



4

Basic Configuration and Administration

CERTIFICATION OBJECTIVES

- | | | | |
|------|---|------|---------------------------|
| 4.01 | Adding, Deleting, and Modifying User Accounts | 4.06 | RPM Sources |
| 4.02 | The Basic User Environment | 4.07 | Network Configuration |
| 4.03 | Red Hat Filesystem Basics | 4.08 | The Basic Boot Process |
| 4.04 | Filesystem Management and the Automounter | 4.09 | Virtual Consoles |
| 4.05 | The Red Hat Package Manager Command and Utility | 4.10 | Other Configuration Tools |
| | | ✓ | Two-Minute Drill |
| | | Q&A | Self Test |

After installation is complete, you still have work to do. To customize the system to meet your needs, user accounts need to be set up, filesystems must be configured, and additional packages may need to be added or removed.

This chapter will get you started with the basics that every RHCE (and many RHCT) level administrator should know. At the end of this chapter, you should know how to manage user accounts and environments, configure and mount filesystems, use RPM to manage packages, manage system daemons, and configure virtual consoles, keyboards, and mice.

CERTIFICATION OBJECTIVE 4.01

Adding, Deleting, and Modifying User Accounts

The default Red Hat installation gives you just a single login account: root. You should set up some regular user accounts. Even if you're going to be the only user on the

INSIDE THE EXAM

Administrative Skills

When you take the Red Hat exams, as long as you don't cheat, it generally does not matter how you come to a solution. For example, you get the same credit whether you add users by directly editing `/etc/passwd`, using commands such as `useradd`, or using GUI tools such as the Red Hat User Manager (**redhat-config-users**). As in the real world, it is the result that matters.

When available, I present more than one method of configuring your Linux computer. Unless otherwise directed on the exam, the method you select is up to you. In general, I

believe it is better and faster to configure Linux directly from the command line interface. But if you get nervous on an exam and forget how to use `chkconfig` to enable a service at runlevels 3 and 5, it's okay to use Red Hat's GUI Service Configuration Utility. GUI tools just take more time to start, to manage, and to execute.

Many of the skills in this chapter apply only to the RHCE. If you're using this book to study for the RHCT, consult the syllabus for the Red Hat RH133 course at www.redhat.com for more information.

system, it's a good idea to create at least one nonadministrative account to do your day-to-day work. Then you can use the root account only when it's necessary to administer the system. Accounts can be added to Red Hat Enterprise Linux systems using various utilities, including the vi text editor (the manual method), the **useradd** command (the command line method), and the Red Hat User Manager utility (the graphical method).

exam

Watch

As discussed earlier, it's faster to log in as root (and not just the superuser). While you'll be doing most of the work on the Red Hat exams as root, it's quite possible that you'll be asked to create

accounts for regular users (and groups) to configure a workstation. As RHCT requirements are part of the RHCE exam, this applies to both exams.

User Account Categories

There are three basic types of Linux user accounts: administrative (root), regular, and service. The administrative root account is automatically created when you install Linux, which has administrative privileges for all services on your Linux computer. A cracker who has a chance to take control of this account can take full control of your system.

Nevertheless, there are times when it is appropriate to log in as an administrator (that is, as the root user), such as during most of the Red Hat exams. Red Hat Enterprise Linux builds in safeguards for root users. Log in as the root user, and then run the **alias** command. You'll see entries such as:

```
alias rm='rm -i'
```

which prompt for confirmation before the **rm** command deletes a file. Unfortunately, a command such as **rm -rf directoryname** would supersede this safety setting.

Regular users have the necessary privileges to perform standard tasks on a Linux computer. They can access programs such as word processors, databases, and Web browsers. They can store files in their own home directories. Since regular users do not normally have administrative privileges, they cannot accidentally delete critical operating system configuration files. You can assign a regular account to most users, safe in the knowledge that they can't disrupt your system with the privileges they have on that account.

Services such as Apache, Squid, mail, games, and printing have their own individual service accounts. These accounts exist to allow each of these services

4 Chapter 4: Basic Configuration and Administration

to interact with your computer. Normally, you won't need to change any service account, but if you see that someone has logged in through one of these accounts, be wary. Someone may have broken into your system.

on the
Joe

To review recent logins, run the `utmpdump /var/log/wtmp | less` command. If the login is from a remote location, it will be associated with a specific IP address outside your network.

Basic Command Line Tools

There are two basic ways you can add users through the command line interface. You can add users directly by editing the `/etc/passwd` file in a text editor such as `vi`. Alternatively, you can use text commands customized for the purpose.

exam

Watch

You'll probably need to know how to use the `vi` editor during the Troubleshooting and System Maintenance exam. While the `joe` text editor also works

in linux rescue mode, `vi` is more of a standard. I therefore focus on the `vi` editor throughout this book.

Adding Users Directly

Open the `/etc/passwd` file in the text editor of your choice. If you've added regular users to your system, you'll normally see them listed at the bottom of this file. Scroll around this file, and you should see a series of lines like the following:

```
mj:x:500:500:Michael Jang:/home/mj:/bin/bash
```

Each column in `/etc/passwd`, delineated by a colon, has a purpose, which I describe in Table 4-1.

You can create a new user in `/etc/passwd` by copying and then editing a line associated with an existing user. Just substitute the information of your choice to create the new user. Make sure that you at least assign a new username and user ID.

Assuming you've assigned a new home directory for your new user, you have to remember to create that directory. For example, if your new user is `ez`, you'll probably want to create the `/home/ez` directory. You'll also need to make sure that `ez` has ownership permissions on that directory and all of the files that you're going to put in that directory.

