Excerpts of Chapters from

A Practical Guide to Fedora™ and Red Hat® Enterprise Linux®

SEVENTH EDITION

Mark G. Sobell

ISBN-13: 978-0-13-347743-6

Upper Saddle River, NJ • Boston • Indianapolis • San Francisco
New York • Toronto • Montreal • London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

BLANK

3

Step-by-Step Installation

In This Chapter

Objectives

After reading this chapter you should be able to:

- Run a live session and use gnome-disks to view and change disk partitioning
- Install Fedora from a live session
- Install Fedora/RHEL using an Install Image
- Modify system behavior using boot parameters
- Modify partitions during installation
- Select software during installation
- List the requirement and considerations for a dual-boot configuration

Chapter 2 covered planning the installation of Fedora/RHEL: determining the requirements; planning the layout of the hard disk; obtaining the files you need for the installation, including how to download and burn or write Install, Live, and Network Images to installation media; and collecting information about the system. This chapter focuses on installing Fedora/RHEL. Frequently the installation is quite simple, especially if you have done a good job of planning. Sometimes you might run into a problem or have a special circumstance; this chapter gives you tools to use in these cases. Read as much of this chapter as you need to; once you have installed Fedora/RHEL, continue with Chapter 4, which covers getting started using the Fedora/RHEL desktop and command line.

THE NEW ANACONDA INSTALLER

The new Anaconda installer, introduced in Fedora 18, has been totally rewritten. The Anaconda user interface changed from a linear (wizard) model to a hub-and-spoke model. Using this model means Anaconda can perform background processing (e.g., you can still be entering information while Anaconda is installing packages) and you can skip screens that are not pertinent to your installation. See fedoraproject.org/wiki/Anaconda/NewInstaller for more information.

Chapter 17 explains how to set up a virtual system

tip To install Fedora/RHEL on a virtual system, you build the virtual system and then follow the instructions in this chapter for installing the operating system. See page 663 for instructions on setting up a QEMU/KVM virtual machine and page 671 for setting up a VMware virtual machine.

Upgrading a Fedora system

tip The new Anaconda installer will not upgrade a Fedora system from one release to the next. You must use FedUp (FEDora UPgrader), which works on Fedora 17 and later. Visit the Fedora Web page at fedoraproject.org/wiki/FedUp for more information.

RUNNING A FEDORA LIVE SESSION

As discussed in Chapter 2, a live session is a Linux session you run on a computer without installing Linux on the computer. When you reboot after a live session, the computer is untouched. If you are running Windows, after a live session Windows boots the way it did before the live session. If you choose, you can install Fedora from a live session. RHEL does not offer a live session.

A live session gives you a chance to preview Fedora without installing it. Boot from a Live Image to begin a live session and work with Fedora as explained in Chapter 4. When you are finished, remove the installation medium and reboot the system. The system will boot as it did before you ran the live session.

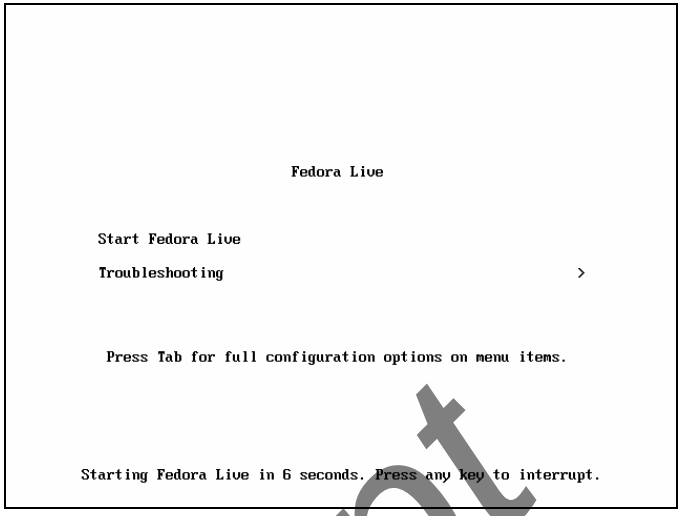

Figure 3-1 The Live Image Boot menu

Preserving files

Because a live session does not write to the hard disk (other than using a swap partition, if one is available), none of the work you save will be available once you reboot. You can use a USB flash drive, email, or another method to transfer files you want to preserve to another system. The liveusb-creator utility (page 50) can create persistent storage on a USB drive that you can write to from a live session.

BOOTING THE SYSTEM

Before Fedora can display the desktop of a live session or install itself on a hard disk, the Linux operating system must be read into memory (booted). This process can take a few minutes on older, slower systems and systems with minimal RAM (memory).

In most cases, you can boot Fedora to run a live session that displays a desktop without doing anything after you boot from a Live Image. To begin, insert the installation medium holding a Live Image (the standard GNOME **Fedora Desktop Live Media**) into the system and turn on or reset the system. Refer to "BIOS setup" on page 31 if the system does not boot from the CD, DVD, or USB drive. Or refer to "Modifying Boot Parameters (Options)" on page 70 if Fedora does not boot or displays an error message.

A few moments after you start the system, Fedora displays a screen that says **Starting Fedora Live in 10 seconds. Press any key to interrupt.** and counts down from 10 (Figure 3-1). Next the system displays some messages and then a graphical outline of the Fedora logo that gets filled in as the system boots. The screen goes blank and then Fedora displays a Welcome to Fedora window. In this window you can click **Try Fedora** or **Install to Hard Drive**.

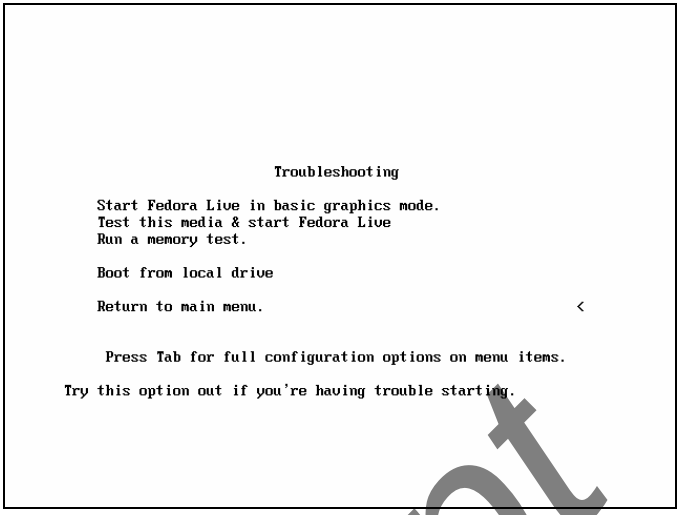

Figure 3-2 The Troubleshooting menu

Checking the installation medium

The first time you use an installation medium, it is a good idea to check it for defects. To do so, interrupt the automatic boot by pressing the SPACE bar while Fedora is counting down. Use the DOWN ARROW key to highlight the **Troubleshooting** line and press RETURN (the mouse will not work yet). Fedora displays the Troubleshooting menu (Figure 3-2). Use the DOWN ARROW key to highlight **Test this media & start Fedora Live**; press RETURN.

Anaconda displays a line labeled **Checking** that shows the percent of the disk it has checked as it verifies the contents of the installation medium. If the installation medium is good, the system boots.

Memory test

Selecting **Run a memory test** from the Troubleshooting menu runs memtest86+, a GPL-licensed, stand-alone memory test utility for x86-based computers. Press **C** to configure the test; press ESCAPE to exit and reboot. See www.memtest.org for more information.

Fedora provides comprehensive installation documentation

tip Visit docs.fedoraproject.org/en-US/Fedora/19/html/Installation_Quick_Start_Guide to display the Fedora Installation Quick Start Guide. To display the complete Fedora Installation Guide, visit docs.fedoraproject.org/en-US/Fedora/19/html/Installation_Guide.

GNOME

If you are booting from Fedora Desktop Live Media (what this book refers to as *a Live Image*), the system will run the GNOME desktop manager; Fedora automatically logs in as the user named **liveuser** and displays the Welcome to Fedora window. This window has two large buttons: **Try Fedora** and **Install to Hard Drive**. When you click **Try Fedora**, GNOME displays a window that explains how to install Fedora to the hard disk. Click **Close** and Fedora displays the GNOME desktop (Figure 3-3, page 60).

KDE

If you are booting from a Fedora KDE Live Image, the system will run the KDE desktop manager. When you boot from this disk, Fedora next displays a KDE startup screen and then the KDE desktop—there is no need to log in.

optional

Seeing What Is Going on

If you are curious and want to see what Fedora is doing as it boots from a Live Image, remove **quiet**, which controls kernel messages, and **rhgb** (Red Hat graphical boot), which controls messages from the graphical installer, from the boot parameters. See Figure 3-8 on page 70; the list of parameters on the screen will be different from those in the figure. With the Fedora Live Image Boot menu displayed (Figure 3-1) and the Start Fedora Live line highlighted, press TAB to display the boot command-line parameters. Use the LEFT ARROW key to back up over—but not remove—any words to the right of **rhgb**. Press BACKSPACE or DEL to back up over and erase **rhgb** and **quiet** from the boot command line. Press RETURN. Now as Fedora boots, it displays information about what it is doing. Text scrolls on the screen, although sometimes too rapidly to read. When you boot Fedora from an Install Image and when you boot RHEL, this information is displayed by default: You do not have to change the command line.

Installing Fedora/RHEL

You can install Fedora from a live session (preceding) or install Fedora/RHEL from an Install Image. Installing from a live session is simpler but does not give you the flexibility installing from an Install Image does. For example, you cannot select the language Anaconda uses, nor can you choose which software packages you will install when you install from a live session.

See what is on the hard disk (and back it up) before installing Linux

caution Unless you are certain the hard disk you are installing Fedora/RHEL on has nothing on it (it is a new disk) or you are sure the disk holds no information of value, it is a good idea to examine the contents of the disk before you start the installation. You can use the gnome-disks disk utility (page 78) from a live session for this purpose. Back up whatever is on the disk, even if it is in a partition the installation will not write to.

An Install Image holds many of the software packages Fedora/RHEL supports. You can install whichever package groups you like from an Install Image without connecting to the Internet. However, without an Internet connection, you will not be able to update the software once the system is installed.

A Live Image holds a limited set of software packages. Once you install from a Live Image, you must connect to the Internet to update the software on the system and to download and install additional packages.

To begin most installations, insert the medium holding a Live or an Install Image into the system and turn on or reset the system. For hard disk and network-based installations, you can use an Install or a Network Image.

Refer to "BIOS setup" on page 31 if the system does not boot from the installation medium.

Figure 3-3 The GNOME 3 Live desktop

Installing from a Live Session (Fedora)

Bring up a live GNOME session as explained on page 56. GNOME will display a GNOME 3 desktop that has the word **Activities** in the upper-left corner of the screen.

The Anaconda installer does not modify the live environment

tip The changes you make as you install Fedora from a live session do not affect the live session. For example, when you set up networking for the system you are installing, the network connection for the live environment does not change.

From a GNOME 3 desktop, click **Activities**; GNOME displays icons along the left side of the screen and a text box with the words **Type to search** in it (Figure 3-3). Near the bottom of the icons is an icon depicting a hard drive with a green tick on top of it. Left-click this icon when you want to install Fedora. Continue reading at "Using Anaconda" on page 63.

Installing from an Install Image

To install Fedora/RHEL from an Install Image, insert the installation medium and turn on or reset the system. After a few moments, the system displays the Fedora/RHEL Boot menu (Figure 3-4) and a message that says **Automatic boot in 60 seconds**. Refer to "BIOS setup" on page 31 if the system does not boot from the installation medium. Refer to "Modifying Boot Parameters (Options)" on page 70 if Fedora/RHEL does not boot or displays an error message.

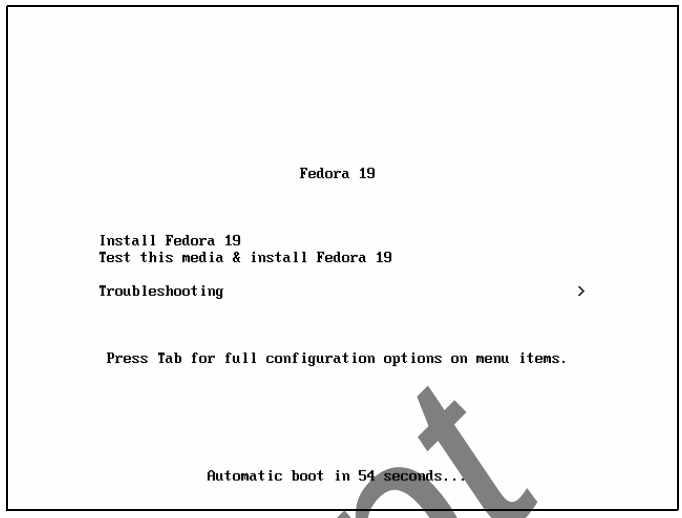

Figure 3-4 The Install Image Boot menu

Press the SPACE bar within 60 seconds to stop the countdown. If you do not press the SPACE bar, after 60 seconds Fedora/RHEL begins a graphical installation.

The Fedora Install Image Boot menu has the following selections:

Install Fedora — Installs a Fedora/RHEL system using the Anaconda graphical installer.

Test this media & install Fedora — Tests the installation medium and installs a Fedora/RHEL system using the Anaconda graphical installer.

Troubleshooting — Displays the Troubleshooting menu (Figure 3-5).

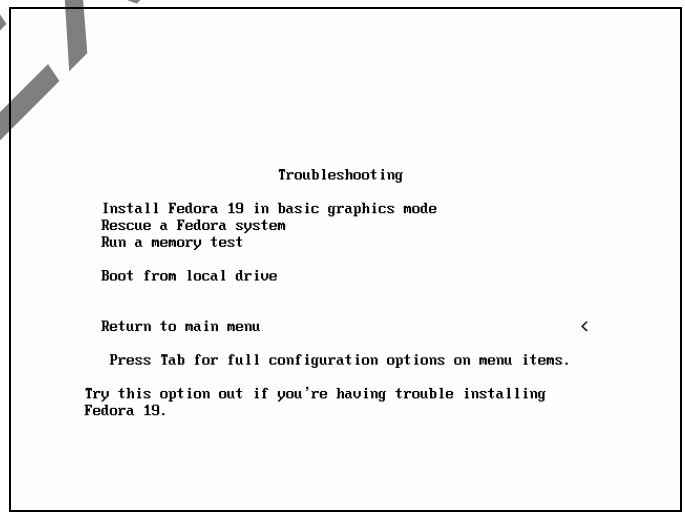

Figure 3-5 The Install Image Troubleshooting menu

The Troubleshooting menu has the following selections.

Install Fedora in basic graphics mode — Installs a graphical Fedora/RHEL system using the graphical installer. Anaconda does not attempt to determine the type of display attached to the system; it uses a basic video driver that works with most displays. Choose this selection if the installation fails before Anaconda displays the Welcome to Fedora screen (page 63).

Rescue a Fedora system — Brings up a minimal Fedora/RHEL system but does not install it. After detecting the system's disks and partitions, the system enters single-user/rescue mode and allows you to mount an existing Linux filesystem. For more information refer to "Rescue an Installed System" on page 456.

Run a memory test — Runs the memory test described on page 58.

Boot from local drive — Boots the system from the hard disk. This selection frequently has the same effect as booting the system without the installation medium in place (depending on how the BIOS [page 31] is set up).

Return to main menu — Displays the Fedora Install Image Boot menu.

THE ANACONDA INSTALLER

Anaconda, which is written in Python and C, identifies the hardware, loads drivers, probes for the devices it will use during installation, builds the filesystems, starts the X server, and installs the Fedora/RHEL operating system. Anaconda can run in graphical interactive mode (default), limited textual mode (see the **text** on page 72), or automated mode (see "Editing a Kickstart Script" on page 83).

Exactly which screens Anaconda displays depends on whether you are installing Fedora from a live session, an Install Image, or a Network Image; whether you are installing RHEL; which paths you pick through the installation process; and which parameters you specify on the boot command line. Unless you tell it not to, Anaconda probes the video card and monitor, and starts a native X server.

While it is running, Anaconda opens the virtual consoles (page 121) shown in Table 3-1. You can display a virtual console by pressing CONTROL-ALT-F*x*, where *x* is the virtual console number and F*x* is the function key that corresponds to the virtual console number.

Table 3-1 Virtual console assignments during installation

Virtual console	Install Image	Live Image
1	Installation dialog	GUI interactive installation
2	Shell	Login prompt (log in as **liveuser**)
3	Installation log	Installation log
4	Storage log	Login prompt (log in as **liveuser**)
5	Program log	Program log

Table 3-1 Virtual console assignments during installation (continued)

Virtual console	Install Image	Live Image
6	Login prompt	Login prompt (log in as **liveuser**)
7	GUI interactive installation	Nothing

At any time during the installation, you can switch to virtual console 2 (by pressing CONTROL-ALT-F2) and give commands to see what is going on. Do not give any commands that change any part of the installation process. To switch back to the graphical installation screen, press CONTROL-ALT-F1 (Live Image) or CONTROL-ALT-F7 (Install Image).

TESTING THE INSTALLATION MEDIUM

The first time you use an installation medium, select **Test this media & install Fedora** from the Boot menu and press RETURN to boot the system. Fedora displays a few lines before displaying a line that starts with **Checking**. This line shows the percent of the disk Anaconda has checked as it verifies the contents of the installation medium. If the installation medium is good, the system boots. If you use the same medium again, you do not have to test it: Select **Install Fedora**.

A DVD might fail the media test if the software that was used to burn the disk did not include padding. If a DVD fails the media test, try booting using the **nodma** parameter. See page 70 for information on adding parameters to the boot command line. If the DVD passes the media test when you boot the system using the **nodma** parameter, the DVD is good; reboot the system without this parameter before installing Fedora/RHEL. If you install Linux after having booted using this parameter, the kernel will be set up to always use this parameter. As a consequence, the installation and operation of the system can be slow.

USING ANACONDA

Welcome to Fedora

Anaconda displays the Welcome to Fedora screen after it obtains enough information to start the X Window System. Scroll and click to highlight the language you want to use for installation. This language is not necessarily the same language the installed system will display. Click **Continue**. Anaconda displays the Installation Summary screen.

THE INSTALLATION SUMMARY SCREEN

Anaconda takes a while to load the information the Installation Summary screen displays (Figure 3-6, next page). While it is loading, some items are grayed out. You must wait until Anaconda finishes loading this information and the items are no longer grayed out before you can select these items. ***You might need to scroll down to display the Storage item.***

Typically an orange bar that holds an exclamation point in a triangle and the words **Please complete items marked with this icon before continuing to the next step** appears at the bottom of the window. While this bar is displayed the button labeled **Begin Installation** is grayed out to indicate it will not work. The only items you must

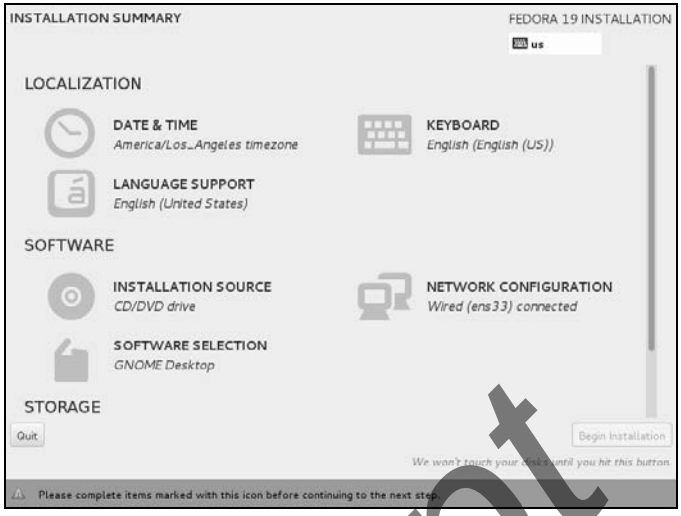

Figure 3-6 The Installation Summary screen (scroll to display the Storage item)

open and modify are those that display this icon. The items displayed when you are installing Fedora from a live session are a subset of those displayed when you install from an Install Image.

Click on an item to display a screen that allows you to work with that item. Click **Done** to return to the Installation Summary screen. The items that appear in the Installation Summary screen are divided into three groups: Localization, Software (not from a live installation), and Storage.

You do not need to open most items on the Installation Summary screen

tip Each item on the Installation Summary screen displays a note about what Anaconda has chosen for that item. You do not need to open an item if it is set properly. For example, if the Date & Time item shows **America/Los_Angeles timezone** and that is correct, you do not need to open this item. Frequently the only item you need to open is Installation Destination (page 67).

LOCALIZATION

Date & Time The Date & Time screen displays a world map that allows you to specify the local time zone. Click a city in the local time zone; Anaconda displays the name of the city in the spin box labeled **City**. Alternately, you can use the spin boxes labeled **Region** and **City** to specify a city.

By default, Anaconda sets up the new system to use NTP (Network Time Protocol) to keep the system clock accurate. If you do not want to use NTP, click the switch labeled Network Time to set it to **Off** and use the widgets at the bottom of the screen to set the system clock.

Keyboard The Keyboard Layout screen allows you to specify the type of keyboard that will be attached to the installed system. The box on the left side of this screen displays available keyboard layouts. The entry at the top of the list is the default layout to be used

both during installation and for the installed system. Use the buttons at the bottom of this box to add layouts and remove, move, and test the highlighted layout.

You can specify a keyboard for use during installation only, but you can specify only keyboards that are in the list on the Keyboard Layout screen. Use the small keyboard button to make this change. The button is located at the upper-right of all installation screens.

Language Support

Not available with a live installation. In the Language Support screen, put ticks in the check boxes adjacent to the languages you want to install.

SOFTWARE

Not available with a live installation.

Installation Source

Typically you install Fedora from the medium you booted from. If that is the case you do not need to open the Installation Source screen. This screen allows you to

- Specify as the installation source either the medium you booted from (**Auto-detected installation media**) or a location (URL) on the network
- Choose to verify the medium you booted from
- Install updates as Anaconda installs the system
- Specify additional repositories (page 533) to use during installation.

The top part of the Installation Source screen is labeled **Which installation source would you like to use?** and has two radio buttons. By default, **Auto-detected installation media** is selected and displays the name of the device that holds and the label on the medium. Click the button labeled **Verify** to test that the image on the installation medium is good.

Test the installation medium

caution It is possible for data to become corrupted while fetching an installation image; it is also possible for a transient error to occur while writing an image to a recordable medium. When you boot from an installation medium, the button labeled **Verify** on the Installation Source screen allows you to verify that the image on the installation medium does not contain any errors. Testing the installation medium takes a few minutes but can save you hours of aggravation if the installation fails due to a bad medium.

The second radio button in this section is labeled **On the network** and allows you to specify a Web, FTP, or NFS URL. When you select **Closest mirror** from the drop-down list, Anaconda finds the closest Fedora mirror and downloads packages from that site; after you click **Done**, Anaconda takes a moment to locate the closest mirror. If you reopen the Installation Source screen after selecting **Closest mirror**, Anaconda reverts to the installation medium and not installing updates.

By default Anaconda does not install updates while it installs Fedora/RHEL: You update the software after you install the system. Remove the tick from the check box labeled **Don't install the latest available software updates** to cause Anaconda to update software as it installs it.

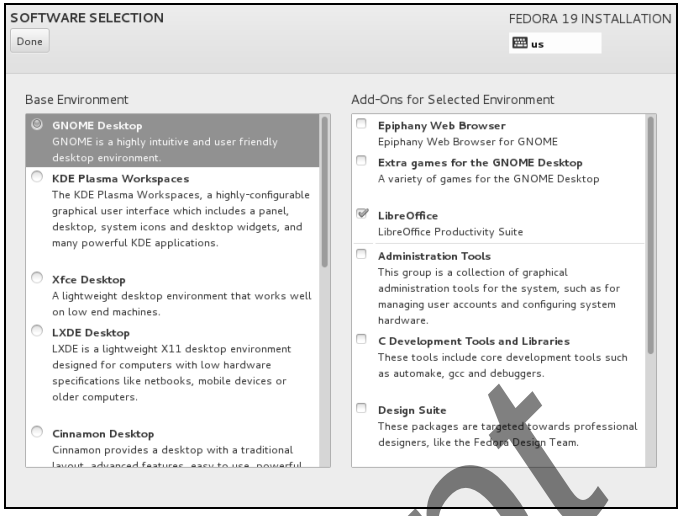

Figure 3-7 The Software Selection screen

The bottom portion of the screen allows you to specify additional repositories (page 542) that Anaconda can use during installation.

