shell command line to perform a number of different tasks, including

- Searching files or directories with programs using pattern-matching, or *expressions*; these commands include the GNU `gawk` (linked as `awk`) and the `grep` family of commands, including `egrep` and `fgrep`.
- Getting data from and sending data to a file or command, known as *input* and *output redirection*.
- Feeding or filtering a program's output to another command (called using *pipes*).

A shell can also have built-in *job-control* commands to launch the command line as a background process, suspend a running program, selectively retrieve or kill running or suspended programs, and perform other types of process control.

Multiple commands can be run on a single command line using a semicolon to separate commands:

```
$ w ; free ; df
  6:02pm  up 4 days, 24 min,  1 user,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU  WHAT
bball    pts/0    shuttle.home.org  1:14pm  0.00s  0.57s  0.01s  w
total       used       free     shared    buffers     cached
Mem:        190684     184420       6264         76      17620     142820
-/+ buffers/cache:      23980     166704
Swap:      1277156       2516    1274640
Filesystem           1k-blocks      Used Available Use% Mounted on
/dev/hda1             11788296   4478228   6711248  41% /
none                     95340         0     95340   0% /dev/shm
```

This example displays the output of the `w`, `free`, and `df` commands. Long shell command lines can be extended inside shell scripts or at the command line by using the backslash character (`\`). For example,

```
$ echo ""this is a long \
> command line and"" ; echo ""shows that multiple commands \
> may be strung out.""
this is a long command line and
shows that multiple commands may be strung out.
```

The first three lines of this example are a single command line. In that single line are two instances of the `echo` command. Note that when you use the backslash as a line-continuation character, it must be the last character on the command line (or in your shell script, as you will see later on in this chapter).

Using the basic features of the shell command line is easy, but mastering use of all features can be difficult. Entire books have been devoted to using shells, writing shell scripts, and using pattern-matching expressions. The following sections provide an overview of some features of the shell command line relating to writing scripts.

Grokking `grep`

If you plan to develop shell scripts to expand the capabilities of pattern-matching commands such as `grep`, you will benefit from learning more about using expressions. One of the definitive guides to using the pattern-matching capabilities of Unix and Linux commands is *Mastering Regular Expressions* by Jeffrey E. F. Freidl (O'Reilly), ISBN: 0-596-52812-4.

Shell Pattern-Matching Support

The shell command line allows you to use strings of specially constructed character patterns for wildcard matches. This is a different simpler capability than that supported by GNU utilities such as `grep`, which can use more complex patterns, known as *expressions*, to search through files or directories or to filter data input to or out of commands.

The shell's pattern strings can be simple or complex, but even using a small subset of the available characters in simple wildcards can yield constructive results at the command line. Some common characters used for shell pattern matching are

- `*`—Matches any character. For example, to find all files in the current directory ending in `.txt`, you could use

  ```
  $ ls *.txt
  ```

- `?`—Matches a single character. For example, to find all files in the current directory ending in the extension `.d?c` (where `?` could be 0–9, a–z, or A–Z),

  ```
  $ ls *.d?c
  ```

- `[xxx]` or `[x-x]`—Matches a range of characters. For example, to list all files in a directory with names containing numbers,

  ```
  $ ls *[0-9]*
  ```

- `\x`—Matches or escapes a character such as `?` or a tab character. For example, to create a file with a name containing question mark,

  ```
  $ touch foo\?
  ```

Note that the shell might not interpret some characters or regular expressions in the same manner as a Linux command, and mixing wildcards and regular expressions in shell scripts can lead to problems unless you're careful. For example, finding patterns in text is best left to regular expressions used with commands such as `grep`; simple wildcards should be used for filtering or matching filenames on the command line. And although both Linux command expressions and shell scripts can recognize the backslash as an escape character in patterns, the dollar sign (`$`) will have two wildly different meanings (single-character pattern matching in expressions and variable assignment in scripts).

CAUTION

Make sure you read your command carefully when using wildcards; an all-too-common error is to type something like `rm –rf * .txt` with a space between the `*` and the `.txt`. By the time you wonder why the command is taking so long, Bash will already have deleted most of your files. The problem is that it will treat the `*` and the `.txt` separately. `*` will match everything, so Bash will delete all your files.

Redirecting Input and Output

You can create, overwrite, and append data to files at the command line, using a process called *input* and *output redirection*. The shell recognizes several special characters for this process, such as >, <, or >>.

In this example, the output of the `ls` command is redirected to create a file named `textfiles.listing`:

```
$ ls *.txt >textfiles.listing
```

Use output redirection with care because it is possible to overwrite existing files. For example, specifying a different directory but using the same output filename will overwrite the existing `textfiles.listing`:

```
$ ls /usr/share/doc/mutt-1.4/*.txt >textfiles.listing
```

Fortunately, most shells are smart enough to recognize when you might do something foolish. Here, the `bash` shell warns that the command is attempting to redirect output to a directory:

```
$ mkdir foo
$ ls >foo
bash: foo: Is a directory
```

Output can be appended to a file without overwriting existing content by using the append operator, >>. In this example, the directory listing will be appended to the end of `textfiles.listing` instead of overwriting its contents:

```
$ ls /usr/share/doc/mutt-1.4/*.txt >>textfiles.listing
```

You can use *input redirection* to feed data into a command by using the < like this:

```
$ cat < textfiles.listing
```

You can use the shell *here* operator, <<, to specify the end of input on the shell command line:

```
$ cat >simple_script <<DONE
> echo ""this is a simple script""
> DONE
$ cat simple_script
echo ""this is a simple script""
```

In this example, the shell will feed the `cat` command you are typing (input) until the pattern `DONE` is recognized. The output file `simple_script` is then saved and its contents verified. This same technique can be used in scripts to create content based on the output of various commands and define an end-of-input or delimiter.

Piping Data

Many Linux commands can be used in concert in a single, connected command line to transform data from one form to another. Stringing Linux commands together in this fashion is known as using or creating *pipes*. Pipes are created on the command line with the bar operator (¦). For example, a pipe can be used to perform a complex task from a single command line like this:

```
 $ find /d2 -name '*.txt' -print ¦ xargs cat ¦ \
tr ' ' '\n' ¦  sort ¦ uniq >output.txt
```

This example takes the output of the `find` command to feed the `cat` command (via `xargs`) the name all text files under the `/d2` command. The content of all matching files is then fed through the `tr` command to change each space in the data stream into a carriage return. The stream of words is then sorted, and identical adjacent lines are removed using the `uniq` command. The output, a raw list of words, is then saved in the file named `output.txt`.

Background Processing

The shell allows you to start a command and then launch it into the background as a process by using an ampersand (`&`) at the end of a command line. This technique is often used at the command line of an X terminal window to start a client and return to the command line. For example, to launch another terminal window using the `xterm` client,

```
$ xterm &
[3] 1437
```

The numbers echoed back show a number (`3` in this example), which is a *job* number, or reference number for a shell process, and a *Process ID* number, or PID (1437 in this example). The `xterm` window session can be killed by using the shell's built-in `kill` command, along with the job number like this:

```
$ kill %3
```

Or the process can be killed by using the `kill` command, along with the PID, like so:

```
$ kill 1437
```

Background processing can be used in shell scripts to start commands that take a long time, such as backups:

```
# tar -czf /backup/home.tgz /home &
```

Writing and Executing a Shell Script

Why should you write and use shell scripts? Shell scripts can save you time and typing, especially if you routinely use the same command lines multiple times every day.

Although you could also use the history function (press the Up or Down keys while using `bash` or use the `history` command), a shell script can add flexibility with command-line argument substitution and built-in help.

Although a shell script won't execute faster than a program written in a computer language such as C, a shell program can be smaller in size than a compiled program. The shell program does not require any additional library support other than the shell or, if used, existing commands installed on your system. The process of creating and testing shell scripts is also generally simpler and faster than the development process for equivalent C language commands.

NOTE

Hundreds of commands included with Ubuntu are actually shell scripts, and many other good shell script examples are available over the Internet—a quick search will yield numerous links to online tutorials and scripting guides from fellow Linux users and developers. For example, the `startx` command, used to start an X Window session from the text console, is a shell script used every day by most users. To learn more about shell scripting with `bash`, see the Advanced Bash-Scripting Guide, listed in the "Reference" section at the end of this chapter. You'll also find *Sams Teach Yourself Shell Programming in 24 Hours* a helpful guide to learning more about using the shell to build your own commands.

When you are learning to write and execute your first shell scripts, start with scripts for simple, but useful tasks. Begin with short examples, and then expand the scripts as you build on your experience and knowledge. Make liberal use of comments (lines preceded with a pound # sign) to document each section of your script. Include an author statement and overview of the script as additional help, along with a creation date or version number. Write shell scripts using a text editor such as `vi` because it does not automatically wrap lines of text. Line wrapping can break script syntax and cause problems. If you use the `nano` editor, include its `-w` flag to disable line wrap.

In this section, you learn how to write a simple shell script to set up a number of *aliases* (command synonyms) whenever you log on. Instead of typing all the aliases every time you log on, you can put them in a file by using a text editor, such as `vi`, and then execute the file. Normally these changes are saved in systemwide shell configuration files under the `/etc` directory to make the changes active for all users or in your `.bashrc`, `.cshrc` (if you use `tcsh`), or `.bash_profile` files in your home directory.

Here is what is contained in `myenv`, a sample shell script created for this purpose (for `bash`):

```
#!/bin/sh
alias ll='ls -l'
alias ldir='ls -aF'
alias copy='cp'
```

This simple script creates command *aliases*, or convenient shorthand forms of commands, for the `ls` and `cp` commands. The `ll` alias provides a long directory listing: The `ldir` alias is the `ls` command, but prints indicators (for directories or executable files) in listings. The `copy` alias is the same as the `cp` command. You can experiment and add your own options or create aliases of other commands with options you frequently use.

You can execute `myenv` in a variety of ways under Linux. As shown in this example, you can make `myenv` executable by using the `chmod` command and then execute it as you would any other native Linux command:

```
$ chmod +x myenv
```

This line turns on the executable permission of `myenv`, which can be checked with the `ls` command and its `-l` option like this:

```
$ ls -l myenv
-rwxrwxr-x    1 winky    winky          11 Aug 26 17:38 myenv
```

Running the New Shell Program

You can run your new shell program in several ways. Each method will produce the same results, which is a testament to the flexibility of using the shell with Linux. One way to run your shell program is to execute the file `myenv` from the command line as if it were a Linux command:

```
$ ./myenv
```

A second way to execute `myenv` under a particular shell, such as `pdksh`, is as follows:

```
$ pdksh myenv
```

This invokes a new `pdksh` shell and passes the filename `myenv` as a parameter to execute the file. A third way will require you to create a directory named `bin` in your home directory, and to then copy the new shell program into this directory. You can then run the program without the need to specify a specific location or to use a shell. You do this like so:

```
$ mkdir bin
$ mv myenv bin
$ myenv
```

This works because Ubuntu is set up by default to include the executable path `$HOME/bin` in your shell's environment. You can view this environment variable, named `PATH`, by piping the output of the `env` command through `fgrep` like so:

```
$ env ¦ fgrep PATH
/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin: \
/usr/X11R6/bin:/sbin:/home/paul/bin
```

As you can see, the user (paul in this example) can use the new bin directory to hold executable files. Another way to bring up an environment variable is to use the echo command along with the variable name (in this case, $PATH):

```
$ echo $PATH
/usr/kerberos/bin:/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:/home/bball/bin
```

CAUTION

Never put . in your $PATH in order to execute files or a command in the current directory—this presents a serious security risk, especially for the root operator, and even more so if . is first in your $PATH search order. Trojan scripts placed by crackers in directories such as /tmp can be used for malicious purposes, and will be executed immediately if the current working directory is part of your $PATH.

Storing Shell Scripts for Systemwide Access

After you execute the command myenv, you should be able to use ldir from the command line to get a list of files under the current directory and ll to get a list of files with attributes displayed. However, the best way to use the new commands in myenv is to put them into your shell's login or profile file. For Ubuntu, and nearly all Linux users, the default shell is bash, so you can make these commands available for everyone on your system by putting them in the /etc/bashrc file. Systemwide aliases for tcsh are contained in files with the extension .csh under the /etc/profile.d directory. The pdksh shell can use these command aliases as well.

NOTE

To use a shell other than bash after logging in, use the chsh command from the command line or the system-config-users client during an X session. You'll be asked for your password (or the root password if using system-config-users), as well as the location and name of the new shell (refer to Table 11.1). The new shell will become your default shell, but only if its name is in the list of acceptable system shells in /etc/shells.

Interpreting Shell Scripts Through Specific Shells

The majority of shell scripts use a *shebang line* (#!) at the beginning to control the type of shell used to run the script; this bang line calls for an sh-incantation of bash:

```
#!/bin/sh
```

A shebang line (it is short for "sharp" and "bang", two names for # and !) tells the Linux kernel that a specific command (a shell, or in the case of other scripts, perhaps awk or Perl) is to be used to interpret the contents of the file. Using a shebang line is common

practice for all shell scripting. For example, if you write a shell script using `bash`, but want the script to execute as if run by the Bourne shell, `sh`, the first line of your script will contain `#!/bin/sh`, which is a link to the `bash` shell. Running `bash` as `sh` will cause `bash` to act as a Bourne shell. This is the reason for the symbolic link `sh`, which points to `bash`.

The Shebang Line

The shebang line is a magic number, as defined in `/usr/share/magic`—a text database of magic numbers for the Linux `file` command. Magic numbers are used by many different Linux commands to quickly identify a type of file, and the database format is documented in the section five manual page named `magic` (read by using `man 5 magic`). For example, magic numbers can be used by the Linux `file` command to display the identity of a script (no matter what filename is used) as a shell script using a specific shell or other interpreter such as `awk` or Perl.

You might also find different or new environment variables available to your scripts by using different shells. For example, if you launch `csh` from the `bash` command line, you will find several new variables or variables with slightly different definitions, such as

```
$ env
...
VENDOR=intel
MACHTYPE=i386
HOSTTYPE=i386-linux
HOST=thinkpad.home.org
```

On the other hand, `bash` might provide these variables or variables of the same name with a slightly different definition, such as

```
$ env
...
HOSTTYPE=i386
HOSTNAME=thinkpad.home.org
```

Although the behavior of a shebang line is not defined by POSIX, variations of its use can be helpful when you are writing shell scripts. For example, as described in the `wish` man page, you can use a shell to help execute programs called within a shell script without needing to hard code pathnames of programs. The `wish` command is a windowing Tool Control Language (`tcl`) interpreter that can be used to write graphical clients. Avoiding the use of specific pathnames to programs increases shell script portability because not every Unix or Linux system has programs in the same location.

For example, if you want to use the `wish` command, your first inclination might be to write

```
#!/usr/local/bin/wish
```

Although this will work on many other operating systems, the script will fail under Linux because `wish` is located under the `/usr/bin` directory. However, if you write the command line this way,

```
#!/bin/sh
exec wish "$@"
```

You can use the `wish` command (as a binary or a shell script itself); your first inclination might be to write in Linux.

Using Variables in Shell Scripts

When writing shell scripts for Linux, you work with three types of variables:

- **Environment variables**—Part of the system environment, you can use them in your shell program. New variables can be defined, and some of them, such as `PATH`, can also be modified within a shell program.
- **Built-in variables**—These are variables such as options used on the command (interpreted by the shell as a *positional argument*) are provided by Linux. Unlike environment variables, you cannot modify them.
- **User variables**—Defined by you when you write a shell script. You can use and modify them at will within the shell program.

A major difference between shell programming and other programming languages is that in shell programming, variables are not *typed*—that is, you do not have to specify whether a variable is a number or a string, and so on.

Assigning a Value to a Variable

Say that you want to use a variable called `lcount` to count the number of iterations in a loop within a shell program. You can declare and initialize this variable as follows:

Command	Environment
`lcount=0`	`pdksh` and `bash`
`set lcount=0`	`tcsh`

NOTE

Under `pdksh` and `bash`, you must ensure that the equal sign (=) does not have spaces before and after it.

To store a string in a variable, you can use the following:

Command	Environment
`myname=Sanjiv`	`pdksh` and `bash`
`set myname=Sanjiv`	`tcsh`

Use the preceding variable form if the string doesn't have embedded spaces. If a string has embedded spaces, you can do the assignment as follows:

Command	Environment
`myname="Sanjiv Guha"`	`pdksh` and `bash`
`set myname="Sanjiv Guha"`	`tcsh`

Accessing Variable Values

You can access the value of a variable by prefixing the variable name with a `$` (dollar sign). That is, if the variable name is `var`, you can access the variable by using `$var`.

If you want to assign the value of `var` to the variable `lcount`, you can do so as follows:

Command	Environment
`lcount=$var`	`pdksh` and `bash`
`set lcount=$var`	`tcsh`

Positional Parameters

It is possible to pass options from the command line or from another shell script to your shell program.

These options are supplied to the shell program by Linux as *positional parameters*, which have special names provided by the system. The first parameter is stored in a variable called `1` (number 1) and can be accessed by using `$1` within the program. The second parameter is stored in a variable called `2` and can be accessed by using `$2` within the program, and so on. One or more of the higher numbered positional parameters can be omitted while you're invoking a shell program.

Understanding how to use these positional parameters and how to access and use variables retrieved from the command line is necessary when developing more advanced shell programs.

A Simple Example of a Positional Parameter

For example, if a shell program `mypgm` expects two parameters—such as a first name and a last name—you can invoke the shell program with only one parameter, the first name. However, you cannot invoke it with only the second parameter, the last name.

Here is a shell program called `mypgm1`, which takes only one parameter (a name) and displays it on the screen:

```
#!/bin/sh
#Name display program
if [ $# -eq 0 ]
then
   echo "Name not provided"
else
   echo "Your name is "$1
fi
```

If you execute `mypgm1`, as follows,

```
$ bash  mypgm1
```

you get the following output:

```
Name not provided
```

However, if you execute `mypgm1`, as follows,

```
$ bash  mypgm1 Sanjiv
```

you get the following output:

```
Your name is Sanjiv
```

The shell program `mypgm1` also illustrates another aspect of shell programming: the built-in variables provided to the shell by the Linux kernel. In `mypgm1`, the built-in variable `$#` provides the number of positional parameters passed to the shell program. You learn more about working with built-in variables in the next major section of this chapter.

Using Positional Parameters to Access and Retrieve Variables from the Command Line

Using positional parameters in scripts can be helpful if you need to use command lines with piped commands requiring complex arguments. Shell programs containing positional parameters can be even more convenient if the commands are infrequently used. For example, if you use your Ubuntu system with an attached voice modem as an answering machine, you can write a script to issue a command that retrieves and plays the voice messages. The following lines convert a saved sound file (in `.rmd` or voice-phone format) and pipe the result to your system's audio device:

```
#!/bin/sh
# play voice message in /var/spool/voice/incoming
rmdtopvf /var/spool/voice/incoming/$1 ¦ pvfspeed -s 8000 ¦ \
pvftobasic >/dev/audio
```

A voice message can then easily be played back using this script (perhaps named `pmm`):

```
$ pmm name_of_message
```

Shell scripts that contain positional parameters are often used for automating routine and mundane jobs, such as system log report generation, file system checks, user resource accounting, printer use accounting, and other system, network, or security administration tasks.

Using a Simple Script to Automate Tasks

You could use a simple script, for example, to examine your system log for certain keywords. If the script is run via your system's scheduling table, `/etc/crontab`, it can help automate security monitoring. By combining the output capabilities of existing Linux commands with the language facilities of the shell, you can quickly build a useful script to perform a task normally requiring a number of command lines. For example, you can create a short script, named `greplog`, like this:

```
#!/bin/sh
#    name:  greplog
#      use:  mail grep of designated log using keyword
# version:  v.01 08aug02
#
#  author: bb
#
# usage: greplog [keyword] [logpathname]
#
#  bugs: does not check for correct number of arguments

# build report name using keyword search and date
log_report=/tmp/$1.logreport.`date '+%m%d%y'`

# build report header with system type, hostname, date and time
echo "============================================================" \
      >$log_report
echo "           S Y S T E M   M O N I T O R   L O G" >>$log_report
echo uname -a >>$log_report
echo "Log report for" `hostname -f` "on" `date '+%c'`      >>$log_report
echo "============================================================" \
      >>$log_report ; echo "" >>$log_report

# record log search start
echo "Search for->" $1 "starting" `date '+%r'` >>$log_report
echo "" >>$log_report
```

```
# get and save grep results of keyword ($1) from logfile ($2)
grep -i $1 $2 >>$log_report

# build report footer with time
echo "" >>$log_report
echo "End of" $log_report at `date '+%r'` >>$log_report

# mail report to root
mail -s "Log Analysis for $1" root <$log_report

# clean up and remove report
rm $log_report
exit 0
```

In this example, the script creates the variable `$log_report`, which will be the filename of the temporary report. The keyword (`$1`) and first argument on the command line is used as part of the filename, along with the current date (with perhaps a better approach to use `$$` instead of the date, which will append the script's PID as a file extension). Next, the report header containing some formatted text, the output of the `uname` command, and the hostname and date is added to the report. The start of the search is then recorded, and any matches of the keyword in the log are added to the report. A footer containing the name of the report and the time is then added. The report is mailed to root with the search term as the subject of the message, and the temporary file is deleted.

You can test the script by running it manually and feeding it a keyword and a pathname to the system log, `/var/log/messages`, like this:

```
# greplog FAILED /var/log/messages
```

Note that your system should be running the `syslogd` daemon. If any login failures have occurred on your system, the root operator might get an email message that looks like this:

```
Date: Thu, 23 Oct 2003 16:23:24 -0400
From: root <root@stinkpad.home.org>
To: root@stinkpad.home.org
Subject: FAILED

================================================================
          S Y S T E M   M O N I T O R   L O G
Linux stinky 2.4.22-1.2088.nptl #1 Thu Oct 9 20:21:24 EDT 2003 i686 i686 i386
+GNU/Linux
Log report for stinkpad.home.org on Thu 23 Oct 2003 04:23:24 PM EDT
================================================================

Search for-> FAILED starting 04:23:24 PM
```

```
Oct 23 16:23:04 stinkpad login[1769]: FAILED LOGIN 3 FROM (null) FOR bball,
+Authentication failure

End of /tmp/FAILED.logreport.102303 at 04:23:24 PM
```

To further automate the process, you can include command lines using the script in another script to generate a series of searches and reports.

Built-in Variables

Built-in variables are special variables provided to shell by Linux that can be used to make decisions within a shell program. You cannot modify the values of these variables within the shell program.

Some of these variables are

- `$#`—Number of positional parameters passed to the shell program
- `$?`—Completion code of the last command or shell program executed within the shell program (returned value)
- `$0`—The name of the shell program
- `$*`—A single string of all arguments passed at the time of invocation of the shell program

To show these built-in variables in use, here is a sample program called `mypgm2`:

```
#!/bin/sh
#my test program
echo "Number of parameters is $#"
echo "Program name is $0"
echo "Parameters as a single string is $*"
```

If you execute `mypgm2` from the command line in `pdksh` and `bash` as follows,

```
$ bash mypgm2 Sanjiv Guha
```

you get the following result:

```
Number of parameters is 2
Program name is mypgm2
Parameters as a single string is Sanjiv Guha
```

Special Characters

Some characters have special meaning to Linux shells; these characters represent commands, denote specific use for surrounding text, or provide search parameters. Special characters provide a sort of shorthand by incorporating these rather complex meanings into a simple character. Some special characters are shown in Table 11.2.

TABLE 11.2 Special Shell Characters

Character	Explanation
$	Indicates the beginning of a shell variable name
¦	Pipes standard output to next command
#	Starts a comment
&	Executes a process in the background
?	Matches one character
*	Matches one or more characters
>	Output redirection operator
<	Input redirection operator
`	Command substitution (the backquote or backtick—the key above the Tab key on most keyboards)
>>	Output redirection operator (to append to a file)
<<	Wait until following end-of-input string (HERE operator)
[ ]	Range of characters
[a-z]	All characters a through z
[a,z] or [az]	Characters a or z
Space	Delimiter between two words

Special characters are very useful to you when you're creating shell scripts, but if you inadvertently use a special character as part of variable names or strings, your program will behave incorrectly. As you learn in later parts of this section, you can use one of the special characters in a string if you precede it with an *escape character* (/, or backslash) to indicate that it isn't being used as a special character and shouldn't be treated as such by the program.

A few special characters deserve special note. They are the double quotes ("), the single quotes ('), the backslash (\), and the backtick (`)—all discussed in the following sections.

Use Double Quotes to Resolve Variables in Strings with Embedded Spaces

If a string contains embedded spaces, you can enclose the string in double quotes (") so that the shell interprets the whole string as one entity instead of more than one.

For example, if you assigned the value of `abc def` (`abc` followed by one space, followed by `def`) to a variable called `x` in a shell program as follows, you would get an error because the shell would try to execute `def` as a separate command:

Command	Environment
`x=abc def`	pdksh and bash
`set x=abc def`	tcsh

The shell will execute the string as a single command if you surround the string in double quotes as follows:

Command	Environment
`x="abc def"`	`pdksh` and `bash`
`set x="abc def"`	`tcsh`

The double quotes resolve all variables within the string. Here is an example for `pdksh` and `bash`:

```
var="test string"
newvar="Value of var is $var"
echo $newvar
```

Here is the same example for `tcsh`:

```
set var="test string"
set newvar="Value of var is $var"
echo $newvar
```

If you execute a shell program containing the preceding three lines, you get the following result:

```
Value of var is test string
```

Using Single Quotes to Maintain Unexpanded Variables

You can surround a string with single quotes (`'`) to stop the shell from expanding variables and interpreting special characters. When used for the latter purpose, the single quote is an *escape character*, similar to the backslash, which you learn about in the next section. Here, you learn how to use the single quote to avoid expanding a variable in a shell script. An unexpanded variable maintains its original form in the output.

In the following examples, the double quotes in the preceding examples have been changed to single quotes.

`pdksh` and `bash`:

```
var='test string'
newvar='Value of var is $var'
echo $newvar
```

`tcsh`:

```
set var = 'test string'
set newvar = 'Value of var is $var'
echo $newvar
```

If you execute a shell program containing these three lines, you get the following result:

```
Value of var is $var
```

As you can see, the variable `var` maintains its original format in the results, rather than having been expanded.