TABLE 4-1 The Anatomy of `/etc/passwd`

Field	Example	Purpose
Username	mj	The user logs in with this name. Usernames can include hyphens (-) or underscores (_). However, they should not start with a number or include uppercase letters. RHEL limits usernames to 32 characters.
Password	x	The password. You should see either an “x,” an asterisk (*), or a seemingly random group of letters and numbers. An “x” points to <code>/etc/shadow</code> for the actual password. An asterisk means the account is disabled. A random group of letters and numbers represents the encrypted password.
User ID	500	The unique numeric user ID (UID) for that user. By default, Red Hat starts user IDs at 500.
Group ID	500	The numeric group ID (GID) associated with that user. By default, Red Hat Enterprise Linux creates a new group for every new user. Some other Linux and Unix systems assign all users to the default Users group (GID=100).
User info	Michael Jang	You can enter any information of your choice in this field. Standard options include the user’s full name, telephone number, e-mail address, or physical location. You can leave this blank.
Home Directory	/home/mj	By default, RHEL places new home directories in <code>/home/username</code> .
Login Shell	/bin/bash	By default, RHEL assigns users to the bash shell. You can change this to any legal shell that you have installed.

You can then populate that new user’s home directory. A default environment is available in the `/etc/skel` directory. You can copy the contents of that directory to the new user’s home directory. The `/etc/skel` directory is covered in more detail later in this chapter.

Adding Users to a Group

Every Linux user is assigned to a group. By default in RHEL 3, every user gets their own private group. By default, they are the only member of that group, as defined in the `/etc/group` configuration file. Open that file in the editor of your choice. You should see lines similar to the following:

```
mj:x:500:
vp:x:501:
managers:x:1000:mj, vp
```

TABLE 4-2 The Anatomy of /etc/group

Field	Example	Purpose
Groupname	mj	Each user gets their own group, with the same name as their username. You can also create unique groupnames.
Password	x	The password. You should see either an “x” or a seemingly random group of letters and numbers. An “x” points to /etc/gshadow for the actual password. A random group of letters and numbers represents the encrypted password.
Group ID	500	The numeric group ID (GID) associated with that user. By default, Red Hat Enterprise Linux creates a new group for every new user.
Group members	mj,vp	Lists the usernames that are members of the group. If blank, and there is a username that is identical to the groupname, that user is the only member of that group.

The contents are straightforward. The users `mj` and `vp` are members of their own groups, as well as the managers group. The four columns in each `/etc/group` line are described in Table 4-2.

Adding Users at the Command Line

Alternatively, you can automate this process with the `useradd` command. If you wanted to add a new user named `pm`, you could just type `useradd pm` to add this user to the `/etc/passwd` file. By default, it creates a home directory, `/home/pm`, adds the standard files from the `/etc/skel` directory, and assigns the default shell, `/bin/bash`. But `useradd` is versatile. It includes a number of command options, shown in Table 4-3.

TABLE 4-3 useradd Command Options

Option	Purpose
<code>-u UID</code>	Overrides the default assigned UID. By default, in RHEL this starts at 500 and can continue sequentially to 65536.
<code>-g GID</code>	Overrides the default assigned GID. By default, RHEL the same GID and UID numbers to each user. The GID you assign must be either 100 (users) or already otherwise exist.
<code>-c info</code>	Enters the comment of your choice about the user, such as his or her name.
<code>-d dir</code>	Overrides the default home directory for the user, <code>/home/username</code> .
<code>-s shell</code>	Overrides the default shell for the user, <code>/bin/bash</code> .

Assigning a Password

You can now use the `passwd username` command to assign a new password to that user. For example, the `passwd pm` command lets you assign a new password to user pm. You're prompted to enter a password twice. RHEL is configured to discourage passwords that are based on dictionary words or are shorter than six characters for security reasons. Nevertheless, such passwords are legal, and such a password is accepted by the `passwd` command when you type it in a second time.



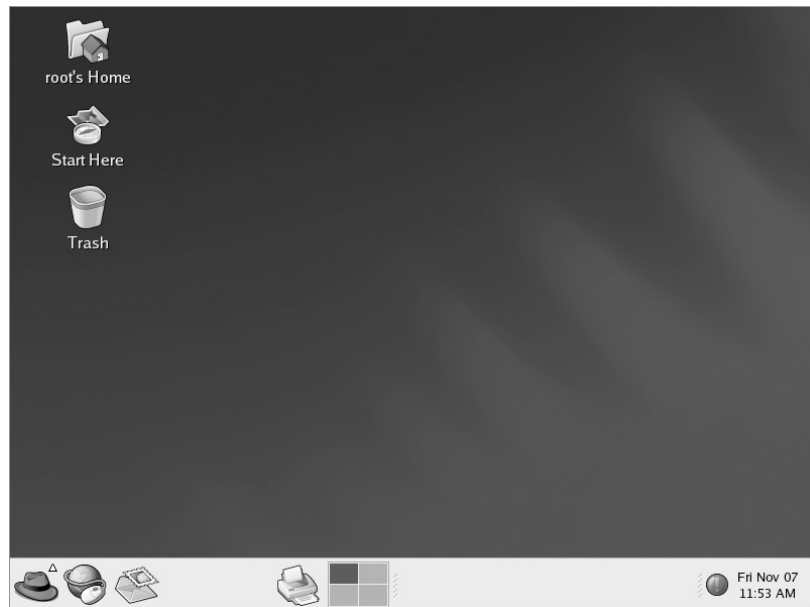
Good passwords are important. Any cracker who may have tapped into your network can try to match the password of any of your users. A password cracking program may be able to find dictionary word passwords in a matter of minutes. In contrast, it may take hours to crack a more complex password such as lla45lms (which could stand for “I live at 45l Main Street”).

The Red Hat User Manager

As of this writing, the Red Hat User Manager can be run only from the Linux graphical user interface (GUI). If you're not in a GUI, and have installed the needed software, the `startx` command moves you into the GUI, as shown in Figure 4-1.