Network Configuration

If the local system is connected to a network that runs a DHCP server (page 491), Anaconda configures the network automatically and you do not have to open the Network Configuration screen. This screen

- Displays information about the network interface and connection Anaconda is using
- Allows you to turn off the network connection
- Allows you to (re)configure the network
- Allows you to specify a hostname for the installed system

When you click the button labeled **Configure**, Anaconda displays a window that allows you to modify the network connection.

Enter the hostname of the installed system in the text box labeled **Hostname** at the bottom of the screen.

Software Selection

The Software Selection screen (Figure 3-7) allows you to select which environmental package group and add-on package groups you want to install. An installation from a Network Image allows you to select from more package groups than does installation from an Install Image. By default, Anaconda installs the GNOME desktop and the LibreOffice productivity suite.

The left side of the Software Selection screen is labeled **Base Environment** and allows you to specify an environmental package group that specifies a desktop environment (e.g., GNOME, KDE, LXDE) or a system function (e.g., Web server, Infrastructure Server). You can make only one selection from this column. All the desktops install appropriate networking software and the Firefox Web browser. The Basic Desktop

selection allows you to specify more than one desktop on the right side of the screen. The Minimal Install selection installs as few software packages as possible for a textual system (no GUI). The right side of the screen allows you to select package groups that are typically used with the base environment you selected on the left. (Fedora; RHEL selections differ slightly).

Use a Minimal Install for older, slower hardware

tip The last entry in the Base Environment column is Minimal Install. Select this entry for older, slower hardware. This selection installs a system that has no GUI, just a command-line (textual) interface. If the system does not have enough memory to perform a graphical installation, perform a textual installation using the **text** boot parameter (page 72) and VNC (page 72) or Kickstart (pages 71 and 83).

You can select only package groups, not individual packages

tip To streamline the installation process, the new Anaconda allows you to specify which package groups you want to install but not which packages. You must add or remove individual packages after the installation is complete. Visit ohjeezlinux.wordpress.com/2013/02/07/anaconda-packaging-retrospective for background information. Use Kickstart (pages 71 and 83) if you require more granular control over package selection.

STORAGE

Installation Destination

You might need to scroll down to display the Installation Destination selection. At a minimum, Anaconda requires you to confirm where you want to install Fedora/RHEL, although this screen gives you many more choices.

By default, with a single empty disk, Anaconda installs Fedora/RHEL on the whole disk. If the disk is not empty and there is enough free space (page 36) on the disk, Anaconda installs Fedora/RHEL in the free space.

If this default is acceptable, open the Installation Destination screen; a tick will appear over the single disk (similar to Figure 3-9 on page 73). Click **Done**; Anaconda displays the Installation Options dialog. This dialog allows you to choose whether you want to review and/or modify the default disk setup or whether you want to continue without reviewing it. The bottom of the screen has a check box that allows you to encrypt the data on the disk (page 30). For the simplest installation, do not make any change; click **Continue**.

If there is not enough free space to install Fedora/RHEL, if the computer has more than one hard disk, if you want to reclaim disk space from an exiting partition, or if you want to customize the disk partitions, see "Advanced Disk Configuration" on page 72.

BEGIN COPYING FILES

After making the changes you want (and those that Anaconda requires) to the Installation Summary items, click the button labeled **Begin Installation**. While it is installing Fedora/RHEL, Anaconda places a Kickstart file (page 83) that corresponds to your selections in **/root/anaconda-ks.cfg**.

Installing Fedora/RHEL can take a while. The amount of time depends on the hardware you are installing the operating system on and the number of software packages you are installing.

Blank

4

Introduction to Fedora and Red Hat Enterprise Linux

In This Chapter

Objectives

After reading this chapter you should be able to:

- Log in on the Fedora desktop
- Understand **root** privileges
- Change the desktop background
- Use the Nautilus File Manager to work with files
- Correct mistakes on a command line
- Explain what you can do using a window titlebar
- Update and install software
- Find documentation using man, info, and yelp
- Open a terminal emulator and run programs from the command line
- Create and run a simple shell script

One way or another you are sitting in front of a computer that is running Fedora or RHEL (Red Hat Enterprise Linux). After describing the conventions this book uses and **root** (Superuser) privileges, this chapter takes you on a tour of the system to give you some ideas about what you can do with it. The tour does not go into depth about choices, options, menus, and so on; that is left for you to experiment with and to explore in greater detail throughout later chapters of this book.

The tour covers how to log in on the system, the GNOME 3 Standard and Classic desktops, working with the desktop, the Nautilus File Manager, the Settings window, getting help, and installing software. The final part of the chapter introduces some command-line utilities and describes how to work from the command line, concluding with a section on solving problems logging in and password security.

Be sure to read the warning about the dangers of misusing the powers of **root** (Superuser) in the next section. While heeding that warning, feel free to experiment with the system: Give commands, create files, click objects, choose items from menus, follow the examples in this book, and have fun.

CURBING YOUR POWER (SUPERUSER/root PRIVILEGES)

While you are logged in as the user named **root**, you are referred to as *Superuser* or *administrator;* you are working with **root** privileges and have extraordinary systemwide powers. Running the su or sudo utility can give you similar privileges. When working with **root** privileges, you can read from or write to almost any file on the system, execute programs that ordinary users cannot, and more. On a multiuser system you might not be able to run certain programs. Nevertheless, someone—the *system administrator*—knows the **root** password or is a member of the administrative group named **wheel**, and that person maintains the system. When you are running Linux on your own computer, you will assign a password to **root** when you install Linux. Refer to "Running Commands with **root** Privileges" on page 422 for more information.

Do not experiment while you are working with root privileges

caution Feel free to experiment when you are *not* working with **root** privileges. When you *are* working with **root** privileges, do only what you have to do and make sure you know exactly what you are doing. After you have completed the task at hand, revert to working as yourself. When working with **root** privileges, you can damage the system to such an extent that you will need to reinstall Linux to get it working again.

LOGGING IN ON THE SYSTEM

When you boot a default Fedora/RHEL system, gdm (GNOME display manager) displays a Login screen (Figure 4-1) on the system console. In the middle of the screen is a list of names of users who can log in on the system. At the upper-right are icons that allow you to restart or shut down the system, adjust the volume, and display the

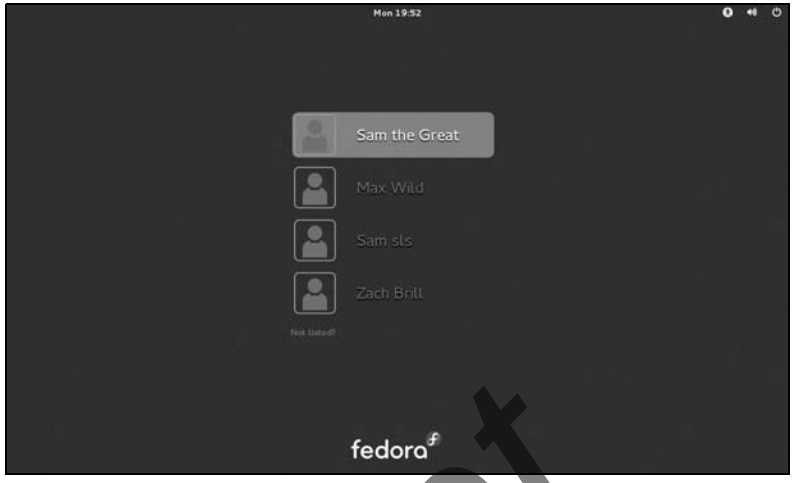

Figure 4-1 The GNOME Login screen

universal access menu that allows you to change accessibility preferences (e.g., make the text larger or display a high-contrast desktop).

To log in, click your name; gdm displays the Password screen. Enter your password and press RETURN. If gdm displays an error message, try clicking your name and entering your password again. Make sure the CAPS LOCK key is not on (gdm displays a message if it is) because the routine that verifies your entry is case sensitive. See page 135 if you need help with logging in and page 112 if you want to change your password. The system takes a moment to set things up and then displays a workspace.

THE GNOME 3 STANDARD AND CLASSIC DESKTOPS

Fedora 15 introduced GNOME 3 and the GNOME shell. GNOME 3 can run in standard mode (Standard desktop) or classic mode (Classic desktop). The GNOME 3 Standard desktop is radically different from the GNOME 2 desktop, following the trend toward simpler, more graphical desktops that have more icons and fewer menus. All versions of Fedora from 15 onward run the GNOME 3 Standard desktop by default. Figure 4-2 on page 93 and Figure 4-3 on page 94 show commonly used screens on a GNOME 3 Standard desktop. See the next section for more information.

By default, RHEL 7 runs the GNOME 3 Classic desktop, a group of GNOME Shell extensions that give GNOME 3 a GNOME 2 look and feel, including the Applications and Places menus. Figure 4-5 on page 97 shows the GNOME Classic desktop. For more information refer to "The GNOME Classic Desktop" on page 96.

This book uses the Linux textual interface (CLI) for most tasks, including setting up servers and programming in bash, SQL, and Python. Where this book describes a GUI utility, it explains how to open the utility/window by pressing ALT-F2 to display the Enter a Command window and typing the name of the utility. Alternately, you can press the SUPER (Windows) key to display the Search text box and enter the name

of utility there. These techniques work from both the GNOME 3 Standard desktop (Fedora) and Classic desktop (RHEL). For more information refer to "The Search Text Box and the Enter a Command Window" on page 98.

You can run the Classic desktop on Fedora

tip If you like, you can install and run the Classic desktop on Fedora; see "Installing the Classic Desktop on GNOME 3" on page 95.

optional

gnome-tweak-tool You can use gnome-tweak-tool (**gnome-tweak-tool** package) to change many aspects of how GNOME windows look and work. Visit wiki.gnome.org/GnomeTweakTool for more information.

THE GNOME 3 STANDARD DESKTOP (FEDORA)

This section briefly describes how to use the GNOME 3 Standard desktop that Fedora installs by default. For more information visit wiki.gnome.org/GnomeShell/CheatSheet. The extensions.gnome.org Web site lists extensions to GNOME 3. See "The GNOME Classic Desktop" on page 96 if you are running RHEL.

When you log in on a system running GNOME 3 the workspace is empty. On the Top panel (the bar at the top of the workspace) are objects: the word **Activities**, a clock, speaker and networking icons, and your name (or Live System User). You can click one of these objects to work with it. The Top panel remains constant as the information displayed below it changes.

THE ACTIVITIES OVERVIEW SCREEN

Click **Activities** at the top left of a workspace (or press the SUPER [Windows] key) to display the *Activities Overview* screen (Figure 4-2). This screen holds the Search text box (near the top), icons of favorite applications (on the left), and a section holding shrunken views of workspaces (on the right); each of these views contain live thumbnails of running applications. When only one workspace exists, you need to move the cursor to the right side of the screen to display this section. Shrunken views of the windows in the active workspace appear in the center of the screen. If the active workspace is empty, the center of the screen is blank.

Figure 4-2 shows an Activities Overview screen in which the active workspace holds windows that are running Firefox and Files (Nautilus; page 102). Because the shrunken Firefox window is selected (highlighted), it has an **X** in a circle at its upper-right corner. Click this **X** to close the window and terminate Firefox. Click the Firefox or Files window to close the Activities Overview screen and display the active workspace with the window you clicked in the foreground.

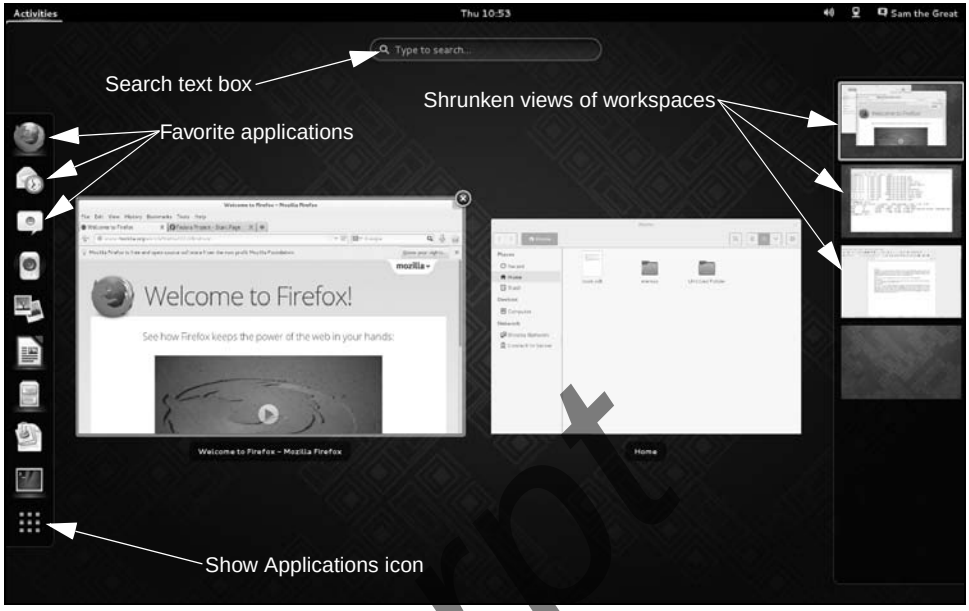

Figure 4-2 GNOME Activities Overview screen

Making a workspace active

Still looking at Figure 4-2, you can also click one of the shrunken workspaces along the right side of the screen to make that workspace active. Click the empty workspace at the bottom of the stack to open a new, empty workspace.

Running an application from the Favorites list

There are several ways to start an application. If the icon of the application you want to run is in the Favorites list, click that icon; GNOME closes the Activities Overview screen and opens in the active workspace the application whose icon you clicked.

Running an application from the Search text box (SUPER key)

You can also type the name of the (graphical) application (e.g., gnome-terminal or gnome-disks) or the name of the window the application displays (e.g., Terminal or Disks) in the Search text box, and press RETURN to open that application in the active workspace. If the Search text box is not visible, press the SUPER (Windows) key to display it.

Running an application from the Enter a Command window (ALT-F2 key)

Finally, you can press ALT-F2 to display the Enter a Command window, type the name of the application or utility you want to run, and press RETURN to run an application. You must run a textual application from a terminal emulator such as gnome-terminal (which you can start from an Enter a Command window). See page 98 for more information.

THE APPLICATIONS OVERVIEW SCREEN

At the bottom of the list of favorite application icons on the left of the Activities Overview screen is the Show Applications icon (it holds nine small squares; see Figure 4-2).

Figure 4-3 The GNOME Applications Overview screen

Click this icon to display the *Applications Overview* screen (Figure 4-3). This screen displays application icons. Clicking the button labeled **Frequent** at the bottom of the screen displays only icons of applications you use frequently. Clicking **All** displays a scrollable list of application icons. Not all applications, or utilities, have icons; use the Enter a Command window (page 98) to run applications that do not have icons.

Left-click an icon to close the Applications Overview screen and open the associated application in the active workspace. Right-click an icon to display a context menu that can include Add to Favorites (if the icon is not in the list of favorites), Remove from Favorites (if it is already in the list of favorites), New Window (closes the Applications Overview screen and opens the application in a new window in the active workspace even if it is already opened), and a line that identifies an already open instance of the application (closes the Applications Overview screen and opens the application in the workspace in which it is already running).

The Application Switcher: Changing the Input Focus

The window with the *input focus* is the one that receives keyboard characters and commands you type. In addition to using the Activities Overview screen, you can change which window has the input focus using the keyboard; this process is called *window cycling*.

When you press ALT-TAB when the desktop is displayed, GNOME displays in the center of the workspace the Application Switcher, a box that holds icons representing the programs running in the windows on the desktop. It also shifts the input focus to the window that was active just before the currently active window, making it easy to switch back and forth between two windows. When you hold ALT and press TAB multiple times, the focus moves from window to window. Holding ALT and SHIFT and repeatedly pressing TAB cycles in the other direction.

INSTALLING THE CLASSIC DESKTOP ON GNOME 3 (FEDORA)

The following instructions explain how to install and run the Classic desktop under Fedora. RHEL 7 runs the Classic desktop by default.

1. With GNOME running the GNOME 3 Standard desktop, click the word **Activities** at the upper left of the screen. Type **terminal** and press RETURN; GNOME displays a terminal emulator.
2. With the cursor over the terminal emulator, type the following command; terminate the command using a RETURN.

```
$ su -c 'yum -y install gnome-classic-session'
Password:
...
Complete!
```

   The su utility will prompt for the **root** password. Once you enter the **root** password and press RETURN, yum will display a lot of information as it installs the **gnome-classic-session** package. When yum displays **Complete!** and the shell displays its prompt (**$**), the package is installed. For more information on using yum to install packages see page 534.

3. *From a live session only* (or if **Log Out** does not appear in the menu described in the next step), give the following additional commands from the terminal emulator. Substitute your name for **sam** in the first two commands and supply the **root** password when you are prompted.

```
$ su -c 'useradd sam'
$ su -c 'passwd sam'
Changing password for user sam.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.

$ su -c 'systemctl restart gdm'
```

   After a few moments GNOME logs you in as Live System User again (or allows you to log in as yourself).

4. Log off by clicking your name or **Live System User** at the upper right of the screen, clicking **Log Out** from the menu GNOME displays, and then clicking the button labeled **Log Out** from the small window GNOME displays. (If **Log Out** does not appear in the menu, follow the previous step and then repeat this step.) After a few moments GNOME displays a Login screen (Figure 4-1, page 91).
5. Click your name (not Live System User) on the Login screen; GNOME displays the Password screen (Figure 4-4, next page). Just below the text box labeled **Password** is a small triangle followed by the word **Session**. Click the

Figure 4-4 The GNOME Password screen showing the Session menu

triangle to display the Session menu. Click **GNOME Classic**; GNOME moves the white dot to the entry you just clicked.

6. Type your password in the text box labeled **Password** and click **Sign In**. After a few moments GNOME displays a Classic desktop (Figure 4-5). If you followed step 3 to add a user, gnome-initial-setup will take you through setting up that user; see page 68.

Once you have selected GNOME Classic when you log in, it is your default desktop; you do not have to select it next time you log in.

THE GNOME CLASSIC DESKTOP (RHEL AND OPTIONALLY FEDORA)

This section briefly describes how to use the GNOME Classic desktop that is installed by default on RHEL. Refer to "The GNOME 3 Standard Desktop" on page 92 if you are running Fedora. If you want to install and run a Classic desktop under Fedora refer to "Installing the Classic Desktop on GNOME 3" on the previous page.

Panels and objects

When you log in, the GNOME Classic desktop displays a workspace that includes the Top and Bottom panels (bars) that help you get your work done easily and efficiently (Figure 4-5). Each of the panels can hold several icons and words called objects.

The Applications menu

You can run an application from the Applications menu that appears at the left end of the Top panel (Figure 4-5). Click **Applications** to display the Applications menu. When you move the mouse pointer over one of the selections in this menu and leave it there for a moment (this action is called *hovering*), the menu displays a submenu. Move the cursor to and click one of the items in the submenu to select and run it.

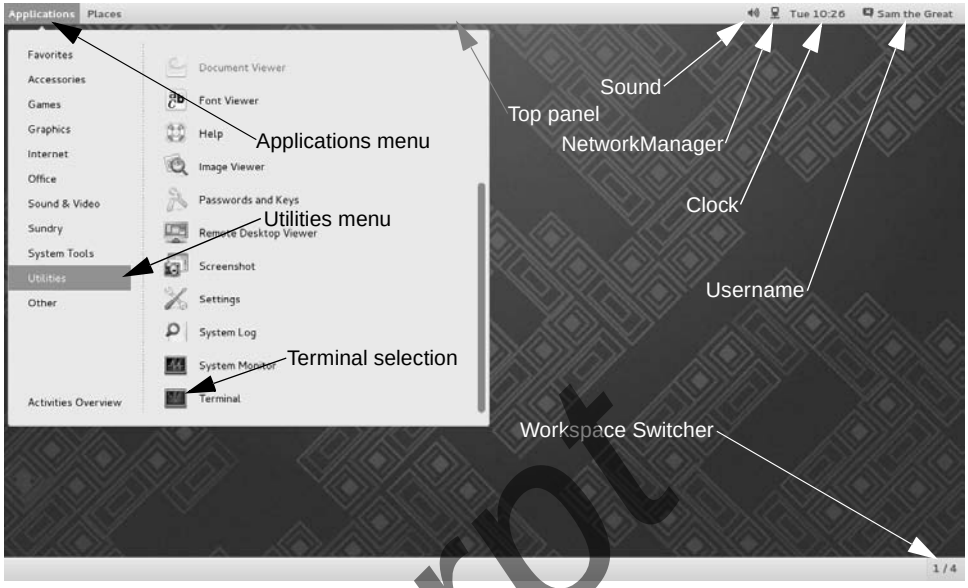

Figure 4-5 The GNOME Classic desktop showing the **Applications⇨Utilities** menu

The Places menu

The Places menu has no submenus; click an entry in this menu to select it. The Places menu on the GNOME Classic desktop is the same as Places in the Nautilus File Browser Sidebar; see page 103 for more information.

WORKING WITH THE DESKTOP

This section discusses several ways you can work with the desktop. It applies to both the GNOME 3 Standard and Classic desktops.

TERMINOLOGY

In addition to this section, see the Glossary, which defines data entry *widgets* (page 1281) such as the *combo box* (page 1243), *drop-down list* (page 1248), *list box* (page 1258), and *text box* (page 1277).

Workspace

A *workspace* is a screen that holds windows of one or more applications. The Activities Overview screen (page 92) and the Application Switcher (page 94) enable you to display any of the running applications and its workspace.

Active workspace

The *active workspace* is the workspace that displays the windows you can work with immediately. Only one workspace can be active at a time.

Desktop

The *desktop*, which is not displayed all at once, is the collection of all workspaces.

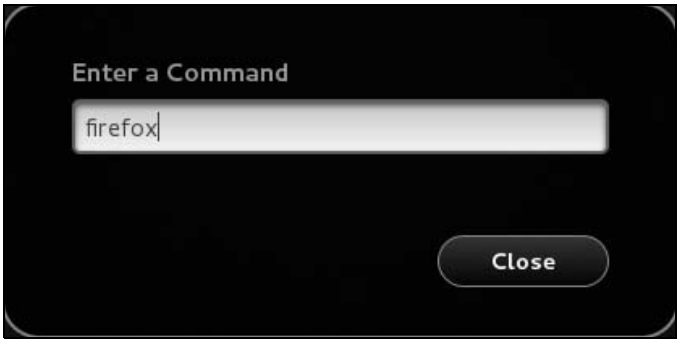

Figure 4-6 The Enter a Command window

Panel Panels are bars that can appear on the desktop and hold objects. The Top panel appears at the top of the screen and the Bottom panel, if it is present, appears at the bottom of the screen.

Object An *object* is a word, icon, or menu that you can select by clicking.

Click and right-click

tip This book uses the term **click** when you need to click the *left* mouse button. It uses the term **right-click** when you need to click the *right* mouse button. See page 110 for instructions on adapting the mouse for left-handed use.

The Search Text Box and the Enter a Command Window

You can start a graphical application (open a window) by pressing the SUPER (Windows) key to display the Search text box (Figure 4-2, page 93) or by pressing ALT-F2 to display the Enter a Command window (Figure 4-6) and typing the name of the application. Try typing **firefox** and pressing RETURN to start Firefox. The new window opens on the active workspace.

You must run a textual application (utility) from a terminal emulator

tip A textual application or utility requires a textual environment such as a terminal emulator to run. The Search text box and Enter a Command window do not provide a textual environment and so cannot run a textual application. However, you can run gnome-terminal, a graphical application that provides a textual environment using either of these methods. You can then run the textual application from the terminal emulator that gnome-terminal displays.

Application Menus

Many applications have an Application menu that allows you to set preferences and get help with the application. While the application is active (has the focus), click the object (the name of the window) for the application on the Top panel; GNOME

Figure 4-7 The Software (gpk-application) Application menu

opens the Application menu for the application. Figure 4-7 shows the Application menu for the Software (gpk-application) application. See "Updating, Installing, and Removing Software Packages" on page 116 for more information on this application.