Using the Backslash As an Escape Character

As you learned earlier, the backslash (`\`) serves as an escape character that stops the shell from interpreting the succeeding character as a special character. Say that you want to assign a value of `$test` to a variable called `var`. If you use the following command, the shell reads the special character `$` and interprets `$test` as the value of the variable `test`. No value has been assigned to `test`; a null value is stored in `var` as follows:

Command	Environment
`var=$test`	`pdksh` and `bash`
`set var=$test`	`tcsh`

Unfortunately, this assignment may work for `bash` and `pdksh`, but it returns an error of "undefined variable" if you use it with `tcsh`. Use the following commands to correctly store `$test` in `var`:

Command	Environment
`var=\$test`	`pdksh` and `bash`
`set var = \$test`	`tcsh`

The backslash before the dollar sign (`\$`) signals the shell to interpret the `$` as any other ordinary character and not to associate any special meaning to it. You could also use single quotes (`'`) around the `$test` variable to get the same result.

Using the Backtick to Replace a String with Output

You can use the backtick (`` ` ``) character to signal the shell to replace a string with its output when executed. This special character can be used in shell programs when you want the result of the execution of a command to be stored in a variable. For example, if you want to count the number of lines in a file called `test.txt` in the current directory and store the result in a variable called `var`, you can use the following command:

Command	Environment
`` var=`wc -l test.txt` ``	`pdksh` and `bash`
`` set var = `wc -l test.txt` ``	`tcsh`

Comparison of Expressions in pdksh and bash

Comparing values or evaluating the differences between similar bits of data—such as file information, character strings, or numbers—is a task known as *comparison of expressions*. Comparison of expressions is an integral part of using logic in shell programs to accomplish tasks. The way the logical comparison of two operators (numeric or string) is done varies slightly in different shells. In `pdksh` and `bash`, a command called `test` can be used to achieve comparisons of expressions. In `tcsh`, you can write an expression to accomplish the same thing.

The following section covers comparison operations using the `pdksh` or `bash` shells. Later in the chapter, you learn how to compare expressions in the `tcsh` shell.

The `pdksh` and `bash` shell syntax provide a command named `test` to compare strings, numbers, and files. The syntax of the `test` command is as follows:

```
test expression
```

or

```
[ expression ]
```

Both forms of the `test` commands are processed the same way by `pdksh` and `bash`. The `test` commands support the following types of comparisons:

- String comparison
- Numeric comparison
- File operators
- Logical operators

String Comparison

The following operators can be used to compare two string expressions:

`=`—To compare whether two strings are equal

`!=`—To compare whether two strings are not equal

`-n`—To evaluate whether the string length is greater than zero

`-z`—To evaluate whether the string length is equal to zero

Next are some examples using these operators when comparing two strings, `string1` and `string2`, in a shell program called `compare1`:

```
#!/bin/sh
string1="abc"
string2="abd"
if [ $string1 = $string2 ]; then
```

```
   echo "string1 equal to string2"
else
   echo "string1 not equal to string2"
fi

if [ $string2 != string1 ]; then
   echo "string2 not equal to string1"
else
   echo "string2 equal to string2"
fi

if [ $string1 ]; then
   echo "string1 is not empty"
else
   echo "string1 is empty"
fi

if [ -n $string2 ]; then
   echo "string2 has a length greater than zero"
else
   echo "string2 has length equal to zero"
fi

if [ -z $string1 ]; then
   echo "string1 has a length equal to zero"
else
  echo "string1 has a length greater than zero"
fi
```

If you execute `compare1`, you get the following result:

```
string1 not equal to string2
string2 not equal to string1
string1 is not empty
string2 has a length greater than zero
string1 has a length greater than zero
```

If two strings are not equal in size, the system pads out the shorter string with trailing spaces for comparison. That is, if the value of `string1` is `abc` and that of `string2` is `ab`, `string2` will be padded with a trailing space for comparison purposes—it will have a value of `ab`.

Number Comparison

The following operators can be used to compare two numbers:

- `-eq`—To compare whether two numbers are equal
- `-ge`—To compare whether one number is greater than or equal to the other number
- `-le`—To compare whether one number is less than or equal to the other number
- `-ne`—To compare whether two numbers are not equal
- `-gt`—To compare whether one number is greater than the other number
- `-lt`—To compare whether one number is less than the other number

The following shell program compares three numbers, `number1`, `number2`, and `number3`:

```
#!/bin/sh
number1=5
number2=10
number3=5

if [ $number1 -eq $number3 ]; then
   echo "number1 is equal to number3"
else
   echo "number1 is not equal to number3"
fi

if [ $number1 -ne $number2 ]; then
   echo "number1 is not equal to number2"
else
   echo "number1 is equal to number2"
fi

if [ $number1 -gt $number2 ]; then
   echo "number1 is greater than number2"
else
   echo "number1 is not greater than number2"
fi

if [ $number1 -ge $number3 ]; then
   echo "number1 is greater than or equal to number3"
else
   echo "number1 is not greater than or equal to number3"
fi

if [ $number1 -lt $number2 ]; then
   echo "number1 is less than number2"
```

```
else
   echo "number1 is not less than number2"
fi

if [ $number1 -le $number3 ]; then
   echo "number1 is less than or equal to number3"
else
   echo "number1 is not less than or equal to number3"
fi
```

When you execute the shell program, you get the following results:

```
number1 is equal to number3
number1 is not equal to number2
number1 is not greater than number2
number1 is greater than or equal to number3
number1 is less than number2
number1 is less than or equal to number3
```

File Operators

The following operators can be used as file comparison operators:

- `-d`—To ascertain whether a file is a directory
- `-f`—To ascertain whether a file is a regular file
- `-r`—To ascertain whether read permission is set for a file
- `-s`—To ascertain whether a file exists and has a length greater than zero
- `-w`—To ascertain whether write permission is set for a file
- `-x`—To ascertain whether execute permission is set for a file

Assume that a shell program called `compare3` is in a directory with a file called `file1` and a subdirectory `dir1` under the current directory. Assume that `file1` has a permission of `r-x` (read and execute permission) and `dir1` has a permission of `rwx` (read, write, and execute permission). The code for the shell program would look like this:

```
#!/bin/sh
if [ -d $dir1 ]; then
   echo "dir1 is a directory"
else
   echo "dir1 is not a directory"
fi

if [ -f $dir1 ]; then
   echo "dir1 is a regular file"
```

```
else
   echo "dir1 is not a regular file"
fi

if [ -r $file1 ]; then
   echo "file1 has read permission"
else
   echo "file1 does not have read permission"
fi

if [ -w $file1 ]; then
   echo "file1 has write permission"
else
   echo "file1 does not have write permission"
fi

if [ -x $dir1 ]; then
   echo "dir1 has execute permission"
else
   echo "dir1 does not have execute permission"
fi
```

If you execute the shell program, you get the following results:

```
dir1 is a directory
file1 is a regular file
file1 has read permission
file1 does not have write permission
dir1 has execute permission
```

Logical Operators

Logical operators are used to compare expressions using Boolean logic, which compares values using characters representing `NOT`, `AND`, and `OR`.

- `!`—To negate a logical expression
- `-a`—To logically `AND` two logical expressions
- `-o`—To logically `OR` two logical expressions

This example named `logic` uses the file and directory mentioned in the previous `compare3` example.

```
#!/bin/sh
if [ -x file1 -a -x dir1 ]; then
   echo file1 and dir1 are executable
```

```
else
   echo at least one of file1 or dir1 are not executable
fi

if [ -w file1 -o -w dir1 ]; then
   echo file1 or dir1 are writable
else
   echo neither file1 or dir1 are executable
fi

if [ ! -w file1 ]; then
   echo file1 is not writable
else
   echo file1 is writable
fi
```

If you execute `logic`, it will yield the following result:

```
file1 and dir1 are executable
file1 or dir1 are writable
file1 is not writable
```

Comparing Expressions with `tcsh`

As stated earlier, the method for comparing expressions in `tcsh` is different from the method used under `pdksh` and `bash`. The comparison of expression demonstrated in this section uses the syntax necessary for the `tcsh` shell environment.

String Comparison

The following operators can be used to compare two string expressions:

- `==`—To compare whether two strings are equal
- `!=`—To compare whether two strings are not equal

The following examples compare two strings, `string1` and `string2`, in the shell program `compare1`:

```
#!/bin/tcsh
set string1 = "abc"
set string2 = "abd"

if  (string1 == string2)  then
   echo "string1 equal to string2"
```

```
else
   echo "string1 not equal to string2"
endif

if  (string2 != string1)  then
   echo "string2 not equal to string1"
else
   echo "string2 equal to string1"
endif
```

If you execute `compare1`, you get the following results:

```
string1 not equal to string2
string2 not equal to string1
```

Number Comparison

These operators can be used to compare two numbers:

- >=—To compare whether one number is greater than or equal to the other number
- <=—To compare whether one number is less than or equal to the other number
- >—To compare whether one number is greater than the other number
- <—To compare whether one number is less than the other number

The next examples compare two numbers, `number1` and `number2`, in a shell program called `compare2`:

```
#!/bin/tcsh
set number1=5
set number2=10
set number3=5

if  ($number1 > $number2)  then
   echo "number1 is greater than number2"
else
   echo "number1 is not greater than number2"
endif

if  ($number1 >= $number3) then
   echo "number1 is greater than or equal to number3"
else
   echo "number1 is not greater than or equal to number3"
endif

if  ($number1 < $number2)  then
   echo "number1 is less than number2"
```

```
else
   echo "number1 is not less than number2"
endif

if  ($number1 <= $number3) then
   echo "number1 is less than or equal to number3"
else
   echo "number1 is not less than or equal to number3"
endif
```

When executing the shell program `compare2`, you get the following results:

```
number1 is not greater than number2
number1 is greater than or equal to number3
number1 is less than number2
number1 is less than or equal to number3
```

File Operators

These operators can be used as file comparison operators:

- `-d`—To ascertain whether a file is a directory
- `-e`—To ascertain whether a file exists
- `-f`—To ascertain whether a file is a regular file
- `-o`—To ascertain whether a user is the owner of a file
- `-r`—To ascertain whether read permission is set for a file
- `-w`—To ascertain whether write permission is set for a file
- `-x`—To ascertain whether execute permission is set for a file
- `-z`—To ascertain whether the file size is zero

The following examples are based on a shell program called `compare3`, which is in a directory with a file called `file1` and a subdirectory `dir1` under the current directory. Assume that `file1` has a permission of `r-x` (read and execute permission) and `dir1` has a permission of `rwx` (read, write, and execute permission).

The following is the code for the `compare3` shell program:

```
#!/bin/tcsh
if  (-d dir1) then
   echo "dir1 is a directory"
else
   echo "dir1 is not a directory"
endif
```

```
if (-f dir1)  then
   echo "file1 is a regular file"
else
   echo "file1 is not a regular file"
endif

if (-r file1) then
   echo "file1 has read permission"
else
   echo "file1 does not have read permission"
endif

if (-w file1) then
   echo "file1 has write permission"
else
   echo "file1 does not have write permission"
endif

if (-x dir1) then
   echo "dir1 has execute permission"
else
   echo "dir1 does not have execute permission"
endif

if (-z file1) then
   echo "file1 has zero length"
else
   echo "file1 has greater than zero length"
endif
```

If you execute the file `compare3`, you get the following results:

```
dir1 is a directory
file1 is a regular file
file1 has read permission
file1 does not have write permission
dir1 has execute permission
file1 has greater than zero length
```

Logical Operators

Logical operators are used with conditional statements. These operators are used to negate a logical expression or to perform logical `AND`s and `OR`s:

- `!`—To negate a logical expression
- `&&`—To logically `AND` two logical expressions
- `||`—To logically `OR` two logical expressions

This example named `logic` uses the file and directory mentioned in the previous `compare3` example.

```
#!/bin/tcsh
if ( -x file1 && -x dir1 ) then
   echo file1 and dir1 are executable
else
   echo at least one of file1 or dir1 are not executable
endif

if ( -w file1 ¦¦ -w dir1 ) then
   echo file1 or dir1 are writable
else
   echo neither file1 or dir1 are executable
endif

if ( ! -w file1 ) then
   echo file1 is not writable
else
   echo file1 is writable
endif
```

If you execute `logic`, it will yield the following result:

```
file1 and dir1 are executable
file1 or dir1 are writable
file1 is not writable
```

The `for` Statement

The `for` statement is used to execute a set of commands once each time a specified condition is true. The `for` statement has a number of formats. The first format used by `pdksh` and `bash` is as follows:

```
for curvar in list
do
    statements
done
```

This form should be used if you want to execute `statements` once for each value in `list`. For each iteration, the current value of the list is assigned to `vcurvar`. `list` can be a variable containing a number of items or a list of values separated by spaces. The second format is as follows:

```
for curvar
do
    statements
done
```

In this form, the `statements` are executed once for each of the positional parameters passed to the shell program. For each iteration, the current value of the positional parameter is assigned to the variable `curvar`.

This form can also be written as follows:

```
for curvar in $@
do
    statements
done
```

Remember that `$@` gives you a list of positional parameters passed to the shell program, quoted in a manner consistent with the way the user originally invoked the command.

Under `tcsh`, the `for` statement is called `foreach`. The format is as follows:

```
foreach curvar (list)
    statements
end
```

In this form, `statements` are executed once for each value in `list`, and, for each iteration, the current value of `list` is assigned to `curvar`.

Suppose that you want to create a backup version of each file in a directory to a subdirectory called `backup`. You can do the following in `pdksh` and `bash`:

```
#!/bin/sh
for filename in *
do
   cp $filename backup/$filename
   if [ $? -ne 0 ]; then
      echo "copy for $filename failed"
   fi
done
```

In the preceding example, a backup copy of each file is created. If the copy fails, a message is generated.

The same example in `tcsh` is as follows:

```
#!/bin/tcsh
foreach filename (`/bin/ls`)
   cp $filename backup/$filename
   if ($? != 0) then
      echo "copy for $filename failed"
   endif
end
```

The `while` Statement

The `while` statement can be used to execute a series of commands while a specified condition is true. The loop terminates as soon as the specified condition evaluates to false. It is possible that the loop will not execute at all if the specified condition initially evaluates to false. You should be careful with the `while` command because the loop will never terminate if the specified condition never evaluates to false.

Endless Loops Have Their Place in Shell Programs

Endless loops can sometimes be useful. For example, you can easily construct a simple command that constantly monitors the 802.11b link quality of a network interface by using a few lines of script:

```
#!/bin/sh
while :
 do
   /sbin/iwconfig eth0 ¦  grep Link ¦ tr '\n' '\r'
 done
```

The script outputs the search, and then the `tr` command formats the output. The result is a simple animation of a constantly updated single line of information:

```
Link Quality:92/92  Signal level:-11 dBm  Noise level:-102 dBm
```

This technique can also be used to create a graphical monitoring client for X that outputs traffic information and activity about a network interface:

```
#!/bin/sh
xterm -geometry 75x2 -e \
bash -c \
"while :; do \
     /sbin/ifconfig eth0 ¦ \
     grep 'TX bytes' ¦
      tr '\n' '\r' ; \
done"
```

The simple example uses a `bash` command-line script (enabled by `-c`) to execute a command line repeatedly. The command line pipes the output of the `ifconfig` command through `grep`, which searches `ifconfig`'s output and then pipes a line containing the string `"TX bytes"` to the `tr` command. The `tr` command then removes the carriage return at the end of the line to display the information inside an `/xterm` X11 terminal window, automatically sized by the `-geometry` option:

```
RX bytes:4117594780 (3926.8 Mb)  TX bytes:452230967 (431.2 Mb)
```

Endless loops can be so useful that Linux includes a command that will repeatedly execute a given command line. For example, you can get a quick report about a system's hardware health by using the `sensors` command. But instead of using a shell script to loop the output endlessly, you can use the `watch` command to repeat the information and provide simple animation:

```
$ watch "sensors -f ¦ cut -c 1-20"
```

In pdksh and bash, the following format is used for the while flow control construct:

```
while expression
do
    statements
done
```

In tcsh, the following format is used:

```
while (expression)
    Statements
end
```

If you want to add the first five even numbers, you can use the following shell program in pdksh and bash:

```
#!/bin/bash
loopcount=0
result=0
while [ $loopcount -lt 5 ]
do
   loopcount=`expr $loopcount + 1`
   increment=`expr $loopcount \* 2`
   result=`expr $result + $increment`
done

echo "result is $result"
```

In tcsh, this program can be written as follows:

```
#!/bin/tcsh
set loopcount = 0
set result = 0
while ($loopcount < 5)
   set loopcount = `expr $loopcount + 1`
   set increment = `expr $loopcount \* 2`
   set result = `expr $result + $increment`

end

echo "result is $result"
```

The until Statement

The until statement can be used to execute a series of commands until a specified condition is true.

The loop terminates as soon as the specified condition evaluates to `True`.

In `pdksh` and `bash`, the following format is used:

```
until expression
do
    statements
done
```

As you can see, the format of the `until` statement is similar to that of the `while` statement, but the logic is different: In a `while` loop, you execute until an expression is `False`, but in an `until` loop, you loop until the expression is `True`.

If you want to add the first five even numbers, you can use the following shell program in `pdksh` and `bash`:

```
#!/bin/bash
loopcount=0
result=0
until [ $loopcount -ge 5 ]
do
   loopcount=`expr $loopcount + 1`
   increment=`expr $loopcount \* 2`
   result=`expr $result + $increment`
done

echo "result is $result"
```

The example here is identical to the example for the `while` statement, except that the condition being tested is just the opposite of the condition specified in the `while` statement.

The `tcsh` shell does not support the `until` statement.

The `repeat` Statement (`tcsh`)

The `repeat` statement is used to execute only one command a fixed number of times.

If you want to print a hyphen (`-`) 80 times with one hyphen per line on the screen, you can use the following command:

```
repeat  80 echo '-'
```

The `select` Statement (`pdksh`)

The `select` statement is used to generate a menu list if you are writing a shell program that expects input from the user online. The format of the `select` statement is as follows:

```
select  item in itemlist
do
    Statements
done
```

`itemlist` is optional. If it isn't provided, the system iterates through the `item` entries one at a time. If `itemlist` is provided, however, the system iterates for each entry in `itemlist` and the current value of `itemlist` is assigned to `item` for each iteration, which then can be used as part of the statements being executed.

If you want to write a menu that gives the user a choice of picking a `Continue` or a `Finish`, you can write the following shell program:

```
#!/bin/ksh
select  item in Continue Finish
do
   if [ $item = "Finish" ]; then
      break
   fi
done
```

When the `select` command is executed, the system displays a menu with numeric choices to the user—in this case, `1` for `Continue` and `2` for `Finish`. If the user chooses `1`, the variable `item` contains a value of `Continue`; if the user chooses `2`, the variable `item` contains a value of `Finish`. When the user chooses `2`, the `if` statement is executed and the loop terminates.

The `shift` Statement

The `shift` statement is used to process the positional parameters, one at a time, from left to right. As you'll remember, the positional parameters are identified as `$1`, `$2`, `$3`, and so on. The effect of the `shift` command is that each positional parameter is moved one position to the left and the current `$1` parameter is lost.

The `shift` statement is useful when you are writing shell programs in which a user can pass various options. Depending on the specified option, the parameters that follow can mean different things or might not be there at all.

The format of the `shift` command is as follows:

```
shift  number
```

The parameter `number` is the number of places to be shifted and is optional. If not specified, the default is `1`; that is, the parameters are shifted one position to the left. If specified, the parameters are shifted `number` positions to the left.

The `if` Statement

The `if` statement evaluates a logical expression to make a decision. An `if` condition has the following format in `pdksh` and `bash`:

```
if [ expression ]; then
    Statements
elif [ expression ]; then
    Statements
else
    Statements
fi
```

The `if` conditions can be nested. That is, an `if` condition can contain another `if` condition within it. It isn't necessary for an `if` condition to have an `elif` or `else` part. The `else` part is executed if none of the expressions that are specified in the `if` statement and are optional if subsequent `elif` statements are true. The word `fi` is used to indicate the end of the `if` statements, which is very useful if you have nested `if` conditions. In such a case, you should be able to match `fi` to `if` to ensure that all `if` statements are properly coded.

In the following example for `bash` or `pdksh`, a variable `var` can have either of two values: `Yes` or `No`. Any other value is invalid. This can be coded as follows:

```
if [ $var = "Yes" ]; then
   echo "Value is Yes"
elif [ $var = "No" ]; then
   echo "Value is No"
else
   echo "Invalid value"
fi
```

In `tcsh`, the `if` statement has two forms. The first form, similar to the one for `pdksh` and `bash`, is as follows:

```
if (expression) then
    Statements
else if (expression) then
    Statements
else
    Statements
endif
```

The `if` conditions can be nested—that is, an `if` condition can contain another `if` condition within it. It isn't necessary for an `if` condition to have an `else` part. The `else` part is executed if none of the expressions specified in any of the `if` statements are true. The

optional `if` part of the statement (`else if (expression) then`) is executed if the condition following it is true and the previous `if` statement is not true. The word `endif` is used to indicate the end of the `if` statements, which is very useful if you have nested `if` conditions. In such a case, you should be able to match `endif` to `if` to ensure that all `if` statements are properly coded.

Using the example of the variable `var` having only two values, `Yes` and `No`, here is how it would be coded with `tcsh`:

```
if ($var == "Yes") then
   echo "Value is Yes"
else if ($var == "No" ) then
   echo "Value is No"
else
   echo "Invalid value"
endif
```

The second form of the `if` condition for `tcsh` is as follows:

```
if (expression) command
```

In this format, only a single command can be executed if the expression evaluates to true.

The `case` Statement

The `case` statement is used to execute statements depending on a discrete value or a range of values matching the specified variable. In most cases, you can use a `case` statement instead of an `if` statement if you have a large number of conditions.

The format of a `case` statement for `pdksh` and `bash` is as follows:

```
case str in
   str1 ¦ str2)
      Statements;;
   str3¦str4)
      Statements;;
   *)
      Statements;;
esac
```

You can specify a number of discrete values—such as `str1`, `str2`, and so on—for each condition, or you can specify a value with a wildcard. The last condition should be `*` (asterisk) and is executed if none of the other conditions are met. For each of the specified conditions, all the associated statements until the double semicolon (`;;`) are executed.

You can write a script that will echo the name of the month if you provide the month number as a parameter. If you provide a number that isn't between 1 and 12, you will get an error message. The script is as follows:

```
#!/bin/sh

case $1 in
   01 ¦ 1) echo "Month is January";;
   02 ¦ 2) echo "Month is February";;
   03 ¦ 3) echo "Month is March";;
   04 ¦ 4) echo "Month is April";;
   05 ¦ 5) echo "Month is May";;
   06 ¦ 6) echo "Month is June";;
   07 ¦ 7) echo "Month is July";;
   08 ¦ 8) echo "Month is August";;
   09 ¦ 9) echo "Month is September";;
   10) echo "Month is October";;
   11) echo "Month is November";;
   12) echo "Month is December";;
   *) echo "Invalid parameter";;
esac
```

You need to end the statements under each condition with a double semicolon (`;;`). If you do not, the statements under the next condition will also be executed.

The format for a `case` statement for `tcsh` is as follows:

```
switch (str)
   case str1¦str2:
      Statements
      breaksw
   case str3¦str4:
      Statements
      breaksw
   default:
      Statements
      breaksw
endsw
```

You can specify a number of discrete values—such as `str1`, `str2`, and so on—for each condition, or you can specify a value with a wildcard. The last condition should be `default` and is executed if none of the other conditions are met. For each of the specified conditions, all the associated statements until `breaksw` are executed.

The example that echoes the month when a number is given, shown earlier for `pdksh` and `bash`, can be written in `tcsh` as follows:

```
#!/bin/tcsh

set month = 5
switch  ( $month )
   case 1: echo "Month is January" ;  breaksw
   case 2: echo "Month is February" ;  breaksw
   case 3: echo "Month is March" ;  breaksw
   case 4: echo "Month is April" ;  breaksw
   case 5: echo "Month is May" ;  breaksw
   case 6: echo "Month is June" ;  breaksw
   case 7: echo "Month is July" ;  breaksw
   case 8: echo "Month is August"  ;  breaksw
   case 9: echo "Month is September" ;  breaksw
   case 10: echo "Month is October" ;  breaksw
   case 11: echo "Month is November" ;  breaksw
   case 12: echo "Month is December" ;  breaksw
   default: echo "Oops! Month is Octember!" ;  breaksw
endsw
```

You need to end the statements under each condition with `breaksw`. If you do not, the statements under the next condition will also be executed.

The `break` and `exit` Statements

You should be aware of two other statements: the `break` statement and the `exit` statement.

The `break` statement can be used to terminate an iteration loop, such as a `for`, `until`, or `repeat` command.

`exit` statements can be used to exit a shell program. You can optionally use a number after `exit`. If the current shell program has been called by another shell program, the calling program can check for the code (the `$?` or `$status` variable, depending on shell) and make a decision accordingly.

Using Functions in Shell Scripts

As with other programming languages, shell programs also support *functions*. A function is a piece of a shell program that performs a particular process; you can reuse the same function multiple times within the shell program. Functions help eliminate the need for duplicating code as you write shell programs.