FIGURE 4-1

The Red Hat
GNOME GUI



Pay attention to the figure of the Red Hat in the lower-left corner of the desktop; it starts program menus like the GNOME foot, the KDE K, or the Microsoft Windows Start buttons. Red Hat refers to it as the “Main Menu” button, which is the term I’ll use throughout this book.

on the
book

Red Hat has text-based versions of several of its graphical tools. I believe that they will develop text-based versions of GUI tools such as the Red Hat User Manager (`redhat-config-users`) sometime in the near future. But that does not help you on the current Red Hat exams.

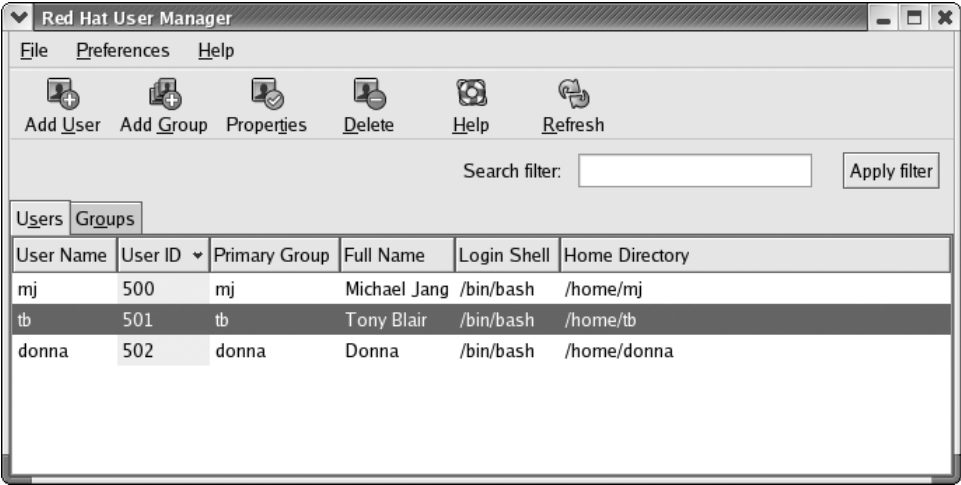
If you’re already in a Linux GUI, you can start the Red Hat User Manager in one of two ways. Enter `redhat-config-users` from a command line interface in the GUI, or click Main Menu | System Settings | Users and Groups. (This notation indicates you should click the Main Menu button—the “Red Hat”—in the lower-left corner of the Taskbar, then click System Settings followed by Users in the menus that appear.) Figure 4-2 shows the Red Hat User Manager window.

on the
book

If you have not logged in as the root user, you’ll be prompted for a root password before opening administrative tools such as the Red Hat User Manager.

FIGURE 4-2

The Red Hat User Manager



EXERCISE 4-1

Adding a User with the Red Hat User Manager

To add a user with the Red Hat User Manager:

1. Start the Red Hat User Manager. You can do this in one of two ways. Right-click on the desktop, and click New Terminal in the pop-up menu that appears. Type **redhat-config-users** in the command line terminal. Alternatively, click Main Menu | System Settings | Users and Groups.
2. In the Red Hat User Manager, click the Add User button, or click File | Add User. This will open the Create New User window, as shown here:

III 4-1

The screenshot shows the 'Create New User' window with the following details:

- User Name:** vp
- Full Name:** Vladimir Putin
- Password:** [masked]
- Confirm Password:** [masked]
- Login Shell:** /bin/bash
- ☒ **Create home directory**
- Home Directory:** /home/vp
- ☒ **Create a private group for the user**
- ☐ **Specify user ID manually**
- UID:** 500
- Buttons:** Cancel, OK

3. Complete the form. All entries are required, except Full Name. The entries are fairly self-explanatory; see the previous sections for more information on each field. The password should be at least six characters and should contain a mix of upper- and lowercase letters, numbers, and symbols to keep it more secure from the standard password cracking programs.

4. Enter the identical password in the Confirm Password field.
 5. Click OK when you are done.
 6. Repeat the process as desired for any additional new users you're expected to support.
-

User Account Management Tips

Although creating user accounts may seem to be a straightforward process, there are a few things that you should consider when configuring new users:

- By default, RHEL 3 configures individual private group IDs (GID) for each user. As this associates each user with their own exclusive group, this is also known as the Red Hat User Private Group scheme. In the default Red Hat scenario, everyone has a unique private GID (which corresponds to their UID), and regular users do not have access to other users' home directories. These users can still share access to special directories; see Chapter 11 for more information.
- If your configuration doesn't require each user to have his or her own GID, or if you have no need for the security associated with the User Private Group scheme, you can assign all of your users to the Users group, with a GID of 100. In many cases, this can make system administration easier.
- Discourage the use of shared accounts, where several people use a single account. Shared accounts are almost always unnecessary and are easily compromised.
- If you'll be using the Network File System (NFS), make sure all users have the same UID on every system on the network. This can be problematic. One alternative is the Network Information System (NIS). NIS supports a centralized database of users and passwords for users on all participating computers. For more information on NIS, see Chapter 9.



The Red Hat standard where every user is a member of his or her own exclusive group is known as the User Private Group scheme, which I describe in more detail in Chapter 11.

Deleting a User Account

Removing user accounts is a pretty straightforward process. The easiest way to delete a user account is with the **userdel** command. By default, this command does not delete that user's home directory. Alternatively, the **userdel -r username** command deletes that user's home directory along with all of the files stored in that home directory.

exam

Watch

If you know both the text and GUI tools to perform a task, use the text method. It almost always saves time.