CONTEXT MENUS

A *context menu* has choices that apply specifically to the window or object you click. The choices differ from window to window and from object to object. Some windows do not have context menus. Frequently a right-click displays a context menu.

Right-click to display a context menu

tip Right-click an object to display its context menu. Each object displays its own context menu, although similar objects have similar context menus. Most context menus have either a Preferences or Properties selection.

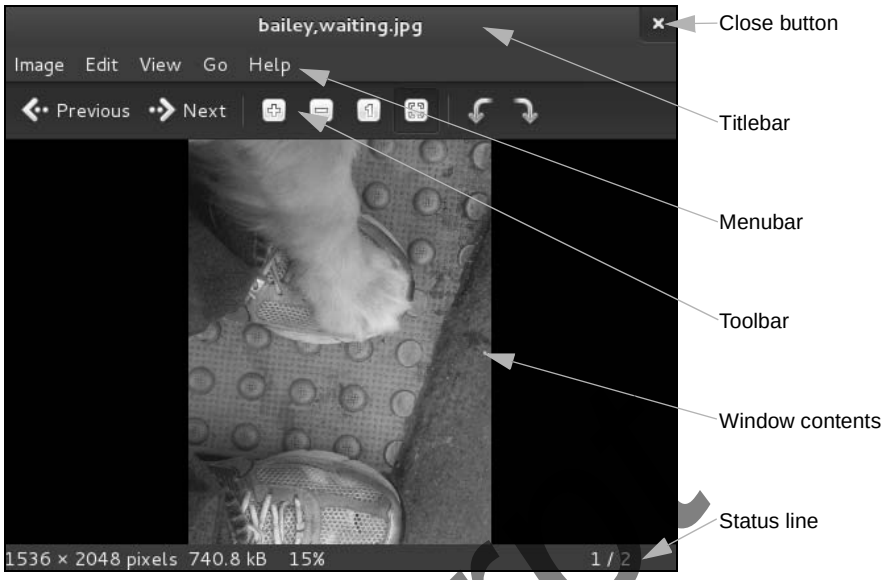

Figure 4-8 A typical window

WINDOWS

Window — In a workspace, a *window* is a region that runs, or is controlled by, a particular program.

Titlebar — A *titlebar* (Figure 4-8) appears at the top of most windows and controls the window it is attached to. You can double-click the titlebar to maximize and restore a window. Some windows can be restored only by pressing SUPER-DOWN ARROW (SUPER is the Windows key) or by dragging down from an empty spot on the Top panel. Clicking the close button (X) closes the window and usually terminates the program running in it. To reposition the window, left-click the titlebar and drag the window to the desired location.

Window Operations menu — The Window Operations menu contains operations you can perform on any window. Right-click the titlebar to display this menu. This menu allows you to minimize, maximize, resize, or close a window; move the window within the workspace; move a window to another workspace; keep the window on top of or below other windows; and cause the window to always be visible on the displayed workspace. It also shows the keyboard shortcuts for some of these actions. SUPER refers to the Windows key (e.g., **Super+Up** means hold the Windows key down and press the UP ARROW key).

Menubar — The *menubar* (Figure 4-8) holds icons you can use to work with the window contents.

Toolbar — A *toolbar* (Figure 4-8) usually appears near the top of a window and contains icons, text, applets, menus, and more. Many kinds of toolbars exist. The titlebar is not a toolbar.

Resizing a window

To resize a window, position the mouse pointer over an edge of the window; the pointer turns into an arrow pointing to a line. When the pointer is an arrow pointing to a line, you can click and drag the side of a window. When you position the mouse pointer over a corner of the window, you can resize both the height and the width of the window simultaneously. Some windows are not resizeable and some windows do not allow you to resize them from all four sides.

Moving a window

To move a window, click and drag the titlebar. Dragging a window to the top of the screen maximizes the window.

CUTTING AND PASTING OBJECTS USING THE CLIPBOARD

There are two similar ways to cut/copy and paste objects and text both within and between windows. In the first method, you use the clipboard, technically called the *copy buffer*, to copy or move objects or text. To do so, you explicitly copy an object or text to the buffer and then paste it somewhere else. Applications that follow the user interface guidelines use CONTROL-X to cut, CONTROL-C to copy, and CONTROL-V to paste. Application context menus frequently provide these options.

You might not be familiar with the second method to copy and paste text—using the *selection* or *primary* buffer, which always contains the text you most recently selected (highlighted). You cannot use this method to copy objects. Clicking the middle mouse button (click the scroll wheel on a mouse that has one) pastes the contents of the selection buffer at the location of the mouse pointer. If you are using a two-button mouse, click both buttons at the same time to simulate clicking the middle button.

With both of these techniques, start by highlighting an object or text to select it. You can drag a box around multiple objects to select them or drag the mouse pointer over text to select it. Double-click to select a word or triple-click to select a line or a paragraph.

Next, to use the clipboard, explicitly copy (CONTROL-C) or cut (CONTROL-X) the objects or text. If you want to use the selection buffer, skip this step.

To paste the selected objects or text, position the mouse pointer where you want to put it and then either press CONTROL-V (clipboard method) or press the middle mouse button (selection buffer method).

Use SHIFT-CONTROL-C and SHIFT-CONTROL-V within a terminal emulator

tip

The CONTROL-C, CONTROL-X, and CONTROL-V characters do not work in a terminal emulator window because the shell running in the window intercepts them before the terminal emulator can receive them. However, you can use SHIFT-CONTROL-C and SHIFT-CONTROL-V in place of CONTROL-C and CONTROL-V, respectively. There is no keyboard shortcut for CONTROL-X. You can also use the selection buffer in this environment or use copy/paste from the **Edit** selection on the menubar or from the context (right-click) menu.

When using the clipboard, you can give as many commands as you like between the CONTROL-C or CONTROL-X and CONTROL-V, as long as you do not press CONTROL-C or CONTROL-X again. When using the selection buffer, you can give other commands after selecting text and before pasting it, as long as you do not select (highlight) other text.

Logging Off

Log off by clicking your name or **Live System User** at the upper-right corner of the screen, click **Log Out** from the menu GNOME displays, and then click the button labeled **Log Out** from the small window GNOME displays. Select **Power Off** to shut down or restart the system, among other options. Alternately, you can give the command **gnome-session-quit** from an Enter a Command window (ALT-F2) or a terminal emulator.

Using the Nautilus File Manager

Nautilus (the Files program) is the simple, powerful GNOME file manager. You can use it to create, open, view, move, and copy files and folders as well as to execute applications and scripts.

Terms: folder and directory

Nautilus displays the File Browser window, which displays the contents of a folder. The terms *folder* and *directory* are synonymous; "folder" is frequently used in graphical contexts, whereas "directory" might be used in textual or command-line contexts. This book uses these terms interchangeably.

Term: File Browser

This book sometimes uses the terms *File Browser window* and *File Browser* when referring to the Nautilus File Browser window.

Opening Nautilus

Give the command **nautilus** from an Enter a Command window (ALT-F2) or a terminal emulator to open a Nautilus File Browser window that shows the files in your home directory (Figure 4-9). From a GNOME 3 desktop you can type **nautilus** or **files** in the Search text box in the Activities Overview screen (page 92). Alternately, you can click the filing cabinet icon in the Applications Overview screen (page 93). From a Classic desktop you can select Home from the Places menu.

The Nautilus File Browser Window

The right side of a File Browser window shows the View pane that holds file objects; the left side shows the Sidebar that holds the Places, Devices, and Network lists.

Application menu

With a File Browser window active, click **Files** on the Top panel to display the Files Application menu. This menu allows you to set preferences for, open a location bar in, and close the File Browser window.

View pane

The View pane displays file icons or a list of filenames. Select the view you prefer by clicking the dots or lines button near the right end of the toolbar.

Sidebar

The Sidebar allows you to work with Nautilus in different ways. Close or display the Sidebar by pressing F9. To change the horizontal size of the Sidebar, drag the handle (Figure 4-9) on its right side. See "The Sidebar" on the next page for more information.

Toolbar

The left end of the toolbar allows you to display different directories in the View pane. Select **Application menu⇨Enter Location** to open a location bar in which you

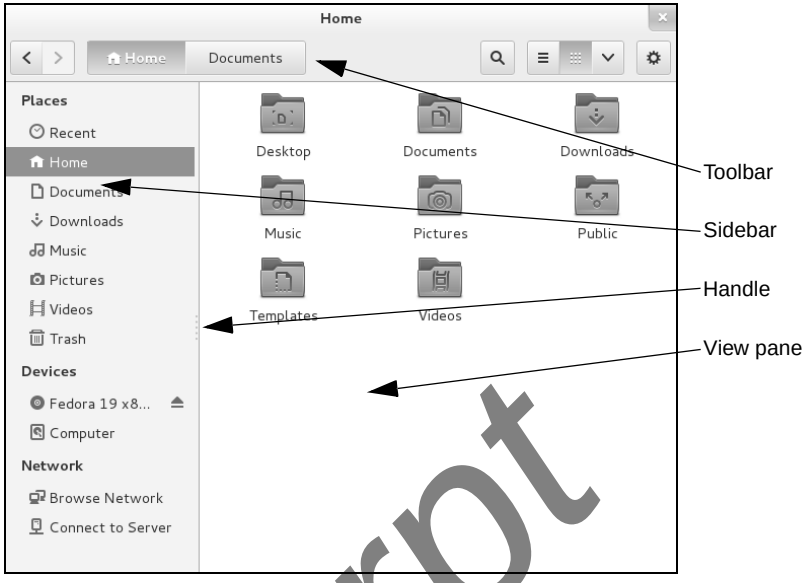

Figure 4-9 A Nautilus File Browser window

can enter a pathname. The right end of the toolbar holds icons that allow you to search for files, control the type of display Nautilus presents in the View pane, and display view options and location options.

THE SIDEBAR

The Places section of the Sidebar holds objects that you click to change what Nautilus displays in the View pane. The Home, Documents, and Downloads objects display your directories with corresponding names. The Computer object displays the local filesystem. The Browse Network object displays systems and files on the local network. The Connect to Server object opens a window that allows you to enter a URL to connect to an SSH, FTP, or other type of server.

Trash

Selecting **Move to Trash** from an object's context (right-click) menu moves the object to the **Trash** directory. Because files in the trash take up space on the hard disk (just as all files do), it is a good idea to remove them periodically. To display the **Trash** directory, double-click the Trash icon in the Sidebar.

Emptying the trash

Select **Empty Trash** from the Trash icon context (right-click) menu to permanently remove all files from the trash. (This selection is grayed out if there are no files in the trash.) Alternately, you can right-click an object in the Trash File Browser window and select **Delete** to remove only that object (file), or you can select **Restore** to move the file back to its original location. You can drag and drop files to and from the trash just as you can with any folder.

Deleting files

You can delete files without first sending them to the trash. Add **Delete** to the context menu by selecting Preferences from the Files Application menu, then selecting the Behavior tab, and putting a tick in the check box labeled **Include a Delete command that bypasses Trash**.

Blank

7

The Linux Utilities

In This Chapter

Objectives

After reading this chapter you should be able to:

- Use basic utilities to list files and display text files
- Copy, move, and remove files
- Display the beginning or end of a file
- Search, sort, print, categorize, and compare text files
- Compress, decompress, and archive files
- Count the number of letters, words, and lines in a file
- Locate utilities on the system
- Change the modification time of a file
- Display information about users and the system
- Record a session in a file
- Edit text files

When Linus Torvalds introduced Linux and for a long time thereafter, Linux did not have a GUI (graphical user interface): It ran on character-based terminals only, using a CLI (command-line interface), also referred to as a textual interface. All the tools ran from a command line. Today the Linux GUI is important, but many people—especially system administrators—run many command-line utilities. Command-line utilities are often faster, more powerful, or more complete than their GUI counterparts. Sometimes there is no GUI counterpart to a textual utility; some people just prefer the hands-on feeling of the command line. This chapter describes a number of frequently used utilities and concludes with tutorials on the vim and nano text editors. See "Working from the Command Line" on page 119 for an introduction to the command line.

When you work with a CLI, you are working with a shell (Chapters 5, 9, and 27). When you are working on the command line it is important to quote special characters as explained on page 142.

More utilities are covered throughout the book

tip This chapter introduces a few important utilities that are good to know as you start using Linux. More utilities are covered throughout the book. See the inside of the front and back covers for a complete list.

BASIC UTILITIES

One of the advantages of Linux is that it comes with thousands of utilities that perform myriad functions. You will use utilities whenever you work with Linux, whether you use them directly by name from the command line or indirectly from a graphical menu or icon. The following sections discuss some of the most basic and important utilities that are available from a CLI. Some of the more important utilities are also available from a GUI; others are available only from a GUI.

Run these utilities from a command line

tip This chapter describes command-line, or textual, utilities. You can experiment with these utilities from a terminal, a terminal emulator within a GUI (page 120), or a virtual console (page 121).

LPI cat: JOINS AND DISPLAYS FILES

The cat utility joins and copies files to standard output. The name of the cat utility is derived from *catenate*, which means to join together, one after the other. You can also use cat to display virtual system files in the **/proc** directory hierarchy as explained on page 512.

ARGUMENTS

The arguments are the pathnames of one or more files that cat processes. You can use cat to display the contents of one or more text files on the screen. More precisely, cat copies the files to standard output, which by default is attached to the screen.

```
$ cat practice
This is a small file that I created
using:  a text editor!

End.
```

If you do not specify an argument or if you specify a hyphen (–) in place of a filename, cat reads from standard input. The following example shows cat working without an argument; the **<** symbol causes the shell to redirect standard input to cat to come from **practice**. The shell passes no arguments to cat.

```
$ cat < practice
This is a small file that I created
using:  a text editor!

End.
```

You can use cat to create short, simple files. See "The Keyboard and Screen as Standard Input and Standard Output" on page 152 and "Redirection" on page 153 for more examples of redirecting the standard input and standard output of cat.

OPTIONS

Number The **–n** (**––number**) option numbers all lines as they are written to standard output while the **–b** (**––number-nonblank**) numbers only non-blank lines.

```
$ cat -n practice
     1  This is a small file that I created
     2  using:  a text editor!
     3
     4  End.
```

Tabs The **–T** (**––show-tabs**) option displays TABs as **^I**.

```
$ cat -T practice
This is a small file that I created
using:^Ia text editor!

End.
```

You can combine options like this

```
$ cat -bT practice
```

Or, using long options, like this

```
$ cat --number-nonblank --show-tabs practice
```

Nonprinting The **–v** (**––show-nonprinting**) option displays CONTROL characters using the caret notation (e.g., **^M** means CONTROL-M) and displays characters that have the high bit set (META characters) using the **M-** notation (e.g., **M-M** means META-M). This option does not convert TABs and LINEFEEDs. Use **–T** (**––show-tabs**) if you want to display TABs as **^I**. LINEFEEDs cannot be displayed as anything but themselves; otherwise, the line could be too long.

RELATED UTILITIES

tac The tac (cat spelled backwards) utility works the same way as cat works except it reverses the order of the lines in each file.

rev The rev (reverse) utility works the same way as cat works except it reverses the order of the characters in each line.

LPI date: DISPLAYS THE SYSTEM TIME AND DATE

Without any arguments, the date utility displays the date and time known to the local system. If you set up a locale database (page 368), date uses that database to substitute in its output terms appropriate to the locale for your account. Use timedatectl (page 613) to set the system clock.

```
$ date
Mon Aug 19 14:39:55 PDT 2013
```

The hardware clock and the system clock

tip The hardware clock is the time kept in the BIOS (page 31). This clock is battery powered and keeps time even when the computer is off. The system clock runs only while the system is running and is what Linux uses when it needs to know the time. On some systems the system clock is adjusted periodically by NTP (Network Time Protocol; page 315) so it stays accurate.

ARGUMENTS

The following example formats and specifies the contents of the output of date. See the date man page for a complete list of format sequences.

```
$ date +"%A %B %d"
Monday August 19
```

OPTIONS

Date The **–d** ***datestring*** (**––date**=***datestring***) option displays the date specified by ***datestring***, not the date known to the system. According to the date man page, "the ***datestring*** is a mostly free-format date string" such as **2pm next thursday**. See **DATE STRING** in the date man page for details about the syntax of ***datestring***. This option does not change the system clock.

UTC The **–u** (**––utc** *or* **––universal**) option displays or sets the time and date using UTC (Universal Coordinated Time; page 1279). UTC is also called GMT (Greenwich Mean Time).

```
$ date -u
Mon Aug 19 21:40:19 UTC 2013
```

RELATED UTILITIES

timedtectl The timedtectl utility, which is part of systemd (page 438), displays more information about the system clock. You can also use it to set the system clock. See page 613.

cal The cal utility displays a calendar for the month and year you specify. Without any arguments it displays a calendar for the current month.

LE+ echo: DISPLAYS ARGUMENTS

The echo utility copies its arguments, followed by a NEWLINE, to standard output. The Bourne Again Shell has an echo builtin that works similarly to the echo utility.

The echo builtin/utility is a good tool for learning about the shell and other Linux utilities. Some examples on page 166 use echo to illustrate how special characters, such as the asterisk, work. Throughout Chapters 5, 9, and 27, echo helps explain how shell variables work and how you can send messages from shell scripts to the screen. You can even use echo to create a short file by redirecting its output:

```
$ echo "This is a short file." > short
$ cat short
This is a short file.
```

ARGUMENTS

The arguments can include quoted strings, ambiguous file references, and shell variables. The shell recognizes and expands unquoted special characters in arguments.

The following example shows echo copying its arguments to standard output. The second command includes an unquoted asterisk (*; page 166) that the shell expands into a list of files in the working directory before passing that list as arguments to echo; echo copies this list to standard output.

```
$ echo This is a sentence.
This is a sentence.
$ echo star: *
star: memo memo.0714 practice
$ ls
memo  memo.0714  practice
```

OPTION

NEWLINE The **–n** option suppresses the NEWLINE that normally terminates the output of echo. This option is useful in shell scripts when you want to prompt a user and have the response appear on the same line as the prompt. See page 1041 for an example of a shell script that uses this feature.

LPI hostname: DISPLAYS THE SYSTEM NAME

The hostname utility displays the name of the system you are working on. Use this utility if you are not sure that you are logged in on the correct machine. See also **/etc/hostname** on page 507.

```
$ hostname
guava
```

OPTION

The following option works with hostname.

FQDN The **–f** (**––fqdn**) option displays the *FQDN* (page 1250) of the local system.

RELATED UTILITY

hostnamectl The hostnamectl utility, which is part of systemd (page 438), displays more information about the local system. You can also use it to change the hostname.

LE less Is more: DISPLAY A TEXT FILE ONE SCREEN AT A TIME

You can use the less or more utilities, called *pagers,* to view a long text file one screen at a time. Each of these utilities pauses after displaying a screen of text. You can then press the SPACE bar to display the next screen of text.

Although less and more are very similar, they have subtle differences. The less utility, for example, allows you to move backward while viewing a file in some situations where more does not. Whereas more must read an entire file before displaying anything, less does not have to and so starts more quickly than more when displaying a large file. At the end of the file less displays an **END** message and waits for you to press **q** before returning you to the shell. In contrast, more returns you directly to the shell. While using both utilities you can press **h** to display a Help screen that lists commands you can use while paging through a file. Give the command **less /etc/services** to experiment with paging through a file.

ARGUMENTS

Both less and more take the names of files you want to view as arguments. If you do not specify an argument or if you specify a hyphen (**–**; less only) in place of a filename, less and more read from standard input.

OPTIONS

The following options work with less.

Clear screen The **–c** (**––clear-screen**) option paints each new screen from the top down; by default less scrolls the display.

EOF The **–e** (**––quit-at-eof**) option causes less to exit when it reaches the second EOF (when you press SPACE while less is displaying **END** at the end of a file) so you do not need to type **q** to quit.

Truncate The **–S** (**––chop-long-lines**) option truncates lines wider than the screen. By default less wraps long lines.

No initialization The **–X** (**––no-init**) option prevents less from sending terminal initialization and deinitialization strings to the terminal. By default, less clears the screen when it finishes; this option causes the last page of what you were viewing to remain on the screen.

optional You can set the **LESS** environment variable (page 1032) in your **~/.bash_profile** file (page 330) to set options for less each time you call it and when it is called from another program such as man. For example, the following line in the **.bash_profile** file in your home directory will cause less to always run with the **–X** option.

```
export LESS='-X'
```

RELATED UTILITY

most The most utility (part of the **most** package; see page 534 for installation instructions) is newer and more capable than less and more. See the most man page for details.

LPI ls: DISPLAYS INFORMATION ABOUT FILES

The ls utility displays information about one or more files. It lists the information alphabetically by filename unless you use an option to change the order.

When you do not provide an argument, ls displays the names of the visible files (those with filenames that do not begin with a period; page 180) in the working directory.

ARGUMENTS

The arguments are one or more pathnames of any ordinary, directory, or device files. The shell expands ambiguous file references (page 165) in the arguments.

When you specify an ordinary file as an argument, ls displays information about that one file. When the argument is a directory, ls displays the contents of the directory. It displays the name of the directory only when needed to avoid ambiguity, such as when the listing includes more than one directory.

```
$ ls memos.zach/130715.1
memos.zach/130715.1

$ ls memos.zach
130712.1  130714.2  130714.3  130715.1

$ ls memos.*
memos.max:
130619.1  130621.2  130622.1

memos.sam:
130811.2  130811.3  130812.1

memos.zach:
130712.1  130714.2  130714.3  130715.1
```

LE OPTIONS

Options determine the type of information ls displays, and the manner and order in which it displays the information. When you do not use an option, ls displays a short list that contains just the names of files, in alphabetical order. See page 145 for examples of how options affect ls output.

All The **–a** (**––all**) option includes hidden filenames (those filenames that begin with a period; page 180) in the listing. Without this option ls does not list information about files with hidden filenames unless you specify the name of a hidden file as an argument. The * ambiguous file reference does not match a leading period in a filename, so you must use this option or explicitly specify a filename (ambiguous or not) that begins with a period to display information about files with hidden filenames.

LE Directory The **–d** (**––directory**) option displays directories without displaying their contents.

```
$ ls -ld /
dr-xr-xr-x. 27 root root 4096 07-16 18:37 /
```

This option is useful when you want to find out, for example, the name of the user who owns a directory in a large directory such as **/etc**. Instead of scrolling through

the output of **ls –l /etc** looking for a directory, you can give the command **ls –ld /etc/*filename*** (e.g., **ls –ld /etc/cron.d**).

Human readable The **–h** (**––human-readable**) option, when specified with the **–l** option, displays sizes in K (kilobyte), M (megabyte), and G (gigabyte) blocks, as appropriate. This option works with the **–l** and **–s** options only. It displays powers of 1,024. Use **––si** to display powers of 1,000.

```
$ ls -lh /usr/bin
...
-rwxr-xr-x. 1 root root     105K 05-08 00:45 tac
-rwxr-xr-x. 1 root root      68K 05-08 00:45 tail
-rwxr-xr-x. 1 root root      24K 08-01 01:08 tailf
-rwxr-xr-x. 1 root root      29K 05-02 08:55 talk
-rwxr-xr-x. 1 root root     340K 06-04 08:23 tar
...
```

Long The **–l** (lowercase "el"; **––format=long**) option lists more information about each file. See page 191 for an explanation of this output. If standard output for a directory listing is sent to the screen, this option displays the number of blocks used by all files in the listing on a line before the listing.

LE Recursive The **–R** (**––recursive**) option recursively lists directory hierarchies.

Reverse The **–r** (**––reverse**) option displays the list of filenames in reverse sorted order.

LE Size The **–s** (**––size**) option displays the number of 1,024-byte blocks allocated to the file. The size precedes the filename. With the **–l** option, this option displays the size in column 1 and shifts other items one column to the right. If standard output for a directory listing is sent to the screen, this option displays the number of blocks used by all files in the listing on a line before the listing. You can include the **–h** option to make the file sizes easier to read. The following example recursively lists the **memos** directory hierarchy in reverse size order.

```
$ ls -lRrs memos
memos:
4 drwxrwxr-x. 2 sam sam 4096 04-29 14:11 memos.zach
4 drwxrwxr-x. 2 sam sam 4096 04-29 14:32 memos.sam
4 drwxrwxr-x. 2 sam sam 4096 04-30 14:15 memos.max

memos/memos.zach:
4 -rw-rw-r--. 1 sam sam 4064 04-29 14:11 130715.1
4 -rw-rw-r--. 1 sam sam 3933 04-29 14:11 130714.3
4 -rw-rw-r--. 1 sam sam 3317 04-29 14:11 130714.2
...
```


LE+ rm: REMOVES A FILE (DELETES A LINK)

The rm utility removes hard and/or symbolic links to one or more files. Frequently this action results in the file being deleted. See page 203 for information on links.