The following is the format of a function in pdksh and bash:

```
func(){
   Statements
}
```

You can call a function as follows:

```
func param1 param2 param3
```

The parameters *param1*, *param2*, and so on are optional. You can also pass the parameters as a single string—for example, $@. A function can parse the parameters as if they were positional parameters passed to a shell program from the command line as command-line arguments, but instead use values passed inside the script. For example, the following script uses a function named Displaymonth() that displays the name of the month or an error message if you pass a month number out of the range 1 to 12. This example works with pdksh and bash:

```
#!/bin/sh
Displaymonth() {
   case $1 in
      01 ¦ 1) echo "Month is January";;
      02 ¦ 2) echo "Month is February";;
      03 ¦ 3) echo "Month is March";;
      04 ¦ 4) echo "Month is April";;
      05 ¦ 5) echo "Month is May";;
      06 ¦ 6) echo "Month is June";;
      07 ¦ 7) echo "Month is July";;
      08 ¦ 8) echo "Month is August";;
      09 ¦ 9) echo "Month is September";;
      10) echo "Month is October";;
      11) echo "Month is November";;
      12) echo "Month is December";;
      *) echo "Invalid parameter";;
   esac
}
Displaymonth 8
```

The preceding program displays the following output:

```
Month is August
```

Related Ubuntu and Linux Commands

You can use these commands and tools when using the shell or writing shell scripts:

`chsh`—Command used to change one's login shell

`kibitz`—Allows two-person interaction with a single shell

`mc`—A visual shell named the GNU Midnight Commander

`nano`—An easy-to-use text editor for the console

`shar`—Command used to create shell archives

`vi`—The `vi` (actually `vim`) text editor

Reference

- http://www.linuxgazette.com/issue70/ghosh.html—A Linux Gazette article on "Bootstrapping Linux," which includes much detail on the BIOS and MBR aspects of the boot process.
- http://www.lostcircuits.com/advice/bios2/1.shtml—the LostCircuits BIOS guide; much detail on the meaning of all those BIOS settings.
- http://www.rojakpot.com/default.aspx?location=1—The BIOS Optimization Guide; details on tweaking those BIOS settings to your heart's content.
- http://www-106.ibm.com/developerworks/linux/library/l-slack.html—A link through IBM's website to an article on booting Slackware Linux, along with a sidebar about the history of System V versus BSD `init` scripts.
- `/usr/src/linux/init/main.c`—The best place to learn about how Linux boots. Fascinating reading, really. Get it from the source!
- http://sunsite.dk/linux-newbie/—Home page for the Linux Newbie Administrator Guide—A gentle introduction to Linux System Administration.
- The GRUB Manual—The still yet-to-be-completed manual can be found at http://www.gnu.org/software/grub/manual/. On your Ubuntu system, `info grub` provides a plethora of information and a sample `grub.conf` file (`/boot/grub/menu.lst` is a symbolic link to `/boot/grub/grub.conf`; use either name) can be found in `/usr/doc/grub`.
- "LILO User's Guide"—Werner Almesberger's definitive technical tome on the LInux LOader, or LILO, and how it works on Intel-based PCs. Look under the `/usr/share/doc/lilo*/doc` directory for the file `User_Guide.ps`, which can be viewed using the gv client. LILO has been dropped from Ubuntu; GRUB is now the default boot loader supported in the distribution.
- "Grub, Glorious Grub"—Hoyt Duff, *Linux Format*, August 2001; pp. 58–61. A tutorial on the GRUB boot leader.

- http://www.linuxbase.org/spec/refspecs/LSB_1.0.0/gLSB/sysinit.html—The Linux Standard Base description of system initialization; this is the standard.
- http://www.gnu.org/software/bash/bash.html—The `bash` home page at the GNU Software Project.
- http://www.tldp.org/LDP/abs/html/—Mendel Cooper's "Advanced Bash-Scripting Guide."
- http://www.linuxnewbie.org/nhf/intel/shells/basic.html—Learn basic shell commands at this site.
- http://www.cs.princeton.edu/~jlk/lj/—"The New Korn Shell—ksh93," an article by David G. Korn, Charles J. Northrup, and Jeffery Korn regarding the `korn` shell.
- http://web.cs.mun.ca/~michael/pdksh/—The `pdksh` home page.
- http://lug.umbc.edu/~mabzug1/bash-httpd.html—The Bash-httpd FAQ, with links to the latest version of the `bash` web server, `bash-httpd-0.02`.
- http://www.tcsh.org/—Find out more about `tcsh` here.
- http://www.zsh.org/—Examine `zsh` in more detail here.

This page intentionally left blank

CHAPTER 12

System-Monitoring Tools

IN THIS CHAPTER

To keep your system in optimum shape, you need to be able to monitor it closely. Such monitoring is imperative in a corporate environment where uptime is vital and any system failures can cost real money. Whether it is checking processes for any errant daemons, or keeping a close eye on CPU and memory usage, Ubuntu provides a wealth of utilities designed to give you as little or as much feedback as you want. This chapter looks at some of the basic monitoring tools, along with some tactics designed to keep your system up longer. Some of the monitoring tools cover network connectivity, memory, and hard drive usage, but all should find a place in your sysadmin toolkit. Finally, you will learn how to manipulate active system processes, using a mixture of graphical and command-line tools.

Console-Based Monitoring

Those familiar with Unix system administration already know about the `ps` or process display command commonly found on most flavors of Unix. Because Linux is closely related to Unix, it also benefits from this command and allows you to quickly see the current running processes on the system, as well as who owns them and how resource-hungry they are.

Although the Linux kernel has its own distinct architecture and memory management, it also benefits from enhanced use of the `/proc` file system, the virtual file system found on many Unix flavors. Through the `/proc` file system, you can communicate directly with the kernel to get a deep view of what is currently happening. Developers tend to use the `/proc` file system as a way of getting information out from the kernel and for their programs to manipulate it

into more human-readable formats. The /proc file system is beyond the scope of this book, but if you want to get a better idea of what it contains you should head on over to http://en.tldp.org/LDP/Linux-Filesystem-Hierarchy/html/proc.html for an excellent and very in-depth guide.

Processes can also be controlled at the command line, which is important because you might sometimes have only a command-line interface. Whenever an application or command is launched, either from the command line or a clicked icon, the process that comes from the kernel is assigned an identification number called a *process ID* or *PID* for short. This number is shown in the shell if the program is launched via the command line.

```
$ gedit &

[1] 6193
```

In this example, gedit has been launched in the background, and the (bash) shell reported a shell job number ([1] in this case). A job number or job control is a shell-specific feature that allows a different form of process control, such as sending or suspending programs to the background and retrieving background jobs to the foreground (see your shell's man pages for more information if you are not using bash).

The second number displayed (6193 in this example) represents the process ID. You can get a quick list of your processes by using the ps command like this:

```
# ps
  PID TTY          TIME CMD

 6176 pts/0    00:00:00 bash

 6193 pts/0    00:00:00 gedit

 6197 pts/0    00:00:00 ps
```

Note that not all output from the display is shown here. But as you can see, the output includes the process ID, abbreviated as PID, along with other information, such as the name of the running program. As with any Unix command, many options are available; the proc man page has a full list. A most useful option is aux, which provides a friendly list of all the processes. You should also know that ps works not by polling memory, but through the interrogation of the Linux /proc or process file system. (ps is one of the interfaces mentioned at the beginning of this section.)

The /proc directory contains quite a few files—some of which include constantly updated hardware information (such as battery power levels, and so on). Linux administrators often pipe the output of ps through a member of the grep family of commands to display information about a specific program, perhaps like this:

```
$ ps aux ¦ grep gedit
andrew    6193  0.4  1.4  42772 14992 pts/0    S    20:27   0:00 gedit
```

This example returns the owner (the user who launched the program) and the PID, along with other information, such as the percentage of CPU and memory usage, size of the command (code, data, and stack), time (or date) the command was launched, and name of the command. Processes can also be queried by PID like this:

```
$ ps 6193
  PID TTY      STAT   TIME COMMAND

 6193 pts/0    S      0:00 gedit
```

You can use the PID to stop a running process by using the shell's built-in `kill` command. This command asks the kernel to stop a running process and reclaim system memory. For example, to stop the `gedit` client in the example, use the `kill` command like this:

```
$ kill 6193
```

After you press Enter (or perhaps press Enter again), the shell might report

```
[1]+  Terminated              gedit
```

Note that users can `kill` only their own processes, but `root` can kill them all. Controlling any other running process requires root permission, which should be used judiciously (especially when forcing a `kill` by using the `-9` option); by inadvertently killing the wrong process through a typo in the command, you could bring down an active system.

Using the `kill` Command to Control Processes

The `kill` command is a basic Unix system command. You can communicate with a running process by entering a command into its interface, such as when you type into a text editor. But some processes (usually system processes rather than application processes) run without such an interface, and you need a way to communicate with them as well, so we use a system of signals. The `kill` system accomplishes that by sending a signal to a process, and you can use it to communicate with any process. The general format of the `kill` command is

```
# kill option PID
```

A number of signal options can be sent as words or numbers, but most are of interest only to programmers. One of the most common is

```
# kill PID
```

This tells the process with PID to stop; you supply the actual PID.

```
# kill -9 PID
```

is the signal for `kill` (`9` is the number of the SIGKILL signal); use this combination when the plain `kill` shown previously does not work.

```
# kill -SIGHUP PID
```

is the signal to "hang up"—stop—and then clean up all associated processes as well. (Its number is `-1`.)

As you become proficient at process control and job control, you will learn the utility of a number of `kill` options. A full list of signal options can be found in the `signal` man page.

Using Priority Scheduling and Control

No process can make use of the system's resources (CPU, memory, disk access, and so on) as it pleases. After all, the kernel's primary function is to manage the system resources equitably. It does this by assigning a priority to each process so that some processes get better access to system resources and some processes might have to wait longer until their turn arrives. Priority scheduling can be an important tool in managing a system supporting critical applications or in a situation in which CPU and RAM usage must be reserved or allocated for a specific task. Two legacy applications included with Ubuntu include the `nice` and `renice` commands. (`nice` is part of the GNU `sh-utils` package, whereas `renice` is inherited from BSD Unix.)

The `nice` command is used with its `-n` option, along with an argument in the range of `-20` to `19`, in order from highest to lowest priority (the lower the number, the higher the priority). For example, to run the `gkrellm` client with a low priority, use the `nice` command like this:

```
$ nice -n 12 gkrellm &
```

The `nice` command is typically used for disk- or CPU-intensive tasks that might be obtrusive or cause system slowdown. The `renice` command can be used to reset the priority of running processes or control the priority and scheduling of all processes owned by a user. Regular users can only numerically increase process priorities (that is, make tasks less important) with this command, but the root operator can use the full `nice` range of scheduling (`-20` to `19`).

System administrators can also use the `time` command to get an idea of how much time and what proportion of a system's resources are required for a task, such as a shell script. (Here, `time` is used to measure the duration of elapsed time; the command that deals with civil and sidereal time is the `date` command.) This command is used with the name of another command (or script) as an argument like this:

```
# time -p find / -name core -print
/dev/core
/proc/sys/net/core

real 61.23
```

```
user 0.72

sys 3.33
```

Output of the command displays the time from start to finish, along with the user and system time required. Other factors you can query include memory, CPU usage, and file system input/output (I/O) statistics. See the `time` command's man page for more details.

Nearly all graphical process-monitoring tools include some form of process control or management. Many of the early tools ported to Linux were clones of legacy Unix utilities. One familiar monitoring (and control) program is `top`. Based on the `ps` command, the `top` command provides a text-based display of constantly updated console-based output showing the most CPU-intensive processes currently running. It can be started like this:

```
$ top
```

After you press Enter, you see a display as shown in Figure 12.1. The `top` command has a few interactive commands: Pressing `h` displays the help screen; pressing `k` prompts you to enter the `pid` of a process to kill; pressing `n` prompts you to enter the `pid` of a process to change its nice value. The `top` man page describes other commands and includes a detailed description of what all the columns of information `top` can display actually represent; have a look at top's well-written `man` page.

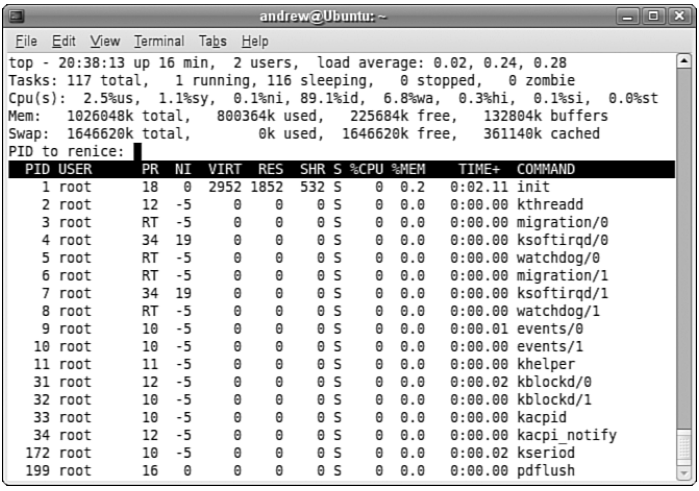


FIGURE 12.1 The *`top`* command can be used to monitor and control processes. Here, we are prompted to *`renice`* a process.

The `top` command displays quite a bit of information about your system. Processes can be sorted by PID, age, CPU or memory usage, time, or user. This command also provides process management, and system administrators can use its `k` and `r` keypress commands to kill and reschedule running tasks, respectively.

The `top` command uses a fair amount of memory, so you might want to be judicious in its use and not leave it running all the time. When you've finished with it, simply press `q` to quit `top`.

Displaying Free and Used Memory with `free`

Although `top` includes some memory information, the `free` utility displays the amount of free and used memory in the system in kilobytes (the `-m` switch displays in megabytes). On one system, the output looks like this:

```
$ free
             total       used       free     shared    buffers     cached
Mem:       1026048     808388     217660          0     132896     362360
-/+ buffers/cache:     313132     712916
Swap:      1646620          0    1646620
andrew@Ubuntu:~$
```

This output describes a machine with 1GB of RAM memory and a swap partition of 1.6GB. Note that none of the swap is being used, and that the machine is not heavily loaded. Linux is very good at memory management and "grabs" all the memory it can in anticipation of future work.

TIP

A useful trick is to employ the `watch` command; it repeatedly reruns a command every two seconds by default. If you use

```
$ sudo watch free
```

you can see the output of the `free` command updated every two seconds.

Another useful system-monitoring tool is `vmstat` (*virtual memory statistics*). This command reports on processes, memory, I/O, and CPU, typically providing an average since the last reboot; or you can make it report usage for a current period by telling it the time interval in seconds and the number of iterations you desire, like

```
$ vmstat 5 10
```

which runs `vmstat` every five seconds for 10 iterations.

Use the `uptime` command to see how long it has been since the last reboot and to get an idea of what the load average has been; higher numbers mean higher loads.

Disk Space

Besides system load, it's important to keep an eye on the amount of free hard drive space that your computer has remaining.

It's very easy to do this, mainly by using the `df` command, like so:

```
$ df
```

Just using the command as is can give you a lot of gobbledegook like so,

```
Filesystem           1K-blocks      Used Available Use% Mounted on

/dev/sda1             36835176   6829896  28134112  20% /

varrun                  513024       160    512864   1% /var/run

varlock                 513024         0    513024   0% /var/lock

udev                    513024        72    512952   1% /dev

devshm                  513024         0    513024   0% /dev/shm

lrm                     513024     34696    478328   7% /lib/modules/\
2.6.22-12-generic/volatile
```

Here you can see each drive as mounted on your system, as well as the used space, the available space, the percentage of the total usage of the disk, and finally where it is mounted on your system.

Unless you're good at math, you may find it difficult to work out exactly what the figures mean in megabytes and gigabytes, so it's recommended that you use the `-h` switch to make the output human readable, like so:

```
Filesystem            Size  Used Avail Use% Mounted on

/dev/sda1              36G  6.6G   27G  20% /

varrun                501M  160K  501M   1% /var/run

varlock               501M     0  501M   0% /var/lock

udev                  501M   72K  501M   1% /dev

devshm                501M     0  501M   0% /dev/shm

lrm                   501M   34M  468M   7% /lib/modules/2.6.22-12-generic/volatile
```

Disk Quotas

Disk quotas are a way to restrict the usage of disk space either by user or by groups. Although rarely—if ever—used on a local or standalone workstation, quotas are definitely a way of life at the enterprise level of computing. Usage limits on disk space not only conserve resources, but also provide a measure of operational safety by limiting the amount of disk space any user can consume.

Disk quotas are more fully covered in Chapter 10, "Managing Users."

Graphical Process and System Management Tools

The GNOME and KDE desktop environments offer a rich set of network and system-monitoring tools. Graphical interface elements, such as menus and buttons, and graphical output, including metering and real-time load charts, make these tools easy to use. These clients, which require an active X session and (in some cases) root permission, are included with Ubuntu.

If you view the graphical tools locally while they are being run on a server, you must have X properly installed and configured on your local machine. Although some tools can be used to remotely monitor systems or locally mounted remote file systems, you have to properly configure pertinent X11 environment variables, such as `$DISPLAY`, to use the software or use the `ssh` client's `-X` option when connecting to the remote host.

A good graphical process and system management tool is GKrellM, which provides a neat interface and a host of additional plugins. You have to use this command to retrieve GKrellM:

```
$ sudo apt-get install gkrellm
```

or you can use `synaptic` and after installed it can be found under Applications, System Tools. GKrellM is shown in Figure 12.2.

Some of the graphical system- and process-monitoring tools that come with Ubuntu include the following:

- `vncviewer`—AT&T's open source remote session manager (part of the `Xvnc` package), which can be used to view and run a remote desktop session locally. This software (discussed in more detail in Chapter 15, "Remote Access with SSH and Telnet") requires an active, background, X session on the remote computer.
- `gnome-nettool`—A GNOME-developed tool that enables system administrators to carry out a wide range of diagnostics on network interfaces, including port scanning and route tracing.
- `ethereal`—This graphical network protocol analyzer can be used to save or display packet data in real time and has intelligent filtering to recognize data signatures or patterns from a variety of hardware and data captures from third-party data capture programs, including compressed files. Some protocols include AppleTalk, Andrew File System (AFS), AOL's Instant Messenger, various Cisco protocols, and many more.

FIGURE 12.2 GKrellM allows you to monitor most system processes on Ubuntu.

- `gnome-system-monitor`—This tool is a simple process monitor offering three views: a list view, a moving graph, and a storage status overview. To access it, choose System, Administration, System Monitor (see Figure 12.3).

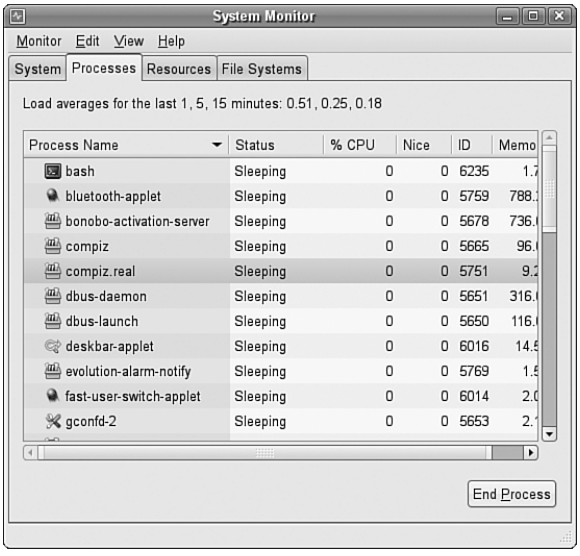


FIGURE 12.3 The Process Listing view of the System Monitor.

From the Process Listing view (chosen via the Processes tab in the upper-left portion of the window), select a process and click on More Info at the bottom left of the screen to display details on that process at the bottom of the display. You can select from three views to filter the display, available in the drop-down View list: All Processes, My Processes (those you alone own), or Active Processes (all processes that are active).

Choose Hidden Processes under the Edit command accessible from the top of the display to show any hidden processes (those that the kernel does not enable the normal monitoring tools to see). Select any process and kill it with End Process.

You can `renice` the processes by selecting Edit, Change Priority. The View selection from the menu bar also provides a memory map. In the Resource Monitor tab, you can view a moving graph representing CPU and memory usage (see Figure 12.4).

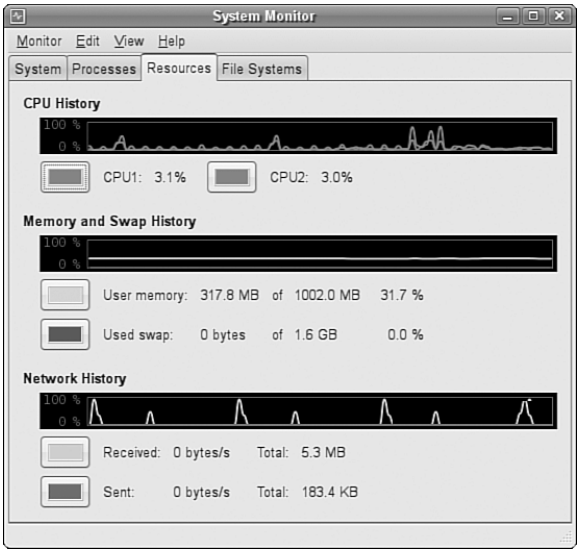


FIGURE 12.4 The Graph view of the System Monitor. It shows CPU usage, Memory/Swap usage, and disk usage. To get this view, select the Resource Monitor tab.

Finally, you can see the status of your file system and storage devices by clicking the File Systems tab shown in Figure 12.5.

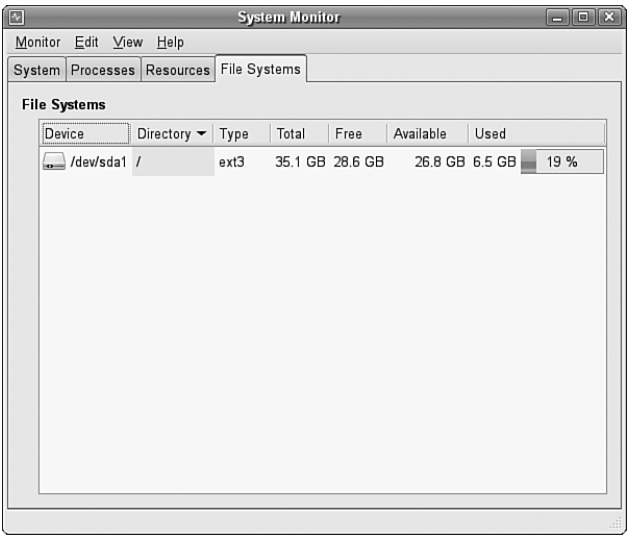


FIGURE 12.5 Keep a close eye on your free disk space.

KDE Process- and System-Monitoring Tools

KDE provides several process- and system-monitoring clients. Integrate the KDE graphical clients into the desktop taskbar by right-clicking on the taskbar and following the menus.

These KDE monitoring clients include the following:

- `kdf`—A graphical interface to your system's file system table that displays free disk space and enables you to mount and unmount file systems with a pointing device.
- `ksysguard`—Another panel applet that provides CPU load and memory use information in animated graphs.

Reference

- http://and.sourceforge.net/—Home page of the `and` auto-`nice` daemon, which can be used to prioritize and reschedule processes automatically.
- http://sourceforge.net/projects/schedutils/—Home page for various projects offering scheduling utilities for real-time scheduling.
- http://freetype.sourceforge.net/patents.html—A discussion of the FreeType bytecode interpreter patents.
- http://www.ethereal.com/—Home page for the Ethereal client.
- http://www.realvnc.com/—The home page for the Virtual Network Computing remote desktop software, available for a variety of platforms, including Ubuntu. This software has become so popular that it is now included with nearly every Linux distribution.

This page intentionally left blank

CHAPTER 13

Backing Up

IN THIS CHAPTER

This chapter examines the practice of safeguarding data through backups, restoring that same data if necessary, and recovering data in case of a catastrophic hardware or software failure. After reading this chapter, you will have a full understanding of the reasons for sound backup practices. You can use the information in this chapter to make intelligent choices about which strategies are best for you. The chapter also shows you how to perform some types of data recovery and system restoration on your own and when to seek professional assistance.

Choosing a Backup Strategy

Backups are always trade-offs. Any backup will consume time, money, and effort on an ongoing basis; backups must be monitored, validated, indexed, stored, and new media continuously purchased. Sound expensive? The cost of not having backups is the loss of your critical data. Re-creating the data from scratch will cost time and money, and if the cost of doing it all again is greater than the cost associated with backing up, you should be performing backups. At the where-the-rubber-meets-the-road level, backups are nothing more than insurance against financial loss for you or your business.

Your first step in formulating and learning to use an effective backup strategy is to choose the strategy that is right for you. First, you must understand some of the most common (and not so common) causes of data loss so that you are better able to understand the threats your system faces. Then, you need to assess your own system, how it is used and by whom, your available hardware and software resources, and your budget constraints. The following

sections look at each of these issues in detail, as well as offering some sample backup systems and discussing their use.

Why Data Loss Occurs

Files disappear for any number of reasons: They can be lost because the hardware fails and causes data loss; your attention might wander and you accidentally delete or overwrite a file. Some data loss occurs as a result of natural disasters and other circumstances beyond your control. A tornado, flood, or earthquake could strike, the water pipes could burst, or the building could catch on fire. Your data, as well as the hardware, would likely be destroyed in such a disaster. A disgruntled employee might destroy files or hardware in an attempt at retribution. And any equipment might fail, and it all will fail at some time—most likely when it is extremely important for it not to fail.

A Case in Point

A recent Harris poll of Fortune 500 executives found that roughly two-thirds of them had problems with their backups and disaster-recovery plans. How about you?

Data can also be lost because of malfunctions that corrupt the data as it attempts to write to the disk. Other applications, utilities, and drivers might be poorly written, buggy (the phrase most often heard is "still beta quality"), or might suffer some corruption and fail to correctly write that all-important data you have just created. If that happened, the contents of your data file would be indecipherable garbage of no use to anyone.

All of these accidents and disasters offer important reasons for having a good backup strategy; however, the most frequent cause of data loss is human error. Who among us has not overwritten a new file with an older version or unintentionally deleted a needed file? This applies not only to data files, but also to configuration files and binaries. While perusing the mail lists or the Usenet newsgroup postings, stories about deleting entire directories such as `/home`, `/usr`, or `/lib` seem all too common. Incorrectly changing a configuration file and not saving the original in case it has to be restored (which it does more often than not because the person reconfigured it incorrectly) is another common error.

TIP

To make a backup of a configuration file you are about to edit, use the `cp` command:

```
$ cp filename filename.original
```

And to restore it:

```
$ cp filename.original filename
```

Never edit or move the `*.original file`, or the original copy will be lost.

Proper backups can help you recover from these problems with a minimum of hassle, but you have to put in the effort to keep backups current, verify their intactness, and practice restoring the data in different disaster scenarios.