This is a lot faster than the GUI method, where you start the X Window System, open the Red Hat User Manager, select the user, and then click Delete. While it's probably easier for a less experienced user to remember the GUI method, text commands are faster.

Modifying a User Account

As a Linux administrator, you may want to add some limitations to your user accounts. The easiest way to illustrate some of what you can do is through the Red Hat User Manager. Start the Red Hat User Manager, select a currently configured user, and then click Properties to open the User Properties dialog box.

Click the Account Info tab for the account expiration information shown in Figure 4-3. As shown in the figure, you can limit the life of an account so that it expires on a specific date, or you can disable an account by locking it. Click the Password Info tab.

As shown in Figure 4-4, you can set several characteristics related to an individual user's password. Even when you set good passwords, frequent password changes can

FIGURE 4-3

Managing user account life

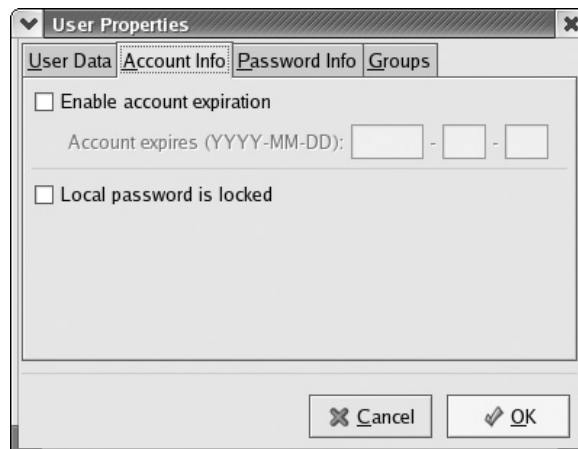
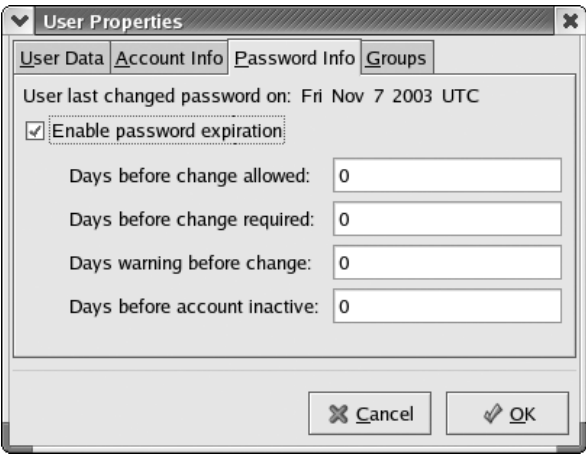


FIGURE 4-4

Configuring password information

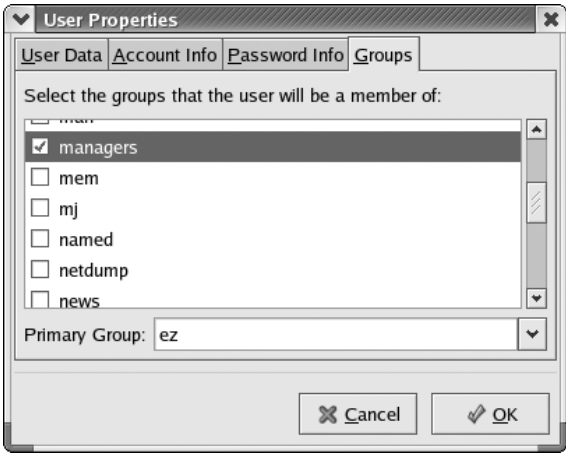


help provide additional security. The categories shown in the figure are self-explanatory; a 0 in any of these text boxes means that the limit does not apply.

Click the Groups tab. Users can belong to more than one group in Linux. Under the Groups properties tab shown in Figure 4-5, you can assign the subject user to other groups. For example, if you want to collect the files supporting the managers in your company together in a directory, you can give access to these files to the group named managers. Alternatively, you can then assign members of that project team to the project group through the Groups tab.

FIGURE 4-5

Assigning groups



exam**Watch**

You may not have access to a GUI during the troubleshooting portion of either Red Hat exam. Therefore, you need to know how to manage users independent of GUI tools such as the Red Hat User Manager. In any case, text-based tools are almost always faster.

Regular User Management Commands

While the Red Hat User Manager GUI utility is convenient, it's often faster to perform the associated administrative functions at the command line interface. I've described some of these commands such as **useradd** and **userdel**. Two other key user administration commands are **usermod** and **chage**.

usermod

The **usermod** command modifies various settings in `/etc/passwd`. It also allows you to set an expiration date for an account or an additional group. For example, the following command sets the account associated with user `test1` to expire on June 8, 2004:

```
# usermod -e 2004-06-08 test1
```

The following command makes user `test1` a member of the special group:

```
# usermod -G special test1
```

chage

You can use the **chage** command to manage the expiration date of a password. Password aging information is stored in the `/etc/shadow` file. In order, the columns in `/etc/shadow` are shown in Table 4-4. The associated **chage** switch is shown with each description entry. For example, if you wanted to require that user `test1` keep a password for at least two days, use the **chage test1 -m 2** command.

Limiting Access to su

As described earlier, I've recommended that you use the root account during the Red Hat exams. However, I assume you still need to configure Linux computers in real life. One security measure that can help is to limit regular users who can access the root account via the **su** command. This takes two basic steps.