ARGUMENTS

The arguments are the pathnames of the files whose links rm will remove. Removing the only (last) hard link to a file deletes the file (page 208). Removing a symbolic link deletes the symbolic link only.

Be careful when you use rm with ambiguous file references

caution Because this utility enables you to remove a large number of files with a single command, use rm cautiously, especially when you are working with ambiguous file references. *Never use* rm *with ambiguous file references while you are working with* **root** *privileges.* If you have any doubts about the effect of an rm command with an ambiguous file reference, first use echo with the same file reference and evaluate the list of files the reference generates. Alternately, you can use the rm **–i** (**––interactive**) option.

OPTIONS

Force The **–f** (**––force**) option, without asking for your consent, removes files for which you do not have write access permission. This option suppresses informative messages if a file does not exist.

Interactive The **–i** (**––interactive**) option prompts you before removing each file. If you use **–r** (**––recursive**) with this option, rm also prompts you before examining each directory.

```
$ rm -ri memos
rm: descend into directory 'memos'? y
rm: descend into directory 'memos/memos.max'? y
rm: remove regular file 'memos/memos.max/130621.2'? y
rm: remove regular file 'memos/memos.max/130619.1'? n
...
```

You can create an alias (page 392) for **rm –i** and put it in a startup file (page 180) so rm always runs in interactive mode. The **–i** option is set up by default for the **root** user under Fedora/RHEL.

Recursive The **–r** (**––recursive**) option deletes the contents of the specified directory, including all its subdirectories, and the directory itself. Use this option with caution; do not use it with wildcards (***** and **?**).

Verbose The **–v** (**––verbose**) option displays the name of each file as it is removed.

RELATED UTILITIES

shred The shred utility overwrites a file to hide its contents and make it very difficult to recover the data that was stored in it. The **–u** (**––remove**) option causes shred to delete the file after overwriting it. The following command overwrites the file named **myfile** three times and then deletes it.

```
$ shred -u myfile
```

testdisk The testdisk utility (**testdisk** package) scans, repairs, and sometimes can recover disk partitions. In some cases can undelete files. See the testdisk man page for details.

WORKING WITH FILES

This section describes utilities that copy, move, print, search through, display, sort, compare, and identify files.

Filename completion

tip After you enter one or more letters of a filename (as an argument) on a command line, press TAB, and the shell will complete as much of the filename as it can. When only one filename starts with the characters you entered, the shell completes the filename and places a SPACE after it. You can keep typing or you can press RETURN to execute the command at this point. When the characters you entered do not uniquely identify a filename, the shell completes what it can and waits for more input. If pressing TAB does not change the display, press TAB again to display a list of possible completions. For more information refer to "Pathname Completion" on page 389.

LE+ cp: COPIES FILES

The cp utility copies one or more files. Use scp (page 695) or rsync (page 696) to copy files from one system to another (or to make local copies). See page 237 for a description of the related mv (move) utility.

ARGUMENTS

With two arguments that are pathnames, neither of which specifies a directory, cp copies the file named by the first argument to the file named by the second argument.

```
$ cp memo memo.cp
```

With two or more arguments that are pathnames, the last of which is an existing directory, cp copies the files named by all but the last argument to the directory named by the last argument.

```
$ cp memo1 memo2 memo4 memo.dir
```

cp can destroy a file

caution If the destination file exists *before* you give a cp command, cp overwrites it. Because cp overwrites (and destroys the contents of) an existing destination file without warning, you must take care not to cause cp to overwrite a file that you need. The cp **–i** (interactive) option prompts you before it overwrites a file.

The following example assumes the file named **orange.2** exists before you give the cp command. The user answers **y** to overwrite the file.

```
$ cp -i orange orange.2
cp: overwrite 'orange.2'? y
```

OPTIONS

Archive The **–a** (**––archive**) option attempts to preserve the owner, group, permissions, access date, and modification date of source file(s) while copying a directory recursively.

Backup The **–b** (**––backup**) option makes a backup copy of a file that would be removed or overwritten by cp. The backup copy has the same name as the destination file with a

tilde (~) appended to it. When you use both **–b** and **–f**, cp makes a backup copy when you try to copy a file over itself. For more backup options, search for **Backup options** in the **coreutils** info page.

Force The **–f** (**––force**) option causes cp to try to remove the destination file (when it exists but cannot be opened for writing) before copying the source file. This option is useful when the user copying a file does not have write permission to an existing file but does have write permission to the directory containing the file. Use this option with **–b** to back up a destination file before removing or overwriting it.

Interactive The **–i** (**––interactive**) option prompts you whenever cp would overwrite a file. If you respond with a string that starts with **y** or **Y**, cp copies the file. If you enter anything else, cp does not copy the file.

```
$ cp -i * ../memos.helen
cp: overwrite '../memos.helen/130619.1'? y
cp: overwrite '../memos.helen/130622.1'? n
```

Preserve The **–p** (**––preserve**[=***attr***]) option creates a destination file with the same owner, group, permissions, access date, modification date, and ACLs as the source file. The **–p** option does not take an argument.

Without ***attr***, **––preserve** works as described above. The ***attr*** is a comma-separated list that can include **mode** (permissions), **ownership** (owner and group), **timestamps** (access and modification dates), **links** (hard links), and **all** (all attributes).

Recursive The **–R** *or* **–r** (**––recursive**) option recursively copies directory hierarchies including ordinary files. The last argument must be the name of a directory.

Verbose The **–v** (**––verbose**) option displays the name of each file as cp copies it.

```
$ cp -rv memos memos.bak
'memos' -> 'memos.bak'
'memos/memos.max' -> 'memos.bak/memos.max'
'memos/memos.max/130619.1' -> 'memos.bak/memos.max/130619.1'
'memos/memos.max/130622.1' -> 'memos.bak/memos.max/130622.1'
'memos/memos.sam' -> 'memos.bak/memos.sam'
'memos/memos.sam/130811.3' -> 'memos.bak/memos.sam/130811.3'
'memos/memos.sam/130811.2' -> 'memos.bak/memos.sam/130811.2'
...
```


LE+ cut: SELECTS CHARACTERS OR FIELDS FROM INPUT LINES

The cut utility selects characters or fields from lines of input and writes them to standard output. Character and field numbering start with 1. Although limited in functionality, cut is easy to learn and use and is a good choice when columns and fields can be specified without using pattern matching.

ARGUMENTS

Arguments to cut are pathnames of ordinary files. If you do not specify an argument or if you specify a hyphen (–) in place of a pathname, cut reads from standard input.

OPTIONS

Characters

The **–c** ***clist*** (**––characters=*clist***) option selects the characters given by the column numbers in ***clist***. The value of ***clist*** is one or more comma-separated column numbers or column ranges. A range is specified by two column numbers separated by a hyphen. A range of ***–n*** means columns **1** through ***n***; ***n–*** means columns ***n*** through the end of the line.

The following example displays the permissions of the files in the working directory. The **–c2-10** option selects characters 2 through 10 from each input line.

```
$ ls -l | cut -c2-10
otal 2944
rwxr-xr-x
rw-rw-r--
rw-rw-r--
rw-rw-r--
rw-rw-r--
```

Input delimiter

The **–d** ***dchar*** (**––delimiter=*dchar***) option specifies ***dchar*** as the input field delimiter. This option also specifies ***dchar*** as the output field delimiter unless you use the **––output-delimiter** option. The default delimiter is a TAB character. Quote ***dchar*** as necessary to protect it from shell expansion.

Fields

The **–f** ***flist*** (**––fields=*flist***) option selects the fields specified by ***flist***. The value of ***flist*** is one or more comma-separated field numbers or field ranges. A range is specified by two field numbers separated by a hyphen. A range of ***–n*** means fields **1** through ***n***; ***n–*** means fields ***n*** through the last field. The field delimiter is a TAB character unless you use the **–d** option to change it.

The following example displays a list of full names as stored in the fifth field (**–f5**) of the **/etc/passwd** file. The **–d** option specifies that the colon character is the field delimiter.

```
$ cut -d: -f5 /etc/passwd
Sam the Great
Sam the Great
Zach Brill
Max Wild
...
```

The next command outputs the size and name of each file in the working directory. The **–f** option selects the fifth and ninth fields from the input lines. The **–d** option tells cut to use SPACEs, not TABs, as delimiters. The tr utility (page 258) with the **–s** option changes sequences of two or more SPACE characters to a single SPACE; otherwise, cut counts the extra SPACE characters as separate fields.

```
$ ls -l | tr -s '  ' ' ' | cut -f5,9 -d' '

259 countout
9453 headers
1474828 memo
1474828 memos_save
```

```
7134 tmp1
4770 tmp2
13580 typescript
```

Output delimiter

The **––output-delimiter=***ochar* option specifies ***ochar*** as the output field delimiter. By default, the output field delimiter is the same as the input field delimiter (the TAB character unless you change it using **–d**). Quote ***ochar*** as necessary to protect it from shell expansion.

diff: DISPLAYS THE DIFFERENCES BETWEEN TWO TEXT FILES

The diff (difference) utility displays line-by-line differences between two text files. By default diff displays the differences as instructions that you can use to edit one of the files to make it the same as the other.

The sdiff utility is similar to diff but its output might be easier to read; its output is the same as that of the diff **–y** (**––side-by-side**) option. Use the diff3 utility to compare three files and cmp to compare nontext (binary) files.

ARGUMENTS

The diff utility commonly takes the pathnames of two ordinary files it is to compare as arguments.

When you call diff without any options, it produces a series of lines containing Add (**a**), Delete (**d**), and Change (**c**) instructions. Each of these lines is followed by the lines from the file you need to add to, delete from, or change, respectively, to make the files the same. A less than symbol (**<**) precedes lines from ***file1***. A greater than symbol (**>**) precedes lines from ***file2***. The diff output appears in the format shown in Table 7-1. A pair of line numbers separated by a comma represents a range of lines; a single line number represents a single line.

Table 7-1 diff output

Instruction	Meaning (to change file1 to file2)
line1 a *line2*,*line3* > lines from file2	Append *line2* through *line3* from **file2** after *line1* in **file1**
line1,*line2* d *line3* < lines from file1	Delete *line1* through *line2* from **file1**
line1,*line2* c *line3*,*line4* < lines from file1 --- > lines from file 2	Change *line1* through *line2* in **file1** to *line3* through *line4* from **file2**

The diff utility assumes you will convert the file named by the first argument to the file named by the second argument. The line numbers to the left of each of the **a**, **c**, or **d** instructions always pertain to the first file; the line numbers to the right of the instructions apply to the second file. To display instructions to convert the files in the opposite order, reverse the arguments.

Blank

10

System Administration: Core Concepts

In This Chapter

Objectives

After reading this chapter you should be able to:

- Explain the need for and the responsibility of a privileged user (**root**)
- Gain **root** privileges using su and sudo
- Describe the startup sequence using systemd
- Manage which services start at boot time
- List four characteristics of a well-maintained system
- Start and stop services on a running system
- Boot into single-user mode for system maintenance
- Shut down a running system
- Secure a system by applying updates, monitoring logs, and controlling access to files using SELinux, setuid permission, and PAM
- Use system administration tools to monitor and maintain the system
- List common steps for installing, configuring, and securing a server
- Configure a system using a static IP address or using DHCP

The job of a system administrator is to keep one or more systems in a useful and convenient state for users. On a Linux system, the administrator and user might both be you, with you and the computer being separated by only a few feet. Alternately, the system administrator might be halfway around the world, supporting a network of systems, with you being one of thousands of users. Or a system administrator can be one person who works part-time taking care of a system and perhaps is also a user of the system. In some cases several administrators might work together full-time to keep many systems running.

A well-maintained system:

- Runs quickly enough so users do not get frustrated waiting for the system to respond or complete a task
- Has enough storage to accommodate the reasonable needs of users
- Provides a working environment appropriate to each user's abilities and requirements
- Is secure from malicious and accidental acts altering its performance or compromising the security of the data it holds and exchanges with other systems
- Is backed up regularly, with recently backed-up files readily available to users
- Has recent copies of the software that users need to get their jobs done
- Is easier to administer than a poorly maintained system

In addition, a system administrator should be available to help users with all types of system-related problems—from logging in to obtaining and installing software updates to tracking down and fixing obscure network issues.

Part III of this book breaks system administration into nine chapters.

- Chapter 9 covers bash (Bourne Again Shell) to a depth that you can use it interactively to administer a system and begin to understand complex administration shell scripts.
- Chapter 10 covers the core concepts of system administration, including working with **root** (Superuser) privileges, system operation, configuration tools and other useful utilities, general information about setting up and securing a server (including a section on DHCP), and PAM.
- Chapter 11 covers files, directories, and filesystems from an administrator's point of view.
- Chapter 12 covers installing software on the system, including the use of yum, BitTorrent, and curl.
- Chapter 13 discusses how to set up local and remote printers that use the CUPS printing system.

- Chapter 14 explains how to rebuild the Linux kernel and work with GRUB, the Linux boot loader.
- Chapter 15 covers additional system administrator tasks and tools, including setting up users and groups, backing up files, scheduling tasks, printing system reports, and general problem solving.
- Chapter 16 goes into detail about how to set up a LAN, including setting up and configuring network hardware and configuring software.
- Chapter 17 describes how to set up virtual machines locally (gnome-boxes, KVM/QEMU, and VMware) and in the cloud (AWS).

Because Linux is readily configurable and runs on a wide variety of platforms, this chapter cannot discuss every system configuration or every action you might have to take as a system administrator. Instead, this chapter seeks to familiarize you with the concepts you need to understand and the tools you will use to maintain a Linux system. Where it is not possible to go into depth about a subject, the chapter provides references to other sources.

This chapter assumes you are familiar with the following terms:

block device (page 1239)
daemon (page 1245)
device (page 1246)
device filename (page 1247)
disk partition (page 1247)
environment (page 1249)
filesystem (page 1250)
fork (page 1250)
kernel (page 1257)
login shell (page 1259)
mount (page 1261)
process (page 1267)
root filesystem (page 1271)
runlevel (page 1271)
signal (page 1273)
spawn (page 1274)
system console (page 1276)
X server (page 1281)

If there is a problem, check the log files

tip If something on the system is not working as expected, check the log files in **/var/log**. This directory holds many files and subdirectories. If you cannot connect to a server, also check the log files on the server.

If something does not work, see if the problem is caused by SELinux

tip If a server or other system software does not work properly, especially if it displays a permissions-related error message, the problem might lie with SELinux. To see whether SELinux is the cause of the problem, put SELinux in permissive mode and run the software again. If the problem goes away, you need to modify the SELinux policy. Remember to turn SELinux back on. See the tip on page 473 and refer to "Setting the Targeted Policy using system-config-selinux" on page 475.

If you cannot access a remote system, check the firewall

tip If a server does not appear to work or you cannot access a remote system, make sure the firewall is not the problem. On a non-production system, use systemctl to turn the firewall off (page 900) and see if the problem goes away. Then turn the firewall back on and open only the necessary port using firewall-cmd (page 906).

LE RUNNING COMMANDS WITH root PRIVILEGES

Some commands can damage the filesystem or crash the operating system. Other commands can invade users' privacy or make the system less secure. To keep a Linux system up and running as well as secure, most systems are configured not to permit ordinary users to execute some commands and access certain files. Linux provides several ways for a trusted user to execute these commands and access these files. A user running with these privileges is sometimes referred to as an *administrator, privileged user,* or *Superuser*.

Administrator

security During installation (page 68) and whenever you add or modify a user (pages 112 and 598), you have the opportunity to specify that user as an *administrator*. Two characteristics give an administrator special privileges. First, an administrator is a member of the **wheel** group (page 429) and as such can use sudo to authenticate using her password; she does not need to know the **root** password. Second, **polkit** (www.freedesktop.org/wiki/Software/polkit) is set up so that an administrator can do some kinds of administrative work on the desktop (e.g., updating software and adding a printer) without needing to enter a password.

LE THE SPECIAL POWERS OF A PRIVILEGED USER

root privileges A user running with **root** privileges has the following powers—and more.

- Some commands, such as those that add new users, partition hard drives, and change system configuration, can be executed only by a user working with **root** privileges. Such a user can configure tools, such as sudo, to give specific users permission to perform tasks that are normally reserved for a user running with **root** privileges.
- Read, write, and execute file access and directory access permissions do not affect a user with **root** privileges. A user with **root** privileges can read from, write to, and execute all files, as well as examine and work in all directories.

  Exceptions to these privileges exist. For example, SELinux mandatory access can be configured to limit **root** access to a file. Also, a user working with **root** privileges cannot make sense of an encrypted file without possessing the key.
- Some restrictions and safeguards that are built into some commands do not apply to a user with **root** privileges. For example, a user with **root** privileges can change any user's password without knowing the old password.

Console security

security Linux is not secure from a person who has physical access to the computer. Additional security measures, such as setting boot loader and BIOS passwords, can help secure the computer. However, if someone has physical access to the hardware, as system console users typically do, it is very difficult to secure a system from that user.

prompt When you are running with **root** privileges in a command-line environment, by convention the shell displays a special prompt to remind you of your status. By default, this prompt is (or ends with) a hashmark (#).

Least privilege

caution When you are working on any computer system, but especially when you are working as the system administrator (working with **root** privileges), perform any task while using the least privilege possible. When you can perform a task logged in as an ordinary user, do so. When you must run a command with **root** privileges, do as much as you can as an ordinary user, log in or use su or sudo so you have **root** privileges, complete the part of the task that has to be done with **root** privileges, and revert to being an ordinary user as soon as you can. Because you are more likely to make a mistake when you are rushing, this concept becomes even more important when you have less time to apply it.

LE GAINING root PRIVILEGES

Classically a user gained **root** privileges by logging in as the user named **root** or by giving an su (substitute user) command and providing the **root** password. More recently the use of sudo has taken over this classic technique of gaining **root** privileges. With sudo, the user logs in as herself, gives an **sudo** command, and provides her own password (not the **root** password) to gain **root** privileges. "Advantages of sudo" on page 428 discusses some of the advantages of sudo.

Graphical environment When an ordinary user executes a privileged command in a graphical environment, the system prompts for the **root** password or the user's password, depending on how the system is set up.

Some distributions lock the **root** account by not assigning a **root** password. On these systems, you cannot gain **root** privileges using a technique that requires you to supply the **root** password unless you unlock the **root** account as explained on page 438. Fedora/RHEL assigns a **root** password when the system is installed, so you can use these techniques from the start.

There is a root account, but no root password

tip As installed, some systems (not Fedora/RHEL) lock the **root** account by not providing a **root** password. This setup prevents anyone from logging in to the **root** account (except when you bring the system up in single-user mode [page 450]). There is, however, a **root** account (a user with the username **root**—look at the first line in **/etc/passwd**). This account/user owns files (give the command **ls –l /usr/bin**) and runs processes (give the command **ps –ef** and look at the left column of the output). The **root** account is critical to the functioning of a Linux system.

When properly set up, the sudo utility enables you to run a command as though it had been run by a user logged in as **root**. This book uses the phrase **working** (or **run**) **with root privileges** to emphasize that, although you might not be logged in as **root**, when you use su or sudo you have the powers of the **root** user.

The following list describes some of the ways you can gain or grant **root** privileges. Some of these techniques depend on you supplying the password for the **root**

account. Again, if the **root** account is locked, you cannot use these techniques unless you unlock the **root** account (set up a **root** password), as explained on page 438. Other techniques depend on the **sudoers** file being set up to allow you to gain **root** privileges (page 433). If this file is not set up in this manner, you cannot use these techniques.

- When you bring the system up in single-user/rescue mode (page 450), you log in as the user named **root**.
- You can give an su (substitute user) command while you are logged in as yourself. When you then provide the **root** password, you will be running with **root** privileges. See page 425.
- The sudo utility allows specified users to run selected commands with **root** privileges while they are logged in as themselves. You can set up sudo to allow certain users to perform specific tasks that require **root** privileges without granting these users systemwide **root** privileges. See page 428.
- Once the system is up and running in multiuser mode (page 452), you can log in as **root**. When you then supply the **root** password, you will be running with **root** privileges.
- Some programs ask for a password (either your password or the **root** password, depending on the command and the configuration of the system) when they start or when you ask them to perform certain tasks. When you provide a password, the program runs with **root** privileges. You stop running as a privileged user when you quit using the program. This setup keeps you from remaining logged in with **root** privileges when you do not need or intend to be.

LE+ Setuid file

- Any user can create a *setuid* (set user ID) file. Setuid programs run on behalf of the owner of the file and have all the access privileges the owner has. While you are working with **root** privileges, you can change the permissions of a file owned by **root** to setuid. When an ordinary user executes a file that is owned by **root** and has setuid permissions, the program has *effective **root** privileges*. In other words, the program can do anything a program running with **root** privileges can do that the program normally does. The user's privileges do not change. When the program finishes running, all user privileges are as they were before the program started. Setuid programs owned by **root** are both extremely powerful and extremely dangerous to system security, which is why a system contains very few of them. Examples of setuid programs that are owned by **root** include passwd, at, and crontab. For more information refer to "Setuid and Setgid Permissions" on page 196 and "Real UID Versus Effective UID" (next).

Logging in

The **/etc/securetty** file controls which terminals (ttys) a user can log in on as **root**.

Using the **/etc/security/access.conf** file, PAM controls the who, when, and how of logging in. Initially this file contains only comments. See page 476, the comments in the file, and the **access.conf** man page for details.

root-owned setuid programs are extremely dangerous

security Because **root**-owned setuid programs allow someone who does not know the **root** password and cannot use sudo to gain **root** privileges, they are tempting targets for a malicious user. Also, programming errors that make normal programs crash can become **root** exploits in setuid programs. A system should have as few of these programs as possible. You can disable setuid programs at the filesystem level by mounting a filesystem with the **nosuid** option (page 522). You can also use SELinux (page 472) to disable setuid programs. See page 458 for a find command that lists all setuid files on the local system. Future releases of Fedora/RHEL will remove most setuid files; see fedoraproject.org/wiki/Features/RemoveSETUID.

Do not allow root access over the Internet

security Prohibiting a user from logging in as **root** over a network is the default policy of Fedora/RHEL. The **/etc/securetty** file must contain the names of all devices you want a user to be able to log in on as **root**.

You can, however, log in as **root** over a network using ssh (page 685). As shipped by Fedora/RHEL, PAM configuration for ssh does not call the modules that use the **securetty** or **access.conf** configuration files. Also, in **/etc/ssh/sshd_config**, Fedora/RHEL sets **PermitRootLogin** to **yes** (it is set by default) to permit **root** to log in using ssh (page 705). For a more secure system, change **PermitRootLogin** to **no**.

LE REAL UID VERSUS EFFECTIVE UID

UID and username A UID (user ID) is the number the system associates with a username. UID 0 is typically associated with the username **root**. To speed things up, the kernel keeps track of a user by UID, not username. Most utilities display the associated username in place of a UID.

The kernel associates two UIDs with each process: a *real UID* and an *effective UID*. The third column of the **/etc/passwd** file (or NIS/LDAP) specifies your real UID. When you log in, your real UID is associated with the process running the login shell. Because you have done nothing to change it, the effective UID associated with the process running the login shell is the same as the real UID associated with that process.

process privilege The kernel determines which privileges a process has based on that process' effective UID. For example, when a process asks to open a file or execute a program, the kernel looks at the effective UID of the process to determine whether it is allowed to do so. When a user runs a setuid program (page 196), the program runs with the UID of the owner of the program, not the user running the program.

terminology: root privileges When this book uses the phrase **run with root privileges** or **gain root privileges,** it means run with an effective UID of 0 (**root**).

LE+ USING su TO GAIN root PRIVILEGES

When you install Fedora/RHEL, you assign a password to the **root** account. Thus you can use su to gain **root** privileges without any further setup.