Assessing Your Backup Needs and Resources

By now you have realized that some kind of plan is needed to safeguard your data, and, like most people, you are overwhelmed by the prospect. Entire books, as well as countless articles and whitepapers, have been written on the subject of backing up and restoring data. What makes the topic so complex is that each solution is truly individual.

Yet, the proper approach to making the decision is very straightforward. You start the process by asking

- What data must be safeguarded?
- How often does the data change?

The answers to these two questions determine how important the data is, determine the volume of the data, and determine the frequency of the backups. This in turn will determine the backup medium. Only then can the software be selected that will accommodate all these considerations. (You learn about choosing backup software, hardware, and media later in this chapter.)

Available resources are another important consideration when selecting a backup strategy. Backups require time, money, and personnel. Begin your planning activities by determining what limitations you face for all of these resources. Then, construct your plan to fit those limitations, or be prepared to justify the need for more resources with a careful assessment of both backup needs and costs.

TIP

If you are not willing or capable of assessing your backup needs and choosing a backup solution, there exists a legion of consultants, hardware vendors, and software vendors who would love to assist you. The best way to choose one in your area is to query other UNIX and Linux system administrators (located through user groups, discussion groups, or mail lists) who are willing to share their experiences and make recommendations. If you cannot get a referral, ask the consultant for references and check them out.

Many people also fail to consider the element of time when formulating their plan. Some backup devices are faster than others, and some recovery methods are faster than others. You need to consider that when making choices.

To formulate your backup plan, you need to determine the frequency of backups. The necessary frequency of backups should be determined by how quickly the important data on your system changes. On a home system, most files never change, a few change daily,

and some change weekly. No elaborate strategy needs to be created to deal with that. A good strategy for home use is to back up (to any kind of removable media) critical data frequently and back up configuration and other files weekly.

At the enterprise level on a larger system with multiple users, a different approach is called for. Some critical data is changing constantly, and it could be expensive to re-create; this typically involves elaborate and expensive solutions. Most of us exist somewhere in between these extremes. Assess your system and its use to determine where you fall in this spectrum.

Backup schemes and hardware can be elaborate or simple, but they all require a workable plan and faithful follow-through. Even the best backup plan is useless if the process is not carried out, data is not verified, and data restoration is not practiced on a regular basis. Whatever backup scheme you choose, be sure to incorporate in it these three principles:

- **Have a plan**—Design a plan that is right for your needs and have equipment appropriate to the task. This involves assessing all the factors that affect the data you are backing up. We will get into more detail later in the chapter.
- **Follow the plan**—Faithfully complete each part of your backup strategy, and then verify the data stored in the backups. Backups with corrupt data are of no use to anyone. Even backup operations can go wrong.
- **Practice your skills**—Practice restoring data from your backup systems from time to time, so when disaster strikes, you are ready (and able) to benefit from the strength of your backup plan. (For restoring data, see the section "Using Backup Software.") Keep in mind that it is entirely possible that the flaws in your backup plan will only become apparent when you try restoring!

Sound Practices

You have to create your own best-backup plan, but here are some building blocks that go into the foundation of any sound backup program:

- Maintain more than one copy of critical data.
- Label the backups.
- Store the backups in a climate-controlled and secure area.
- Use secure, offsite storage of critical data. Many companies choose bank vaults for their offsite storage, and this is highly recommended.
- Establish a backup policy that makes sense and can be followed religiously. Try to back up your data when the system is consistent (that is, no data is being written), which is usually overnight.
- Keep track of who has access to your backup media, and keep the total number of people as low as possible. If you can, allow only trusted personnel near your backups.

- Routinely verify backups and practice restoring data from them.
- Routinely inspect backup media for defects and regularly replace them (after destroying the data on them if it is sensitive).

Evaluating Backup Strategies

Now that you are convinced you need backups, you need a strategy. It is difficult to be specific about an ideal strategy because each user or administrator's strategy will be highly individualized, but here are a few general examples:

- **Home user**—At home, the user has the Ubuntu installation DVD that takes an hour or so to reinstall, so the time issue is not a major concern. The home user will want to back up any configuration files that have altered, keep an archive of any files that have been downloaded, and keep an archive of any data files created while using any applications. Unless the home user has a special project in which constant backups are useful, a weekly backup is adequate. The home user will likely use a DVD-RW drive or other removable media for backups.
- **Small office**—Many small offices tend to use the same strategy as the home user, but are more likely to back up critical data daily and use manually changed tape drives. If they have a tape drive with adequate storage, they will likely have a full system backup as well because restoring from the tape is quicker than reinstalling from the CDs. They also might be using CD-RW or DVD writers for backups. Although they will use scripts to automate backups, most of it is probably done by hand.
- **Small enterprise**—Here is where backups begin to require higher-end equipment such as autoloading tape drives with fully automated backups. Commercial backup software usually makes an introduction at this level, but a skillful system administrator on a budget can use one of the basic applications discussed in this chapter. Backups are highly structured and supervised by a dedicated system administrator.
- **Large enterprise**—These are the most likely settings for the use of expensive, proprietary, highly automated backup solutions. At this level, data means money, lost data means lost money, and delays in restoring data means money lost as well. These system administrators know that backups are necessary insurance and plan accordingly.

Does all this mean that enterprise-level backups are better than those done by a home user? Not at all. The "little guy" with Ubuntu can do just as well as the enterprise operation at the expense of investing more time in the process. By examining the higher-end strategies, we can apply useful concepts across the board.

NOTE

If you are a new sysadmin, you might be inheriting an existing backup strategy. Take some time to examine it and see if it meets the current needs of the organization. Think about what backup protection your organization really needs, and determine if the current strategy meets that need. If it does not, change the strategy. Consider whether the current policy is being practiced by the users, and, if not, why it is not.

Backup Levels

UNIX uses the concept of backup levels as a shorthand way of referring to how much data is backed up in relation to a previous backup. It works this way:

A level 0 backup is a full backup. The next backup level would be 1.

Backups at the other numbered levels will back up everything that has changed since the last backup at that level or a numerically higher level (the `dump` command, for example, offers 10 different backup levels). For example, a level 3 followed by a level 4 will generate an incremental backup from the full backup, whereas a level 4 followed by a level 3 will generate a differential backup between the two.

The following sections examine a few of the many strategies in use today. Many strategies are based on these sample schemes; one of them can serve as a foundation for the strategy you construct for your own system.

Simple Strategy

If you need to back up just a few configuration files and some small data files, copy them to a USB stick, engage the write-protect tab, and keep it someplace safe. If you need just a bit more backup storage capacity, you can copy the important files to a Zip disk (100,250 and 750MB in size), CD-RW disk (up to 700MB in size), or DVD-RW disk (up to 8GB for data).

In addition to configuration and data files, you should archive each user's home directory, as well as the entire `/etc` directory. Between the two, that backup would contain most of the important files for a small system. Then, you can easily restore this data from the backup media device you have chosen, after a complete reinstall of Ubuntu, if necessary.

Experts believe that if you have more data than can fit on a floppy disk, you really need a formal backup strategy. Some of those are discussed in the following sections. We use a tape media backup as an example.

Full Backup on a Periodic Basis

This backup strategy involves a backup of the complete file system on a weekly, bi-weekly, or other periodic basis. The frequency of the backup depends on the amount of data being backed up, the frequency of changes to the data, and the cost of losing those changes.

This backup strategy is not complicated to perform, and it can be accomplished with the swappable disk drives discussed later in the chapter. If you are connected to a network, it is possible to mirror the data on another machine (preferably offsite); the `rsync` tool is particularly well suited to this task. Recognize that this does not address the need for archives of the recent state of files; it only presents a snapshot of the system at the time the update is done.

Full Backups with Incremental Backups

This scheme involves performing a full backup of the entire system once a week, along with a daily incremental backup of only those files that have changed in the previous day, and it begins to resemble what a sysadmin of a medium to large system would traditionally use.

This backup scheme can be advanced in two ways. In one way, each incremental backup can be made with reference to the original full backup. In other words, a level 0 backup is followed by a series of level 1 backups. The benefit of this backup scheme is that a restoration requires only two tapes (the full backup and the most recent incremental backup). But because it references the full backup, each incremental backup might be large (and grow ever larger) on a heavily used system.

Alternatively, each incremental backup could reference the previous incremental backup. This would be a level 0 backup followed by a level 1, followed by a level 2, and so on. Incremental backups are quicker (less data each time), but require every tape to restore a full system. Again, it is a classic trade-off decision.

Modern commercial backup applications such as Amanda or BRU assist in organizing the process of managing complex backup schedules and tracking backup media. Doing it yourself using the classic dump or employing shell scripts to run `tar` requires that the system administrator handle all the organization herself. For this reason, complex backup situations are typically handled with commercial software and specialized hardware that are packaged, sold, and supported by vendors.

Mirroring Data or RAID Arrays

Given adequate (and often expensive) hardware resources, you can always mirror the data somewhere else, essentially maintaining a real-time copy of your data on hand. This is often a cheap, workable solution if no large amounts of data are involved. The use of RAID arrays (in some of their incarnations) provides for a recovery if a disk fails.

Note that RAID arrays and mirroring systems will just as happily write corrupt data as valid data. Moreover, if a file is deleted, a RAID array will not save it. RAID arrays are best suited for protecting the current state of a running system, not for backup needs.

Making the Choice

Only you can decide what is best for your situation. After reading about the backup options in this book, put together some sample backup plans; run through a few likely scenarios and assess the effectiveness of your choice.

In addition to all the other information you have learned about backup strategies, here are a couple of good rules of thumb to remember when making your choice:

- If the backup strategy and policy is too complicated (and this holds true for most security issues), it will eventually be disregarded and fall into disuse.
- The best scheme is often a combination of strategies; use what works.

Choosing Backup Hardware and Media

Any device that can store data can be used to back it up, but that is like saying that anything with wheels can take you on a cross-country trip. Trying to fit a gigabyte worth of data on a big stack of CD-RWs is an exercise in frustration, and using an expensive automated tape device to save a single copy of an email is a waste of resources.

Many people use what hardware they already have for their backup operations. Many consumer-grade workstations have a CD-RW drive, but they typically do not have the abundant free disk space necessary for performing and storing multiple full backups.

In this section, you learn about some of the most common backup hardware available and how to evaluate its appropriateness for your backup needs. With large storage devices becoming increasingly affordable (160GB IDE drives can be had for around $100) and prices falling on DVD recorders, decisions about backup hardware for the small business and home users have become more interesting.

Removable Storage Media

Choosing the right media for you isn't as easy as it used to be back when floppy drives were the only choice. Today, most machines have CD-ROM drives that can read but not write CDs, which rules them out for backup purposes. Instead, USB hard drives and solid-state "pen" drives have taken over the niche previously held by floppy drives: you can get a 256MB drive for under $10, and you can even get capacities up to 16GB for good prices if you shop around. If your machine supports them (or if you have purchased a card reader), you can also use Compact Flash devices, which come in sizes up to 8GB in the Flash memory versions and 4GB for Hitachi Microdrives. Both USB hard drives and solid-state drives are highly portable. Support for these drives under Ubuntu is very good, accommodating these drives by emulating them as SCSI drives—the system usually sees them as `/dev/scd1`. Watch for improved support and ever falling prices in the future. A 500GB USB hard drive will cost about $150. The biggest benefits of USB drives are data transfer speed and portability.

FireWire Drives

FireWire (IEEE-1394) hard drives are similar to USB drives; they just use a different interface to your computer. Many digital cameras and portable MP3 players use FireWire. Kernel support is available if you have this hardware. The cost of FireWire devices is now essentially the same as USB because many external drives come with both USB and FireWire standard.

CD-RW and DVD+RW/-RW Drives

Compared to floppy drives and some removable drives, CD-RW drives and their cousins, DVD+RW/-RW drives, can store large amounts of data and are useful for a home or small business. Once very expensive, CD writers and media are at commodity prices today, although automated CD changing machines, necessary for automatically backing up large amounts of data, are still quite costly. A benefit of CD and DVD storage over tape devices is that the archived uncompressed file system can be mounted and its files accessed randomly just like a hard drive (you do this when you create a data CD, refer to Chapter 10, "Multimedia Applications"), making the recovery of individual files easier.

Each CD-RW disk can hold 650MB–700MB of data (the media comes in both capacities at roughly the same cost); larger chunks of data can be split to fit on multiple disks. Some backup programs support this method of storage. Once burned and verified, the shelf life for the media is at least a decade or longer. Prices increase with writing speed, but a serviceable CD-RW drive can be purchased for less than $100.

DVD+RW/-RW is similar to CD-RW, but it is more expensive and can store up to 8GB of uncompressed data per disk. These drives are selling for less than $50.

Network Storage

For network backup storage, remote arrays of hard drives provide one solution to data storage. With the declining cost of mass storage devices and the increasing need for larger storage space, network storage (NAS, or Network Attached Storage) is available and supported in Linux. These are cabinets full of hard drives and their associated controlling circuitry, as well as special software to manage all of it. These NAS systems are connected to the network and act as a huge (and expensive) mass storage device.

More modest and simple network storage can be done on a remote desktop-style machine that has adequate storage space (up to eight 250GB IDE drives is a lot of storage space, easily accomplished with off-the-shelf parts), but then that machine (and the local system administrator) has to deal with all the problems of backing up, preserving, and restoring its own data, doesn't it? Several hardware vendors offer such products in varying sizes.

Tape Drive Backup

Tape drives have been used in the computer industry from the beginning. Tape drive storage has been so prevalent in the industry that the `tar` command (the most commonly used command for archiving) is derived from the words Tape ARchive. Modern tape

drives use tape cartridges that can hold 70GB of data (or more in compressed format). Capacities and durability of tapes vary from type to type and range from a few gigabytes to hundreds of gigabytes with commensurate increases in cost for the equipment and media. Autoloading tape-drive systems can accommodate archives that exceed the capacity of the file systems.

TIP

Older tape equipment is often available in the used equipment market and might be useful for smaller operations that have outgrown more limited backup device options.

Tape equipment is well supported in Linux and, when properly maintained, is extremely reliable. The tapes themselves are inexpensive, given their storage capacity and the ability to reuse them. Be aware, however, that tapes do deteriorate over time and, being mechanical, tape drives can and will fail.

CAUTION

Neglecting to clean, align, and maintain tape drives puts your data at risk. The tapes themselves are also susceptible to mechanical wear and degradation. Hardware maintenance is part of a good backup policy. Do not ever forget that it is a question of when—not if—hardware will fail.

Using Backup Software

Because there are thousands of unique situations requiring as many unique backup solutions, it comes as no surprise that Linux offers many backup tools. Along with command-line tools such as `tar` and `dd`, Ubuntu also provides a graphical archiving tool, File Roller, that can create and extract files from archives. Finally, Ubuntu provides support for the Amanda backup application—a sophisticated backup application that works well over network connections and can be configured to automatically back up all the computers on your network. Amanda works with drives as well as tapes. The book *Unix Backup and Recovery* by W. Curtis Preston includes a whole chapter on setting up and using Amanda, and this chapter is available online at http://www.backupcentral.com/amanda.html.

NOTE

The software in a backup system must support the hardware, and this relationship can determine which hardware or software choices you make. Many sysadmins choose a particular backup software not because they prefer it to other options, but because it supports the hardware they own.

The price seems right for free backup tools, but consider the software's ease of use and automation when assessing costs. If you must spend several hours implementing, debugging, documenting, and otherwise dealing with overly elaborate automation scripts, the real costs go up.

tar: The Most Basic Backup Tool

The tar tool, the bewhiskered old man of archiving utilities, is installed by default. It is an excellent tool for saving entire directories full of files. For example, here is the command used to back up the /etc directory:

```
sudo tar cvf etc.tar /etc
```

Here, the options use tar to create an archive, be verbose in the message output, and use the filename etc.tar as the archive name for the contents of the directory /etc.

Alternatively, if the output of tar is sent to the standard output and redirected to a file, the command appears as follows:

```
sudo tar cv /etc > etc.tar
```

and the result is the same.

All files in the /etc directory will be saved to a file named etc.tar. With an impressive array of options (see the man page), tar is quite flexible and powerful in combination with shell scripts. With the -z option, it can even create and restore gzip compressed archives while the -j option works with bzipped files.

Creating Full and Incremental Backups with tar

If you want to create a full backup,

```
sudo tar cjvf fullbackup.tar.bz2 /
```

will create a bzip2 compressed tarball (the j option) of the entire system.

To perform an incremental backup, you must locate all the files that have been changed since the last backup. For simplicity, assume that you do incremental backups on a daily basis. To locate the files, use the find command:

```
sudo find / -newer name_of_last_backup_file ! -a –type f –print
```

When run alone, find will generate a list of files systemwide and print it to the screen. The ! -a -type eliminates everything but regular files from the list; otherwise, the entire directory would be sent to tar even if the contents was not all changed.

Pipe the output of our find command to tar as follows:

```
sudo find / -newer name_of_last_backup_file ! –type d -print ¦\
 tar czT - backup_file_name_or_device_name
```

Here, the T - option gets the filenames from a buffer (where the - is the shorthand name for the buffer).

NOTE

The `tar` command can back up to a raw device (one with no file system) as well as a formatted partition. For example,

```
sudo tar cvzf /dev/hdd  /boot  /etc /home
```

backs up those directories to device `/dev/hdd` (not `/dev/hda1`, but to the unformatted device itself).

The `tar` command can also back up over multiple floppy disks:

```
sudo tar czvMf /dev/fd0 /home
```

will back up the contents of `/home` and spread the file out over multiple floppies, prompting you with this message:

```
Prepare volume #2 for `/dev/fd0' and hit return:
```

Restoring Files from an Archive with `tar`

The `xp` option in `tar` will restore the files from a backup and preserve the file attributes as well, and `tar` will create any subdirectories it needs. Be careful when using this option because the backups might have been created with either relative or absolute paths. You should use the `tvf` option with `tar` to list the files in the archive before extracting them so that you will know where they will be placed.

For example, to restore a `tar` archive compressed with `bzip2`,

```
sudo tar xjvf ubuntutest.tar.bz2
```

To list the contents of a `tar` archive compressed with `bzip2`,

```
sudo tar tjvf ubuntutest.tar.bz2
drwxrwxr-x paul/paul         0 2003-09-04 18:15:05 ubuntutest/
-rw-rw-r-- paul/paul       163 2003-09-03 22:30:49
                           ℜubuntutest/ubuntu_screenshots.txt
-rw-rw-r-- paul/paul       840 2003-09-01 19:27:59
                           ℜubuntutest/a_guideline.txt
-rw-rw-r-- paul/paul      1485 2003-09-01 18:14:23 ubuntutest/style-sheet.txt
-rw-rw-r-- paul/paul       931 2003-09-01 19:02:00 ubuntutest/ubuntu_TOC.txt
```

Note that because the pathnames do not start with a backslash, they are relative pathnames and will install in your current working directory. If they were absolute pathnames, they would install exactly where the paths state.

The GNOME File Roller

The GNOME desktop file archiving graphical application File Roller (`file-roller`) will view, extract, and create archive files using `tar`, `gzip`, `bzip`, `compress`, `zip`, `rar`, `lha`, and several other compression formats. Note that File Roller is only a front-end to the command-line utilities that actually provide these compression formats; if they are not installed, File Roller cannot use that format.

CAUTION

File Roller will not complain if you select a compression format that is not supported by installed software until after you attempt to create the archive. Install any needed compression utilities first.

File Roller is well-integrated with the GNOME desktop environment to provide convenient drag-and-drop functionality with the Nautilus file manager. To create a new archive, select Archive, New to open the New Archive dialog box and navigate to the directory where you want the archive to be kept. Type your archive's name in the Selection: /root text box at the bottom of the New Archive dialog box. Use the Archive type drop-down menu to select a compression method. Now, drag the files that you want to be included from Nautilus into the empty space of the File Roller window, and the animated icons will show that files are being included in the new archive. When you are done, a list of files will be shown in the previously blank File Roller window (see Figure 13.1). To save the archive, simply select Archive, Close. Opening an archive is as easy as using the Archive, Open dialog to select the appropriate archive file.

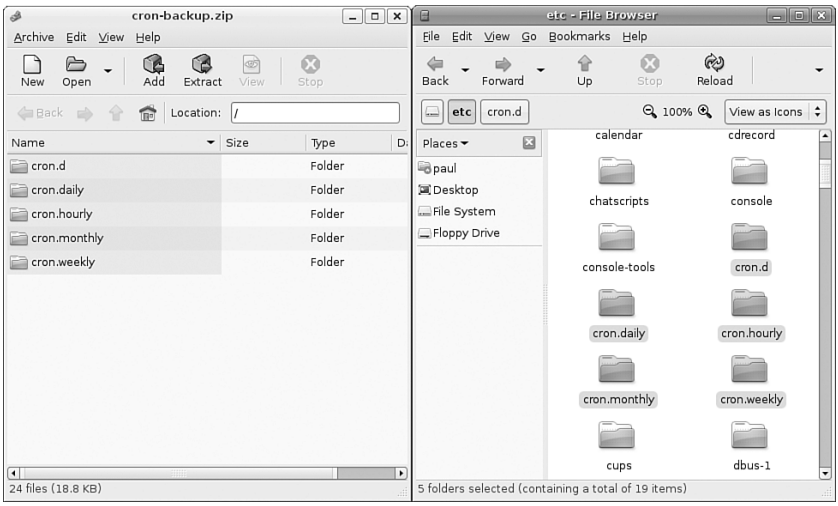


FIGURE 13.1 Drag and drop files to build an archive with the GNOME File Roller.

Ubuntu provides you with the KDE `ark` and `kdat` GUI tools for backups; they are installed only if you select the KDE desktop during installation, but you can search through Synaptic to find them. Archiving has traditionally been a function of the system administrator and not seen as a task for the individual user, so no elaborate GUI was believed necessary. Backing up has also been seen as a script driven, automated task in which a GUI is not as useful. Although that's true for sysadmins, home users usually want something a little more attractive and easier to use, and that's the exact gap filled by `ark`.

The KDE `ark` Archiving Tool

You launch `ark` by launching it from the command line. It is integrated with the KDE desktop (like File Roller is with GNOME), so it might be a better choice if you use KDE. This application provides a graphical interface to viewing, creating, adding to, and extracting from archived files as shown in Figure 13.2. Several configuration options are available with `ark` to ensure its compatibility with MS Windows. You can drag and drop from the KDE desktop or Konqueror file browser to add or extract files, or you can use the `ark` menus.

As long as the associated command-line programs are installed, `ark` can work with `tar`, `gzip`, `bzip2`, `zip`, and `lha` files (the latter four being compression methods used to save space by compaction of the archived files).

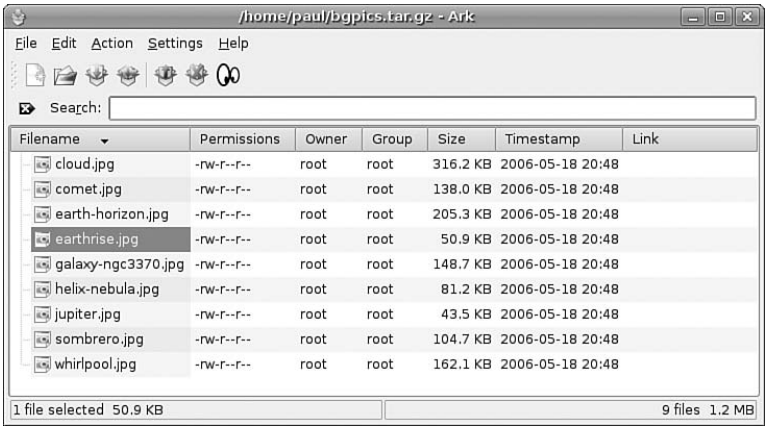


FIGURE 13.2 Here, the contents of a `.zip` file containing some pictures are displayed.

Existing archives are opened after launching the application itself. You can add files and directories to the archive or delete them from the archive, as shown in Figure 13.3. After opening the archive, you can extract all of its contents or individual files. You can also perform searches using patterns (all `*.jpg` files, for example) to select files.

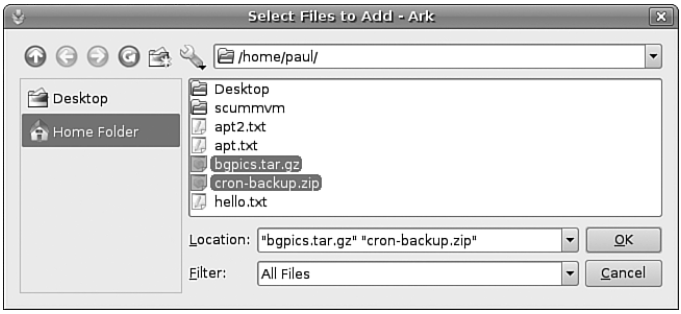


FIGURE 13.3 Adding files to ark is a matter of selecting them in the usual Open File dialog. Here, several files are being selected to add to the new archive.

Choosing New from the File menu creates new archives. You then type the name of the archive, providing the appropriate extension (`.tar`, `.gz`, and so on), and then proceed to add files and directories as you desire.

Using the Amanda Backup Application

Provided with Ubuntu, Amanda is a powerful, network backup application created by the University of Maryland at College Park. Amanda is a robust backup and restore application best suited to unattended backups with an autoloading tape drive of adequate capacity. It benefits from good user support and documentation.

Amanda's features include compression and encryption. It is intended for use with high-capacity tape drives, floptical, CD-R, and CD-RW devices.

Amanda uses GNU `tar` and `dump`; it is intended for unattended, automated tape backups, and is not well-suited for interactive or ad hoc backups. The support for tape devices in Amanda is robust, and file restoration is relatively simple. Although Amanda does not support older Macintosh clients, it will use Samba to back up Microsoft Windows clients, as well as any UNIX client that can use GNU tools (which includes Mac OS X). Because Amanda runs on top of standard GNU tools, file restoration can be made using those tools on a recovery disk even if the Amanda server is not available. File compression can be done on either the client or server, thus lightening the computational load on less powerful machines that need backing up.