TABLE 4-4 The Anatomy of /etc/shadow

Column	Field	Description
1	Username	Username
2	Password	Encrypted password; requires an “x” in the second column of /etc/passwd
3	Password history	Date of the last password change, in number of days after January 1, 1970
4	mindays	Minimum number of days which you must keep a password (-m)
5	maxdays	Maximum number of days after which a password has to be changed (-M)
6	warndays	Number of days before password expiration - when a warning is given (-W)
7	inactive	Number of days after password expiration when an account is made inactive (-I)
8	disabled	Number of days after password expiration when an account is disabled (-E)

First, you’ll need to add the users who you want to allow access to the **su** command. Make them a part of the wheel group. By default, this line in /etc/group looks like:

```
wheel:x:10:root
```

You can add the users of your choice to the end of this line directly, with the **usermod -G wheel username** command, or with the Red Hat User Manager.

Next, you’ll need to make your Pluggable Authentication Modules look for this group. You can do so by activating the following command in your /etc/pam.d/su file:

```
# auth required /lib/security/$ISA/pam_wheel.so use_uid
```

CERTIFICATION OBJECTIVE 4.02

The Basic User Environment

Each user on your Red Hat Enterprise Linux system has an *environment* when logged on to the system. The environment defines directories where Linux looks for programs to run, the look of the login prompt, the terminal type, and more. This section explains how you can configure the default environment for your users.

Home Directories and /etc/skel

By default, when you create a new user in RHEL 3, they get a default set of configuration files in their home directories. These defaults are hidden files stored in the /etc/skel directory.

Home Directories

The home directory is where users start when they first log on to a RHEL system. The home directory for most users is /home/*username*, where *username* is the user's login name. Users should normally have write permission in their own home directory, so they're free to read and write their own files. In Chapter 5, you'll learn how to configure disk quotas, so users don't take more than their fair share of disk space.

/etc/skel

The /etc/skel directory contains default environment files for new accounts. The **useradd** command and the Red Hat User Manager copy these files to the home directory when you create a new account. The contents of /etc/skel may vary depending on what you have installed. Standard files from my RHEL 3 /etc/skel are described in Table 4-5.

If you've installed more than the default software on RHEL, you may see additional configuration files in the /etc/skel directory. For example, if you've installed the appropriate RPMs, you may see configuration files associated with emacs and the z shell (zsh) in this directory.

TABLE 4-5 Default Home Directory Files from /etc/skel

Files	Purpose
.bashrc	The basic bash configuration file. May include a reference to the general /etc/bashrc configuration file. Can include commands you want to run when you start the bash shell, appropriate for aliases such as rm='rm -i' .
.bash_logout	A file executed when you exit a bash shell. Can include commands appropriate for this purpose, such as clearing your screen.
.bash_profile	Configures the bash startup environment. Appropriate place to add environment variables or modify the directories in your PATH.
.gnome*	Several directories that include startup settings for the GNOME desktop environment. For example, details of desktop icons such as Trash are stored in .gnome-desktop/Trash.
.gtkrc	Adds the Bluecurve theme for the default Red Hat GUI.
.kde	A directory that includes autostart settings for the K Desktop Environment. Not copied to user home directories if you haven't installed KDE on this computer.



Linux includes many hidden files, which start with a dot (.). To list these files, run the `ls -a` command. For example, if you wanted to list the files in the `/etc/skel` directory, run the `ls -a /etc/skel` command.

As the system administrator, you can edit these files or place your own customized files in `/etc/skel`. When new users are created, these files are propagated to the new users' home directories.



Adding files to `/etc/skel` may be a convenient way to distribute files such as corporate policies to new users.

Window Manager Configuration File Locations

RHEL 3 comes with several window managers. At some point you will want to configure one or more of them for use on your system. Most window manager configuration files are stored in the `/etc/X11/windowmanager` directory, where *windowmanager* is the name of the specific window manager. This includes window managers such as `twm` (Tom's Window Manager), `xdm` (X Display Manager), and `gdm` (GNOME Display Manager). The exception is KDE, where the default window manager configuration files are stored in the `/etc/kde/kdm` directory.



Red Hat Linux 9 stores default KDE window manager configuration files in the `/etc/skel/.kde` directory.

CERTIFICATION OBJECTIVE 4.03

Red Hat Filesystem Basics

There are as many, if not more, filesystem types as there are operating systems. RHEL 3 can work with many of these formats.

At the heart of every RHEL 3 installation are the filesystems on which it relies. Linux supports a rich variety of filesystem types. For Linux, we can divide filesystems into two categories: "standard" formatting and journaling. While this is an oversimplification, it suffices to describe the filesystems important to Linux. To me, a standard filesystem is an older Linux filesystem or more static systems.



*There are a large number of filesystem types, well described in the **Filesystems HOWTO** at www.tldp.org. Strictly speaking, there is no “standard” filesystem, and in this book, I include a “balanced tree” filesystem (ReiserFS) as a journaling filesystem, just to be more consistent with the Red Hat outlines.*

The filesystems I describe in this book are just a small list of those available for RHEL 3. If you have the kernel source RPMs loaded on your system, you can find a list of the filesystems supported by your kernel. Navigate to the `/usr/src/linux-2.4` directory. Run the **make menuconfig** command and use your arrow keys to navigate to the filesystems section.

Standard Formatting Filesystems

Linux is a clone of Unix. The Linux filesystems were developed from the Unix filesystems available at the time. The first Linux operating systems used the Extended Filesystem (ext). Until the past couple of years, Red Hat Linux operating systems formatted their partitions by default to the Second Extended Filesystem (ext2).

I also include some other filesystems that you might see in RHEL 3. These “standard” filesystems don’t include journaling features. I illustrate a sample of these filesystem types in Table 4-6.