The su (substitute user) utility can spawn a shell or execute a program with the identity and privileges (effective UID) of a specified user, including **root**:

- Follow **su** on the command line with the name of a user; if you are working with **root** privileges or if you know the user's password, the newly spawned shell will take on the identity (effective UID) of that user.
- When you give an su command without an argument, su defaults to spawning a shell with **root** privileges (you have to supply the **root** password). That shell runs with an effective UID of 0 (**root**).

SPAWNING A root SHELL

When you give an su command to work with **root** privileges, su spawns a new shell, which displays the **#** prompt. That shell runs with an effective UID of 0 (**root**). You can return to your normal status (and your former shell and prompt) by terminating this shell: Press CONTROL-D or give an **exit** command.

LE who, whoami The who utility, when called with the arguments **am i**, displays the real UID (translated to a username) of the process that called it. The whoami utility displays the effective UID (translated to a username) of the process that called it. As the following example shows, the su utility (the same is true for sudo) changes the effective UID of the process it spawns but leaves the real UID unchanged.

```
$ who am i
sam      pts/2        2013-06-12 13:43 (192.168.206.1)
$ whoami
sam
$ su
Password:
# who am i
sam      pts/2        2013-06-12 13:43 (192.168.206.1)
# whoami
root
```

LE id Giving an su command without any arguments changes your effective user and group IDs but makes minimal changes to the environment. For example, **PATH** has the same value as it did before you gave the su command. The id utility displays the effective UID and GID of the process that called it as well as the groups the process is associated with. In the following example, the information that starts with **context** pertains to SELinux:

```
$ pwd
/home/sam
$ echo $PATH
/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/home/sam/.local/bin:/home/sam/bin
$ id
uid=1000(sam) gid=1400(pubs) groups=1400(pubs),10(wheel) context=unconfined_u: ...
$ su
Password:
# pwd
/home/sam
# echo $PATH
/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/home/sam/.local/bin:/home/sam/bin
```

```
# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u: ...
# exit
exit
$
```

When you give the command **su –** (you can use **–l** or **––login** in place of the hyphen), su provides a **root** login shell: It is as though you logged in as **root**. Not only do the shell's effective user and group IDs match those of **root**, but the environment is the same as when you log in as **root**. The login shell executes the appropriate startup files (page 329) before displaying a prompt, and the working directory is set to what it would be if you had logged in as **root** (**/root**). **PATH** is also set as though you had logged in as **root**. However, as who shows, the real UID of the new process is not changed from that of the parent process.

```
$ su -
Password:
# pwd
/root
# echo $PATH
/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin
# who am i
sam      pts/2        2013-06-12 13:43 (192.168.206.1)
```

EXECUTING A SINGLE COMMAND

You can use su with the **–c** option to run a command with **root** privileges, returning to the original shell when the command finishes executing. In the following example, Sam tries to display the **/etc/shadow** file while working as himself, a nonprivileged user. The cat utility displays an error message. When he uses su to run cat to display the file, su prompts him for a password, he responds with the **root** password, and the command succeeds. The quotation marks are necessary because **su –c** takes the command it is to execute as a single argument.

```
$ cat /etc/shadow
cat: /etc/shadow: Permission denied

$ su -c 'cat /etc/shadow'
Password:
root:$6$il96HvSfmvep.m2F$2RI1LZ ... fYc3wYZFQ/:15861:0:99999:7:::
bin:*:15839:0:99999:7:::
daemon:*:15839:0:99999:7:::
adm:*:15839:0:99999:7:::
...
```

The next example first shows that Sam is not permitted to kill (page 465) a process. With the use of **su –c** and the **root** password, however, Sam is working with **root** privileges and is permitted to kill the process.

```
$ kill -15 4982
-bash: kill: (4982) - Operation not permitted
$ su -c "kill -15 4982"
Password:
$
```

The final example combines the **–** and **–c** options to show how to run a single command with **root** privileges in the **root** environment:

```
$ su -c pwd
Password:
/home/sam
$ su - -c pwd
Password:
/root
```

Root privileges, PATH, and security

security The fewer directories you keep in **PATH** when you are working with **root** privileges, the less likely you will be to execute an untrusted program while working with **root** privileges. If possible, keep only the default directories, along with **/usr/bin** and **/usr/sbin**, in **root**'s **PATH**. *Never include the working directory in **PATH** (as . or : : anywhere in **PATH**, or : as the last element of **PATH**).* For more information refer to "PATH: Where the Shell Looks for Programs" on page 359.

LE+ USING sudo TO GAIN root PRIVILEGES

If sudo (www.sudo.ws) is not set up so you can use it, and a **root** password exists and you know what it is, see "Administrator and the wheel group" on the next page for instructions on setting up sudo so you can use it to gain **root** privileges.

ADVANTAGES OF sudo

Using sudo rather than the **root** account for system administration offers many advantages.

- When you run sudo, it requests *your* password—not the **root** password—so you have to remember only one password.
- The sudo utility logs all commands it executes. This log can be useful for retracing your steps for system auditing and if you make a mistake.
- The sudo utility logs the username of a user who issues an sudo command. On systems with more than one administrator, this log tells you which users have issued sudo commands. Without sudo, you would not know which user issued a command while working with **root** privileges.
- The sudo utility allows implementation of a finer-grained security policy than does the use of su and the **root** account. Using sudo, you can enable specific users to execute specific commands—something you cannot do with the classic **root** account setup.
- Using sudo makes it harder for a malicious user to gain access to a system. When there is an unlocked **root** account, a malicious user knows the username of the account she wants to crack before she starts. When the **root** account is locked, the user has to determine the username *and* the password to break into a system.

Blank

rootpw Causes sudo to accept only the **root** password in response to its prompt. Because sudo issues the same prompt whether it is asking for your password or the **root** password, turning this flag on might confuse users. Default is **off**, causing sudo to accept the password of the user running sudo.

shell_noargs Causes sudo, when called without any arguments, to spawn a **root** shell without changing the environment. Default is **off**. This option causes the same behavior as the sudo **–s** option.

timestamp_timeout=*mins*
The ***mins*** is the number of minutes that the sudo timestamp (page 429) is valid. Set ***mins*** to **–1** to cause the timestamp to be valid forever; set to 0 (zero) to cause sudo to always prompt for a password. Default is 5.

tty_tickets Causes sudo to authenticate users on a per-tty basis, not a per-user basis. Default is **on**.

umask=*val* The ***val*** is the umask (page 469) that sudo uses to run the command that the user specifies. Set ***val*** to **0777** to preserve the user's umask value. Default is **0022**.

LOCKING THE root ACCOUNT (REMOVING THE root PASSWORD)

If you decide you want to lock the **root** account, give the command **su –c 'passwd –l root'**. This command renders the encrypted password in **/etc/shadow** invalid by prepending two exclamation points (**!!**) to it. You can unlock the account again by removing the exclamation points or by giving the command shown in the following example.

Unlocking the **root** account

If you decide you want to unlock the **root** account after locking it, give the following command. This command assumes you can use sudo to gain **root** privileges and unlocks the **root** account by assigning a password to it:

```
$ sudo passwd root
[sudo] password for sam:
Changing password for user root.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```


LPI THE systemd init DAEMON

The **init** daemon is the system and service manager for Linux. As explained on page 374, it is the first true process Linux starts when it boots and, as such, has a PID of 1 and is the ancestor of all processes. The **init** daemon has been around since the early days of UNIX, and many people have worked to improve it. The first Linux **init** daemon was based on the UNIX System V **init** daemon and is referred to as SysVinit (System V **init** daemon).

Because SysVinit does not deal well with modern hardware, including hotplug (page 516) devices, USB hard and flash drives, and network-mounted filesystems,

Fedora 15 replaced it with the systemd **init** daemon, which is described in this section. Several other replacements for SysVinit are also available. Ubuntu uses Upstart (upstart.ubuntu.com), Solaris uses SMF (Service Management Facility), and MacOS uses **launchd**.

The name *systemd* comprises *system,* which systemd manages, followed by *d*. Under UNIX/Linux, daemon names frequently end in **d**: systemd is the system daemon. At boot time, systemd renames itself **init**, so you will not see a process named systemd. However, **init** is simply a link to systemd:

```
$ ls -l /sbin/init
lrwxrwxrwx. 1 root root 22 05-28 12:17 /sbin/init -> ../lib/systemd/systemd
```

The name is also a play on words with *System D,* a reference to the French débrouillard (to untangle) or démerder. System D is a manner of responding to challenges that requires fast thinking, adapting, and improvising.

The systemd **init** daemon is a drop-in replacement for SysVinit; most of the administration tools that worked with SysVinit and Upstart work with systemd. Although systemd is relatively new, most of the user interfaces pertinent to administrators will remain stable (www.freedesktop.org/wiki/Software/systemd/InterfaceStabilityPromise).

MORE INFORMATION

Local Use apropos to list man pages that pertain to systemd (**apropos systemd**). Some of the most interesting of these are **systemd**, systemctl, **systemd.unit**, and **systemd.special**.

Web systemd home page: www.freedesktop.org/wiki/Software/systemd
Fedora systemd home page: fedoraproject.org/wiki/Systemd
SysVinit to systemd conversion notes:
fedoraproject.org/wiki/SysVinit_to_Systemd_Cheatsheet
Blog about systemd by its creator, Lennart Poettering:
0pointer.de/blog/projects/systemd.html
systemd compatibility with SysV:
www.freedesktop.org/wiki/Software/systemd/Incompatibilities
systemd stability promise:
www.freedesktop.org/wiki/Software/systemd/InterfaceStabilityPromise
cgroups (control groups): www.kernel.org/doc/Documentation/cgroups/cgroups.txt

SERVICE UNITS AND TARGET UNITS

The systemd **init** daemon is based on the concept of units, each of which has a name and type. Typically information about a unit is stored in a file that has the same name as the unit (e.g., **dbus.service**). The types of units are service, socket, device, mount, automount, target, snapshot, timer, swap, and path. This section discusses service and target units, which are critical to controlling daemons and runlevel under systemd.

Service unit — A service unit refers to a daemon (service) that systemd controls, including those controlled natively by systemd and those controlled by systemd via SysVinit init scripts. The systemd **init** daemon controls almost all services natively.

Target unit — A target unit groups other units. Of concern in this section are targets that control the system runlevel. By default, Fedora activates **graphical.target**, which brings the system to a runlevel that equates to what was formerly called runlevel 5 (multiuser graphical mode). Activating **multi-user.target** brings the system to what was formerly called runlevel 3 (multiuser textual mode). Table 10-1 on page 449 lists all predefined target units (runlevels).

Terminology: server, service, daemon — A *daemon*, such as **atd** or **cupsd**, provides a *service* that runs on a *server*. The daemon itself is also sometimes referred to as a *server*. These three terms can be used interchangeably.

RUNLEVELS

The systemd **init** daemon does not support runlevels the way SysVinit did. It supports *target units*, which parallel runlevels but are different. To ease the transition, this book continues to use the term *runlevel* to refer to target units. One difference between SysVinit runlevels and systemd target units is that the former can be changed only when the system changes runlevels whereas the latter can be activated by any of a large group of triggers. Another difference is that a systemd-based system can activate more than one target unit at a time, allowing the system to be in more than one "runlevel" at a time. For example, **graphical.target** pulls in **multi-user.target** so they are both active at the same time.

systemd runlevels differ from SysVinit runlevels

tip — For consistency and clarity during the transition from SysVinit to systemd, this book refers to systemd *target units* as *runlevels*. Target units are not true runlevels, but they perform a function similar to the function performed by SysVinit runlevels.

A systemd-based system can activate more than one target unit at a time, allowing the system to be in more than one "runlevel" at a time. A system running SysVinit can be in only one runlevel at a time.

WANTS AND REQUIRES

Under systemd, the terms *wants* and *requires* specify units that are to be activated when the unit that wants or requires the other unit is activated. A unit that *requires* another unit will not start if the other unit is not available. Wants is similar to requires, except a unit that *wants* another unit will not fail if the wanted unit is not available.

DISPLAYING PROPERTIES

The following systemctl **show** command displays the Requires properties of the **graphical.target** unit. It shows that **graphical.target** requires **multi-user.target**:

```
$ systemctl show --property "Requires" graphical.target
Requires=multi-user.target
```

This relationship causes systemd not to start **graphical.target** if **multi-user.target** is not available. It means **graphical.target** requires units that **multi-user.target** requires and wants units that **multi-user.target** wants. Because of this relationship, **multi-user.target** (runlevel) is active at the same time **graphical.target** (runlevel) is active.

You can also use the systemctl **show** command to display the Wants properties of a target:

```
$ systemctl show --property "Wants" multi-user.target
Wants=vmtoolsd.service abrt-xorg.service rpcbind.service ...
lines 1-1/1 (END) q
```

The list is long. Although systemctl passes the output through a pager, it runs off the right edge of the screen. When the command displays (**END**), press **q** to return to the shell prompt. Sending the output through the fmt text formatter with a line-length specification of 10 displays the list one service per line:

```
$ systemctl show --property "Wants" multi-user.target | fmt -10
Wants=vmtoolsd.service
abrt-xorg.service
rpcbind.service
remote-fs.target
sm-client.service
sssd.service
...
```

To see whether a target wants a specific service, send the output of the previous command through grep:

```
$ systemctl show --property "Wants" multi-user.target | fmt -10 | grep atd
Wants=atd.service
```

The output shows that **multi-user.target** wants the **atd.service** service. Because **graphical.target** requires **multi-user.target** and **multi-user.target** wants **atd.service**, systemd will start **atd.service** when the system enters the runlevel defined by **graphical.target**.

/etc/systemd/system HIERARCHY: CONTROLS SERVICES AND THE PERSISTENT RUNLEVEL

The services wanted by a runlevel target appear in directories named ***.wants** under the **/etc/systemd/system** directory:

```
$ ls -ld /etc/systemd/system/*.wants
...
drwxr-xr-x. 2 root root 4096 06-04 19:29 /etc/systemd/system/graphical.target.wants
drwxr-xr-x. 2 root root 4096 06-05 19:44 /etc/systemd/system/multi-user.target.wants
...
```

The following command lists the runlevel targets that want **atd.service**:

```
$ ls /etc/systemd/system/*.wants/atd.service
/etc/systemd/system/multi-user.target.wants/atd.service
```

As explained in the previous section, you can also display this information using the systemctl **show** command.

The directory hierarchy with its root at **/etc/systemd/system** controls the persistent runlevel of the system. That is, this directory hierarchy specifies which daemons will be started when the system boots or otherwise changes runlevel or when the daemon is activated for another reason.

All service unit files are kept in the **/lib/systemd/system** directory. All plain files in the **/etc/systemd/system** hierarchy are links to files in the **/usr/lib/systemd/system** or**/lib/systemd/system** hierarchy. For example, **atd.service** shown in the preceding example is a link to the appropriate service unit file in **/usr/lib/systemd/system**:

```
$ ls -l /etc/systemd/system/multi-user.target.wants/atd.service
lrwxrwxrwx. 1 root root 35 07-02 12:56 /etc/systemd/system/multi-user.target.wants/atd.service ->
/usr/lib/systemd/system/atd.service
```

The service unit file provides systemd with information it needs to start the service and specifies the system runlevel target that wants the service:

```
$ cat /lib/systemd/system/atd.service
[Unit]
Description=Job spooling tools
After=syslog.target systemd-user-sessions.service

[Service]
EnvironmentFile=/etc/sysconfig/atd
ExecStart=/usr/sbin/atd -f $OPTS

[Install]
WantedBy=multi-user.target
```

When you instruct systemd to start a service when the system boots (make the service persistent), it places a link to the service file in the directory specified by WantedBy in the service file. Continuing with the example, systemd places a link to **atd.service** in the **multi-user.target.wants** directory as shown previously. When you instruct systemd to not start a service when the system boots, it removes the link. "Setting the Persistent State of a Daemon" on page 445 explains how to use systemctl to make these changes.

The persistent (default) runlevel is also controlled by a link:

```
$ ls -l /etc/systemd/system/default.target
lrwxrwxrwx. 1 root root 36 06-04 19:34 /etc/systemd/system/default.target ->
/lib/systemd/system/graphical.target
```

See "Setting the Persistent Runlevel" on page 444 for more information.

CUSTOM SERVICE FILES

As explained in the previous section, files in the **/etc/systemd/system** directory hierarchy are symbolic links to files in the **/lib/systemd/system** directory hierarchy. The systemd **init** daemon treats files in the **/etc/systemd/system** directory hierarchy and files in the **/lib/systemd/system** directory hierarchy the same way, with files in **/etc/systemd/system** overriding files with the same name in **/lib/systemd/system**. An important difference between these directory hierarchies is that **/lib/systemd/system** is managed by yum/RPM while **/etc/systemd/system** is managed by the system administrator.

Put custom service files in the **/etc/systemd/system** hierarchy. If you want to modify a service file, copy it from the **/lib/systemd/system** hierarchy to the **/etc/systemd/system** hierarchy and edit the copy. The custom file in the **/etc/systemd/system** hierarchy will not be overwritten by yum/RPM and will take precedence over a file with the same name in the **/lib/systemd/system** hierarchy.

DETERMINING WHETHER systemd RUNS A DAEMON NATIVELY

To ease migration from SysVinit to systemd and to provide compatibility with software intended for other distributions, systemd can control daemons via SysVinit scripts (page 448). You can use the systemctl **status** command to determine whether systemd is controlling a daemon natively or via a SysVinit script. Following, systemctl displays the status of **cups** and **network**:

```
$ systemctl status cups.service
cups.service - CUPS Printing Service
   Loaded: loaded (/usr/lib/systemd/system/cups.service; enabled)
   Active: active (running) since Wed 2013-06-05 17:29:38 PDT; 2h 30min ago

$ systemctl status network.service
network.service - LSB: Bring up/down networking
   Loaded: loaded (/etc/rc.d/init.d/network)
   Active: inactive (dead)
```

The systemctl utility does not require a period and unit type (**.service** in the preceding example) following a unit name (**cups** and **network** in the preceding example). The lines that start with **Loaded** name the file controlling each daemon (service). The **cups** daemon is controlled by **/usr/lib/systemd/system/cups.service**. The location of the file (the **/usr/lib/systemd** hierarchy) and its filename extension (**.service**) indicate systemd is running the daemon natively. The **network** daemon is controlled by **/etc/rc.d/init.d/network**. The location of the file (the **init.d** directory) and the lack of a filename extension indicate systemd is running the daemon via a SysVinit script. See fedoraproject.org/wiki/User:Johannbg/QA/Systemd/compatability for a list of services that have been ported to systemd (and are run natively by systemd).

LPI service Although it is deprecated, you can use service to display information similar to that displayed by systemctl **status**. However, service does not accept a period and unit type.

```
$ service cups status
Redirecting to /bin/systemctl status  cups.service
cups.service - CUPS Printing Service
   Loaded: loaded (/usr/lib/systemd/system/cups.service; enabled)
   Active: active (running) since Wed 2013-06-05 17:29:38 PDT; 2h 30min ago
...

$ service network status
Configured devices:
lo Wired_connection_1 ens33
Currently active devices:
lo ens33
```

SETTING AND CHANGING RUNLEVELS

The runlevel specifies which daemons are running and which interfaces are available on a system. See "Runlevels" on page 440 and Table 10-1 on page 449 for more information. This section describes how to set the persistent (default) runlevel (the runlevel the system boots to) and how to change the current runlevel.

LPI SETTING THE PERSISTENT RUNLEVEL

Under systemd no true runlevels exist; see "Runlevels" on page 449. For example, the default runlevel under Fedora is **graphical.target** and has an alternative name of **runlevel5.target**. Under SysVinit this runlevel is referred to as runlevel 5 (multiuser graphical mode). The **/etc/systemd/system/default.target** file is a link to the file that specifies the target the system will boot to by default:

```
$ ls -l /etc/systemd/system/default.target
lrwxrwxrwx. 1 root root 36 06-04 19:34 /etc/systemd/system/default.target ->
/lib/systemd/system/graphical.target
```

The following command shows **runlevel5.target** is a link to **graphical.target**:

```
$ ls -l /lib/systemd/system/runlevel5.target
lrwxrwxrwx. 1 root root 16 06-04 20:02 /lib/systemd/system/runlevel5.target ->
graphical.target
```

The **multi-user.target** file (and the link to it, **runlevel3.target**) causes the system to boot to the multiuser runlevel (multiuser textual mode; SysVinit runlevel 3). The following command replaces the link shown in the previous example and will cause the system enter multiuser textual mode each time it is booted. This command does not change the current runlevel. (Newer versions of systemd use the command **systemctl set-default** ***target***, where ***target*** is replaced with the new runlevel target, in place of the following ln command.)

```
# ln -sf /lib/systemd/system/multi-user.target /etc/systemd/system/default.target
```

The ln **–s** (symbolic) option creates a symbolic link, and the **–f** (force) option overwrites any existing file.

The systemctl **list-units** command with the **––type=target** option lists all active target units. Before rebooting the system, this command shows that both **graphical.target** and **multi-user.target** are active.

```
$ systemctl list-units --type=target | egrep 'multi-user|graphical'
graphical.target    loaded active active Graphical Interface
multi-user.target   loaded active active Multi-User System
```

After giving the ln command and rebooting the system, **graphical.target** is no longer active but **multi-user.target** is:

```
$ systemctl list-units --type=target | egrep 'multi-user|graphical'
multi-user.target   loaded active active Multi-User System
```

See page 452 for instructions on how to boot to a specific runlevel using target units.

LPI CHANGING THE CURRENT RUNLEVEL

The systemctl **isolate** command changes the current runlevel of the system. The following command changes the runlevel to multiuser graphical mode (**graphical.target**). When the system is rebooted, it will return to the default runlevel as specified by the link at **/etc/systemd/system/default.target**.

```
# systemctl isolate graphical.target
```

After a moment systemctl shows that, again, both **graphical.target** and **multi-user.target** are active:

```
$ systemctl list-units --type=target | egrep 'multi-user|graphical'
graphical.target    loaded active active Graphical Interface
multi-user.target   loaded active active Multi-User System
```

The preceding examples were run from an ssh login. If you give these commands from a terminal emulator running on the console, the system will log you out each time it changes runlevels.

LPI CONFIGURING DAEMONS (SERVICES)

Two states are important when you consider a daemon: its current state and its persistent state (the state it will be in after the system is booted). Possible states are running and stopped. When you install a package that includes a daemon, the daemon is stopped and is not set up to start when the system is booted.

chkconfig and service

The systemctl utility controls the current and persistent states of daemons that run natively under systemd. The service and chkconfig utilities originally controlled daemons that run under SysVinit and were upgraded to control daemons that run under Upstart. Now these utilities have been retrofitted to control daemons that run under systemd, which they do by calling systemctl. You can still use service to control the current state of a daemon and chkconfig to control the persistent state of a daemon.

The next sections describe the systemctl commands that control daemons.

SETTING THE PERSISTENT STATE OF A DAEMON

The systemctl **disable** command causes a daemon not to start when the system is booted. This command has no effect on the current state of the daemon. You do not need to specify the **.service** part of the service name.