CAUTION

Amanda does not support `dump` images larger than a single tape and requires a new tape for each run. If you forget to change a tape, Amanda continues to attempt backups until you insert a new tape, but those backups will not capture the data as you intended them to. Do not use too small a tape or forget to change a tape, or you will not be happy with the results.

There is no GUI interface for Amanda. Configuration is done in the time-honored UNIX tradition of editing text configuration files located in `/etc/amanda`. The default installation in Ubuntu includes a sample `cron` file because it is expected that you will be using `cron` to run Amanda regularly. The client utilities are installed with the package `am-utils`; the Amanda server must be obtained from the Amanda website. As far as backup schemes are concerned, Amanda calculates an optimal scheme on-the-fly and schedules it accordingly. It can be forced to adhere to a traditional scheme, but other tools are possibly better suited for that job.

The `man` page for Amanda (the client is `amdump`) is well written and useful, explaining both the configuration of Amanda as well as detailing the several programs that actually make up Amanda. The configuration files found in `/etc/amanda` are well commented; they provide a number of examples to assist you in configuration.

The program's home page is http://www.amanda.org. There, you will find information on subscribing to the mail list, as well as links to Amanda-related projects and a FAQ.

Alternative Backup Software

Commercial and other freeware backup products do exist; BRU and Veritas are good examples of effective commercial backup products. Here are some useful free software backup tools that are not installed with Ubuntu:

- `flexbackup`—This backup tool is a large file of Perl scripts that makes `dump` and `restore` easier to use. `flexbackup`'s command syntax can be accessed by using the command with the `-help` argument. It also can use `afio`, `cpio`, and `tar` to create and restore archives locally or over a network using `rsh` or `ssh` if security is a concern. Its home page is http://www.flexbackup.org/.
- `afio`—This tool creates `cpio`-formatted archives, but handles input data corruption better than `cpio` (which does not handle data input corruption very well at all). It supports multi-volume archives during interactive operation and can make compressed archives. If you feel the need to use `cpio`, you might want to check out `afio` at http://freshmeat.net/projects/afio/.
- `cdbackup`—Designed for the home or small office user, `cdbackup` will work with any backup and will restore software that can read from `stdin`, write to `stdout`, and can handle linear devices such as tape drives. It makes it easier to use CD-Rs as the storage medium. Similar applications are available elsewhere as well; the home page for this application is at http://www.muempf.de/index.html.

Many other alternative backup tools exist, but covering all of them is beyond the scope of this book. Two good places to look for free backup software are Freshmeat (http://www.freshmeat.net) and Google (http://www.google.com/linux).

Copying Files

Often, when you have only a few files that you need to protect from loss or corruption, it might make better sense to simply copy the individual files to another storage medium rather than to create an archive of them. You can use the `tar`, `cp`, `rsync`, or even the `cpio` commands to do this, as well as a handy file management tool known as `mc`. Using `tar` is the traditional choice because older versions of `cp` did not handle symbolic links and permissions well at times, causing those attributes (characteristics of the file) to be lost; `tar` handled those file attributes in a better manner. `cp` has been improved to fix those problems, but `tar` is still more widely used. `rsync` has recently been added to Ubuntu and is an excellent choice for mirroring sets of files, especially when done over a network.

To illustrate how to use file copying as a backup technique, the examples here show how to copy (not archive) a directory tree. This tree includes symbolic links and files that have special file permissions we need to keep intact.

Copying Files Using `tar`

One choice for copying files into another location would be to use the `tar` command where you would create a `tar` file that would be piped to `tar` to be uncompressed in the new location. To accomplish this, first change to the source directory. Then, the entire command resembles

```
sudo tar cvf - files ¦ (cd target_directory ; tar xpf -)
```

where *files* are the filenames you want to include; use `*` to include the entire current directory.

Here is how the command shown works: You have already changed to the source directory and executed `tar` with the `cvf -` arguments that tell `tar` to

`c`—Create an archive.

`v`—Verbose; lists the files processed so we can see that it is working.

`f`—The filename of the archive will be what follows. (In this case, it is `-`.)

`-`—A buffer; a place to hold our data temporarily.

The following `tar` commands can be useful for creating file copies for backup purposes:

`l`—Stay in the local file system (so you do not include remote volumes).

`atime-preserve`—Do not change access times on files, even though you are accessing them now, to preserve the old access information for archival purposes.

The contents of the `tar` file (held for us temporarily in the buffer, which is named `-`) are then piped to the second expression, which will extract the files to the target directory. In shell programming (refer to Chapter 15, "Automating Tasks"), enclosing an expression in parentheses causes it to operate in a subshell and be executed first.

First we change to the target directory, and then

x—Extract files from a `tar` archive.

p—Preserve permissions.

f—The filename will be `-`, the temporary buffer that holds the `tar`ed files.

Compressing, Encrypting, and Sending `tar` Streams

The file copy techniques using the `tar` command in the previous section can also be used to quickly and securely copy a directory structure across a LAN or the Internet (using the `ssh` command). One way to make use of these techniques is to use the following command line to first compress the contents of a designated directory, and then decompress the compressed and encrypted archive stream into a designated directory on a remote host:

```
$ tar cvzf - data_folder ¦ ssh remote_host `( cd ~/mybackup_dir; tar xvzf - )'
```

The `tar` command is used to create, list, and compress the files in the directory named `data_folder`. The output is piped through the `ssh` (secure shell) command and sent to the remote computer named *`remote_host`*. On the remote computer, the stream is then extracted and saved in the directory named */mybackup_dir*. You will be prompted for a password in order to send the stream.

Copying Files Using `cp`

To copy files, we could use the `cp` command. The general format of the command when used for simple copying is

```
$ cp -a source_directory  target_directory
```

The `-a` argument is the same as giving `-dpR`, which would be

`-d`—Dereferences symbolic links (never follows symbolic links) and copies the files that they point to instead of copying the links.

`-p`—Preserves all file attributes if possible. (File ownership might interfere.)

`-R`—Copies directories recursively.

The `cp` command can also be used to quickly replicate directories and retain permissions by using the `-avR` command-line options. Using these options preserves file and directory permissions, gives verbose output, and recursively copies and re-creates subdirectories. A log of the backup can also be created during the backup by redirecting the standard output like this:

```
sudo cp -avR directory_to_backup  destination_vol_or_dir  1>/root/backup_log.txt
```

or

```
sudo cp -avR ubuntu /test2 1>/root/backup_log.txt
```

This example makes an exact copy of the directory named `/ubuntu` on the volume named `/test2`, and saves a backup report named `backup_log.txt` under `/root`.

Copying Files Using `mc`

The Midnight Commander (available in the Universe repository, under the package "mc"; see Chapter 7, "Managing Software" for how to enable the Universe and Multiverse repositories) is a command-line file manager that is useful for copying, moving, and archiving files and directories. The Midnight Commander has a look and feel similar to the Norton Commander of DOS fame. By executing `mc` at a shell prompt, a dual-pane view of the files is displayed. It contains drop-down menu choices and function keys to manipulate files. It also uses its own virtual file system, enabling it to mount FTP directories and display the contents of `tar` files, gzipped tar files (`.tar.gz` or `.tgz`), `bzip` files, `DEB` files, and `RPM` files, as well as extract individual files from them. As if that was not enough, `mc` contains a File Undelete virtual file system for `ext2/3` partitions. By using `cd` to "change directories" to an FTP server's URL, you can transfer files using the FTP protocol. The default font chosen for Ubuntu makes the display of `mc` ugly when used in a `tty` console (as opposed to an `xterm`), but does not affect its performance.

Figure 13.4 shows a shot of the default dual-panel display. Pressing the `F9` key drops down the menu, and pressing `F1` displays the Help file. A "feature" in the default GNOME terminal intercepts the `F10` key used to exit `mc`, so use F9 instead to access the menu item to quit, or simply click on the menu bar at the bottom with your mouse. The configuration files are well documented, and it would appear easy to extend the functionality of `mc` for your system if you understand shell scripting and regular expressions. It is an excellent choice for file management on servers not running X.

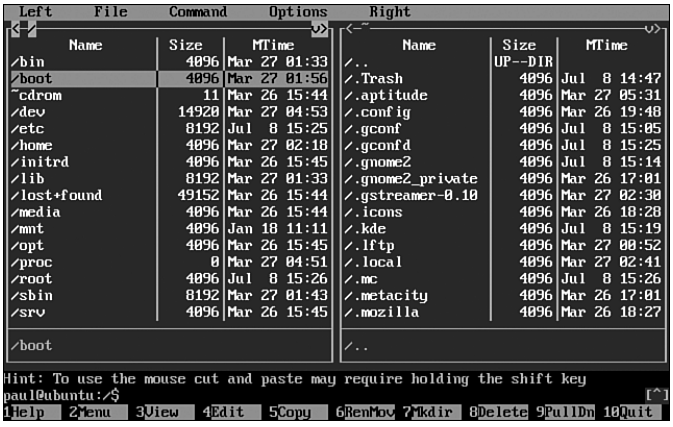


FIGURE 13.4 The Midnight Commander is a highly versatile file tool. If it does not display properly on your screen, launch it with the `-a` argument to force ASCII mode.

System Rescue

There will come a time when you need to engage in system rescue efforts. This need arises when the system will not even start Linux so that you can recover any files. This problem is most frequently associated with the boot loader program or partition table, but it could be that critical system files have been inadvertently deleted or corrupted. If you have been making backups properly, these kinds of system failures are easily, though not quickly, recoverable through a full restore. Still, valuable current data might not have been backed up since the last scheduled backup, and the backup archives are found to be corrupt, incomplete, or missing. A full restore also takes time you might not have. If the problem causing the system failure is simply a damaged boot loader, a damaged partition table, a missing library, or misconfiguration, a quick fix can get the system up and running and the data can then be easily retrieved.

In this section, we will first examine a way to back up and restore the boot loader itself or, having failed to do that, restore it by hand. Then we will look at a few alternatives to booting the damaged system so that we can inspect it, fix it, or retrieve data from it.

The Ubuntu Rescue Disc

Ubuntu provides a rescue disc hidden in the installation DVD. To use it, insert the disc and reboot the computer, booting from the DVD just as you did when you installed Ubuntu originally.

If necessary, you can perform all the operations discussed in this section from rescue mode.

Backing Up and Restoring the Master Boot Record

The Master Boot Record (MBR) is the first 512 bytes of a hard disk. It contains the boot loader code in the first 446 bytes and the partition table in the next 64 bytes; the last two bytes identify that sector as the MBR. The MBR can become corrupted, so it makes sense to back it up.

This example uses the `dd` command as `root` to back up the entire MBR. If the boot loader code changes from the time you make this image and restore the old code, the system will not boot when you restore it all; it is easy enough to keep a boot floppy handy and then re-run LILO if that is what you are using.

To copy the entire MBR to a file, use this:

```
sudo dd if=/dev/hda of=/tmp/hdambr bs=512 count=1
```

To restore the entire MBR, use this:

```
sudo dd if=/tmp/hdambr of=/dev/hda bs=512 count=1
```

To restore only the partition table, skipping the boot loader code, use this:

```
sudo dd if=/tmp/hdambr of=/dev/hda bs=1 skip=446 count=66
```

Of course, it would be prudent to move the copy of the MBR to a floppy or other appropriate storage device. (The file is only 512 bytes in size.) You will need to be able to run `dd` on the system in order to restore it (which means that you will be using the Ubuntu rescue disc as described later, or any equivalent to it).

Booting the System from a Generic Boot Floppy

If you failed to make a boot floppy or cannot locate the one you did make, any Linux boot floppy (a slightly older version or one borrowed from a friend) can be pressed into service as long as it has a reasonably similar kernel version. (The major and minor numbers match—for example, 2.6.5 would likely work with any 2.6 system, but not with a 2.4 system.) You would boot your system by manually specifying the root and boot partitions as described previously. Although you are almost guaranteed to get some error messages, you might at least be able to get a base system running enough to replace files and recover the system.

TIP

In both preceding cases, it is assumed that you do not need any special file system or device drivers to access the root partition. If you do, add the `initrd=` argument to the LILO line pointing to the appropriate `initrd` file on your system. If you do not know the exact name of the `initrd` file, you are out of luck with LILO, so learn to use a GRUB boot floppy as well.

Using a GRUB Boot Floppy

The GRand Unified Boot loader (GRUB) can attempt to boot a system from a floppy without a viable custom-made boot floppy. The image for the floppy can be downloaded from ftp://alpha.gnu.org/gnu/grub/grub-0.95-i386-pc.ext2fs and copied to a floppy using `dd`. (`rawrite.exe` would be used on a Microsoft system.) Or, if you have a boot floppy from an existing system using GRUB, that one will work as well.

GRUB has its own command shell, file system drivers, and search function (much like command completion in the bash shell). It is possible to boot using the GRUB floppy, examine the drive partitions, and search for the kernel and `initrd` image as well, using them to boot the system. Worthy of a chapter all its own, the GRUB documentation is extensive: In addition to `info grub` (the `info` system is similar to the `man` system for documentation), the GRUB documents contain a tutorial worth reading. The GRUB boot loader is shown in Figure 13.5.

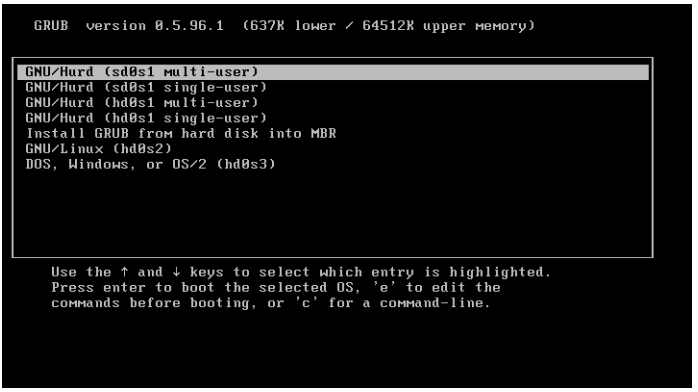


FIGURE 13.5 The GRUB boot loader gives you incredible flexibility in booting even unfamiliar systems.

Using the Recovery Facility

As your computer starts, Ubuntu will show a message prompting you to press Escape to see the GRUB menu. If you do that, you'll see a list of bootable operating systems, which will include Recovery Mode options for your Ubuntu install. If you boot into recovery mode, you'll get automatic root access to your machine where can fix your problems safely.

Alternatively, you can insert your Ubuntu installation disc and use the recovery mode from there. Keep in mind that these recovery modes are designed to be text-only systems for serious system recovery purposes; don't expect them to be hand-held, and don't expect much in the way of ease of use!

Relevant Ubuntu Commands

The following commands are useful in performing backup, recovery, and restore operations in Ubuntu:

`amdump`—Amanda is a network-based backup system, consisting of 18 separate commands, for use with Linux.

`ark`—A KDE desktop GUI archiving utility.

`cp`—The copy command.

`scp`—The secure shell copy command.

`cpio`—A data archive utility.

`dd`—A data copy and conversion utility.

`gzip`—The GNU compression utility.

`tar`—The GNU tape archive utility.

Reference

- http://www.tldp.org/LDP/solrhe/Securing-Optimizing-Linux-RH-Edition-v1.3/whywhen.html—A thorough discussion with examples of using `dump` and `restore` for backups.
- http://en.tldp.org/LDP/solrhe/Securing-Optimizing-Linux-RH-Edition-v1.3/chap29sec306.html—Making automatic backups with `tar` using `cron`.
- http://kmself.home.netcom.com/Linux/FAQs/backups.html—The Linux Backups mini FAQ contains some useful, although brief, comments on backup media, compression, encryption, and security.
- http://www.tldp.org/—The Linux Documentation Project offers several useful HOWTO documents that discuss backups and disk recovery.
- http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/Bootdisk-HOWTO.html#AEN1483—Here is a list of LILO Boot error codes to help you debug a cranky system that will not boot.
- http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/Ftape-HOWTO.html—This is a HOWTO for the floppy tape device driver.
- http://www.linux-usb.org/USB-guide/x498.html—The USB Guide for mass storage devices. If you have a USB device and need to know if it is supported, check here.
- http://www.backupcentral.com/amanda.html—This is the Amanda chapter of *Unix Backup and Recovery* (written by John R. Jackson and published by O'Reilly and Associates). The chapter is available online and covers every aspect of using Amanda. The site features a handy search tool for the chapter.
- http://twiki.org/cgi-bin/view/Wikilearn/RsyncingALargeFileBeginner—Rsyncing a large file to "repair" a local ISO image that does not pass the `md5sum` check.
- http://www.lycoris.org/sections.php?op=viewarticle&artid=8—Lycoris ISO rsync mini HOWTO. A step-by-step tutorial on using `rsync` to sync files.
- http://www.mikerubel.org/computers/rsync_snapshots/—Automated snapshot-style backups using `rsync`.
- http://www.mondorescue.org/—Mondo Rescue is a bare-metal backup/rescue tool independent of Ubuntu, using CD, DVD, tape, or NFS; it can produce bootable CDs to restore the system.
- http://www.ccp14.ac.uk/ccp14admin/linux-server/mondorescue/dvd_mondo.html—A HOWTO for using MondoRescue to back up on a DVD.
- http://www.linuxorbit.com/modules.php?op=modload&name=Sections&file=index&req=viewarticle&artid=222&page=1—A HOWTO using split and `mkisofs` to manually back up large archives to CD.

This page intentionally left blank

CHAPTER 14

Networking

IN THIS CHAPTER

One of the benefits of open source technology in general and Linux is particular is that it is now mature enough to be used effortlessly across several different networking environments as well as the Internet. With strong support for the standard internet protocol TCP/IP, Linux can also talk to all of the UNIX flavors, including Mac OS X, Windows (with the help of Samba), NetWare (IPX) and even older protocols such as DECNET and Banyan Vines. Many organizations use Linux as an Internet gateway, allowing many different clients to access the Internet through Linux, as well as communicate via email and instant messaging.This chapter covers network and Internet connectivity, as most networks invariably end up connected to the Internet in some shape or form. You will learn about how to get the basics right, including configuration and management of network cards (*NICs*) and other network services with Ubuntu. You will also find out how to manage network services from the command line—again an important lesson in case you are ever confined to a command prompt. We will also look at connectivity options, both for inbound and outbound network traffic and the importance of PPP (*Point to Point Protocol*). Also included is an overview of graphical management clients for Ubuntu, which are becoming more and more popular.

Laying the Foundation: The `localhost` Interface

The first thing that needs to happen before you can successfully connect to a network or even to the Internet is to create a `localhost` interface, sometimes also called a *loopback interface*, but more commonly referenced as *`lo`*. The TCP/IP protocol (see "Networking with TCP/IP" later

on in this chapter) uses this interface to assign an IP address to your computer and is needed for Ubuntu to establish a PPP interface.

Checking for the Availability of the Loopback Interface

You should not normally have to manually create a loopback interface as Ubuntu creates one automatically for you during installation. To check that one is set up, you can use the `ifconfig` command to show something similar to this:

```
$ ifconfig
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:13 errors:0 dropped:0 overruns:0 frame:0
          TX packets:13 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:832 (832.0 b)  TX bytes:832 (832.0 b)
```

What you see in this example is evidence that the loopback interface is present and active. The `inet addr` is the IP number assigned to the `localhost`, typically `127.0.0.1` along with the broadcast mask of `255.255.255.0` and that there has been little activity on this interface (`RX` = receive and `TX` = transmit). If your output does not look like the one above, you must hand-configure the `localhost` interface after you finish the rest of this section.

Configuring the Loopback Interface Manually

The `localhost` interface's IP address is specified in a text configuration file that is used by Ubuntu to keep record of various network wide IP addresses. The file is called `/etc/hosts` and usually exists on a system, even if it is empty. The file is used by the Linux kernel and other networking tools to enable them to access local IP addresses and hostnames. If you have not configured any other networking interfaces then you may find that the file only contains one line:

```
127.0.0.1       localhost.localdomain             localhost
```

This line defines the special `localhost` interface and assigns it an IP address of `127.0.0.1`. You might hear or read about terms such as *`localhost`*, *loopback*, and *dummy interface*; all these terms refer to the use of the IP address `127.0.0.1`. The term *loopback interface* indicates that to Linux networking drivers, it looks as though the machine is talking to a network that consists of only one machine; the kernel sends network traffic to and from itself on the same computer. *Dummy interface* indicates that the interface doesn't really exist as far as the outside world is concerned; it exists only for the local machine.

Each networked Ubuntu machine on a LAN will use this same IP address for its `localhost`. If for some reason a Ubuntu computer does not have this interface, edit the

/etc/hosts file to add the localhost entry, and then use the ifconfig and route commands using your sudo permissions to create the interface like this:

```
$ sudo /sbin/ifconfig lo 127.0.0.1
$ sudo /sbin/route add 127.0.0.1 lo
```

These commands will create the localhost interface in memory (all interfaces, such as eth0 or ppp0, are created in memory when using Linux), and then add the IP address 127.0.0.1 to an internal (in-memory) table so that the Linux kernel's networking code can keep track of routes to different addresses.

Use the ifconfig command as shown previously to test the interface.

You should now be able to use ping to check that the interface is responding properly like this (using either localhost or its IP address):

```
$ ping -c 3 localhost
PING localhost (127.0.0.1) 56(84) bytes of data.
64 bytes from localhost (127.0.0.1): icmp_seq=1 ttl=64 time=0.036 ms
64 bytes from localhost (127.0.0.1): icmp_seq=2 ttl=64 time=0.028 ms
64 bytes from localhost (127.0.0.1): icmp_seq=3 ttl=64 time=0.028 ms

--- localhost ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.028/0.030/0.036/0.007 ms
```

The -c option is used to set the number of pings, and the command, if successful (as it was previously), returns information regarding the round-trip speed of sending a test packet to the specified host.

Networking with TCP/IP

The basic building block for any network based on Unix hosts is the Transport Control Protocol/Internet Protocol (*TCP/IP*) suite of three protocols. The suite consists of the Internet Protocol (*IP*), Transport Control Protocol (*TCP*), and Universal Datagram Protocol (*UDP*). IP is the base protocol. The TCP/IP suite is *packet*-based, which means that data is broken into little chunks on the transmit end for transmission to the receiving end. Breaking data up into manageable packets allows for faster and more accurate transfers. In TCP/IP, all data travels via IP packets, which is why addresses are referred to as IP addresses. It is the lowest level of the suite.

TCP is a connection-based protocol. Before data is transmitted between two machines, a connection is established between them. When a connection is made, a stream of data is sent to the IP to be broken into the packets that are then transmitted. At the receiving end, the packets are put back in order and sent to the proper application port. TCP/IP

forms the basis of the Internet; without it the Internet would be a very different place indeed, if it even existed!

On the other hand, UDP is a connectionless protocol. Applications using this protocol just choose their destination and start sending. UDP is normally used for small amounts of data or on fast and reliable networks. If you are interested in the internals of TCP/IP, see the "Reference" section at the end of this chapter for places to look for more information.

Ubuntu and Networking

Chances are that your network card was configured during the installation of Ubuntu. You can however, use the ifconfig command at the shell prompt or Ubuntu's graphical network configuration tools, such as network-admin, to edit your system's network device information or to add or remove network devices on your system. Hundreds of networking commands and utilities are included with Ubuntu—far too many to cover in this chapter and more than enough for coverage in two or three volumes.

Nearly all ethernet cards can be used with Linux, along with many PCMCIA wired and wireless network cards. The great news is that many USB wireless network devices also work just fine with Linux, and more will be supported with upcoming versions of the Linux kernel. Check the Linux USB Project at http://www.linux-usb.org/ for the latest developments or to verify support for your device.

After reading this chapter, you might want to learn more about other graphical network clients for use with Linux. The GNOME `ethereal` client, for example, can be used to monitor all traffic on your LAN or specific types of traffic. Another client, NmapFE, can be used to scan a specific host for open ports and other running services.

TCP/IP Addressing

To understand networking with Linux, you need to know the basics of TCP/IP addressing. Internet IP addresses (also known as *public* IP addresses) are different from those used internally on a local area network, or *LAN*. Internet IP addresses are assigned (for the United States and some other hosts) by the American Registry for Internet Numbers, available at http://www.arin.net/. Entities that need an Internet address apply to this agency to be assigned an address. The agency assigns Internet service providers (*ISPs*) one or more blocks of IP addresses, which the ISPs can then assign to their subscribers.

You will quickly recognize the current form of TCP/IP addressing, known as IPv4 (IP version 4). In this method, a TCP/IP address is expressed of a series of four decimal numbers—a 32-bit value expressed in a format known as dotted decimal format, such as `192.168.120.135`. Each set of numbers is known as an *octet* (eight ones and zeros, such as `10000000` to represent `128`) and ranges from zero to 255.

The first octet usually determines what *class* the network belongs to. There are three classes of networks. The classes are

Class A—Consists of networks with the first octet ranging from 1 to 126. There are only 126 Class A networks—each composed of up to 16,777,214 hosts. (If you are doing the math, there are potentially 16,777,216 addresses, but no host portion of an address can be all zeros or 255s.) The "10." network is reserved for local network use, and the "127." network is reserved for the *loopback* address of 127.0.0.1. Loopback addressing is used by TCP/IP to enable Linux network-related client and server programs to communicate on the same host. This address will not appear and is not accessible on your LAN.

NOTE

Notice that zero is not included in Class A. The zero address is used for network-to-network broadcasts. Also, note that there are two other classes of networks, Classes D and E. Class D networks are reserved for multicast addresses and not for use by network hosts. Class E addresses are deemed experimental, and thus are not open for public addressing.

Class B—Consists of networks defined by the first two octets with the first ranging from 128 to 191. The "128." network is also reserved for local network use. There are 16,382 Class B networks—each with 65,534 possible hosts.

Class C—Consists of a network defined by the first three octets with the first ranging from 192 to 223. The "192." network is another that is reserved for local network use. There are a possible 2,097,150 Class C networks of up to 254 hosts each.

No host portion of an IP address can be all zeros or 255s. These addresses are reserved for broadcast addresses. IP addresses with all zeros in the host portion are reserved for network-to-network broadcast addresses. IP addresses with all 255s in the host portion are reserved for local network broadcasts. Broadcast messages are not typically seen by users.