TABLE 4-6 Linux Standard Filesystem Type

Filesystem Type	Description
ext	The first Linux filesystem, used only on early versions of that operating system.
ext2 (Second Extended)	The foundation for ext3, the default RHEL 3 filesystem. The ext3 filesystem is essentially ext2 with journaling.
swap	The Linux swap filesystem is associated with dedicated swap partitions. You’ve probably created at least one swap partition when you installed RHEL 3.
MS-DOS and VFAT	These filesystems allow you to read MS-DOS–formatted filesystems. MS-DOS lets you read pre-Windows 95 partitions, or regular Windows partitions within the limits of short filenames. VFAT lets you read Windows 9x/NT/2000/XP/2003 partitions formatted to the FAT16 or FAT32 filesystems.
ISO 9660	The standard filesystem for CD-ROMs. It is also known as the High Sierra File System, or HSFs, on other Unix systems.
NTFS	NTFS is the Microsoft Windows NT/2000/XP/2003 filesystem designed for username/password security. Currently supported as a read-only system.
/proc	The /proc filesystem is a Linux <i>virtual</i> filesystem. Virtual means that it doesn’t occupy real disk space. Instead, files are created as needed. /proc is used to provide information on kernel configuration and device status.

TABLE 4-6 Linux Standard Filesystem Type (*continued*)

Filesystem Type	Description
/dev/pts	The /dev/pts filesystem is the Linux implementation of the Open Group's Unix98 PTY support.
NFS	The Network File System. This is the system most commonly used to share files and printers between Linux and Unix computers.
SMB	Server Message Block (SMB) is based on Microsoft and IBM network protocols. Linux can use SMB to share files and printers with Microsoft Windows operating systems. Works with Microsoft's Common Internet File System (CIFS).

Journaling Filesystems

As hard disks and partitions grow in size, Linux users are moving toward filesystems with journaling features. Journaling filesystems have two main advantages. First, it's faster to check them during the boot process. Second, if there's a crash, a journaling filesystem has a log (also known as a journal) which can be used to restore the metadata for the files on the relevant partition.

The default RHEL 3 filesystem is the Third Extended Filesystem, also known as ext3. However, it isn't the only journaling filesystem option available. I list a few of the options available for RHEL 3 in Table 4-7.

TABLE 4-7 Journaling Filesystems

Filesystem Type	Description
ext3	The default filesystem for RHEL 3.
JFS	JFS is IBM's journaled filesystem. This is commonly used on in IBM enterprise servers.
ReiserFS	The Reiser File System is resizable and supports fast journaling. It's more efficient when most of the files are very small and very large. It's based on the concept of "balanced trees." While supported by RHEL 3, it is not the default Red Hat filesystem and is explicitly cited only in the RHCT course (RH133) syllabus. For more information, see www.namesys.com .
xfs	Developed by Silicon Graphics as a journaling filesystem. Supports very large files; as of this writing, xfs files are limited to 9×10^{18} bytes. Do not confuse this filesystem with the X Font Server; both use the same acronym.

CERTIFICATION OBJECTIVE 4.04

Filesystem Management and the Automounter

Before you can use the files in a directory, you need to mount that directory on a partition formatted to some readable filesystem. Linux normally automates this process using the `/etc/fstab` configuration file. When you boot Linux, specified directories are mounted on configured partitions. The mount options require some explanation, especially for removable media.

But there may be problems if connections are lost or media is removed. When you configure a server, you could be mounting directories from a number of remote computers. You could also want temporary access to removable media such as floppies or Zip drives. The automount daemon, also known as the Automounter or `autofs`, can help. It can automatically mount specific directories as needed. It can unmount a directory automatically after a fixed period of time.

`/etc/fstab`

Linux stores information about your local and remotely mounted filesystems in `/etc/fstab`. Open this file in the text editor of your choice. As you can see, different filesystems are configured on each line. A sample `/etc/fstab` might look like the following:

<code>LABEL=/</code>	<code>/</code>	<code>ext3</code>	<code>defaults</code>	<code>1</code>	<code>1</code>
<code>LABEL=/boot</code>	<code>/boot</code>	<code>ext3</code>	<code>defaults</code>	<code>1</code>	<code>2</code>
<code>none</code>	<code>/dev/pts</code>	<code>devpts</code>	<code>gid=5,mode=620</code>	<code>0</code>	<code>0</code>
<code>none</code>	<code>/proc</code>	<code>proc</code>	<code>defaults</code>	<code>0</code>	<code>0</code>
<code>none</code>	<code>/dev/shm</code>	<code>proc</code>	<code>tmpfs</code>	<code>0</code>	<code>0</code>
<code>/dev/hda3</code>	<code>swap</code>	<code>swap</code>	<code>defaults</code>	<code>0</code>	<code>0</code>
<code>LABEL=/usr</code>	<code>/usr</code>	<code>ext3</code>	<code>defaults</code>	<code>1</code>	<code>2</code>
<code>LABEL=/tmp</code>	<code>/tmp</code>	<code>ext3</code>	<code>defaults</code>	<code>1</code>	<code>2</code>
<code>LABEL=/var</code>	<code>/var</code>	<code>ext3</code>	<code>defaults</code>	<code>1</code>	<code>2</code>
<code>LABEL=/home</code>	<code>/home</code>	<code>ext3</code>	<code>defaults</code>	<code>1</code>	<code>2</code>
<code>/dev/cdrom</code>	<code>/mnt/cdrom</code>	<code>udf,iso9660</code>	<code>noauto,owner,kudzu,ro</code>	<code>0</code>	<code>0</code>
<code>/dev/floppy</code>	<code>/mnt/floppy</code>	<code>auto</code>	<code>noauto,owner,kudzu</code>	<code>0</code>	<code>0</code>

As you can see, there are six fields associated with each filesystem, which I describe from left to right in Table 4-8. Remember, this is a configuration file. You can verify partitions that are actually mounted in `/etc/mstab`.