```
# systemctl disable atd.service
rm '/etc/systemd/system/multi-user.target.wants/atd.service'
```

The preceding command removes the link that causes systemd to start **atd** when the system enters multi-user runlevel and, by inheritance, graphical runlevel. See "**/etc/systemd/system** Hierarchy: Controls Services and the Persistent Runlevel" on page 441 for a discussion of these links.

The following commands each verify that the **atd** daemon will not start when the system boots. The first command shows no links in the ***.wants** directories for

Blank

20

sendmail: Setting Up Mail Servers, Clients, and More

In This Chapter

Objectives

After reading this chapter you should be able to:

- Explain what **sendmail** is
- Explain the purpose and give examples of an MUA, an MTA, and an MDA
- Describe the role of SMTP, IMAP, and POP
- Configure the local SMTP service to use a smarthost for outgoing mail
- Configure the local SMTP server to accept incoming mail, relay outgoing mail for specific hosts, and deliver mail directly to the remote systems
- Configure mail aliases and mail forwarding
- Install and configure SpamAssassin on a mail client and a mail server
- Install and configure a Webmail service (SquirrelMail)
- Setup a mailing list (Mailman)
- Install and configure an IMAP server (Dovecot)
- Describe the process and purpose of authenticated relaying

Sending and receiving email require three pieces of software. At each end, there is a client, called an MUA (mail user agent), which is a bridge between a user and the mail system. Common MUAs are Evolution, KMail, Thunderbird, mutt, and Outlook. When you send an email, the MUA hands it to an MTA (mail transfer agent, such as **exim4** or **sendmail**), which transfers it to the destination server. At the destination, an MDA (mail delivery agent, such as **procmail**) puts the mail in the recipient's mailbox file. On Linux systems, the MUA on the receiving system either reads the mailbox file or retrieves mail from a remote MUA or MTA, such as an ISP's SMTP (Simple Mail Transfer Protocol) server, using POP (Post Office Protocol) or IMAP (Internet Message Access Protocol).

SMTP

Most Linux MUAs expect a local MTA such as **sendmail** to deliver outgoing email. On some systems, including those with a dial-up connection to the Internet, the MTA sends email to an ISP's mail server. Because most MTAs use SMTP (Simple Mail Transfer Protocol) to deliver email, they are often referred to as SMTP servers.

You do not need to set up sendmail to send and receive email

tip

Most MUAs can use POP or IMAP to receive email from an ISP's server. These protocols do not require an MTA such as **sendmail**. As a consequence, you do not need to install or configure **sendmail** (or another MTA) to receive email. Although you still need SMTP to send email, the SMTP server can be at a remote location, such as your ISP. Thus you might not need to concern yourself with it either.

In the default Fedora setup, the **sendmail** MTA uses **procmail** as the local MDA. In turn, **procmail** writes email to the end of the recipient's mailbox file. You can also use **procmail** to sort email according to a set of rules, either on a per-user basis or globally. The global filtering function is useful for systemwide filtering to detect spam and for other tasks, but the per-user feature is largely superfluous on a modern system. Traditional UNIX MUAs were simple programs that could not filter mail and thus delegated this function to MDAs such as **procmail**. Modern MUAs, by contrast, incorporate this functionality. Although by default RHEL uses **postfix** as the MTA, this chapter explains how to set up **sendmail** as the MTA.

LPI INTRODUCTION TO sendmail

When the network that was to evolve into the Internet was first set up, it connected a few computers, each serving a large number of users and running several services. Each computer was capable of sending and receiving email and had a unique hostname, which was used as a destination for email.

Today the Internet has a large number of transient clients. Because these clients do not have fixed IP addresses or hostnames, they cannot receive email directly. Users on these systems usually maintain an account on an email server run by their employer or an ISP, and they collect email from this account using POP or IMAP. Unless you own a domain where you want to receive email, you will not need to set up **sendmail** to receive mail from nonlocal systems.

OUTBOUND EMAIL

When used as a server, **sendmail** accepts outbound email and directs it to the system it is addressed to (the destination system). This section describes the different ways you can set up **sendmail** to perform this task.

ACCEPTING EMAIL FOR DELIVERY

You can set up a **sendmail** server so it accepts outbound email from the local system only, from specified systems (such as a LAN), or from all systems. Accepting email from unknown systems makes it likely the server will propagate spam.

DELIVERING EMAIL

Direct connection

You can set up a **sendmail** server so it connects directly to the SMTP server on nonlocal destination systems. This SMTP server then delivers the email. Typically, **sendmail** delivers local email directly.

Smarthost

Alternately, you can set up a **sendmail** server so it sends email bound for nonlocal systems to an SMTP server that relays the mail to its destination. This type of server is called a *smarthost* or *SMTP relay*.

Port 25

By default, SMTP uses port 25. Some Windows viruses use a simple, self-contained SMTP server to propagate themselves. As a partial defense against these types of viruses, some ISPs and larger organizations block all outgoing connections that originate or terminate on port 25, with the exception of connections to their own SMTP server (smarthost). Blocking this port prevents local systems from making a direct connection to send email. These organizations require you to use a smarthost for email bound for nonlocal systems.

INBOUND EMAIL

Although not typical, you can set up **sendmail** to accept email for a registered domain name as specified in the domain's DNS MX record (page 859). However, most mail clients (MUAs) do not interact directly with **sendmail** to receive email. Instead, they use POP or IMAP—protocols that include features for managing mail folders, leaving messages on the server, and reading only the subject of an email without downloading the entire message. If you want to collect email from a system other than the one running the incoming mail server, you might need to set up a POP or IMAP server, as discussed on page 763.

LPI ALTERNATIVES TO sendmail

Over the years, **sendmail** has grown to be enormously complex. Its complexity makes it challenging to configure if you want to set up something more than a simple mail server. Its size and complexity also add to its vulnerability. For optimal security, make sure you run the latest version of **sendmail** and always keep **sendmail** up-to-date. Or you might want to consider using one of the following alternatives.

LPI **exim4** The default MTA under Ubuntu, **exim4** (www.exim.org; **exim** package), was introduced in 1995 by the University of Cambridge. It can be configured in several different ways and includes many features other MTAs lack.

LE+ Postfix Postfix (www.postfix.org; **postfix** package) is an alternative MTA. Postfix is fast and easy to administer but is compatible enough with **sendmail** to not upset **sendmail** users. Postfix has a good reputation for ease of use and security and is a drop-in replacement for **sendmail**.

LPI Qmail Qmail (www.qmail.org) is a direct competitor of Postfix and has the same objectives. By default, Qmail stores email using the **maildir** format as opposed to the **mbox** format that other MTAs use (page 745).

MORE INFORMATION

Web **sendmail**: www.sendmail.org
exim4: www.exim.org, wiki.debian.org/PkgExim4
procmail: www.procmail.org
IMAP and POP3: www.dovecot.org, cyrusimap.org
SpamAssassin: spamassassin.apache.org, wiki.apache.org/spamassassin
Spam database: razor.sourceforge.net
Mailman: www.list.org
SquirrelMail: www.squirrelmail.org
Dovecot: www.dovecot.org
Postfix: www.postfix.org
Qmail: www.qmail.org

Local Dovecot: **/usr/share/doc/dovecot***
man pages: **sendmail**, aliases, makemap, spamassassin spamc **spamd**
SpamAssassin: **/usr/share/doc/spamassassin***, give the following command:

```
$ perldoc Mail::SpamAssassin::Conf
```


LPI SETTING UP A sendmail MAIL SERVER

This section explains how to set up a **sendmail** mail server.

PREREQUISITES

Install the following packages:

- **sendmail**
- **sendmail-cf** (required to configure **sendmail**)

Restart **sendmail** As installed, **sendmail** is active and enabled so it will run when the system is booted. After changing its configuration files, give the following command to restart **sendmail**, causing it to reread its configuration files:

```
# systemctl restart sendmail.service
```

Notes

Firewall An SMTP server normally uses TCP port 25. If an SMTP server system that receives nonlocal mail is running a firewall, you need to open this port. Give the following commands to open the port each time the system boots (permanently) and on the running system; see page 906 for information on firewall-cmd.

```
$ sudo -c 'firewall-cmd --add-port=25/tcp'
$ sudo -c 'firewall-cmd --permanent --add-port=25/tcp'
```

cyrus This chapter covers the IMAP and POP3 servers included in the **dovecot** package. Fedora/RHEL also provides IMAP and POP3 servers in the **cyrus-imapd** package.

FQDN If the local system does not have an *FQDN* (page 1250), **sendmail** will start very slowly. As a consequence, the system might boot slowly. See **/etc/hostname** on page 507 for instructions on how to determine if the system has an FQDN and how to give the local system an FQDN.

LPI JumpStart I: Configuring sendmail on a Client

You might not need to configure sendmail to send email

tip With **sendmail** running, give the command described under "Test" on page 744. As long as **sendmail** can connect to port 25 outbound, you should not need to set up **sendmail** to use an SMTP relay as described in this section. If you receive the mail sent by the test, you can skip this section. However, see "FQDN," above.

This JumpStart configures an outbound **sendmail** server. This server

- Uses a remote SMTP server—typically an ISP—to relay outbound email to its destination (a smarthost or SMTP relay).
- Sends to the SMTP server email originating from the local system only. It does not forward email originating from other systems.
- Does not handle inbound email. As is frequently the case, you need to use POP or IMAP to receive email.

Change **sendmail.mc** To set up this server, you must edit **/etc/mail/sendmail.mc** and restart **sendmail** (page 742).

The **dnl** (page 749) at the start of the following line in **sendmail.mc** indicates that this line is a comment:

```
dnl define(‘SMART_HOST',‘smtp.your.provider')dnl
```

You can ignore the **dnl** at the end of the line. To specify a remote SMTP server, you must open **sendmail.mc** in an editor and change the preceding line, deleting **dnl** from the beginning of the line and replacing **smtp.your.provider** with the FQDN of your

ISP's SMTP server (obtain this name from your ISP). Be careful not to alter the back ticks (‘) and the single quotation marks (') in this line. If your ISP's SMTP server is at **smtp.myisp.com**, you would change the line to

```
define(‘SMART_HOST',‘smtp.myisp.com')dnl
```

Do not alter the back ticks (‘) or the single quotation marks (')

tip Be careful not to alter the back ticks (‘) or the single quotation marks (') in any line in **sendmail.mc**. These symbols control the way the m4 preprocessor converts **sendmail.mc** to **sendmail.cf**; **sendmail** will not work properly if you do not preserve these symbols.

Restart **sendmail** so that it regenerates the **sendmail.cf** file from the **sendmail.mc** file you edited (page 742).

Test Test **sendmail** with the following command:

```
$ echo "my sendmail test" | /usr/sbin/sendmail user@remote.host
```

Replace *user@remote.host* with an email address on *another system* where you receive email. You need to send email to a remote system to make sure that **sendmail** is relaying your email.

LPI JUMPSTART II: CONFIGURING sendmail ON A SERVER

If you want to receive inbound email sent to a registered domain that you own, you need to set up **sendmail** as an incoming mail server. This JumpStart describes how to set up such a server. This server

- Accepts outbound email from the local system only.
- Delivers outbound email directly to the recipient's system, without using an SMTP relay (smarthost).
- Accepts inbound email from any system.

This server does not relay outbound email originating on other systems. Refer to "access: Sets Up a Relay Host" on page 752 if you want the local system to act as a relay. For this configuration to work, you must be able to make outbound connections from and receive inbound connections to port 25.

The line in **sendmail.mc** that limits **sendmail** to accepting inbound email from the local system only is

```
DAEMON_OPTIONS(‘Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

To allow **sendmail** to accept inbound email from other systems, remove the parameter **Addr=127.0.0.1,** from the preceding line:

```
DAEMON_OPTIONS(‘Port=smtp, Name=MTA')dnl
```

E

LPI and CompTIA Certification

In This Appendix

This book is used as the adopted text in many college classes. Because students who take these classes often seek LPI or CompTIA certification, instructors have asked for a mapping of certification objectives to the material covered in this book. This book fully covers LPI's Linux Essentials certification learning goals and provides extensive coverage of CompTIA's Linux+ exam objectives. This appendix maps these learning goals and exam objectives to pages in this book. The following icons are used throughout the book to mark the places where learning goals and exam objectives are discussed.

LE This icon indicates coverage of a topic in the LPI's Linux Essentials certification learning goals.

LPI This icon indicates coverage of a topic in the CompTIA's Linux+ exam objectives.

LE+ This icon indicates coverage of a topic in the CompTIA's Linux+ exam objectives and a topic in the LPI's Linux Essentials certification learning goals.

MORE INFORMATION

LPI Linux Essentials: www.lpi.org/linux-certifications/introductory-programs/linux-essentials
LPI Certification Exams: www.lpi.org/linux-certifications/programs/lpic-1
CompTIA Exams: certification.comptia.org/getCertified/certifications/linux.aspx
CompTIA and LPI partnership: www.lpi.org/linux-certifications/partnership-programs/comptia

LINUX ESSENTIALS

TOPIC 1: THE LINUX COMMUNITY AND A CAREER IN OPEN SOURCE

1.1 LINUX EVOLUTION AND POPULAR OPERATING SYSTEMS

DESCRIPTION: KNOWLEDGE OF LINUX DEVELOPMENT AND MAJOR DISTRIBUTIONS

Key Knowledge Areas

Open Source Philosophy
- Open-Source Software and Licensing page 6

Distributions
- Distribution page 6

Embedded Systems
- Embedded and moble Linux page 6

Partial List of Used Files, Terms, and Utilities

Android
- Embedded and moble Linux page 6

Debian
- Distribution page 6

CentOS
- CentOS page 33

1.2 MAJOR OPEN SOURCE APPLICATIONS

DESCRIPTION: AWARENESS OF MAJOR APPLICATIONS AND THEIR USES

Key Knowledge Areas

Desktop Applications
- Desktop applications page 1151

Server Applications
- DHCP: Configures Network Interfaces page 491
- Chapter 13: Printing with CUPS page 555
- Chapter 18: OpenSSH: Secure Network Communication page 685
- Chapter 19: FTP: Transferring Files Across a Network page 713
- Chapter 20: sendmail: Setting Up Mail Servers, Clients, and More page 739
- Chapter 21: NIS and LDAP page 769
- Chapter 22: NFS: Sharing Directory Hierarchies page 801

- Chapter 23: Samba: Linux and Windows File and Printer Sharing page 827
- Chapter 24: DNS/BIND: Tracking Domain Names and Addresses page 851
- Chapter 26: Apache (**httpd**): Setting Up a Web Server page 931

Mobile Applications
- Embedded and moble Linux page 6

Development Languages
- Chapter 27: Programming the Bourne Again Shell (bash) page 981
- Chapter 28: The Python Programming Language page 1081
- Chapter 29: The MariaDB SQL Database Management System page 1113

Package Management Tools and repositories
- Chapter 12: Finding, Downloading, and Installing Software page 531
- Appendix D: Keeping the System Up-to-Date Using apt-get page 1183

Partial List of Used Files, Terms, and Utilities

OpenOffice.org, LibreOffice, Thunderbird, Firefox, Blender, Gimp, Audacity, ImageMagick
- Desktop applications page 1151

Apache, MySQL, PostgreSQL
- Chapter 26: Apache (**httpd**): Setting Up a Web Server page 931
- Chapter 29: The MariaDB SQL Database Management System page 1113
- Programming languages page 1152

NFS, Samba, OpenLDAP, Postfix, DNS, DHCP
- Chapter 22: NFS: Sharing Directory Hierarchies page 801
- Chapter 23: Samba: Linux and Windows File and Printer Sharing page 827
- LDAP page 786
- Postfix page 742
- Chapter 24: DNS/BIND: Tracking Domain Names and Addresses page 851
- DHCP: Configures Network Interfaces page 491

C, Perl, shell, Python, PHP
- Chapter 27: Programming the Bourne Again Shell (bash) page 981
- Chapter 28: The Python Programming Language page 1081
- Programming languages page 1152

1.3 Understanding Open Source Software and Licensing

Description: Open communities and licensing Open Source Software for business

Key Knowledge Areas

Licensing
- Open-Source Software and Licensing page 6

Free Software Foundation (FSF), Open Source Initiative (OSI)
- GNU Project page 3
- Linux Is More than a Kernel page 6
- FOSS/FLOSS page 7
- GNOME and KDE page 17

Partial List of Used Files, Terms, and Utilities

GPL, BSD, Creative Commons

- GPL page 5
- Berkeley UNIX (BSD) page 3
- Creative Commons page 1245 (Glossary)

Free Software, Open Source Software, FOSS, FLOSS

- FOSS/FLOSS page 7

Open Source business models

- Making money page 7

1.4 ICT Skills and Working in Linux

Description: Basic Information and Communication Technology (ICT) skills and working in Linux

Key Knowledge Areas

Desktop Skills

- Chapter 4: Introduction to Fedora and Red Hat Enterprise Linux page 89

Getting to the Command Line

- Working from the Command Line page 119
- Chapter 7: The Linux Utilities page 215

Industry uses of Linux, Cloud Computing, and Virtualization

- Chapter 17: Setting Up Virtual Machines Locally and in the Cloud page 659

Partial List of Used Files, Terms, and Utilities

Using a browser, privacy concerns, configuration options, searching the Web, and saving content

- Firefox: www.mozilla.org/en-US/firefox/central
- Chrome: www.google.com/intl/en/chrome/browser/features.html
- Opera: www.opera.com

Terminal and Console

- Using a Virtual Console page 121

Password issues

- Users: Changing Your Account Type and Password (GUI) page 112
- Password Security page 136
- passwd: Changing Your Password (CLI) page 137
- Passwords page 625

Privacy issues and tools

- Search the Web for **browser privacy**
- Mozilla: support.mozilla.org/en-US/kb/private-browsing-browse-web-without-saving-info
- lifehacker.com/the-best-browser-extensions-that-protect-your-privacy-479408034

Use of common open-source applications in presentations and projects

- Desktop Applications page 1151

Topic 2: Finding Your Way on a Linux System

2.1 Command Line Basics

Description: Basics of using the Linux command line

Key Knowledge Areas

Basic shell
- Working from the Command Line page 119
- Chapter 5: The Shell page 141

Formatting commands
- The Command Line page 144

Working with Options
- Options page 145

Variables
- Parameters and Variables page 352
- Variables page 1031

Globbing
- Filename Generation/Pathname Expansion page 165
- Pathname Expansion page 412

Quoting
- Special Characters page 142
- Quoting the $ page 354
- Quotation marks page 412

Partial List of Used Files, Terms, and Utilities

echo
- echo: Displays Arguments page 219
- **echo –e** page 1009

history
- History page 376

PATH env variable
- Set **PATH** in **.bash_profile** page 331
- **PATH**: Where the Shell Looks for Programs page 359

which
- which page 255

Nice to Know

Substitutions
- Command Substitution page 410

||, &&, and ; control operators
- Lists page 162
- ; and NEWLINE Separate Commands page 341
- && and || Boolean Control Operators page 343

2.2 Using the Command Line to Get Help

Description: Running Help commands and navigation of the various Help systems

Key Knowledge Areas

man

- man: Displays the System Manual page 128

info

- info: Displays Information About Utilities page 131

Partial List of Used Files, Terms, and Utilities

man

- man: Displays the System Manual page 128

info

- info: Displays Information About Utilities page 131

man pages

- man: Displays the System Manual page 128

/usr/share/doc

- **/usr/share/doc** page 134

locate

- locate: Searches for a File page 256

Nice to Know

apropos, whatis, whereis

- apropos: Searches for a Keyword page 130
- whatis page 130
- whereis page 255

2.3 Using Directories and Listing Files

Description: Navigation of home and system directories and listing files in various locations

Key Knowledge Areas

Files, directories

- Ordinary Files and Directory Files page 177

Hidden files and directories

- Hidden Filenames page 180

Home

- Your Home Directory page 143

Absolute and relative paths

- Absolute Pathnames page 181
- Relative Pathnames page 182

Partial List of Used Files, Terms, and Utilities

Common options for ls

- Options page 221

Recursive listings

- Recursive page 222

cd

- cd: Changes to Another Working Directory page 185

. and ..

- The . and .. Directory Entries page 186

home and ~

- Your Home Directory page 143
- ~ (Tilde) in Pathnames page 182
- Tilde (~) page 359
- Tilde Expansion page 407

2.4 CREATING, MOVING, AND DELETING FILES

DESCRIPTION: CREATE, MOVE, AND DELETE FILES AND DIRECTORIES UNDER THE HOME DIRECTORY

Key Knowledge Areas

Files and directories

- Ordinary Files and Directory Files page 177

Case sensitivity

- Case sensitivity page 179

Simple globbing and quoting

- Filename Generation/Pathname Expansion page 165
- Pathname Expansion page 412
- Special Characters page 166
- Quoting the $ page 394
- Quotation marks page 452

Partial List of Used Files, Terms, and Utilities

mv, cp, rm, touch

- mv: Moves a Directory page 188
- mv: Renames or Moves a File page 237
- mv, cp: Move or Copy Files page 187
- cp: Copies Files page 224
- rm: Removes a Link page 208
- rm: Removes a File (Deletes a Link) page 222
- touch: Changes File Modification and Access Times page 243

mkdir, rmdir

- mkdir: Creates a Directory page 184
- rmdir: Deletes a Directory page 186

TOPIC 3: THE POWER OF THE COMMAND LINE

3.1 ARCHIVING FILES ON THE COMMAND LINE

DESCRIPTION: ARCHIVING FILES IN THE USER HOME DIRECTORY

Key Knowledge Areas

Files, directories

- Ordinary Files and Directory Files page 177

Archives, compression

- Compressing and Archiving Files page 245

Partial List of Used Files, Terms, and Utilities

tar

- tar: Stores or Extracts Files to/from an Archive File page 249
- tar: Archives Files page 603

Common tar options

- Options page 249
- Modifiers page 251

gzip, bzip2

- xz, bzip2, and gzip: Compress and Decompress Files page 245

zip, unzip

- zip page 249
- unzip page 249

Nice to Know

Extracting individual files from archives

- Extract page 250

3.2 SEARCHING AND EXTRACTING DATA FROM FILES

DESCRIPTION: SEARCH AND EXTRACT DATA FROM FILES IN THE HOME DIRECTORY

Key Knowledge Areas

Command line pipes

- Pipelines page 158

I/O redirection

- Redirection page 153

Partial POSIX Regular Expressions (., [ ], *, ?)