These classes are the standard, but a *netmask* also determines what class your network is in. The netmask determines what part of an IP address represents the network and what part represents the host. Common netmasks for the different classes are

Class A—255.0.0.0

Class B—255.255.0.0

Class C—255.255.255.0

Because of the allocation of IP addresses for Internet hosts, it is now impossible to get a Class A network. It is also nearly impossible to get a Class B network (all the addresses have been given out, but some companies are said to be willing to sell theirs), and Class C network availability is dropping rapidly with the current growth of Internet use worldwide. See the following sidebar.

Limits of Current IP Addressing

The current IPv4 address scheme is based on 32-bit numbering and limits the number of available IP addresses to about 4.1 billion. Many companies and organizations (particularly in the United States) were assigned very large blocks of IP addresses in the early stages of the growth of the Internet, which has left a shortage of "open" addresses. Even with careful allocation of Internet-connected host IP addresses and the use of *network address translation* (*NAT*) to provide communication to and from machines behind an Internet-connected computer, the Internet might run out of available addresses.

To solve this problem, a newer scheme named IPv6 (IP version 6) is being implemented. It uses a much larger addressing solution based on 128-bit addresses, with enough room to include much more information about a specific host or device, such as global positioning server (*GPS*) or serial numbering. Although the specific details about the entire contents of the an IPv6 address have yet to be finalized, all Internet-related organizations appear to agree that something must be done to provide more addresses. According to Vint Cerf, one of the primary developers of the TCP/IP protocol, "There will be nearly 2.5 billion devices on the Internet by 2006, and by 2010 half the world's population will have access to the Internet."

You can get a good overview of the differences between IPv4 and IPv6 policies regarding IP address assignments, and the registration process of obtaining IP addresses, by browsing to http://www.arin.net/library/index.html. Read the Linux IPv6 HOWTO by browsing to http://tldp.org/HOWTO/Linux+IPv6-HOWTO/.

Ubuntu supports the use of IPv6 and includes a number of networking tools conforming to IPv6 addressing. Support for IPv6 can be configured by using settings and options in the file named `network` under the `/etc/sysconfig` directory, along with changes to related network configuration files, such as `/etc/hosts`. Many IPv6-based tools, such as `ipcalc6`, `ping6`, and `traceroute6`, are available for Ubuntu. See various files under the `/usr/share/doc/initscripts` directory for more information specific to setting up IPv6 addressing with Linux and Ubuntu.

Migration to IPv6 is slow in coming, however, because the majority of computer operating systems, software, hardware, firmware, and users are still in the IPv4 mindset. Supporting IPv6 will require rewrites to many networking utilities, portions of operating systems currently in use, and firmware in routing and firewall hardware.

Using IP Masquerading in Ubuntu

Three blocks of IP addresses are reserved for use on internal networks and hosts not directly connected to the Internet. The address ranges are from 10.0.0.0 to 10.255.255.255, or 1 Class A network; from 172.16.0.0 to 172.31.255.255, or 16 Class B networks; and from 192.168.0.0 to 192.168.255.255, or 256 Class C networks. Use these IP addresses when building a LAN for your business or home. Which class you choose can depend on the number of hosts on your network.

Internet access for your internal network can be provided by a PC running Ubuntu or other broadband or dial-up router. The host or device is connected to the Internet and is used as an Internet gateway to forward information to and from your LAN. The host

should also be used as a firewall to protect your network from malicious data and users while functioning as an Internet gateway.

A PC used in this fashion typically has at least two network interfaces. One is connected to the Internet with the other connected to the computers on the LAN (via a hub or switch). Some broadband devices also incorporate four or more switching network interfaces. Data is then passed between the LAN and the Internet using *network address translation*, or *NAT*, better known in Linux circles as *IP masquerading*.

NOTE

Do not rely on a single point of protection for your LAN, especially if you use wireless networking, provide dial-in services, or allow mobile (laptop or PDA) users internal or external access to your network. Companies, institutions, and individuals relying on a "moat mentality" have often discovered to their dismay that such an approach to security is easily breached. Make sure that your network operation is accompanied by a security policy that stresses multiple levels of secure access, with protection built into every server and workstation—something easily accomplished when using Linux.

Ports

Most servers on your network have more than one task. For example, web servers have to serve both standard and secure pages. You might also be running an FTP server on the same host. For this reason, applications are provided *ports* to use to make "direct" connections for specific software services. These ports help TCP/IP distinguish services so that data can get to the correct application. If you check the file `/etc/services`, you will see the common ports and their usage. For example, for FTP, HTTP, and Post Office Protocol (email retrieval server), you will see

```
ftp       21/tcp
http      80/tcp     http          # WorldWideWeb HTTP
pop3     110/tcp     pop-3             # POP version 3
```

The ports defined in `/etc/services` in this example are `21` for FTP, `80` for HTTP, and `110` for POP3. Other common port assignments are `25` for *simple mail transport protocol (SMTP)* and `22` for *secure shell (SSH)* remote login. Note that these ports are not set in stone, and you can set up your server to respond to different ports. For example, although port `22` is listed in `/etc/services` as a common default for SSH, the `sshd` server can be configured to listen on a different port by editing its configuration file `/etc/ssh/sshd_config`. The default setting (commented out with a pound sign) looks like this:

```
#Port 22
```

Edit the entry to use a different port, making sure to select an unused port number, such as

```
Port 2224
```

Save your changes, and then restart the `sshd` server. (Refer to Chapter 11, "Automating Tasks," to see how to restart a service.) Remote users must now access the host through port `2224`, which can be done using `ssh`'s `-p` (port) option like so:

```
$ ssh -p 2224 remote_host_name_or_IP
```

Network Organization

Properly organizing your network addressing process grows more difficult as the size of your network grows. Setting up network addressing for a Class C network with fewer than 254 devices is simple. Setting up addressing for a large, worldwide company with a Class A network and many different users can be extremely complex. If your company has fewer than 254 *hosts* (meaning any device that requires an IP address, including computers, printers, routers, switches, and other devices) and all your workgroups can share information, a single Class C network will be sufficient.

Subnetting

Within Class A and B networks, there can be separate networks called *subnets*. Subnets are considered part of the host portion of an address for network class definitions. For example, in the 128. Class B network, you can have one computer with an address of 128.10.10.10 and another with an address of 128.10.200.20; these computers are on the same network (128.10.), but they have different subnets (128.10.10. and 128.10.200.). Because of this, communication between the two computers requires either a router or a switch. Subnets can be helpful for separating workgroups within your company.

Often subnets can be used to separate workgroups that have no real need to interact with or to shield from other groups' information passing among members of a specific workgroup. For example, if your company is large enough to have its own HR department and payroll section, you could put those departments' hosts on their own subnet and use your router configuration to limit the hosts that can connect to this subnet. This configuration prevents networked workers who are not members of the designated departments from being able to view some of the confidential information the HR and payroll personnel work with.

Subnet use also enables your network to grow beyond 254 hosts and share IP addresses. With proper routing configuration, users might not even know they are on a different subnet from their co-workers. Another common use for subnetting is with networks that cover a wide geographic area. It is not practical for a company with offices in Chicago and London to have both offices on the same subnet, so using a separate subnet for each office is the best solution.

Subnet Masks

Subnet masks are used by TCP/IP to show which part of an IP address is the network portion and which part is the host. Subnet masks are usually referred to as *netmasks*. For a pure Class A network, the netmask would be 255.0.0.0; for a Class B network, the

netmask would be 255.255.0.0; and for a Class C network, the netmask would be 255.255.255.0. Netmasks can also be used to deviate from the standard classes.

By using customized netmasks, you can subnet your network to fit your needs. For example, your network has a single Class C address. You have a need to subnet your network. Although this is not possible with a normal Class C subnet mask, you can change the mask to break your network into subnets. By changing the last octet to a number greater than zero, you can break the network into as many subnets as you need.

For more information on how to create customized subnet masks, see Day 6, "The Art of Subnet Masking," in *Sams Teach Yourself TCP/IP Network Administration in 21 Days*. That chapter goes into great detail on how to create custom netmasks and explains how to create an addressing cheat sheet for hosts on each subnet. You can also browse to the Linux Network Administrator's Guide and read about how to create subnets at http://www.tldp.org/LDP/nag2/index.html.

Broadcast, Unicast, and Multicast Addressing

Information can get to systems through three types of addresses: unicast, multicast, and broadcast. Each type of address is used according to the purpose of the information being sent, as explained here:

- **Unicast**—Sends information to one specific host. Unicast addresses are used for Telnet, FTP, SSH, or any other information that needs to be shared in a one-to-one exchange of information. Although it is possible that any host on the subnet/network can see the information being passed, only one host is the intended recipient and will take action on the information being received.
- **Multicasting**—Broadcasts information to groups of computers sharing an application, such as a video conferencing client or online gaming application. All the machines participating in the conference or game require the same information at precisely the same time to be effective.
- **Broadcasting**—Transmits information to all the hosts on a network or subnet. *Dynamic Host Configuration Protocol (DHCP)* uses broadcast messages when the DHCP client looks for a DHCP server to get its network settings, and *Reverse Address Resolution Protocol (RARP)* uses broadcast messages for hardware address to IP address resolution. Broadcast messages use .255 in all the host octets of the network IP address. (10.2.255.255 will broadcast to every host in your Class B network.)

Hardware Devices for Networking

As stated at the beginning of this chapter, networking is one of the strong points of the Linux operating system. This section covers the classes of devices used for basic networking. Note that this section talks about hardware devices, and not Linux networking devices, which are discussed in the section, "Using Network Configuration Tools."

Network Interface Cards

A computer must have a *network interface card (NIC)* to connect to a network. Currently, there are several topologies (ways of connecting computers) for network connections. These topologies range from the old and mostly outdated 10BASE-2 to the much newer and popular wireless Wi-Fi or 802.11 networking.

A Dual-Host No-NIC Network

The truly frugal can also "network" two Linux workstations using a null-modem or LPT cable and PPP, as shown in the PPP HOWTO at http://tldp.org/HOWTO/PPP-HOWTO/direct.html. Just connect two PCs running Linux together with a null-modem cable, and use the `pppd` daemon on each end. If one Linux host is connected to a network or has a dial-up connection, that PC becomes a gateway for the other host. You have to use `pppd`'s `defaultroute` option on the serial-only host when connecting.

Each NIC has a unique address (the hardware address, known as *media access control*, or MAC), which identifies that NIC. This address is six pairs of hexadecimal bits separated by colons (:). A MAC address looks similar to this: 00:60:08:8F:5A:D9. The hardware address is used by DHCP (see "Dynamic Host Configuration Protocol" later in this chapter) to identify a specific host. It is also used by the *Address Resolution Protocol (ARP)* and *Reverse Address Resolution Protocol (RARP)* to map hosts to IP addresses.

This section covers some of the different types of NIC used to connect to your network.

Token Ring

Token ring networking was developed by IBM. As the name implies, the network is set up in a ring. A single "token" is passed from host to host, indicating the receiving host's permission to transmit data.

Token ring has a maximum transfer rate of 16Mbps (16 million bits per second). Unlike 10BASE-2 and 10BASE-5, token ring uses what is called *unshielded twisted pair (UTP)* cable. This cable looks a lot like the cable that connects your phone to the wall. Almost all token ring NICs are recognized by Linux.

10BASE-T

10BASE-T was the standard for a long time. A large number of networks still use it. 10BASE-T also uses UTP cable. Instead of being configured in a ring, 10BASE-T mostly uses a star architecture. In this architecture, the hosts all connect to a central location (usually a hub, which you learn about later in the section titled "Hubs and Switches"). All the data is sent to all hosts, but only the destination host takes action on individual packets. 10BASE-T has a transfer rate of 10Mbps.

10BASE-T has a maximum segment length of 100 meters. There are many manufacturers of 10BASE-T NICs, and most are recognized by Ubuntu.

100BASE-T

100BASE-T was popular around the turn of the millennium, keeping the same ease of administration as 10BASE-T while increasing the speed by a factor of 10. For most networks, the step from 10BASE-T to 100BASE-T is as simple as replacing NICs and hubs. Most 100BASE-T NICs and hubs can also handle 10BASE-T and can automatically detect which is in use. This allows for a gradual network upgrade and usually does not require rewiring your whole network. Nearly every known 100BASE-T NIC and most generic NICs are compatible with Linux, thanks to Donald Becker of http://www.scyld.com/. 100BASE-T requires category 5 unshielded twisted pair cabling.

1000BASE-T

1000BASE-T—usually referred to as *gigabit ethernet*—is the accepted standard in enterprise networking, with most NICs being detected and configured correctly by Ubuntu. Like 100BASE-T NICs, gigabit NICs automatically downgrade if they are plugged in to a slower network. Also like 100BASE-T, gigabit NICs require category 5 unshielded twisted pair cabling; however, many institutions are now deploying category 6 cables because they have much longer range and so are often worth the extra cost. You will find that many newer computers tend to be fitted with gigabit NICs as standard.

Fiber Optic and Gigabit Ethernet

Fiber optic is more commonly used in newer and high-end installations because the cost of upgrading can be prohibitive for older sites.

Fiber optics were originally used on *fiber distributed data interface (FDDI)* networks, similar to token ring in structure except that there are two rings—one is primary, whereas the other is secondary. The primary ring is used exclusively, and the secondary sits idle until there is a break in the primary ring. At this point, the secondary ring takes over, keeping the network alive. FDDI has a speed of 100Mbps and has a maximum ring length of 62 miles. FDDI uses several tokens at the same time that, along with the faster speed of fiber optics, account for the drastic increase in network speed.

As stated, switching to a fiber optic network can be very costly. To make the upgrade, the whole network has to be rewired (as much as U.S. $150 per network connection), and all NICs must be replaced at the same time. Most FDDI NICs are recognized by Linux.

Fiber-related gigabit is termed 1000BASE-X, whereas 1000BASE-T gigabit ethernet uses twisted-pair (see the “Unshielded Twisted Pair” section, later in this chapter).

Wireless Network Interfaces

Wireless networking, as the name states, works without network cables and is an extremely popular option, particularly for those whose spouses do not like wires trailing everywhere! Upgrading is as easy as replacing network cards and equipment, such as routers and switches. Wireless networking equipment can also work along with the traditional wired networking using existing equipment.

It might not be practical to upgrade a desktop or large server to wireless just yet if the wiring is already in place. Wireless networking is still generally slower than a traditional

wired network. However, this situation is changing with wider adoption of newer protocols, such as 802.11g (supporting the common 802.11b and faster but less popular 802.11a), along with the introduction of more compliant and inexpensive wireless NICs. Some 802.11g NICs work at up to 108Mbps, which appears faster than 100BASE-T wired networking on the surface. However, in practice, it is a great deal slower: Unless your networking environment has paper-thin walls, you can usually halve the reported speed of Wi-Fi network devices. 108Mbps works about half the speed of 100BASE-T.

With each new version of Linux, more and more wireless NICs are compatible. That said, it is usually better to get brand name wireless NICs, because you have a better chance of compatibility. Check the http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/ web page for more specific hardware compatibility information. More on wireless networking is discussed later in this chapter.

Network Cable

Currently, three types of network cable exist: coaxial, unshielded twisted pair (UTP), and fiber. Coaxial cable (rarely used today) looks a lot like the coaxial cable used to connect your television to the cable jack or antenna. UTP looks a lot like the cable that runs from your phone to the wall jack (the jacks are a bit wider). Fiber cable looks sort of like the RCA cables used on your stereo or like the cable used on your electrical appliances in your house (two separate segments connected together). The following sections discuss UTP and fiber network cable in more detail.

Unshielded Twisted Pair

Unshielded twisted pair (UTP) uses color-coded pairs of thin copper wire to transmit data. Six categories of UTP exist—each serving a different purpose:

- **Category 1 (Cat1)**—Used for voice transmissions such as your phone. Only one pair is used per line—one wire to transmit and one to receive. An RJ-11 plug is used to connect the cable to your phone and the wall.
- **Category 2 (Cat2)**—Used in early token ring networks. Has a transmission rate of 4Mbps (million bits per second) and has the slowest data transfer rate. An RJ-11 plug is also used for cable connections.
- **Category 3 (Cat3)**—Used for 10BASE-T networks. It has a transmission rate of 10Mbps. Three pairs of cables are used to send and receive signals. RJ-11 or RJ-45 plugs can be used for Cat3 cables, usually deferring to the smaller RJ-11. RJ-45 plugs are similar in design to RJ-11, but are larger to handle up to four pairs of wire and are used more commonly on Cat5 cables.
- **Category 4 (Cat4)**—Used in modern token ring networks. It has a transmission rate of 16Mbps and is less and less common as companies are switching to better alternatives. RJ-45 plugs are used for cable connections.
- **Category 5 (Cat5)**—The fastest of the UTP categories with a transmission rate of up to 1000Mbps. It is used in both 100BASE-T and 1000BASE-T networks and uses four pairs of wire. Cat5 cable came out just as 10BASE-T networks were becoming

popular and isn't much more expensive than Cat3 cable. As a result, most 10BASE-T networks use Cat5 UTP instead of Cat3. Cat5 cable uses RJ-45 plugs.

- **Category 6 (Cat6)**—Also rated at 1000Mbps, this cable is available in two forms: stranded for short runs (25-meter) and solid for up to 100-meter runs, but which should not be flexed.

Fiber Optic Cable

Fiber optic cable (fiber) is usually orange or red in color. The transmission rate is 100Mbps and has a maximum length of 62 miles. Fiber uses a two-pronged plug to connect to devices. A couple of advantages to fiber are that because it uses light instead of electricity to transmit its signal, it is free from the possibility of electromagnetic interference and is also more difficult to tap into and eavesdrop.

Hubs and Switches

Hubs and switches are used to connect several hosts together on a star architecture network. They can have any number of connections; the common sizes are 4, 8, 16, 24, and 48 connections (ports)—each port has a light that comes on when a network connection is made (link light). Their use enables you to expand your network easily; you can just add new hubs or switches when you need to add new connections. Each unit can connect to the other hubs or switches on the network, typically, through a port on the hub or switch called an *uplink* port. This enables two hubs or switches, connected by their uplink ports, to act as one hub or switch. Having a central location where all the hosts on your network can connect allows for easier troubleshooting of problems. If one host goes down, none of the other hosts are affected (depending on the purpose of the downed host). Because hubs and switches are not directly involved with the Linux operating system, compatibility is not an issue.

If you are constructing a small to mid-size network, it is important to consider whether you intend to use either hubs or switches. Hubs and switches are visually the same in that they have rows of network ports. However, under the hood, the difference is quite important. Data is sent as packets of information across the network; with a hub the data is transmitted simultaneously to all the network ports, irrespective of which port the destination computer is attached to.

Switches, however, are more intelligent because they can direct packets of information directly to the correct network port that leads to the destination computer. They do this by "learning" the MAC addresses of each computer that is attached to them. In short, using switches minimizes excess packets being sent across the network, thus increasing network bandwidth. In a small network with a handful of computers, the use of hubs might be perfectly acceptable and you will find that hubs are generally cheaper than switches. However, for larger networks of 15 computers or more, you might want to consider implementing a switched network.

TIP

Troubleshooting network connections can be a challenge, especially on large networks. If a user complains that he has lost his network connection, the hub is a good place to start. If the link light for the user's port is lit, chances are the problem is with the user's network configuration. If the link light is not on, the host's NIC is bad, the cable is not inserted properly, or the cable has gone bad for some reason.

Routers and Bridges

Routers and bridges are used to connect different networks to your network and to connect different subnets within your network. Routers and bridges both serve the same purpose of connecting networks and subnets, but they do so with different techniques. The information in the following sections will help you choose the connection method that best suits your needs.

Bridges

Bridges are used within a network to connect different subnets. A bridge blindly relays all information from one subnet to another without any filtering and is often referred to as a *dumb gateway*. This can be helpful if one subnet in your network is becoming overburdened and you need to lighten the load. A bridge is not very good for connecting to the Internet, however, because it lacks filtering. You really do not want all traffic traveling the Internet to be able to get through to your network.

Routers

Routers can pass data from one network to another, and they allow for filtering of data. Routers are best suited to connect your network to an outside network, such as the Internet. If you have a web server for an internal intranet that you do not want people to access from the Internet, for example, you can use a router's filter to block port 80 from your network. These filters can be used to block specific hosts from accessing the Internet as well. For these reasons, routers are also called *smart gateways*.

Routers range in complexity and price from a Cisco brand router that can cost thousands of dollars to other brands that can be less than a hundred dollars.

Initializing New Network Hardware

All the initial network configuration and hardware initialization for Ubuntu is normally done during installation. At times, however, you will have to reconfigure networking on your system, such as when a host needs to be moved to a different subnet or a different network, or if you replace any of your computer's networking hardware.

Linux creates network interfaces in memory when the kernel recognizes that a NIC or other network device is attached to the system. These interfaces are unlike other Linux interfaces, such as serial communications ports, and do not have a corresponding device file in the `/dev` directory. Unless support for a particular NIC is built in to your kernel, Linux must be told to load a specific kernel module to support your NIC. More than 100

such modules are located in the `/lib/modules/2.6.XX-XX/kernel/net` directory (where *XX-XX* is your version of the kernel).

You can initialize a NIC in several ways when using Linux. When you first install Ubuntu, automatic hardware probing detects and configures your system to use any installed NICs. If you remove the original NIC and replace it with a different make and model, your system will not automatically detect and initialize the device unless you configure Ubuntu to use automatic hardware detection when booting. Ubuntu should detect the absence of the old NIC and the presence of the new NIC at boot time.

If you do not use automatic hardware detection and configuration, you can initialize network hardware by

- Manually editing the `/etc/modprobe.conf` file to prompt the system to recognize and support the new hardware upon reboot
- Manually load or unload the new device's kernel module with the `modprobe` command

The following sections explain the first and last of the preceding methods.

Editing the `/etc/modprobe.conf` File

This file may not be present when you first look for it, so you may need to create a blank file in a text editor. You can manually edit the `/etc/modprobe.conf` file to add a module dependency entry (also known as a *directive*) to support a new NIC or other network device. This entry includes the device's name and its corresponding kernel module. After you add this entry, the Linux kernel recognizes your new networking hardware upon reboot. Ubuntu runs a module dependency check upon booting.

For example, if your system uses a RealTek NIC, you could use an entry like this:

```
alias eth0 8139too
```

The example entry tells the Linux kernel to load the `8139too.o` kernel module to support the `eth0` network device. On the other hand, if you have an Intel Ethernet Pro NIC installed, you would use an entry like this:

```
alias eth0 eepro100
```

Other parameters can be passed to a kernel module using one or more option entries, if need be, to properly configure your NIC. See the `modprobe.conf` man page for more information on using entries. For more specifics regarding NIC kernel modules, examine the module's source code because no man pages exist (a good opportunity for anyone willing to write the documentation).

TIP

Linux kernel and network tools can be used to diagnose problems or troubleshoot problematic NICs. However, if you browse to Don Becker's Linux Ethercard Status, Diagnostic and Setup Utilities page at http://www.scyld.com/ethercard_diag.html, you will find more than two dozen hardware-specific utilities for a variety of PCI and legacy ISA Ethernet network cards. These tools can be extremely helpful if you run into trouble during NIC recognition or configuration.

Using `modprobe` to Manually Load Kernel Modules

You do not have to use an `/etc/modprobe.conf` entry to initialize kernel support for your new network device. As root (using sudo), you can manually load or unload the device's kernel module using the `modprobe` command, along with the module's name. For example, use the following command line to enable the example RealTek NIC:

```
$ sudo modprobe 8139too
```

After you press Enter, you will see this device reported from the kernel's ring buffer messages, which can be displayed by the `dmesg` command. Here's a portion of that command's output:

```
$ dmesg
...
eth0: RealTek RTL8139 Fast Ethernet at 0xce8ee000, 00:30:1b:0b:07:0d, IRQ 11
eth0: Identified 8139 chip type 'RTL-8139C'
eth0: Setting half-duplex based on auto-negotiated partner ability 0000.
...
```

Note that at this point, an IP address or other settings have not been assigned to the device. Linux can use multiple ethernet interfaces, and the first ethernet device will be numbered `eth0`, the second `eth1`, and so on. Each different ethernet device recognized by the kernel might have additional or different information reported, depending on its kernel module. For example,

```
$ dmesg
...
eepro100.c:v1.09j-t 9/29/99 Donald Becker http://cesdis.gsfc.nasa.gov/linux/
➥drive
rs/eepro100.html
eepro100.c: $Revision: 1.36 $ 2000/11/17 Modified by Andrey V. Savochkin <saw@saw
➥.sw.com.sg> and others
PCI: Found IRQ 10 for device 00:0d.0
eth0: Intel Corporation 82557 [Ethernet Pro 100], 00:90:27:91:92:B5, IRQ 10.
 Board assembly 721383-007, Physical connectors present: RJ45
 Primary interface chip i82555 PHY #1.
 General self-test: passed.
 Serial sub-system self-test: passed.
```

```
 Internal registers self-test: passed.
 ROM checksum self-test: passed (0x04f4518b).
...
```

In this example, an Intel Ethernet Pro 100 NIC has been recognized. To disable support for a NIC, the kernel module can be unloaded, but usually only after the device is no longer in use. Read the next section to learn how to configure a NIC after it has been recognized by the Linux kernel and how to control its behavior.

Using Network Configuration Tools

If you add or replace networking hardware after your initial installation, you must configure the new hardware. You can do so using either the command line or the graphical configuration tools. To configure a network client host using the command line, you can use a combination of commands or edit specific files under the `/etc` directory. To configure the hardware through a graphical interface, you can use Ubuntu's graphical tool for X called `network-admin`. This section introduces command-line and graphical software tools you can use to configure a network interface and network settings on your Ubuntu system. You'll see how to control your NIC and manage how your system interacts with your network.

Using the command-line configuration tools can seem difficult if you are new to Linux. For anyone new to networking, the `network-admin` graphical tool is the way to go. Both manual and graphical methods require root access to work. If you do not have root access, get it before trying any of these actions. You should not edit any scripts or settings files used by graphical network administration tools on your system. Your changes will be lost the next time the tool, such as `network-admin`, is run! Either use a manual approach and write your own network setup script, or stick to using graphical configuration utilities.