TABLE 4-8 Description of /etc/fstab by Column from Left to Right

Field Name	Description
Label	Lists the device to be mounted. If mounted, you can find the associated partition in /etc/mtab. For example, if you see /dev/hda2 in this file, you can verify its LABEL with the <code>e2label /dev/hda2</code> command.
Mount Point	Notes the directory where the filesystem will be mounted.
Filesystem Format	Describes the filesystem type. Valid filesystem types include ext, ext2, ext3, msdos, vfat, devpts, proc, tmpfs, udf, iso9660, nfs, smb, and swap.
Mount Options	Covered in the following section.
Dump Value	Either 0 or 1. A value of 1 means that data is automatically saved to disk by the <code>dump(8)</code> command when you exit Linux.
Filesystem Check Order	Determines the order that filesystems are checked by <code>fsck(8)</code> during the boot process. The root directory (/) filesystem should be set to 1, and other local filesystems should be set to 2. Removable filesystems such as /mnt/cdrom should be set to 0, which means that they are not checked during the Linux boot process.

Mount Options

Although **defaults** is the right mount option for most /etc/fstab filesystems, there are other options, as listed in Table 4-9. If you want to use multiple options, separate them by commas. Don't use spaces between options. The list in Table 4-9 is not comprehensive. You can find out more from the mount manual, which you can read by running the following command:

```
# man mount
```

TABLE 4-9 /etc/fstab Mount Options

Mount Option	Description
async	Data is read and written asynchronously.
atime	The inode associated with each file is updated each time the file is accessed.
auto	Searches through /etc/filesystems for the appropriate format for the partition; normally associated with a floppy or removable drive.
defaults	Uses default mount options: rw , suid , dev , exec , auto , nouser , and async .
dev	Permits access to character devices such as terminals or consoles and block devices such as drives.
exec	Allows binaries (compiled programs) to be run on this filesystem.

TABLE 4-9 /etc/fstab Mount Options (*continued*)

Mount Option	Description
noatime	The inode associated with each file is not updated when accessed.
noauto	Requires explicit mounting. Common option for CD and floppy drives.
nodev	Devices on this filesystem are not read or interpreted.
noexec	Binaries (compiled programs) cannot be run on this filesystem so mounted.
nosuid	Disallows setuid or setgid permissions on this filesystem.
nouser	Only root users are allowed to mount the specified filesystem.
remount	Remounts a currently mounted filesystem. Also an option for the mount command.
ro	Mounts the filesystem as read-only.
rw	Mounts the filesystem as read/write.
suid	Allows setuid or setgid permissions on programs on this filesystem.
sync	Reads and writes are done at the same speed (synchronously) on this filesystem.
user	Allows nonroot users to mount this filesystem. By default, this also sets the noexec , nosuid , and nodev options.

Mounting Floppy Disks and Removable Media

To read floppy disks and other removable media, you need to mount the device. If RHEL detects your hardware, it adds the proper settings to /etc/fstab and default mount points to the /mnt directory. The default mount points are straightforward; for example, floppy disks are mounted on /mnt/floppy, CDs are mounted on /mnt/cdrom. If you're mounting an MS-DOS-formatted floppy, you can do so with the following command:

```
# mount -t vfat /dev/fd0 /mnt/floppy
```

The **-t** switch specifies the type of filesystem (vfat). The device file /dev/fd0 represents the first floppy drive; /mnt/floppy is the directory through which you can access the files on the floppy after mounting.

But the command that you actually need may be simpler. Take a look at your /etc/fstab configuration file. You'll probably see something like the following line:

```
/dev/fd0          /mnt/floppy      auto      noauto,owner,kudzu 0 0
```

This `/etc/fstab` line sets the default configuration for the first floppy drive (`/dev/fd0`). Normally, it's mounted on the `/mnt/floppy` directory. The format is `auto`, which means that the **mount** command searches through the `/etc/filesystems` configuration file. As long as `vfat` is part of this file, you do not need to specify the filesystem type. The **mount** command reads the filesystem on the floppy and mounts it with the correct filesystem automatically. So all you really need to mount the floppy is the following command:

```
# mount /dev/fd0 /mnt/floppy
```

But wait—if you know something about `/etc/fstab`, you know that the device and mount points are effectively linked. If you specify either, **mount** looks through `/etc/fstab` for the appropriate device, label, or mount point. Therefore, all you actually need to mount the floppy is either one of the following two commands:

```
# mount /dev/fd0
```

```
# mount /mnt/floppy
```

Similarly, the device for your CD-ROM is normally `/dev/cdrom`. To mount an ISO 9660 CD-ROM, run the following command:

```
# mount -rt iso9660 /dev/cdrom /mnt/cdrom
```

Now you can read the contents of `/mnt/cdrom` as if it were a normal filesystem on your system. You don't have to use the `/mnt/floppy` or `/mnt/cdrom` directories. They're part of the standard Linux filesystem as a matter of convenience. You can mount a floppy or CD on an empty directory of your choice.

But as with a floppy disk, the actual command that you need is simpler. Take a look at the relevant default line from `/etc/fstab`:

```
/dev/cdrom      /mnt/cdrom      udf,iso9660     noauto,owner,kudzu,ro    0 0
```

As you can see, `iso9660` is already specified as the default CD filesystem, and CDs are mounted as read-only (`ro`) by default. Therefore, all you actually need to mount the CD is either one of the following two commands:

```
# mount /dev/cdrom
```

```
# mount /mnt/cdrom
```

To unmount a floppy or CD-ROM, use the **umount** command with the mount point as an argument. *Remember, the command is **umount**, not **unmount**.*

The following commands unmount both our floppy and our CD-ROM:

```
# umount /mnt/floppy
# umount /mnt/cdrom
```

It is important you unmount floppy disks before removing them. Otherwise, the data that you thought you wrote to the disk might still be in the cache. In that case, you would lose that data.