- Appendix A: Regular Expressions page 1139

Partial List of Used Files, Terms, and Utilities

find

- find: Finds Files Based on Criteria page 229

grep

- grep: Searches for a Pattern in Files page 232

less

- less Is more: Display a Text File One Screen at a Time page 220

head, tail
- head: Displays the Beginning of a File page 235
- tail: Displays the Last Part of a File page 241

sort
- sort: Sorts and/or Merges Files page 239

cut
- cut: Selects Characters or Fields from Input Lines page 225

wc
- wc: Displays the Number of Lines, Words, and Bytes in Files page 244

Nice to Know

Partial POSIX Basic Regular Expressions ([^], ^, $)
- Appendix A: Regular Expressions page 1139

Partial POSIX Extended Regular Expressions (+, ( ), |)
- Appendix A: Regular Expressions page 1139

xargs
- xargs: Converts Standard Input to Command Lines page 260

3.3 TURNING COMMANDS INTO A SCRIPT

DESCRIPTION: TURNING REPETITIVE COMMANDS INTO SIMPLE SCRIPTS

Key Knowledge Areas

Basic text editing
- Tutorial: Using vim to Create and Edit a File page 262
- Tutorial: Using nano to Create and Edit a File page 270

Basic shell scripting
- Writing and Executing a Basic Shell Script page 127

Partial List of Used Files, Terms, and Utilities

/bin/sh
- sh Shell page 328

Variables
- Parameters and Variables page 352
- Variables page 1031

Arguments
- Arguments page 145

for loops
- **for...in** page 995
- **for** page 997

echo
- echo: Displays Arguments page 219
- **echo –e** page 1009

Exit status
- **$?**: Exit Status page 1029

Nice to Know

pico, nano, vi (only basics for creating scripts)

- pico, see Desktop applications page 1151
- Tutorial: Using vim to Create and Edit a File page 262
- Tutorial: Using nano to Create and Edit a File page 270

bash

- Chapter 5: The Shell page 141
- Chapter 9: The Bourne Again Shell (bash) page 327
- Chapter 27: Programming the Bourne Again Shell (bash) page 981

if, while, case statements

- **if...then** page 983
- **if...then...else** page 987
- **if...then...elif** page 989
- **for...in** page 995

read and test, and [commands

- read: Accepts User Input page 1041
- test builtin page 983
- [] is a synonym for test page 986
- test builtin page 1000

TOPIC 4: THE LINUX OPERATING SYSTEM

4.1 CHOOSING AN OPERATING SYSTEM

DESCRIPTION: KNOWLEDGE OF MAJOR OPERATING SYSTEMS AND LINUX DISTRIBUTIONS

Key Knowledge Areas

Windows, Mac, Linux differences

- Choosing an Operating System page 19

Distribution life cycle management

- Fedora, RHEL, and CentOS page 33

Partial List of Used Files, Terms, and Utilities

GUI versus command line, desktop configuration

- Choosing an Operating System page 19

Maintenance cycles, Beta and Stable

- beta release page 1239 (Glossary)
- stable release page 1274 (Glossary)

4.2 UNDERSTANDING COMPUTER HARDWARE

DESCRIPTION: FAMILIARITY WITH THE COMPONENTS THAT GO INTO BUILDING DESKTOP AND SERVER COMPUTERS

Key Knowledge Areas

Hardware

- Requirements page 30

Partial List of Used Files, Terms, and Utilities

Hard drives and partitions, motherboards, processors, power supplies, optical drives, peripherals

- Setting Up the Hard Disk page 36
- motherboard page 1261 (Glossary)
- Processor Architecture page 31
- power supply page 1267 (Glossary)
- optical drive page 1264 (Glossary)
- Peripheral, see device page 1246 (Glossary)

Display types

- Interfaces: Installer and Installed System page 32
- Working from the Command Line page 119
- ASCII terminal page 1237 (Glossary)
- graphical display page 1251 (Glossary)

Drivers

- Device files page 515
- Block and Character Devices page 518
- device driver page 1246 (Glossary)

4.3 WHERE DATA IS STORED

DESCRIPTION: WHERE VARIOUS TYPES OF INFORMATION ARE STORED ON A LINUX SYSTEM

Key Knowledge Areas

Kernel

- Chapter 14: Building a Linux Kernel page 579
- kernel page 1257 (Glossary)

Processes

- Process page 150
- Processes page 373
- ps page 466
- process page 1267 (Glossary)

syslog, klog, dmesg

- **rsyslogd**: Logs System Messages page 620
- **klogd**: deprecated; www.linuxjournal.com/article/4058
- dmesg: Displays Kernel Messages page 595

/lib, /usr/lib, /etc, /var/log

- **/lib** page 190
- **/lib64** page 190
- **/usr/lib** page 190
- **/usr/lib64** page 191
- **/etc** page 190
- **/etc** page 506
- **/var/log** page 191
- **/var/log** page 514
- Log Files and Mail for **root** page 626

Partial List of Used Files, Terms, and Utilities

Programs, libraries, packages and package databases, system configuration

- **/lib** page 190
- **/lib64** page 190
- **/usr/lib** page 190
- **/usr/lib64** page 191
- **/usr/bin** page 190
- **/usr/sbin** page 190
- **/etc** page 190
- **/etc** page 506
- library page 1258 (Glossary)
- Software package page 532
- PMS page 532
- Software package formats page 532
- Repositories page 533

Processes and process tables, memory addresses, system messaging, and logging

- Process page 150
- Processes page 373
- ps page 466
- dmesg: Displays Kernel Messages page 595
- **rsyslogd**: Logs System Messages page 620
- D-BUS page 898
- process page 1267 (Glossary)

ps, top, free

- Process Identification page 374
- ps page 466
- top: Lists Processes Using the Most Resources page 612
- free: Displays Memory Usage Information page 253

4.4 YOUR COMPUTER ON THE NETWORK

DESCRIPTION: QUERYING VITAL NETWORKING SETTINGS AND DETERMINING THE BASIC REQUIREMENTS FOR A COMPUTER ON A LOCAL AREA NETWORK (LAN)

Key Knowledge Areas

Internet, network, routers

- Internet page 280
- Introduction to Networking page 280
- Internetworking Through Gateways and Routers page 287

Domain Name Service

- Chapter 24: DNS/BIND: Tracking Domain Names and Addresses page 851

Network configuration

- Chapter 16: Configuring and Monitoring a LAN page 631

Partial List of Used Files, Terms, and Utilities

route

- deprecated (route man page): see ip man page, **route** object instead

resolv.conf
- **/etc/resolv.conf** page 510

IPv4, IPv6
- IPv4 page 292
- IPv6 page 293

ifconfig
- deprecated: (ifconfig man page): see ip man page, **addr** and **link** objects instead

netstat
- netstat: see the netstat man page and wikipedia.org/wiki/netstat

ping
- ping: Tests a Network Connection page 305

Nice to Know

ssh
- ssh: Logs in or Executes Commands on a Remote System page 693

dig
- host and dig: Query Internet Nameservers page 307
- dig page 861
- dig page 862

TOPIC 5: SECURITY AND FILE PERMISSIONS

5.1 BASIC SECURITY AND IDENTIFYING USER TYPES

DESCRIPTION: VARIOUS TYPES OF USERS ON A LINUX SYSTEM

Key Knowledge Areas

Root and Standard Users
- Running Commands with **root** Privileges page 422
- The Special Powers of a Privileged User page 422
- Gaining **root** Privileges page 423
- Real UID Versus Effective UID page 425

System users
- **/etc/passwd** page 508

Partial List of Used Files, Terms, and Utilities

/etc/passwd, /etc/group
- **/etc/passwd** page 508
- **/etc/group** page 506

id, who, w
- id page 426
- who: Lists Users on the System page 254
- who, whoami page 426
- w: Lists Users on the System page 254

sudo
- Using sudo to Gain **root** Privileges page 428

Nice to Know

su

- Using su to Gain **root** Privileges page 425

5.2 Creating Users and Groups

Description: Creating users and groups on a Linux system

Key Knowledge Areas

User and group commands

- useradd: Adds a User Account page 600
- groupadd: Adds a Group page 601
- usermod: Modifies a User Account page 601
- userdel: Removes a User Account page 600
- groupdel and groupmod: Remove and Modify a Group page 601

User IDs

- Real UID Versus Effective UID page 425
- **/etc/passwd** page 508
- user ID page 1279 (Glossary)

Partial List of Used Files, Terms, and Utilities

/etc/passwd, /etc/shadow, /etc/group

- **/etc/passwd** page 508
- **/etc/shadow** page 511
- **/etc/group** page 506

id, last

- id page 426
- last: see the last man page

useradd, groupadd

- useradd: Adds a User Account page 600
- groupadd: Adds a Group page 601

passwd

- Users: Changing Your Account Type and Password (GUI) page 112
- passwd: Changing Your Password (CLI) page 137

Nice to Know

usermod, userdel

- usermod: Modifies a User Account page 601
- userdel: Removes a User Account page 600

groupmod, groupdel

- groupdel and groupmod: Remove and Modify a Group page 601

5.3 Managing File Permissions and Ownership

Description: Understanding and manipulating file permissions and ownership settings

Key Knowledge Areas

File/directory permissions and owners

- Access Permissions page 191

Partial List of Used Files, Terms, and Utilities
ls –l
- **ls –l**: Displays Permissions page 191

chmod, chown
- chmod: Changes File Access Permissions page 193
- chmod: Makes a File Executable page 337
- chown: Changes File Ownership page 195

Nice to Know
chgrp
- chgrp: Changes File Group Association page 195

5.4 SPECIAL DIRECTORIES AND FILES

DESCRIPTION: SPECIAL DIRECTORIES AND FILES ON A LINUX SYSTEM INCLUDING SPECIAL PERMISSIONS

Key Knowledge Areas
System files, libraries
- Important Standard Directories and Files page 189
- library page 1258 (Glossary)

Symbolic links
- Symbolic Links page 206
- Symbolic links page 515
- symbolic link page 1276 (Glossary)

Partial List of Used Files, Terms, and Utilities
/etc, /var
- **/etc** page 190
- /etc page 506
- **/var** page 41
- **/var** page 191

/tmp, /var/tmp and Sticky Bit
- **/tmp** page 190
- **/var** page 191
- Sticky bit page 196
- sticky bit page 1275 (Glossary)

ls –d
- Directory page 221

ln –s
- Size page 222

Nice to Know
Hard links
- Hard Links page 204

Setuid/Setgid
- Setuid and Setgid Permissions page 196
- Setuid file page 424

- Setuid files page 626
- setuid page 1272 (Glossary)
- setgid page 1272 (Glossary)

CERTIFICATION EXAM 1 OBJECTIVES: LX0-101

101 SYSTEM ARCHITECTURE

101.1 DETERMINE AND CONFIGURE HARDWARE SETTINGS

Enable and disable integrated peripherals

Configure systems with or without external peripherals such as keyboards

Differentiate between the various types of mass storage devices

- **/dev** page 503

Set the correct hardware ID for different devices, especially the boot device

Know the differences between coldplug and hotplug devices

- Hotplug page 516

Determine hardware resources for devices

Tools and utilities to list various hardware information (e.g., lsusb, lspci, etc.)

- dmesg: Displays Kernel Messages page 595
- lspci: Lists PCI Information page 635
- lsblk: Lists Block Device Information page 635
- lshw: Lists Hardware Information page 636
- lsusb: Lists USB Devices page 636

Tools and utilities to manipulate USB devices

- Writing to a USB Flash Drive page 50

Conceptual understanding of sysfs, udev, hald, dbus

- udev page 516
- D-BUS page 898

Partial List of Used Files, Terms, and Utilities

/sys

- **/sys** page 190
- **/sys** page 514
- **/sys** page 516

/proc

- **/proc** page 190
- **/proc** page 512
- **proc** page 520

/dev

- Device file page 152
- **/dev** page 190

- **/dev** page 503
- Device files page 515

modprobe
- modprobe page 589

lsmod
- lsmod page 589

lspci
- lspci: Lists PCI Information page 635

lsusb
- lsusb: Lists USB Devices page 636

101.2 BOOT THE SYSTEM

Provide common commands to the boot loader and options to the kernel at boot time
- Modifying Boot Parameters (Options) page 70
- GRUB: The Linux Boot Loader page 590

Demonstrate knowledge of the boot sequence from BIOS to boot completion
- BIOS setup page 31
- CMOS page 31
- Booting the System page 450
- GRUB: The Linux Boot Loader page 590
- BIOS page 590
- BIOS page 1239 (Glossary)

Check boot events in the log file
- dmesg: Displays Kernel Messages page 595

Partial List of Used Files, Terms, and Utilities

/var/log/messages
- **/var/log/messages** page 514
- **/var/log/messages** page 622
- Log Files and Mail for **root** page 626
- **/var/log/messages** page 627

dmesg
- dmesg: Displays Kernel Messages page 595

BIOS
- BIOS setup page 31
- BIOS page 590
- BIOS page 1239 (Glossary)

boot loader
- GRUB: The Linux Boot Loader page 590

kernel
- Chapter 14: Building a Linux Kernel page 579
- kernel page 1257 (Glossary)

init
- **init** daemon page 374

- The **systemd init** Daemon page 438
- SysVinit (**rc**) Scripts: Start and Stop System Services page 448
- **systemd init** daemon page 450

101.3 CHANGE RUNLEVELS AND SHUTDOWN OR REBOOT SYSTEM

Set the default runlevel

- Setting the Persistent Runlevel page 444
- **/etc/inittab** page 508

Change between runlevels including single-user mode

- Changing the Current Runlevel page 445
- telinit page 449
- Booting the System to Single-User/Rescue Mode page 450
- Going to Graphical Multiuser Mode page 452

Shutdown and reboot from the command line

- Bringing the System Down page 454

Alert users before switching runlevels or other major system events

Properly terminate processes

- kill: Aborting a Background Job page 164
- kill: Sends a Signal to a Process page 465
- killall: Kills a Command page 467
- pkill: Kills a Command page 468
- kill: Aborts a Process page 1050

Partial List of Used Files, Terms, and Utilities

/etc/inittab

- **/etc/inittab** page 508

shutdown

- Bringing the System Down page 454

init

- **init** daemon page 374
- The **systemd init** Daemon page 438
- SysVinit (**rc**) Scripts: Start and Stop System Services page 448
- **systemd init** daemon page 450

/etc/init.d

- SysVinit (**rc**) Scripts: Start and Stop System Services page 448

telinit

- telinit page 449

102 LINUX INSTALLATION AND PACKAGE MANAGEMENT

102.1 DESIGN HARD DISK LAYOUT

Allocate filesystems and swap space to separate partitions or disks

- Setting Up the Hard Disk page 36

Tailor the design to the intended use of the system
- Planning the Installation page 29

Ensure the **/boot** partition conforms to the hardware architecture requirements for booting
- Where to put the **/boot** partition page 41
- LBA addressing mode and the **/boot** partition page 590

Partial List of Used Files, Terms, and Utilities

/ (root) filesystem
- **/** (root) page 40
- **/** (root) page 181
- **/** page 189
- root filesystem page 1271 (Glossary)

/var filesystem
- **/var** page 41
- **/var** page 191

/home filesystem
- **/home** page 41
- **/home** page 190

swap space
- (swap) page 40
- swap page 513
- swap space page 1276 (Glossary)

mount points
- Mount Points page 38
- Mount point page 521

partitions
- Partitions page 36
- Partition table page 36
- Primary, Extended, and Logical Partitions page 37
- Default Partitioning page 39
- Manual Partitioning: Planning Partitions page 39
- Example minimum partition sizes page 42
- Manual/Custom Partitioning page 74
- partition page 1265 (Glossary)

102.2 INSTALL A BOOT MANAGER

Providing alternative boot locations and backup boot options

Install and configure a boot loader such as GRUB
- GRUB: The Linux Boot Loader page 590

Interact with the boot loader
- Booting the System to Single-User/Rescue Mode page 450

Partial List of Used Files, Terms, and Utilities

/boot/grub/menu.lst
- Configuring GRUB page 591

grub-install
- grub2-install: Installs the MBR and GRUB Files page 594

MBR
- Reinstalling the MBR page 456
- MBR page 590
- grub2-install: Installs the MBR and GRUB Files page 594

superblock
- superblock page 1276 (Glossary)

/etc/lilo.conf

lilo

102.3 MANAGE SHARED LIBRARIES

Identify shared libraries
- ldd page 487

Identify the typical locations of system libraries

Load shared libraries

Partial List of Used Files, Terms, and Utilities

ldd
- ldd & **libwrap** page 486
- ldd page 487

ldconfig

/etc/ld.so.conf

LD_LIBRARY_PATH

102.4 USE DEBIAN PACKAGE MANAGEMENT

Install, upgrade, and uninstall Debian binary packages
- Using apt-get to Install, Remove, and Update Packages page 1184

Find packages containing specific files or libraries which may or may not be installed

Obtain package information like version, content, dependencies, package integrity, and installation status (whether or not the package is installed)

Partial List of Used Files, Terms, and Utilities

/etc/apt/sources.list
- **sources.list**: Specifies Repositories for apt-get to Search page 1187

dpkg
- apt-get and dpkg page 1184

dpkg-reconfigure

apt-get
- Using apt-get to Install, Remove, and Update Packages page 1184
- Using apt-get to Upgrade the System page 1185
- Other apt-get Commands page 1186

apt-cache

aptitude

102.5 USE RPM AND YUM PACKAGE MANAGEMENT

See Chapter 12: Finding, Downloading, and Installing Software page 531

Install, re-install, upgrade, and remove packages using RPM and YUM

- JumpStart: Installing and Removing Software Packages Using yum page 534
- Updating Packages page 538
- RPM: The RPM Package Manager page 546
- Installing, Upgrading, and Removing Packages page 548

Obtain information on RPM packages such as version, status, dependencies, integrity, and signatures

- Querying Packages and Files page 547

Determine what files a package provides, as well as find which package a specific file comes from

- Finding the Package That Holds an Application or File You Need page 536
- Querying Packages and Files page 547

Partial List of Used Files, Terms, and Utilities

rpm

- RPM: The RPM Package Manager page 546
- Querying Packages and Files page 547

rpm2cpio

/etc/yum.conf

- **yum.conf**: Configures yum page 541

/etc/yum.repos.d/

- yum Repositories page 542

yum

- yum page 533
- JumpStart: Installing and Removing Software Packages Using yum page 534
- Finding the Package That Holds an Application or File You Need page 536
- yum: Keeps the System Up-to-Date page 538

yumdownloader

- Downloading RPM Package Files with yumdownloader page 540
- yumdownloader page 582

103 GNU AND UNIX COMMANDS

103.1 WORK ON THE COMMAND LINE

See Chapter 5: The Shell page 141

See Chapter 9: The Bourne Again Shell (bash) page 327

See Chapter 7: The Linux Utilities page 215

See Chapter 27: Programming the Bourne Again Shell (bash) page 981

Use single shell commands and one line command sequences to perform basic tasks on the command line
- Chapter 5: The Shell page 141
- Chapter 7: The Linux Utilities page 215
- Chapter 9: The Bourne Again Shell (bash) page 327

Use and modify the shell environment including defining, referencing, and exporting environment variables
- Parameters and Variables page 352
- Variables page 1031

Use and edit command history
- History page 376

Invoke commands inside and outside the defined path
- Absolute versus relative pathnames page 149
- **PATH**: Where the Shell Looks for Programs page 359

Partial List of Used Files, Terms, and Utilities

. (dot)
- **.** (Dot) or source: Runs a Startup File in the Current Shell page 332
- exec versus **.** (dot) page 1045

bash
- Chapter 5: The Shell page 141
- Chapter 9: The Bourne Again Shell (bash) page 327
- Chapter 27: Programming the Bourne Again Shell (bash) page 981

echo
- echo: Displays Arguments page 219
- **echo –e** page 1009

env
- env: Runs a Program in a Modified Environment page 1035

exec
- Opening a File Descriptor page 1017
- Duplicating a File Descriptor page 1017
- exec: Executes a Command or Redirects File Descriptors page 1045

export
- declare: Lists and Assigns Attributes to Variables page 357
- readonly and export page 357
- export: Puts Variables in the Environment page 1032

pwd
- pwd page 143

set
- **set ±o**: Turns Shell Features On and Off page 400
- set: Initializes Positional Parameters page 1024

unset
- unset: Removes a Variable page 356

man
- man: Displays the System Manual page 128

uname
- uname: Displays System Information page 470

history
- History page 376

103.2 PROCESS TEXT STREAMS USING FILTERS

Send text files and output streams through text utility filters to modify the output using standard UNIX commands found in the GNU **textutils** package

- Redirection page 153
- Pipelines page 158
- Filters page 161

Partial List of Used Files, Terms, and Utilities

cat
- cat: Joins and Displays Files page 216
- cat page 152
- Redirection page 153

cut
- cut: Selects Characters or Fields from Input Lines page 225

expand

fmt

head
- head: Displays the Beginning of a File page 235

od

join

nl

paste

pr

sed

sort
- sort: Sorts and/or Merges Files page 239

split

tail
- tail: Displays the Last Part of a File page 241

tr
- tr page 159
- tr page 260

unexpand

uniq

wc

- wc: Displays the Number of Lines, Words, and Bytes in Files page 244

103.3 PERFORM BASIC FILE MANAGEMENT

Copy, move, and remove files and directories individually

- rmdir: Deletes a Directory page 186
- mv, cp: Move or Copy Files page 187
- mv: Moves a Directory page 188
- rm: Removes a Link page 208
- rm: Removes a File (Deletes a Link) page 222
- cp: Copies Files page 224
- mv: Renames or Moves a File page 237

Copy multiple files and directories recursively

- cp: Copies Files page 224

Remove files and directories recursively

- rm: Removes a File (Deletes a Link) page 222

Use simple and advanced wildcard specifications in commands

- Filename Generation/Pathname Expansion page 165
- Pathname Expansion page 412

Using find to locate and act on files based on type, size, or time

- find: Finds Files Based on Criteria page 229

Usage of tar, cpio, and dd

- tar: Stores or Extracts Files to/from an Archive File page 249
- tar: Archives Files page 603
- cpio: Archives Files page 605

Partial List of Used Files, Terms, and Utilities

cp

- mv, cp: Move or Copy Files page 187
- cp: Copies Files page 224

find

- find: Finds Files Based on Criteria page 229

mkdir

- mkdir: Creates a Directory page 184

mv

- mv: Moves a Directory page 188
- mv: Renames or Moves a File page 237
- mv, cp: Move or Copy Files page 187

ls

- **ls –l**: Displays Permissions page 191
- ls: Displays Information About Files page 221

rm
- rm: Removes a Link page 208
- rm: Removes a File (Deletes a Link) page 222

rmdir
- rmdir: Deletes a Directory page 186

touch
- touch: Changes File Modification and Access Times page 243

tar
- tar: Stores or Extracts Files to/from an Archive File page 249
- tar: Archives Files page 603

cpio
- cpio: Archives Files page 605

dd

file
- file: Displays the Classification of a File page 229

gzip
- xz, bzip2, and gzip: Compress and Decompress Files page 245

gunzip
- unxz bunzip2 gunzip page 248

bzip2
- xz, bzip2, and gzip: Compress and Decompress Files page 245

file globbing
- Filename Generation/Pathname Expansion page 165
- Pathname Expansion page 412

103.4 USE STREAMS, PIPES, AND REDIRECTS

Redirecting standard input, standard output, and standard error
- Redirecting Standard Output page 154
- Redirecting Standard Input page 155
- Redirecting Standard Error page 333
- redirection page 1269 (Glossary)
- standard input page 1274 (Glossary)
- standard output page 1275 (Glossary)
- standard error page 1274 (Glossary)

Pipe the output of one command to the input of another command
- Pipelines page 158
- Filters page 161
- filter page 1250 (Glossary)
- pipeline page 1266 (Glossary)

Use the output of one command as arguments to another command
- xargs: Converts Standard Input to Command Lines page 260

Send output to both stdout and a file
- tee page 162

Partial List of Used Files, Terms, and Utilities

tee

- tee page 162

xargs

- xargs: Converts Standard Input to Command Lines page 260

103.5 CREATE, MONITOR, AND KILL PROCESSES

Run jobs in the foreground and background

- Running a Command in the Background page 163
- Moving a Job from the Foreground to the Background page 164
- Background process page 375
- background process page 1238 (Glossary)
- foreground process page 1250 (Glossary)

Signal a program to continue running after logout

Monitor active processes

- Process Identification page 374
- ps page 374
- ps page 466

Select and sort processes for display

- Process Identification page 374
- ps page 466
- top: Lists Processes Using the Most Resources page 612

Send signals to processes

- Aborting Execution page 123
- kill: Aborting a Background Job page 164
- kill: Sends a Signal to a Process page 465
- killall: Kills a Command page 467
- pkill: Kills a Command page 468
- Signals page 1047

Partial List of Used Files, Terms, and Utilities

&

- Running a Command in the Background page 163
- Background process page 375
- background process page 1238 (Glossary)
- foreground process page 1250 (Glossary)

bg

- Moving a Job from the Foreground to the Background page 164
- bg: Sends a Job to the Background page 348
- background process page 1238 (Glossary)

fg

- Foreground page 163
- Moving a Job from the Foreground to the Background page 164
- fg: Brings a Job to the Foreground page 347
- foreground process page 1250 (Glossary)

jobs
- Determining the number of a job using jobs page 164
- jobs: Lists Jobs page 346

kill
- kill: Aborting a Background Job page 164
- kill: Sends a Signal to a Process page 465

nohup

ps
- Process Identification page 374
- ps page 466

top
- top: Lists Processes Using the Most Resources page 612

free
- free: Displays Memory Usage Information page 253

uptime
- uptime: Displays System Load and Duration Information page 253

killall
- killall: Kills a Command page 467

103.6 Modify Process Execution Priorities

Know the default priority of a job that is created
- Process Identification page 374
- ps page 466
- top: Lists Processes Using the Most Resources page 612

Run a program with higher or lower priority than the default

Change the priority of a running process

Partial List of Used Files, Terms, and Utilities

nice

ps
- Process Identification page 374
- ps page 466

renice

top
- top: Lists Processes Using the Most Resources page 612

103.7 Search Text Files Using Regular Expressions

See Appendix A: Regular Expressions page 1139

Create simple regular expressions containing several notational elements
- Appendix A: Regular Expressions page 1139
- Regular Expressions page 1101 (Python)

Use regular expression tools to perform searches through a filesystem or file content
- *See preceding entry.*

Partial List of Used Files, Terms, and Utilities

grep

- grep: Searches for a Pattern in Files page 232

egrep

- Extended regular expression page 233

fgrep

sed

regex(7)

103.8 PERFORM BASIC FILE EDITING OPERATIONS USING vi

A Practical Guide to Fedora and Red Hat Enterprise Linux, Seventh Edition *covers the vim editor. All commands discussed here are compatible between vi and vim.*

Tutorial: Using vim to Create and Edit a File page 262

Navigate a document using vi

- Moving the Cursor page 268

Use basic vi modes

- Command and Input Modes page 264

Insert, edit, delete, copy, and find text

- Entering Text page 265
- Deleting Text page 268
- Correcting Text page 268

Partial List of Used Files, Terms, and Utilities

vi

- Tutorial: Using vim to Create and Edit a File page 262

/, ?

h, j, k, l

- Moving the Cursor page 268

i, o, a

- Entering Text page 265
- Entering Additional Text page 268

c, d, p, y, dd, yy

- Deleting Text page 268

ZZ, :w!, :q!, :e!