Command-Line Network Interface Configuration

You can configure a network interface from the command line using the basic Linux networking utilities. You configure your network client hosts with the command line by using commands to change your current settings or by editing a number of system files. Two commands, `ifconfig` and `route`, are used for network configuration. The `netstat` command displays information about the network connections.

`/sbin/ifconfig`

`ifconfig` is used to configure your network interface. You can use it to

- Activate or deactivate your NIC or change your NIC's mode
- Change your machine's IP address, netmask, or broadcast address
- Create an IP alias to allow more than one IP address on your NIC
- Set a destination address for a point-to-point connection

You can change as many or as few of these options as you'd like with a single command. The basic structure for the command is as follows:

```
ifconfig [network device] options
```

Table 14.1 shows a subset of `ifconfig` options and examples of their uses.

TABLE 14.1 `ifconfig` Options

Use	Option	Example
Create alias	`[network device]`	`ifconfig eth0:0_:[number] 10.10.10.10`
Change IP address		`ifconfig eth0 10.10.10.12`
Change the netmask	`netmask [netmask]`	`fconfig eth0 netmask 255.255.255.0`
Change the broadcast	`broadcast [address]`	`ifconfig eth0 broadcast 10.10.10.255`
Take interface down	`down`	`ifconfig eth0 down`
Bring interface up	`up (add IP address)`	`ifconfig eth0 up (ifconfig eth0 10.10.10.10)`
Set NIC promiscuous	`[-]promisc [ifconfig eth0 -promisc]`	`ifconfig eth0 promisc` mode on [off]
Set multicasting mode	`[-]allmulti`	`ifconfig eth0_on [off] allmulti [ifconfig eth0 -allmulti]`
Enable [disable] [address]	`[-]pointopoint eth0_pointopoint`	`ifconfig_point-to-point address 10.10.10.20 [ifconfig eth0 pointopoint_10.10.10.20]`

The `ifconfig` man page shows other options that enable your machine to interface with a number of network types such as AppleTalk, Novell, IPv6, and others. Again, read the man page for details on these network types.

NOTE

Promiscuous mode causes the NIC to receive all packets on the network. It is often used to sniff a network. Multicasting mode enables the NIC to receive all multicast traffic on the network.

If no argument is given, `ifconfig` displays the status of active interfaces. For example, the output of `ifconfig`, without arguments and one active and configured NIC, looks similar to this:

```
$ ifconfig

eth0      Link encap:Ethernet  HWaddr 00:0F:EA:B2:53:85
          inet addr:192.168.2.5  Bcast:192.168.2.255  Mask:255.255.255.0
          inet6 addr: fe80::20f:eaff:feb2:5385/64 Scope:Link
```

```
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:471 errors:0 dropped:0 overruns:0 frame:0
          TX packets:695 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:160637 (156.8 KiB)  TX bytes:86193 (84.1 KiB)
          Interrupt:185 Base address:0x6000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:19 errors:0 dropped:0 overruns:0 frame:0
          TX packets:19 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1336 (1.3 KiB)  TX bytes:1336 (1.3 KiB)
```


The output is easily understood. The `inet` entry displays the IP address for the interface. `UP` signifies that the interface is ready for use, `BROADCAST` denotes that the interface is connected to a network that supports broadcast messaging (`ethernet`), `RUNNING` means that the interface is operating, and `LOOPBACK` shows which device (`lo`) is the loopback address. The *maximum transmission unit (MTU)* on `eth0` is 1500 bytes. This determines the size of the largest packet that can be transmitted over this interface (and is sometimes "tuned" to other values for performance enhancement). `Metric` is a number from 0 to 3 that relates to how much information from the interface is placed in the routing table. The lower the number, the smaller the amount of information.

The `ifconfig` command can be used to display information about or control a specific interface using commands as listed in Table 14.1. For example, to deactivate the first Ethernet device on a host, use the `ifconfig` command, the interface name, and the command `down` like so:

```
$ sudo ifconfig eth0 down
```

You can also configure and activate the device by specifying a hostname or IP address and network information. For example to configure and activate ("bring up") the `eth0` interface with a specific IP address, use the `ifconfig` command like this:

```
$ sudo ifconfig eth0 192.168.2.9 netmask 255.255.255.0 up
```

If you have a host defined in your system's `/etc/hosts` file (see the section "Network Configuration Files" later in this chapter), you can configure and activate the interface according to the defined hostname like this:

```
$ sudo ifconfig eth0 dogdog.hudson.com up
```

Read the next section to see how to configure your system to work with your LAN.

/sbin/route

The second command used to configure your network is the route command. route is used to build the routing tables (in memory) implemented for routing packets as well as displaying the routing information. It is used after ifconfig has initialized the interface. route is normally used to set up static routes to other networks via the gateway or to other hosts. The command configuration is like this:

```
$ route [options] [commands] [parameters]
```

To display the routing table, use the route command with no options. The display will look similar to this:

```
$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.2.0     *               255.255.255.0   U     0      0        0 eth0
default         .               0.0.0.0         UG    0      0        0 eth0
```

In the first column, Destination is the IP address (or, if the host is in /etc/hosts or /etc/networks, the hostname) of the receiving host. The default entry is the default gateway for this machine. The Gateway column lists the gateway that the packets must go through to reach their destination. An asterisk (*) means that packets go directly to the host. Genmask is the netmask. The Flags column can have several possible entries. In our example, U verifies that the route is enabled and G specifies that Destination requires the use of a gateway. The Metric column displays the distance to the Destination. Some daemons use this to figure the easiest route to the Destination. The Ref column is used by some UNIX flavors to convey the references to the route. It isn't used by Linux. The Use column indicates the number of times this entry has been looked up. Finally, the Iface column is the name of the interface for the corresponding entry.

Using the -n option to the route command will give the same information, substituting IP addresses for names and asterisks (*), and looks like this:

```
# /sbin/route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
0.0.0.0         192.168.2.1     0.0.0.0         UG    0      0        0 eth0
```

The route command can add to the table using the add option. With the add option, you can specify a host (-host) or a network (-net) as the destination. If no option is used, the route command assumes that you are configuring the host issuing the command. The most common uses for the route command are to add the default gateway for a host, for a host that has lost its routing table, or if the gateway address has changed. For example, to add a gateway with a specific IP address, you could use the following:

```
$ sudo route add default gw 149.112.50.65
```

Note that you could use a hostname instead of an IP address if desired. Another common use is to add the network to the routing table right after using the `ifconfig` command to configure the interface. Assuming that the `208.59.243.0` entry from the previous examples was missing, replace it using the following command:

```
$ sudo route add -net 208.59.243.0 netmask 255.255.255.0 dev eth0
```

You also can use `route` to configure a specific host for a direct (point-to-point) connection. For example, say that you have a home network of two computers. One of the computers has a modem through which it connects to your business network. You typically work at the other computer. You can use the `route` command to establish a connection through specific hosts using the following command:

```
$ sudo route add -host 198.135.62.25 gw 149.112.50.65
```


The preceding example makes the computer with the modem the gateway for the computer you are using. This type of command line is useful if you have a gateway or firewall connected to the Internet. There are many additional uses for the `route` command, such as manipulating the default packet size. See the man page for those uses.

`/bin/netstat`
The `netstat` command is used to display the status of your network. It has several parameters that can display as much or as little information as you prefer. The services are listed by *sockets* (application-to-application connections between two computers). You can use `netstat` to display the information in Table 14.2.

TABLE 14.2 `netstat` Options

Option	Output
`-g`	Displays the multicast groups configured
`-i`	Displays the interfaces configured by `ifconfig`
`-s`	Lists a summary of activity for each protocol
`-v`	Gives verbose output, listing both active and inactive sockets
`-c`	Updates output every second (good for testing and troubleshooting)
`-e`	Gives verbose output for active connections only
`-C`	Displays information from the route cache and is good for looking at past connections

Several other options are available for this command, but they are used less often. As with the `route` command, the man page can give you details about all options and parameters.

Network Configuration Files

As previously stated, five network configuration files can be modified to make changes to basic network interaction of your system. The files are

`/etc/hosts`—A listing of addresses, hostnames, and aliases

`/etc/services`—Network service and port connections

`/etc/nsswitch.conf`—Linux network information service configuration

`/etc/resolv.conf`—Domain name service domain (search) settings

`/etc/host.conf`—Network information search order (by default, `/etc/hosts` and then DNS)

After these files are modified, the changes are active. As with most configuration files, comments can be added with a hash mark (#) preceding the comment. All of these files have a man page written about them for more information.

Adding Hosts to `/etc/hosts`

The `/etc/hosts` file is a map of IP to hostnames. If you are not using DNS or another naming service, and you are connected to a large network, this file can get quite large and can be a real headache to manage. A small `/etc/hosts` file can look something like this:

```
127.0.0.1       localhost
127.0.1.1       optimus

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
```

The first entry is for the loopback entry. The second is for the name of the machine. If no naming service is in use on the network, the only host that `myhost` will recognize by name is `yourhost`. (IP addresses on the network can still be used.)

Service Settings in `/etc/services`

The `/etc/services` file maps port numbers to services. The first few lines look similar to this (the `/etc/services` file can be quite long, more than 500 lines):

```
# Each line describes one service, and is of the form:
#
# service-name port/protocol [aliases ...]  [# comment]

tcpmux     1/tcp              # TCP port service multiplexer
tcpmux     1/udp              # TCP port service multiplexer
rje        5/tcp              # Remote Job Entry
rje        5/udp              # Remote Job Entry
echo       7/tcp
echo       7/udp
discard    9/tcp      sink null
discard    9/udp      sink null
systat     11/tcp     users
```

Typically, there are two entries for each service because most services can use either TCP or UDP for their transmissions. Usually after `/etc/services` is initially configured, you will not need to change it.

Using `/etc/nsswitch.conf` After Changing Naming Services

This file was initially developed by Sun Microsystems to specify the order in which services are accessed on the system. A number of services are listed in the `/etc/nsswitch.conf` file, but the most commonly modified entry is the `hosts` entry. A portion of the file can look like this:

```
passwd:         compat
group:          compat
shadow:         compat

hosts:          files dns mdns
networks:       files

protocols:      db files
services:       db files
ethers:         db files
rpc:            db files

netgroup:       nis
```

This tells services that they should consult standard Unix/Linux files for `passwd`, `shadow`, and `group` (`/etc/passwd`, `/etc/shadow`, `/etc/group`, respectively) lookups. For `host` lookups, the system checks `/etc/hosts` and if there is no entry, it checks DNS. The commented `hosts` entry lists the possible values for `hosts`. Edit this file only if your naming service has changed.

Setting a Name Server with `/etc/resolv.conf`

`/etc/resolv.conf` is used by DNS, the domain name service. The following is an example of `resolv.conf`:

```
nameserver 192.172.3.8
nameserver 192.172.3.9
search mydomain.com
```

This sets the nameservers and the order of domains for DNS to use. The contents of this file will be set automatically if you use Dynamic Host Configuration Protocol, or DHCP (see the section on "Dynamic Host Configuration Protocol" later in this chapter).

Setting DNS Search Order with `/etc/host.conf`

The `/etc/host.conf` file lists the order in which your machine will search for hostname resolution. The following is the default `/etc/host.conf` file:

```
order hosts, bind
```

In this example, the host checks the `/etc/hosts` file first and then performs a DNS lookup. A couple more options control how the name service is used. The only reason to modify this file is if you use NIS for your name service or you want one of the optional services. The `nospoof` option can be a good option for system security. It compares a standard DNS lookup to a reverse lookup (host-to-IP then IP-to-host) and fails if the two don't match. The drawback is that often when proxy services are used, the lookup fails, so you want to use this with caution.

Using Graphical Configuration Tools

As mentioned earlier, if you are new to networking or still becoming proficient with the command line, the graphical configuration tool is your best method for configuring new hardware in Ubuntu. Like most graphical tools, `network-admin` allows you to fill in the blanks; press the proper buttons, and the tool modifies the required files and issues the proper commands. Remember, you must be root to run `network-admin`.

You can find `network-admin` under System, Administration as the Networking icon.

After it has started, you will see the screen shown in Figure 14.1.

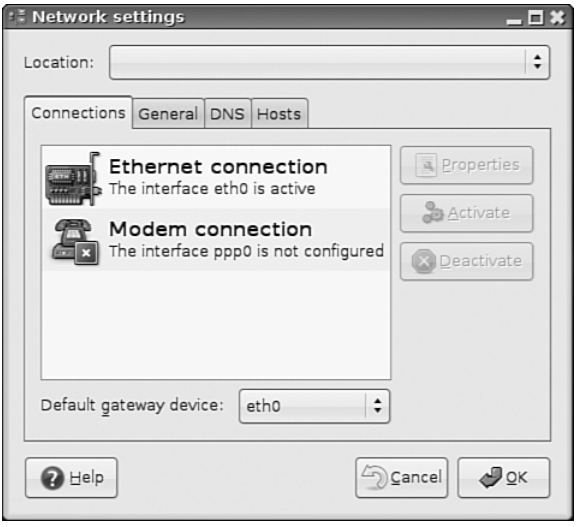


FIGURE 14.1 Use network-admin to configure your network devices.

Click the DNS tab to configure your system's DNS settings, hostname, or DNS search path. Click the Hosts tab, and then either click the Add or Properties button (after selecting a host) to create or edit an entry in your system's `/etc/hosts` file, for example, to add the IP addresses, hostnames, and aliases of hosts on your network. See Figure 14.2 for an example of editing a host entry.

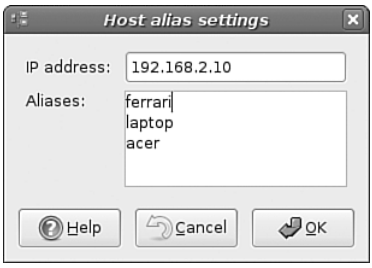


FIGURE 14.2 Highlight an existing entry, and then click the Properties button to change `/etc/hosts` entries in the Hosts tab of the Network Configuration screen.

Ubuntu does a great job of detecting and configuring your network interfaces, leaving you with very little work to do. However, you may need to assign a static IP address, which you do by selecting the appropriate network interface in the Connections tab and clicking the Properties button. This is shown in Figure 14.3, and you can see that you can elect to have a static IP address. Just make sure you enter in all the details and everything should work when you click OK.

FIGURE 14.3 Assign a static IP address to a network interface.

NOTE

Bootp is the initial protocol that DHCP was built on, and it has mostly been replaced by DHCP.

You can also assign locations to your computer, especially handy if you are on a laptop and move between several networks each requiring different configurations. Just select the gray bar at the top of the `network-admin` window and select Create Location. Enter a meaningful name, such as Home and then repeat this again to create another location, Work. Each time you switch between locations, Ubuntu detects that it needs to use configurations specific to those locations, so for instance you might want to use DHCP at

work, but not at home. Simple; just select the Home location, configure your ethernet connection to use a Static IP and you are all set to switch between your home and corporate networks.

One of the other graphical tools used for configuring your network is NetworkManager. We covered this in more detail in Chapter 2 so refer to that chapter if you need more information on NetworkManager.

Dynamic Host Configuration Protocol

As its name implies, *Dynamic Host Configuration Protocol (DHCP)* configures hosts for connection to your network. DHCP allows a network administrator to configure all TCP/IP parameters for each host as he connects to the network after activation of a NIC. These parameters include automatically assigning an IP address to a NIC, setting name server entries in `/etc/resolv.conf`, and configuring default routing and gateway information for a host. This section first describes how to use DHCP to obtain IP address assignment for your NIC, and then how to quickly set up and start a DHCP server using Ubuntu.

NOTE

You can learn more about DHCP by reading RFC2131 "Dynamic Host Configuration Protocol." Browse to http://www.ietf.org/rfc/rfc2131.txt.

How DHCP Works

DHCP provides persistent storage of network parameters by holding identifying information for each network client that might connect to the network. The three most common pairs of identifying information are

- **Network subnet/host address**—Used by hosts to connect to the network at will
- **Subnet/hostname**—Enables the specified host to connect to the subnet
- **Subnet/hardware address**—Enables a specific client to connect to the network after getting the hostname from DHCP

DHCP also allocates to clients temporary or permanent network (IP) addresses. When a temporary assignment, known as a *lease*, elapses, the client can request to have the lease extended, or, if the address is no longer needed, the client can relinquish the address. For hosts that will be permanently connected to a network with adequate addresses available, DHCP allocates infinite leases.

DHCP offers your network some advantages. First, it shifts responsibility for assigning IP addresses from the network administrator (who can accidentally assign duplicate IP addresses) to the DHCP server. Second, DHCP makes better use of limited IP addresses. If a user is away from the office for whatever reason, the user's host can release its IP address for use by other hosts.

Like most things in life, DHCP is not perfect. Servers cannot be configured through DHCP alone because DNS does not know what addresses that DHCP assigns to a host. This means that DNS lookups are not possible on machines configured through DHCP alone; therefore, services cannot be provided. However, DHCP can make assignments based on DNS entries when using subnet/hostname or subnet/hardware address identifiers.

NOTE

The problem of using DHCP to configure servers using registered hostnames is being addressed by Dynamic DNS which, when fully developed, will enable DHCP to register IP addresses with DNS. This will allow you, for example, to register a domain name (such as imalinuxuser.com) and be able to easily access that domain's web server without needing to use static IP addressing of a specific host. The largest hurdle to overcome is the security implication of enabling each host connecting to the system to update DNS. A few companies, such as http://www.dyndns.org/, are already offering Dynamic DNS services and have clients for Linux.

Activating DHCP at Installation and Boot Time

Ubuntu automatically defaults your network interfaces to using DHCP, as it is the simplest way of setting up a network interface. With *dynamic*, or DHCP-assigned IP addressing schemes for your NIC, the broadcast address is set at `255.255.255.255` because `dhclient`, the DHCP client used for IP configuration, is initially unaware of where the DHCP server is located, so the request must travel every network until a server replies.

The instruction to use DHCP for your NIC can be found under `/etc/network/interfaces`, with a line that says `dhcp`.

Other settings specific to obtaining DHCP settings are saved in the file named `dhclient.conf` under the `/etc/dhcp3/dhclient.conf` directory and are documented in the `dhclient.conf` man page. More than 100 options are also documented in the `dhcp-options` man page.

However, using DHCP is not that complicated. If you want to use DHCP and know that there is a server on your network, you can quickly configure your NIC by using the `dhclient` like so:

```
# dhclient
Internet Systems Consortium DHCP Client V3.0.5
Copyright 2004-2006 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/

SIOCSIFFLAGS: No such file or directory
SIOCSIFFLAGS: No such file or directory
```

```
Listening on LPF/eth1/00:11:50:3c:fe:21
Sending on   LPF/eth1/00:11:50:3c:fe:21
Listening on LPF/eth0/00:0f:ea:b2:53:85
Sending on   LPF/eth0/00:0f:ea:b2:53:85
Sending on   Socket/fallback
receive_packet failed on eth1: Network is down
DHCPDISCOVER on eth1 to 255.255.255.255 port 67 interval 7
send_packet: Network is down
DHCPDISCOVER on eth0 to 255.255.255.255 port 67 interval 7
DHCPOFFER from 192.168.0.1
DHCPREQUEST on eth0 to 255.255.255.255 port 67
DHCPACK from 192.168.0.1
bound to 192.168.0.7 — renewal in 118148 seconds.
```

In this example, the first ethernet device, `eth0`, has been assigned an IP address of 192.168.0.7 from a DHCP server at 192.168.0.1. The renewal will take place in about 33 hours.

DHCP Software Installation and Configuration

Installation of the DHCP client and server is fairly straightforward, mainly because Ubuntu already includes `dhclient` in a default installation, but also because installing software is easy using `synaptic` or `apt-get`.

DHCP `dhclient`

DHCP is automatically enabled when you install Ubuntu, so you don't need to worry about having to enable it. The DHCP client, `dhclient`, sends a broadcast message that the DHCP server replies to with networking information for your host. Once it has this, you're done!

You can however, fine-tune how `dhclient` works, and where and how it obtains or looks for DHCP information. You probably will not need to take this additional effort; but if you do, you can create and edit a file named `dhclient.conf`, and save it in the `/etc` directory with your settings.

CAUTION

You shouldn't just go ahead and overwrite your `dhclient.conf`_with any old file, as it could lead you to painful networking problems. Instead, copy the file like so:

```
sudo cp /etc/dhcp3/dhclient.conf/ etc/dhcp3/dhclient.conf.backup
```

That way, if anything goes wrong, you can then copy it back to restore the original settings.

A few of the `dhclient.conf` options include

- `timeout` *`time`* `;`—How long to wait before giving up trying (60 seconds is the default)
- `retry` *`time`* `;`—How long to wait before retrying (five minutes is the default)
- `select-timeout` *`time`* `;`—How long to wait before selecting a DHCP offer (zero seconds is the default)
- `reboot` *`time`* `;`—How long to wait before trying to get a previously set IP (10 seconds is the default)
- `renew` *`date`* `;`—When to renew an IP lease, where *`date`* is in the form of `<weekday> <year>/<month>/<day> <hour>:<minute>:<second>`, such as `4 2004/1/1 22:01:01` for Thursday, January 4, 2004 at 10:01 p.m.

See the `dhclient.conf` man page for more information on additional settings.

DHCP Server

Again, the easiest way to install the DHCP server on your computer is to use either `synaptic` or `apt-get` to retrieve the dchp3-server package. If you are so inclined, you can go to the Internet Software Consortium (ISC) website and download and build the source code yourself (http://www.isc.org/).

If you decide to install from a source downloaded from the ISC website, the installation is very straightforward. Just unpack your `tar` file, run `./configure` from the root of the source directory, run `make`, and finally, if there are no errors, run `make install`. This puts all the files used by the DHCP daemon in the correct places. If you have the disk space, it is best to leave the source files in place until you are sure that DHCP is running correctly; otherwise, you can delete the source tree.

NOTE

For whichever installation method you choose, be sure that a file called `/etc/dhcp3/dhcpd.leases` is created. The file can be empty, but it does need to exist in order for `dhcpd` to start properly.

Using DHCP to Configure Network Hosts

Configuring your network with DHCP can look difficult, but is actually easy if your needs are simple. The server configuration can take a bit more work if your network is more complex and depending on how much you want DHCP to do.

DHCP Server Configuration

Configuring the server takes some thought and a little bit of work. Luckily, the work involves editing only a single configuration file, `/etc/dhcp3/dhcpd.conf`. To start the server at boot time, use the `service`, `ntsysv`, or `bum` commands.

The `/etc/dhcp3/dhcpd.conf` file contains all the information needed to run `dhcpd`. Ubuntu includes a sample `dhcpd.conf` in `/usr/share/doc/dhcp*/dhcpd.conf.sample`. The DHCP server source files also contain a sample `dhcpd.conf` file.

The `/etc/dhcp3/dhcpd.conf` file can be looked at as a three-part file. The first part contains configurations for DHCP itself. The configurations include

- **Setting the domain name**—`option domain-name "example.org"`.
- **Setting DNS servers**—`option domain-name-servers ns1.example.org, ns2.example.org` (IP addresses can be substituted.)
- **Setting the default and maximum lease times**—`default-lease-time 3600` and `max-lease-time 14400`.

Other settings in the first part include whether the server is the primary (authoritative) server and what type of logging DHCP should use. These settings are considered defaults and can be overridden by the subnet and host portion of the configuration in more complex situations.

NOTE

The `dhcpd.conf` file requires semicolons (`;`) after each command statement. If your configuration file has errors or runs improperly, check for this.

The next part of the `dhcpd.conf` deals with the different subnets that your DHCP server serves; this section is quite straightforward. Each subnet is defined separately and can look like this:

```
subnet 10.5.5.0 netmask 255.255.255.224 {
 range 10.5.5.26 10.5.5.30;
 option domain-name-servers ns1.internal.example.org;
 option domain-name "internal.example.org";
 option routers 10.5.5.1;
 option broadcast-address 10.5.5.31;
 default-lease-time 600;
 max-lease-time 7200;
}
```

This defines the IP addressing for the `10.5.5.0` subnet. It defines the IP address ranging from `10.5.5.26` through `10.5.5.30` to be dynamically assigned to hosts that reside on that subnet. This example shows that any TCP/IP option can be set from the subnet portion of the configuration file. It shows which DNS server the subnet will connect to, which can be good for DNS server load balancing, or which can be used to limit the hosts that can be reached through DNS. It defines the domain name, so you can have more than one domain on your network. It can also change the default and maximum lease time.

If you want your server to ignore a specific subnet, the following entry can be used to accomplish this:

```
subnet 10.152.187.0 netmask 255.255.255.0 {
}
```

This defines no options for the `10.152.187.0` subnet; therefore, the DHCP server ignores it.

The last part of your `dhcp.conf` is for defining hosts. This can be good if you want a computer on your network to have a specific IP address or other information specific to that host. The key to completing the host section is to know the hardware address of the host. As you learned in "Hardware Devices for Networking," earlier in this chapter, the hardware address is used to differentiate the host for configuration. Your hardware address can be obtained by using the `ifconfig` command as described previously. The hardware address is on the `eth0` line labeled `"Hwaddr"`.


```
host prowl {
  hardware ethernet 08:00:07:26:c0:a5;
  fixed-address prowl.hudson.com;
}
```

This example takes the host with the hardware address `08:00:07:26:c0:a5` and does a DNS lookup to assign the IP address for `prowl.hudson.com` to the host.

DHCP can also define and configure booting for diskless clients like this:

```
host bumblebee {
  hardware ethernet 0:0:c0:5d:bd:95;
  filename "vmunix.bumblebee";
  server-name "optimus.hudson.com";
}
```

The diskless host `bumblebee` will get its boot information from server `optimus.hudson.com` and use `vmunix.bumblebee` kernel. All other TCP/IP configuration can also be included.

CAUTION

Remember, only one DHCP server should exist on a local network to avoid problems. Your DHCP might not work correctly on a LAN with hosts running outdated legacy operating systems. Often Windows NT servers will have the Windows DHCP server installed by default. Because there is no configuration file for NT to sort through, that DHCP server configures your host before the Linux server if both machines are on the same LAN. Check your NT servers for this situation and disable DHCP on the NT server; afterward, your other DHCP-enabled hosts should configure correctly. Also, check to make sure that there are no conflicts if you use a cable or DSL modem, wireless access point (WAP), or other intelligent router on your LAN that can provide DHCP.