- Ending the Editing Session page 269

104 DEVICES, LINUX FILESYSTEMS, FILESYSTEM HIERARCHY STANDARD

104.1 CREATE PARTITIONS AND FILESYSTEMS

Use various mkfs commands to set up partitions and create various filesystems such as:

ext2

- **ext2** page 519
- **ext2** to **ext3** page 527

ext3
- **ext3** page 519
- **ext3** to **ext2** page 527

xfs
- The XFS Filesystem page 527

reiserfs v3
- **reiserfs** page 520

vfat
- **vfat** page 520

Partial List of Used Files, Terms, and Utilities

fdisk
- fdisk: see the fdisk man page
- See also parted: Reports on and Partitions a Hard Disk page 614

mkfs
- mkfs: Creates a Filesystem page 467

mkswap
- **swap** page 513

104.2 MAINTAIN THE INTEGRITY OF FILESYSTEMS

Verify the integrity of filesystems
- fsck: Checks Filesystem Integrity page 525

Monitor free space and inodes
- df: shows where directory hierarchies are mounted page 804

Repair simple filesystem problems
- fsck: Checks Filesystem Integrity page 525

Partial List of Used Files, Terms, and Utilities

du
- du: Displays Disk Usage Information page 523

df
- df: shows where directory hierarchies are mounted page 804

fsck
- fsck: Checks Filesystem Integrity page 525

e2fsck

mke2fs

debugfs

dumpe2fs

tune2fs
- tune2fs: Changes Filesystem Parameters page 526

xfs tools (such as xfs_metadump and xfs_info)

104.3 CONTROL MOUNTING AND UNMOUNTING OF FILESYSTEMS

Manually mount and unmount filesystems

- mount: Mounts a Filesystem page 520
- umount: Unmounts a Filesystem page 523
- mount: Mounts a Directory Hierarchy page 807
- Mounting Shares page 834

Configure filesystem mounting on bootup

- **fstab**: Keeps Track of Filesystems page 524
- **fstab** file page 807
- **/etc/fstab**: Mounts Directory Hierarchies Automatically page 811

Configure user mountable removable filesystems

- Mount Options page 522

Partial List of Used Files, Terms, and Utilities

/etc/fstab

- **fstab**: Keeps Track of Filesystems page 524
- **fstab** file page 807
- **/etc/fstab**: Mounts Directory Hierarchies Automatically page 811

/media

mount

- mount: Mounts a Filesystem page 520
- mount: Mounts a Directory Hierarchy page 807
- Mounting Shares page 834

umount

- umount: Unmounts a Filesystem page 523

104.4 MANAGE DISK QUOTAS

Set up a disk quota for a filesystem

- Disk Quota System page 629

Edit, check, and generate user quota reports

- quota and repquota page 629

Partial List of Used Files, Terms, and Utilities

quota

- quota and repquota page 629

edquota

- edquota and quotaon page 629

repquota

- quota and repquota page 629

quotaon

- edquota and quotaon page 629

104.5 MANAGE FILE PERMISSIONS AND OWNERSHIP

Manage access permissions on regular and special files as well as directories

- chmod: Changes File Access Permissions page 193

- chmod: Makes a File Executable page 337

Use access modes such as suid, sgid, and the sticky bit to maintain security
- Setuid and Setgid Permissions page 196
- Setuid file page 424
- Setuid files page 626
- setuid page 1272 (Glossary)
- setgid page 1272 (Glossary)

Know how to change the file creation mask
- umask: Specifies the File Permission Mask page 469

Use the group field to grant file access to group members
- **ls –l**: Displays Permissions page 191
- chmod: Changes File Access Permissions page 193
- **/etc/group** page 506

Partial List of Used Files, Terms, and Utilities

chmod
- chmod: Changes File Access Permissions page 193
- chmod: Makes a File Executable page 337

umask
- umask: Specifies the File Permission Mask page 469

chown
- chown: Changes File Ownership page 195

chgrp
- chgrp: Changes File Group Association page 195

104.6 CREATE AND CHANGE HARD AND SYMBOLIC LINKS

Create links
- ln: Creates a Hard Link page 204
- ln: Creates Symbolic Links page 207

Identify hard and/or softlinks
- ls and link counts page 206
- ls and inodes page 206
- hard link page 1252 (Glossary)
- link page 1258 (Glossary)
- symbolic link page 1276 (Glossary)

Copying versus linking files
- cp Versus ln page 205

Use links to support system administration tasks
- ln: Creates a Hard Link page 204
- ln: Creates Symbolic Links page 207

Partial List of Used Files, Terms, and Utilities

ln
- ln: Creates a Hard Link page 204
- ln: Creates Symbolic Links page 207

104.7 FIND SYSTEM FILES AND PLACE FILES IN THE CORRECT LOCATION

Understand the correct locations of files under the FHS
- Important Standard Directories and Files page 189
- Important Files and Directories page 502

Find files and commands on a Linux system
- whereis page 255
- locate: Searches for a File page 256

Know the location and purpose of important files and directories as defined in the FHS
- Important Standard Directories and Files page 189
- Important Files and Directories page 502

Partial List of Used Files, Terms, and Utilities

find
- find: Finds Files Based on Criteria page 229

locate
- locate: Searches for a File page 256

updatedb
- updatedb page 256

whereis
- whereis page 255

which
- which page 255

type
- type: Displays Information About a Command page 1041

/etc/updatedb.conf

CERTIFICATION EXAM 2 OBJECTIVES: LX0-102

105 SHELLS, SCRIPTING, AND DATA MANAGEMENT

105.1 CUSTOMIZE AND USE THE SHELL ENVIRONMENT

See Chapter 9: The Bourne Again Shell (bash) page 327

Set environment variables (e.g., **PATH**) at login or when spawning a new shell
- Startup Files page 329
- Set **PATH** in **.bash_profile** page 331
- Keyword variables page 353
- Keyword Variables page 358

Write bash functions for frequently used sequences of commands
- Functions page 396
- Variables in Functions page 1039

Maintain skeleton directories for new user accounts
- useradd: Adds a User Account page 600

Set command search path with the proper directory
- **PATH**: Where the Shell Looks for Programs page 359

Partial List of Used Files, Terms, and Utilities
/etc/profile
- **/etc/profile** page 330
- **/etc/profile** and **/etc/profile.d** page 509

env
- env: Runs a Program in a Modified Environment page 1035

export
- declare: Lists and Assigns Attributes to Variables page 357
- export: Puts Variables in the Environment page 1032

set
- **set ±o**: Turns Shell Features On and Off page 400
- set: Initializes Positional Parameters page 1024

unset
- unset: Removes a Variable page 356

~/.bash_profile
- **.bash_profile**, **.bash_login**, and **.profile** page 330
- **~/.bash_profile** page 502

~/.bash_login
- **.bash_profile**, **.bash_login**, and **.profile** page 330

~/.profile
- **.bash_profile**, **.bash_login**, and **.profile** page 330

~/.bashrc
- **.bashrc** page 331
- **~/.bashrc** page 502

~/.bash_logout
- **.bash_logout** page 330

functions
- Functions page 396
- Variables in Functions page 1039

alias
- Aliases page 392
- Alias Substitution page 404

lists
- Lists page 162

105.2 Customize or Write Simple Scripts

See Chapter 9: The Bourne Again Shell (bash) page 327

See Chapter 27: Programming the Bourne Again Shell (bash) page 981

Use standard sh syntax (loops, tests)
- Control Structures page 982

Use command substitution
- Command Substitution page 410

Test return values for success or failure or other information provided by a command
- test builtin page 983
- [] is a synonym for test page 986
- test builtin page 1000

Perform conditional mailing to the superuser

Correctly select the script interpreter through the shebang (**#!**) line
- **#!** Specifies a Shell page 338

Manage the location, ownership, execution, and suid-rights of scripts
- Listing setuid files page 458

Partial List of Used Files, Terms, and Utilities

for
- **for...in** page 995
- **for** page 997

while
- **while** page 999

test
- test builtin page 983
- [] is a synonym for test page 986
- test builtin page 1000

if
- **if...then** page 983
- **if...then...else** page 987
- **if...then...elif** page 989

read
- read: Accepts User Input page 1041

seq
- seq page 407

105.3 SQL DATA MANAGEMENT

See Chapter 29: The MariaDB SQL Database Management System page 1113

Use of basic SQL commands
- Notes page 1114

Perform basic data manipulation
- Examples page 1123

Partial List of Used Files, Terms, and Utilities

insert
- INSERT INTO page 1125

update
- UPDATE page 1128

select
- Retrieving Data page 1126
- Joins page 1130

delete
- DELETE FROM page 1128

from
- DELETE FROM page 1128

where
- WHERE page 1127

group by

order by
- ORDER BY page 1126

join
- Joins page 1130

106 User Interfaces and Desktops

106.1 Install and Configure X11

See X Window System page 459

Verify that the video card and monitor are supported by an X server
- Displays page 109

Awareness of the X font server

Basic understanding and knowledge of the X Window configuration file

Partial List of Used Files, Terms, and Utilities

/etc/X11/xorg.conf

xhost
- xhost Grants Access to a Display page 461

DISPLAY
- The **DISPLAY** Variable page 462

xwininfo

xdpyinfo

X
- X Window System page 459

106.2 SET UP A DISPLAY MANAGER

Turn the display manager on or off
Change the display manager greeting
Change default color depth for the display manager
Configure display managers for use by X-stations

Partial List of Used Files, Terms, and Utilities
/etc/inittab
- **/etc/inittab** page 508

xdm configuration files
kdm configuration files
gdm configuration files
- Graphical login page 453
- The **Xorg –nolisten tcp** Option page 460

106.3 ACCESSIBILITY

Keyboard Accessibility Settings (AccessX?)
Visual Settings and Themes
Assistive Technology (ATs)

Partial List of Used Files, Terms, and Utilities
Sticky/Repeat Keys
Slow/Bounce/Toggle Keys
Mouse Keys
High Contrast/Large Print Desktop Themes
Screen Reader
Braille Display
Screen Magnifier
On-Screen Keyboard
Gestures (used at login, for example gdm)
Orca
GOK
emacspeak

107 ADMINISTRATIVE TASKS

107.1 MANAGE USER AND GROUP ACCOUNTS AND RELATED SYSTEM FILES

Add, modify, and remove users and groups
- system-config-users: Manages User Accounts page 598
- Managing User Accounts from the Command Line page 600

Manage user/group info in password/group databases
- Modifying a User page 600
- Working with Groups page 600

- usermod: Modifies a User Account page 601
- groupdel and groupmod: Remove and Modify a Group page 601
- chage page 601

Create and manage special purpose and limited accounts

Partial List of Used Files, Terms, and Utilities

/etc/passwd
- **/etc/passwd** page 508

/etc/shadow
- **/etc/shadow** page 511

/etc/group
- **/etc/group** page 506

/etc/skel
- **/etc/skel** page 600

chage
- chage page 601

groupadd
- groupadd: Adds a Group page 601

groupdel
- groupdel and groupmod: Remove and Modify a Group page 601

groupmod
- groupdel and groupmod: Remove and Modify a Group page 601

passwd
- Users: Changing Your Account Type and Password (GUI) page 112
- passwd: Changing Your Password (CLI) page 137

useradd
- useradd: Adds a User Account page 600

userdel
- userdel: Removes a User Account page 600

usermod
- usermod: Modifies a User Account page 601

107.2 AUTOMATE SYSTEM ADMINISTRATION TASKS BY SCHEDULING JOBS

Manage **cron** and at jobs
- **crond** and anacron: Schedule Routine Tasks page 607
- at: Runs Occasional Tasks page 611

Configure user access to **cron** and at services
- **/etc/at.allow**, **/etc/at.deny**, **/etc/cron.allow**, and **/etc/cron.deny** page 506

Partial List of Used Files, Terms, and Utilities

/etc/cron.{d,daily,hourly,monthly,weekly}
- Crontab Files page 607

/etc/at.deny
- **/etc/at.allow**, **/etc/at.deny**, **/etc/cron.allow**, and **/etc/cron.deny** page 506

/etc/at.allow
- **/etc/at.allow**, **/etc/at.deny**, **/etc/cron.allow**, and **/etc/cron.deny** page 506

/etc/crontab
- **/etc/crontab** page 608

/etc/cron.allow
- **/etc/at.allow**, **/etc/at.deny**, **/etc/cron.allow**, and **/etc/cron.deny** page 506

/etc/cron.deny
- **/etc/at.allow**, **/etc/at.deny**, **/etc/cron.allow**, and **/etc/cron.deny** page 506

/var/spool/cron/*
- Crontab Files page 607

crontab
- User crontab files page 608
- **crond** and anacron: Schedule Routine Tasks page 607

at
- at: Runs Occasional Tasks page 611

atq

atrm

107.3 LOCALIZATION AND INTERNATIONALIZATION

Locale settings
- Locale page 368
- locale page 1258 (Glossary)

Time zone settings
- tzconfig page 372
- tzselect page 372
- **/etc/timezone** page 372

Partial List of Used Files, Terms, and Utilities

/etc/timezone
- **/etc/timezone** page 372

/etc/localtime
- **/etc/localtime** page 373

/usr/share/zoneinfo
- **/usr/share/zoneinfo** page 372

Environment variables:
- **LC_**: Locale Variables page 368
- Environment Variables page 1032

/usr/bin/locale
- locale: Displays Locale Information page 369

tzselect
- tzselect page 372

tzconfig
- tzconfig page 372

date
- date: Displays the System Time and Date page 218

iconv

UTF-8
- **LC_**: Locale Variables page 368
- UTF-8 page 1279 (Glossary)

ISO-8859
- **LC_**: Locale Variables page 368

ASCII
- ASCII page 1237 (Glossary)

Unicode
- Unicode page 1279 (Glossary)

108 ESSENTIAL SYSTEM SERVICES

108.1 MAINTAIN SYSTEM TIME

Set the system date and time
- timedatectl: Reports on and Sets the System Clock page 613

Set the hardware clock to the correct time in UTC

Configure the correct time zone
- Time page 371

Basic NTP configuration

Knowledge of using the pool.ntp.org service

Partial List of Used Files, Terms, and Utilities

/usr/share/zoneinfo
- **/usr/share/zoneinfo** page 372

/etc/timezone
- **/etc/timezone** page 372

/etc/localtime
- **/etc/localtime** page 373

/etc/ntp.conf

date
- date: Displays the System Time and Date page 218
- timedatectl: Reports on and Sets the System Clock page 613

hwclock

ntpd

ntpdate

pool.ntp.org

108.2 SYSTEM LOGGING

Syslog configuration files
- **rsyslog.conf** page 620

syslog
- **rsyslogd:** Logs System Messages page 620

standard facilities, priorities, and actions
- Selectors page 620
- Facilities page 620
- Priorities page 620
- Actions page 621

Partial List of Used Files, Terms, and Utilities
syslog.conf
- **rsyslog.conf** page 620

syslogd
- **rsyslogd:** Logs System Messages page 620

klogd

logger

108.3 MAIL TRANSFER AGENT (MTA) BASICS

See Chapter 20: sendmail: Setting Up Mail Servers, Clients, and More page 739

Create e-mail aliases
- **/etc/aliases** page 746

Configure e-mail forwarding
- **~/.forward** page 747

Knowledge of commonly available MTA programs (Postfix, **sendmail**, Qmail, **exim**) (no configuration)
- Alternatives to sendmail page 741

Partial List of Used Files, Terms, and Utilities
~/.forward
- **~/.forward** page 747

sendmail emulation layer commands

newaliases
- newaliases page 747

mail

mailq
- mailq page 748

Postfix
- Postfix page 742

sendmail
- Introduction to **sendmail** page 740
- Setting Up a **sendmail** Mail Server page 742
- JumpStart I: Configuring **sendmail** on a Client page 743
- JumpStart II: Configuring **sendmail** on a Server page 744
- Working with **sendmail** Messages page 745
- Configuring **sendmail** page 748

exim
- **exim4** page 742

qmail
- Qmail page 742

108.4 MANAGE PRINTERS AND PRINTING

See Chapter 13: Printing with CUPS page 555

Basic CUPS configuration (for local and remote printers)
- The System Configures a Local Printer Automatically page 558
- JumpStart I: Configuring a Printer Using system-config-printer page 558
- JumpStart II: Setting Up a Local or Remote Printer page 560
- Working with the CUPS Web Interface page 565
- Configuring Printers page 566

Manage user print queues
- Managing Print Queues page 572

Troubleshoot general printing problems

Add and remove jobs from configured printer queues
- BSD and System V command-line print utilities page 574

Partial List of Used Files, Terms, and Utilities

CUPS configuration files, tools, and utilities
- JumpStart I: Configuring a Printer Using system-config-printer page 558
- Working with the CUPS Web Interface page 565
- Sharing CUPS Printers page 572

/etc/cups
- Example lpadmin Commands page 570

lpd legacy interface (lpr, lprm, lpq)
- Traditional UNIX Printing page 573

109 NETWORKING FUNDAMENTALS

See Chapter 8: Networking and the Internet page 279

109.1 FUNDAMENTALS OF INTERNET PROTOCOLS

See Network Protocols page 290

Demonstrate an understanding of network masks
- Subnet mask page 298
- network mask page 1263 (Glossary)

Knowledge of the differences between private and public "dotted quad" IP Addresses
- Private address space page 637
- private address space page 1267 (Glossary)

Setting a default route

Knowledge about common TCP and UDP ports (20, 21, 22, 23, 25, 53, 80, 110, 119, 139, 143, 161, 443, 465, 993, 995)

- Each chapter covering a server discusses which ports that server uses.
- Ports page 312
- port page 1266 (Glossary)

Knowledge about the differences and major features of UDP, TCP, and ICMP

- UDP page 290
- UDP: User Datagram Protocol page 292
- TCP page 290
- TCP: Transmission Control Protocol page 291
- ping: Tests a Network Connection page 305
- UDP page 1278 (Glossary)
- TCP page 1276 (Glossary)
- ICMP page 1254 (Glossary)

Knowledge of the major differences between IPv4 and IPv6

- IPv4 page 292
- IPv6 page 293

Partial List of Used Files, Terms, and Utilities

/etc/services

- Network Services page 313
- **/etc/services** page 511

ftp

- Chapter 19: FTP: Transferring Files Across a Network page 713

telnet

- telnet: Logs In on a Remote System page 303

host

- host and dig: Query Internet Nameservers page 307

ping

- ping: Tests a Network Connection page 305

dig

- host and dig: Query Internet Nameservers page 307
- dig page 861
- dig page 862

traceroute

- traceroute: Traces a Route over the Internet page 306

tracepath

109.2 BASIC NETWORK CONFIGURATION

Manually and automatically configure network interfaces

- Configuring the Systems page 636
- NetworkManager: Configures Network Connections page 637

Basic TCP/IP host configuration

Partial List of Used Files, Terms, and Utilities

/etc/hostname

- **/etc/hostname** page 507

/etc/hosts

- Hostnames page 300
- **/etc/hosts** page 507

/etc/resolv.conf

- **/etc/resolv.conf** page 510

/etc/nsswitch.conf

- **nsswitch.conf**: Which Service to Look at First page 495

ifconfig

ifup

ifdown

route

ping

- ping: Tests a Network Connection page 305

109.3 BASIC NETWORK TROUBLESHOOTING

Manually and automatically configure network interfaces and routing tables to include adding, starting, stopping, restarting, deleting, or reconfiguring network interfaces

Change, view, or configure the routing table and correct an improperly set default route manually

Debug problems associated with the network configuration

Partial List of Used Files, Terms, and Utilities

ifconfig

ifup

ifdown

route

host

- host and dig: Query Internet Nameservers page 307

hostname

- hostname: Displays the System Name page 219
- **/etc/sysconfig/network** page 511

dig

- host and dig: Query Internet Nameservers page 307
- dig page 861
- dig page 862

netstat

ping

- ping: Tests a Network Connection page 305

traceroute

- traceroute: Traces a Route over the Internet page 306

109.4 CONFIGURE CLIENT SIDE DNS

See Chapter 24: DNS/BIND: Tracking Domain Names and Addresses page 851

Demonstrate the use of DNS on the local system

- JumpStart I: Setting Up a DNS Cache page 866

Modify the order in which name resolution is done

- Resolver page 854

Partial List of Used Files, Terms, and Utilities

/etc/hosts

- Hostnames page 300
- **/etc/hosts** page 507

/etc/resolv.conf

- **/etc/resolv.conf** page 510

/etc/nsswitch.conf

- **nsswitch.conf**: Which Service to Look at First page 495

110 SECURITY

110.1 PERFORM SECURITY ADMINISTRATION TASKS

Audit a system to find files with the suid/sgid bit set

- Listing setuid files page 458
- Listing setgid files page 459

Set or change user passwords and password aging information

- Users: Changing Your Account Type and Password (GUI) page 112
- passwd: Changing Your Password (CLI) page 137
- Modifying a User page 600
- chage page 601

Being able to use nmap and netstat to discover open ports on a system

Set up limits on user logins, processes, and memory usage

Basic sudo configuration and usage

- Using sudo to Gain **root** Privileges page 428

Partial List of Used Files, Terms, and Utilities

find

- find: Finds Files Based on Criteria page 229

passwd

- Users: Changing Your Account Type and Password (GUI) page 112
- passwd: Changing Your Password (CLI) page 137

lsof

- lsof: Finds Open Files page 624

nmap

chage

- chage page 601

netstat

sudo

- Using sudo to Gain **root** Privileges page 428

/etc/sudoers

- **sudoers**: Configuring sudo page 433

su

- Using su to Gain **root** Privileges page 425

usermod

- usermod: Modifies a User Account page 601

ulimit

110.2 SET UP HOST SECURITY

Awareness of shadow passwords and how they work

- **/etc/shadow** page 511

Turn off network services not in use

- Configuring Daemons (Services) page 445
- service page 443
- chkconfig and service page 445
- system-config-services: Configures Services page 447

Understand the role of TCP wrappers

- TCP Wrappers: Secure a Server (**hosts.allow** and **hosts.deny**) page 485

Partial List of Used Files, Terms, and Utilities

/etc/nologin

- Going to Single-User Mode page 454

/etc/passwd

- **/etc/passwd** page 508

/etc/shadow

- **/etc/shadow** page 511

/etc/xinetd.d/* [deprecated]

/etc/xinetd.conf [deprecated]

/etc/inetd.d/* [deprecated]

/etc/inetd.conf [deprecated]

/etc/inittab

- **/etc/inittab** page 508

/etc/init.d/*

- SysVinit (**rc**) Scripts: Start and Stop System Services page 448

/etc/hosts.allow

- **hosts.allow** and **hosts.deny** page 486

/etc/hosts.deny
- **hosts.allow** and **hosts.deny** page 486

110.3 SECURING DATA WITH ENCRYPTION

Perform basic OpenSSH 2 client configuration and usage
- Configuring OpenSSH Clients page 690
- Running the ssh, scp, and sftp OpenSSH Clients page 689

Understand the role of OpenSSH 2 server host keys
- How OpenSSH Works page 687
- Authorized Keys: Automatic Login page 700

Perform basic GnuPG configuration and usage
- Tutorial: Using GPG to Secure a File page 1169

Understand SSH port tunnels (including X11 tunnels)
- Tunneling/Port Forwarding page 707

Partial List of Used Files, Terms, and Utilities

ssh
- ssh: Logs in or Executes Commands on a Remote System page 693

ssh-keygen
- ssh-keygen page 701

ssh-agent
- ssh-agent: Holds Your Private Keys page 703

ssh-add
- ssh-add page 703

~/.ssh/id_rsa and **id_rsa.pub**
- **id_rsa id_rsa.pub** page 689
- **id_rsa** and **id_rsa.pub** page 701

~/.ssh/id_dsa and **id_dsa.pub**
- **id_dsa id_dsa.pub** page 689

/etc/ssh/ssh_host_rsa_key and **ssh_host_rsa_key.pub**
- **ssh_host_rsa_key, ssh_host_rsa_key.pub** page 688

/etc/ssh/ssh_host_dsa_key and **ssh_host_dsa_key.pub**
- **ssh_host_dsa_key, ssh_host_dsa_key.pub** page 688

~/.ssh/authorized_keys
- **authorized_keys** page 688

/etc/ssh_known_hosts
- **ssh_known_hosts** page 692

gpg
- GnuPG/PGP page 1160
- Tutorial: Using GPG to Secure a File page 1169

~/.gnupg/*
- **~/.gnupg** page 1170