Other Uses for DHCP

A whole host of options can be used in `dhcpd.conf`: Entire books are dedicated to DHCP. The most comprehensive book is *The DHCP Handbook*, available at http://www.dhcp-handbook.com/. You can define NIS domains, configure NETBIOS, set subnet masks, and define time servers, or many other types of servers—to name a few of the DHCP options you can use. The preceding example will get your DHCP server and client up and running.

The DHCP server distribution contains an example of the `dhcpd.conf` file that you can use as a template for your network. The file shows a basic configuration that can get you started with explanations for the options used.

Wireless Networking

As stated earlier, Linux has had support for wireless networking since the first standards were developed in the early 1990s. With computers getting smaller and smaller, the uses for wireless networking increased; meanwhile, the transmission speeds are increasing all the time. There are several different ways to create a wireless network. The following sections introduce you to several Linux commands you can use to initialize, configure, and manage wireless networking on your Ubuntu system.

Support for Wireless Networking in Ubuntu

The Linux kernel that ships with Ubuntu provides extensive support for wireless networking. Related wireless tools for configuring, managing, or displaying information about a wireless connection include

- `iwconfig`—Sets the network name, encryption, transmission rate, and other features of a wireless network interface
- `iwlist`—Displays information about a wireless interface, such as rate, power level, or frequency used
- `iwpriv`—Uses `i` to set optional features, such as roaming, of a wireless network interface
- `iwspy`—Shows wireless statistics of a number of nodes

Support varies for wireless devices, but most modern (that is, post-2005) wireless devices should work with Ubuntu. In general, Linux wireless device software (usually in the form of a kernel module) support the creation of an ethernet device that can be managed by traditional interface tools such as `ifconfig`—with wireless features of the device managed by the various wireless software tools.

For example, when a wireless networking device is first recognized and initialized for use, the driver will most likely report a new device:

```
zd1211rw 5-4:1.0: firmware version 4725

zd1211rw 5-4:1.0: zd1211b chip 050d:705c v4810 \
high 00-17-3f AL2230_RF pa0 g—NS

zd1211rw 5-4:1.0: eth2

usbcore: registered new interface driver zd1211rw
```

This output (from the `dmesg` command) shows that the `eth2` device has been reported. If DHCP is in use, the device should automatically join the nearest wireless subnet and be automatically assigned an IP address. If not, the next step is to use a wireless tool such as `iwconfig` to set various parameters of the wireless device. The `iwconfig` command, along with the device name (`eth2` in this example), will show the status:


```
$ iwconfig eth2
eth2      IEEE 802.11b/g  ESSID:"SKY35120"  Nickname:"zd1211"
          Mode:Managed  Frequency:2.462 GHz  \
Access Point: 00:18:4D:06:8E:2A
          Bit Rate=24 Mb/s
          Encryption key:0EFD-C1AF-5C8D-B2C6-7A89-3790-07A7-AC64-0AB5\
-C36E-D1E9-A230-1DB9-D227-2EB6-D6C8   Security mode:open
          Link Quality=100/100  Signal level=82/100
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

This example shows a 24Mbps connection to a network named `SKY35120`. To change a parameter, such as the transmission rate, use a command-line option with the `iwconfig` command like so:

```
$ sudo iwconfig eth0 rate 11M
```

Other options supported by the `iwconfig` command include `essid`, used to set the NIC to connect to a specific network by `named`; `mode`, used to enable the NIC to automatically retrieve settings from an access point or connect to another wireless host; or `freq`, to set a frequency to use for communication. Additional options include `channel`, `frag`, `enc` (for encryption), `power`, and `txpower`. Details and examples of these options are in the `iwconfig` manual page.

You can then use the `ifconfig` command or perhaps a graphical Ubuntu tool to set the device networking parameters, and the interface will work as on a hardwired LAN. One handy output of the `iwconfig` command is the link quality output, which can be used in shell scripts or other graphical utilities for signal monitoring purposes.

Advantages of Wireless Networking

Advantages of wireless networking are its mobility and potential range. If you have a large enough antenna network, your network can stretch many miles. This would be an expensive network, but one that would easily break out of the brick and mortar confines of the office.

Wireless networking would also be a great advantage to college campuses to eliminate the need to tear through walls to install cabling because more and more students expect to have a network connection in their dorm rooms. Wireless networking cards are very reasonable in price and can easily be issued to each student as he requires them.

Home networkers can also benefit from wireless networking. For those who cannot do wired network modifications to their homes, wireless networking removes the unsightly wires running along baseboards and ceilings that are required to connect computers in different rooms. With a wireless home network, you are not even confined to inside the house. Depending on the transmit power of your router, you can sit out in your backyard and watch clouds drifting by as you type away.

Choosing the right types of wireless devices is an important decision. The next sections discuss some of the basic differences between current protocols used for wireless networking.

Choosing from Among Available Wireless Protocols

The Institute of Electrical and Electronics Engineers (IEEE) started to look seriously at wireless networking in 1990. This is when the 802.11 Standard was first introduced by the Wireless Local Area Networks Standards Working Group. The group based the standard roughly around the architecture used in cellular phone networks. The wireless network is controlled by a base station, which can be just a transmitter attached to the network or, more commonly these days, a router.

Larger networks can use more than one base station. Networks with more than one base station are usually referred to as *distribution systems*. A distribution system can be used to increase coverage area and support roaming of wireless hosts. You can also employ external omnidirectional antennas to increase coverage area, or if required, use point-to-point, or directional antennas to connect distant computers or networks. Right now, the least expensive wireless Linux networks are built using devices (such as access points or NICs) supporting 802.11b, although the faster 802.11g devices tend to get more shelf space. Devices are starting to appear marketed as Pre-N, meaning that they implement a draft standard, while the IEEE carry on debating the full N standard. Significantly more power throughput and range are promised by hardware that supports Pre-N, but this specification has yet to be formally agreed upon and is still some time off.

An early standard, 802.11a, offers greater transmission rates than 802.11b, and a number of 802.11a wireless NICs are available (some products provide up to 72Mbps, but will not work with 802.11b devices). Wireless networking devices based on 802.11g, which has the speed improvement of 802.11a and is compatible with 802.11b, are becoming more widely available. Other wireless protocols include Bluetooth, which provides up to

720Kbps data transfers. Bluetooth is intended for short-range device communications (such as for a printer) and supports a typical range of only 10 meters. Bluetooth is unlike IrDA, which requires line-of-sight (devices that are aimed at each other). Bluetooth use conflicts with 802.11 networks because it also uses the 2.4GHz band. You can find out more by browsing to http://www.bluetooth.com/.

The 802.11 standard specifies that wireless devices use a frequency range of 2400–2483.5MHz. This is the standard used in North America and Europe. In Japan, however, wireless networks are limited to a frequency range of 2471MHz–2479MHz because of Japanese regulations. Within these ranges, each network is given up to 79 non-overlapping frequency channels to use. This reduces the chance of two closely located wireless networks using the same channel at the same time. It also allows for channel hopping, which can be used for security.

Beyond the Network and onto the Internet

Ubuntu supports Internet connections and the use of Internet resources in many different ways. You will find a wealth of Internet-related software included with this book's version of Ubuntu, and you can download hundreds of additional free utilities from a variety of sources. To use them, you must have a working Internet connection.

In this section, you learn how to set up an Internet connection in Ubuntu using a modem and Point-to-Point Protocol (PPP) as well as other connection methods, including Digital Subscriber Line (DSL) and cable modem services. Just a few years ago, getting a dial-up connection working was difficult—hence, an entire chapter of this book was devoted to it. Nowadays, as long as you have a hardware modem, dial-up configuration is simple. The Ubuntu developers and the wider Linux community have made great progress in making connectivity easier.

Although many experienced Linux users continue to use manual scripts to establish their Internet connectivity, new users and experienced system administrators alike will find Ubuntu's graphical network configuration interface, the Internet Connection Wizard, much easier to use. You learn how to use the Internet Connection Wizard in this chapter, as well as how to configure Ubuntu to provide dial-in PPP support. The chapter also describes how to use Roaring Penguin's DSL utilities for managing connectivity through a cable modem connection.

Common Configuration Information

Although Ubuntu enables great flexibility in configuring Internet connections, that flexibility comes at the price of an increase in complexity. To configure Internet connectivity in Ubuntu, you must know more about the details of the connection process than you can learn from the information typically provided by your Internet service provider (ISP). In this section of the chapter, you learn what to ask about and how to use the information.

Some ISPs are unaware of Linux or unwilling to support its use with their service. Fortunately, that attitude is rapidly changing, and the majority of ISPs offer services using

standard protocols that are compatible with Linux, even if they (or their technical support people) aren't aware that their own ISPs are Linux-friendly. You just need to press a little for the information you require.

If you are using a dial-up modem account (referred to in Linux as *PPP* for the Point-to-Point Protocol it uses), your ISP will provide your computer with a static or dynamic IP (Internet Protocol) address. A dynamic IP address changes each time you dial in, whereas a static IP address remains the same. The ISP also might automatically provide your computer with the names of the Domain Name Service (DNS) servers. You need to know the telephone number that your computer will dial in to for making the connection; your ISP supplies that number, too. You will also need a working modem and need to know the device name of the modem (usually `/dev/modem`).

NOTE

Most IP addresses are dynamically assigned by ISPs; ISPs have a pool of addresses, and you get whatever address is available. From the ISP's viewpoint, a small number of addresses can serve a large number of people because not everyone will be online at the same time. For most Internet services, a dynamic IP works well because it is the ISP's job to route that information to you, and it sits in the middle—between you and the service you want to use. But a dynamic IP address changes, and if someone needs to find you at the same address (if you run a website or a file transfer site, for example), an IP that changes every time you log on will not work well. For that, you need a static IP. Because your ISP cannot reuse that IP with its other customers, it will likely charge you more for a static IP than a dynamic IP. The average consumer doesn't need the benefit of a static IP, so he is happy paying less for a dynamically assigned IP. Also, the DNS information can be provided automatically by the ISP by the Dynamic Host Configuration Protocol, or DHCP.

If you are using DSL access or a cable modem, you might have a dynamic IP provided through DHCP, or you might be assigned a static IP. You might automatically be provided with the names of the DNS servers if you use DHCP, or you might have to set up DNS manually (in which case, you have to know the IP addresses of the DNS servers).

In all cases, you have to know your username, your password, and for the configuration of other services, the names of the mail servers and the news server. This information can be obtained from your ISP if you specifically ask for it.

NOTE

The information in this book will help you understand and avoid many connection issues, but you might experience connection problems. Keep the telephone number of the technical help service for your ISP on hand in case you are not able to establish a connection. But be aware that few ISPs offer Linux support, and you might need to seek help from a Linux-savvy friend or a Linux user's group if your special circumstances cannot be handled from the knowledge you gain from this book. Of course, the best place to look is on the Internet. Use Google's Linux page (http://www.google.com/linux/) to research the problem and see if any other users have found fixes or workarounds.

Configuring Digital Subscriber Line Access

Ubuntu also supports the use of a digital subscriber line (DSL) service. Although it refers to the different types of DSL available as xDSL, that name includes ADSL, IDSL, SDSL, and other flavors of DSL service; they can all be configured using the Internet Connection Wizard. DSL service generally provides 256Kbps to 24Mbps transfer speeds and transmits data over copper telephone lines from a central office to individual subscriber sites (such as your home). Many DSL services provide asymmetric speeds with download speeds greater than upload speeds.

NOTE

DSL service is an "always-on" type of Internet service, although you can turn off the connection under Ubuntu using the network configuration tool found under System, Administration, Network. An always-on connection exposes your computer to malicious abuse from crackers who trawl the Internet attempting to gain access to other computer systems. In addition to the capability to turn off such connections, Ubuntu is also preconfigured to not listen on any network ports, which means that any attempts to gain access to your computer will fail as Ubuntu will reject the request. This is the Ubuntu equivalent of putting up a 12-foot steel fence surrounding your computer.

A DSL connection requires that you have an ethernet network interface card (sometimes a USB interface that is not easily supported in Linux) in your computer or notebook. Many users also configure a gateway, firewall, or other computer with at least two network interface cards in order to share a connection with a LAN. We looked at the hardware and protocol issues earlier on in this chapter. Advanced configuration of a firewall or router, other than what was addressed during your initial installation of Ubuntu, is beyond the scope of this book.

Understanding Point-to-Point Protocol over Ethernet

Establishing a DSL connection with an ISP providing a static IP address is easy. Unfortunately, many DSL providers use a type of PPP protocol named Point-to-Point Protocol over Ethernet (*PPPoE*) that provides dynamic IP address assignment and authentication by encapsulating PPP information inside ethernet frames. Roaring Penguin's `rp-pppoe` clients are available from the Roaring Penguin site (www.roaringpenguin.com/penguin/pppoe/rp-pppoe-3.8.tar.gz), and these clients make the difficult-to-configure PPPoE connection much easier to deal with. You can download and install newer versions (see the Roaring Penguin link in the "Reference" section at the end of this chapter).

NOTE

ADSL modems were frequently supplied by ISPs when they originally started to roll out ADSL services. Nowadays however, these modems are optional, which is a good thing as many people choose to purchase a router with an in-built modem to create a dedicated connection. Using a router can save many headaches and will allow you to easily connect more than one computer to an Internet connection. Note that if you are using a cable connection then they usually come with an ethernet cable, in which case you just need a router. Check with your ISP before buying to ensure that whatever router you do end up with can be supported by them. You might find that your ISP even supplies a router as part of the package!

Configuring a PPPoE Connection Manually

You should only need to use these steps if you are using a modem supplied by your ISP, and not a router. The basic steps involved in manually setting up a DSL connection using Ubuntu involve connecting the proper hardware, and then running a simple configuration script if you use `rp-pppoe` from Roaring Penguin.

First, connect your DSL modem to your telephone line, and then plug in your ethernet cable from the modem to your computer's network interface card. If you plan to share your DSL connection with the rest of your LAN, you need at least two network cards—designated `eth0` (for your LAN) and `eth1` (for the DSL connection).

The following example assumes that you have more than one computer and will share your DSL connection on a LAN.

First, log in as `root`, and ensure that your first `eth0` device is enabled and up (perhaps using the `ifconfig` command). Next, bring up the other interface, but assign a null IP address like this:

```
$ sudo /sbin/ifconfig eth1 0.0.0.0 up
```

Now use the `adsl-setup` command to set up your system. Type the command like this:

```
$ sudo /sbin/adsl-setup
```

You will be presented with a text script and be asked to enter your username and the Ethernet interface used for the connection (such as `eth1`). You will then be asked to use "on demand" service or have the connection stay up all the time (until brought down by the root operator). You can also set a timeout in seconds, if desired. You'll then be asked to enter the IP addresses of your ISP's DNS servers if you haven't configured the system's `/etc/resolv.conf` file.

After that, you will be prompted to enter your password two times, and have to choose the type of firewall and IP masquerading to use. (You learned about IP masquerading in the "Using IP Masquerading in Ubuntu" section, earlier in this chapter.) The actual

configuration is done automatically. Using a firewall is essential nowadays, so you should choose this option unless you intend to craft your own set of firewall rules—a discussion of which is beyond the scope of this book. After you have chosen your firewall and IP masquerading setup, you will be asked to confirm, save, and implement your settings. You are also given a choice to allow users to manage the connection, a handy option for home users.

Changes will be made to your system's `/etc/sysconfig/network-scripts/ifcfg-ppp0`, `/etc/resolv.conf`, `/etc/ppp/pap-secrets`, and `/etc/ppp/chap-secrets` files.

After configuration has finished, use the `adsl-start` command to start a connection and DSL session, like this:

```
$ sudo /sbin/adsl-start
```


The DSL connection should be nearly instantaneous, but if problems occur, check to make sure that your DSL modem is communicating with the phone company's central office by examining the status LEDs on the modem. Because this varies from modem to modem, consult your modem user's manual.

Check to make certain that all cables are properly attached, that your interfaces are properly configured, and that you have entered the correct information to the setup script.

If IP masquerading is enabled, other computers on your LAN on the same subnet address (such as `192.168.0.`*`XXX`*) can use the Internet, but must have the same `/etc/resolv.conf` name server entries and a routing entry with the DSL-connected computer as a gateway. For example, if the host computer with the DSL connection has an IP address of `192.168.0.1`, and other computers on your LAN use addresses in the `192.168.0.`*`XXX`* range, use the `route` command on each computer like this:

```
# /sbin/route add default gw 192.168.0.1
```

Note that you can also use a hostname instead if each computer has an `/etc/hosts` file with hostname and IP address entries for your LAN. To stop your connection, use the `adsl-stop` command like this:

```
# /sbin/adsl-stop
```

Configuring Dial-Up Internet Access

Most ISPs provide dial-up connections supporting PPP because it is a fast and efficient protocol for using TCP/IP over serial lines. PPP is designed for two-way networking; TCP/IP provides the transport protocol for data. One hurdle faced by new Ubuntu users is how to set up PPP and connect to the Internet. It is not necessary to understand the details of the PPP protocol in order to use it, and setting up a PPP connection is easy. You can configure the PPP connections manually using the command line or graphically during an X session using Ubuntu's Network Configuration Tool. Each approach produces the same results.

PPP uses several components on your system. The first is a daemon called `pppd`, which controls the use of PPP. The second is a driver called the high-level data link control (*HDLC*), which controls the flow of information between two machines. A third component of PPP is a routine called `chat` that dials the other end of the connection for you when you want it to. Although PPP has many "tunable" parameters, the default settings work well for most people.

Configuring a Dial-Up Connection Manually

Ubuntu includes some useful utilities to get your dial-up connection up and running. In this section we will take a look at two options that will have you on the Internet in no time.

The first way is to configure a connection using `pppconfig`, a command line utility to help you to configure specific dial-up connection settings.

Enter the following command:

```
$ sudo pppconfig
```

Before you connect for the first time you need to add yourself to both the dip and dialout groups by using the commands:

```
$ sudo adduser YOURNAMEHERE dip
$ sudo adduser YOURNAMEHERE dialout
```

Once this has been done it is just a simple matter of issuing the `pon` command to connect, and the `poff` command to disconnect. You can create as many different profiles as you need, and can launch specific ones by using the command `pon profilename`, again using the `poff` command to disconnect.

CAUTION

Many software modems will not work with Linux because the manufacturers will not release programming information about them or provide Linux drivers. An external serial port modem or ISA bus modem will almost always work; USB and PCI modems are still problematic. It is suggested that you do a thorough Google search using your modem's name and model number to see how others have solved problems with that particular modem. Links to software modem compatibility sites appear at the end of this chapter.

An alternative to using `pppconfig` is to use the dialog within `network-admin` to configure and activate/deactivate the dial-up connection. Open up `network-admin` through the System menu under the Administration sub-menu. Select your modem in the list and click properties. The first thing you will need to do is to enable the connection by ticking the Enable this connection box, as shown in Figure 14.4.

Make sure to enter your ISP's phone number as well as your username and password; essential information if you are to succeed at getting on to the Internet!

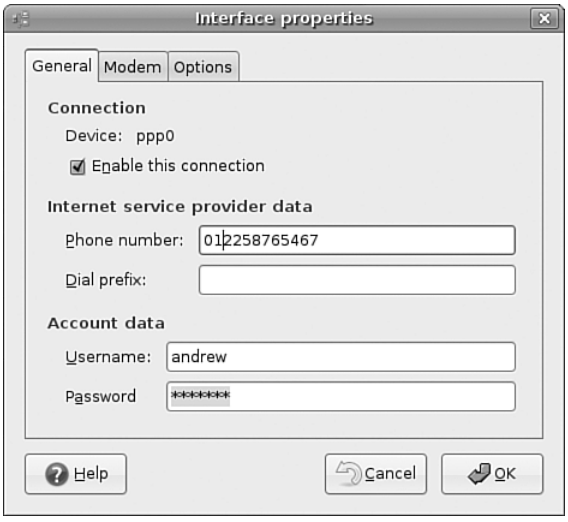


FIGURE 14.4 Simple and to the point, network-admin makes configuring a dial-up connection easy. To start with, enable the connection!

Next, click on the modem tab to specify details about your modem and to also configure the speaker volume (Figure 14.5). Most telephone systems use tone dialing nowadays, so make sure this is selected over the older pulse style dialing. I like to have the speaker volume on low, so it is not too intrusive but it allows me to quickly troubleshoot any problems by listening when I am connecting.

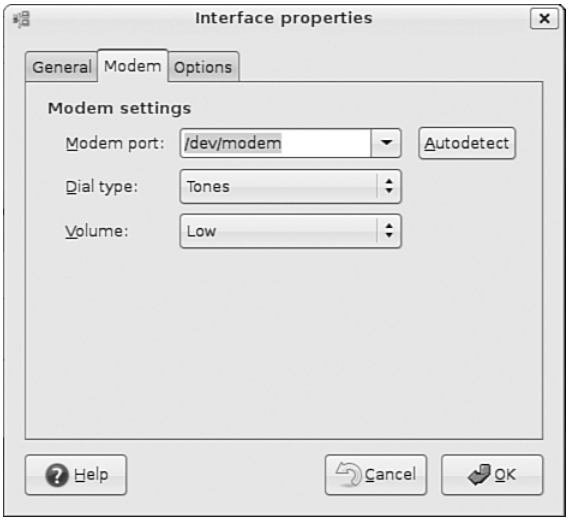


FIGURE 14.5 Keep the speaker volume turned down low otherwise you'll hear incessant screeching during the dial-up process.

The final step is to configure some general options for the connection, mainly to do with how Ubuntu works with the connection. If you are using a laptop, then you will probably want to un-tick Set modem as default route to the Internet whilst you are on a LAN, otherwise you may struggle to see anything! Tick it when you are expecting to use your dial-up connection though, and Ubuntu will use this connection to get out onto the Internet. I would suggest keeping the second option ticked, mainly because the name-servers will be automatically assigned to you when you connect to your ISP and should prevent any problems whilst surfing. Finally, if you have a bad line and are constantly getting kicked off then it might be an idea for you to tick the retry connection option so Ubuntu will attempt to keep you connected should your connection drop out for whatever reason. See Figure 14.6 for a look at the Options tab.

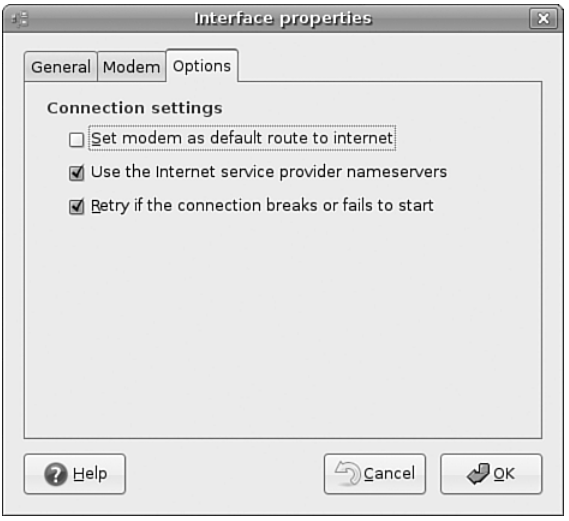


FIGURE 14.6 A couple of choices here will ensure uninterrupted surfing time.

Troubleshooting Connection Problems

The Linux Documentation Project at http://www.tldp.org/ offers many in-depth resources for configuring and troubleshooting these connections. The Internet search engine Google is also an invaluable tool for dealing with specific questions about these connections. For many other useful references, see the “Reference” section at the end of this chapter.

Here are a few troubleshooting tips culled from many years of experience:

- If your modem connects and then hangs up, you are probably using the wrong password or dialing the wrong number. If the password and phone number are correct, it is likely an authentication protocol problem.

- If you get connected but cannot reach websites, it is likely a domain name resolver problem, meaning that DNS is not working. If it worked yesterday and you haven't "adjusted" the associated files, it is probably a problem at the ISP's end. Call and ask.
- Always make certain that everything is plugged in. Check again—and again.
- If the modem works in Windows, but not in Linux no matter what you do, it is probably a software modem no matter what it said on the box.
- If everything just stops working (and you do not see smoke), it is probably a glitch at the ISP or the telephone company. Take a break and give them some time to fix it.
- Never configure a network connection when you have had too little sleep or too much caffeine; you will just have to redo it tomorrow.

Related Ubuntu and Linux Commands

You will use these commands when managing network connectivity in your Ubuntu system:

`dhclient`—Automatically acquire, and then set IP info for a NIC

`ethereal`—GNOME graphical network scanner

`firestarter`—Ubuntu's basic graphical firewalling tool for X

`ifconfig`—Displays and manages Linux networking devices

`iwconfig`—Displays and sets wireless network device parameters

`route`—Displays and manages Linux kernel routing table

`ssh`—The OpenSSH remote-login client and preferred replacement for `telnet`

`network-admin`—Ubuntu's GUI for configuring network connections

Reference

The following websites and books are great resources for more information on the topics covered in this chapter. Networking is complex. The more you take the time to learn, the easier setting up and maintaining your network will be.

General

- http://www.ietf.org/rfc.html—Go here to search for, or get a list of, Request for Comments (RFC).

DHCP

- http://www.oth.net/dyndns.html—For a list of Dynamic DNS service providers, go to this site.
- http://www.isc.org/products/DHCP/dhcpv3-README.html—The DHCP README is available at this site.

Wireless

- http://www.ieee.org—The Institute of Electrical and Electronics Engineers (IEEE) website.
- http://www.mozillaquest.com/Network_02/Wireless_Network_Technology_03_Story-01.html—Wireless networking with Red Hat 7.2.

Books

- *Sams Teach Yourself TCP/IP Network Administration in 21 Days*, Sams Publishing, ISBN: 0-672-31250-6
- *TCP/IP Network Administration*, O'Reilly Publishing, ISBN: 1-56592-322-7
- *Practical Networking*, Que Publishing, ISBN: 0-7897-2252-6
- *Samba Unleashed*, Sams Publishing, ISBN: 0-672-31862-8
- *The DHCP Handbook*, Sams Publishing, ISBN: 0-672-32327-3