Here's another example that I think you're unlikely to see on the Red Hat exams. Assume you're working with a Zip drive of 100MB. The device is set as /dev/hdd and formatted as a single partition (/dev/hdd1) with the Linux Native (ext3) filesystem. The directory /mnt/zip was created and an entry added to /etc/fstab. You can now manually mount, unmount, and eject the zip drive with the following commands:

```
# mount /dev/hdd1 /mnt/zip
```

The command to manually unmount the Zip drive:

```
# umount /dev/hdd1
```

The command to eject the Zip drive (which automatically unmounts it):

```
# eject /dev/hdd1
```

The Automounter

Once you run the **mount** command on a partition, it stays mounted until you unmount it or shut down or reboot your computer. This can cause problems. For example, if you've mounted a floppy and then physically removed the disk, Linux may not have had a chance to write the file to the disk. This situation also applies to Zip or other "hot-swap" removable drives.

Another example is with mounted NFS directories. If the remote computer fails or the connection is lost, your system may become slow or even hang as it looks for the mounted directory.

This is where the Automounter can help. It relies on the **autofs** daemon to mount configured directories as needed, on a temporary basis. In RHEL 3, the relevant configuration files are /etc/auto.master and /etc/auto.misc. If you use the Automounter, keep the /misc directory free. Red Hat configures automounts on this directory by default, and they won't work if there are local files or directories stored there.

/etc/auto.master

The standard `/etc/auto.master` file includes just a series of comments, with one sample command:

```
# /misc /etc/auto.misc --timeout=60
```

You can activate the Automounter by activating this command (or adding a similar command). As you should already know as a Linux expert, the pound sign (#) is a comment character. When you delete the #, the command is activated. This command:

- Configures automounted filesystems on the `/misc` directory
- Points to `/etc/auto.misc` for detailed configuration
- Sets a timeout of 60 seconds; if the automounted directory is not used or accessed in this time, it is automatically unmounted

You can set up the Automounter on other directories. One popular option is to set up the Automounter on the `/home` directory. In this way, you can configure user home directories on remote servers, mounted on demand. Users are given access to their home directories upon login, and with the following command in `/etc/auto.master`, the home directory is automatically unmounted 60 seconds after that user logs off the system:

```
# /home /etc/auto.home --timeout=60
```

This works only if you don't have a `/home` directory on your computer. You'll get a chance to test a variation of this in a lab at the end of this chapter.

/etc/auto.misc

Red Hat conveniently provides standard automount commands in comments in the `/etc/auto.misc` file. It's helpful to analyze this file in detail. I use the default RHEL 3 version of this file. The first four lines are comments, which I skip. The first command is:

```
cd      -fstype=iso9660,ro,nosuid,nodev    :/dev/cdrom
```

In RHEL, this command is active by default. In other words, if you have a CD in the `/dev/cdrom` drive, you can access its files through the Automounter with the `ls /misc/cd` command. The Automounter accesses it using the ISO9660 filesystem. It's mounted read-only (ro); set user ID permissions are not allowed (**nosuid**), and devices on this filesystem are not used.

With the command from `/etc/auto.master`, the CD is unmounted 60 seconds after the last time it's accessed. There are a number of other sample commands in comments. The first command allows you to set up a `/misc/linux` mount point from a shared NFS directory, `/pub/linux`, on the `ftp.example.org` computer.

```
#linux    -ro,soft,intr    ftp.example.org:/pub/linux
```

The next command assumes that the `/boot` directory is stored on the `/dev/hda1` partition. With this command, you don't need to mount `/boot` when you start Linux. Instead, this command allows you to automount it with the **mount `/misc/boot`** command.

```
#boot     -fstype=ext2      :/dev/hda1
```

The following three commands apply to a floppy disk drive on your computer. The first command searches through `/etc/filesystems` to try to match what's on your floppy. The last two commands assume that the floppy is formatted to the Second Extended (`ext2`) filesystem.

```
#floppy    -fstype=auto      :/dev/fd0
#floppy    -fstype=ext2      :/dev/fd0
#e2floppy  -fstype=ext2      :/dev/fd0
```

The next command points to the first partition on the third SCSI drive. The “jaz” at the beginning suggests this is suitable for an Iomega type “Jazz” drive.

```
#jaz       -fstype=ext2      :/dev/sdc1
```

Finally, the last command assumes that you want to apply the Automounter to the IDE drive connected as the slave on the secondary controller. The “removable” at the beginning suggests this is suitable for removable hard drives.

```
#removable -fstype=ext2      :/dev/hdd
```

With the possible exception of the **floppy** commands, you'll need to modify these lines for your own hardware.

Activating the Automounter

Once you've configured the `/etc/auto.master` and related `/etc/auto.misc` configuration files, you can activate the Automounter. As it is governed by the `autofs` daemon, you can activate it with the following command:

```
# service autofs restart
```

With the default command in the `/etc/auto.misc` file, you should now be able to mount a CD on the `/misc/cd` directory, automatically, just by accessing the configured directory. Once you have a CD in the drive, the following command should work:

```
# ls /misc/cd
```

If you were to make `/misc/cd` your current directory, the Automounter would ignore any timeouts. Otherwise, `/misc/cd` is automatically unmounted according to the timeout, which according to the command in `/etc/auto.master`, is 60 seconds.

EXERCISE 4-2

Configuring the Automounter

In this exercise, you'll test the Automounter. You'll need at least a CD. Ideally, you should also have a floppy disk. But there are some preliminaries. You need to make sure that the **autofs** daemon is in operation, modify the appropriate configuration files, and then restart **autofs**. You can then test the Automounter for yourself.

Part of this exercise may be confusing. Where it refers to commands within a configuration file, a pound sign (#) in front indicates it's a comment. If it's on the command line interface, it refers to a command run by the root user, which I've signified in this section with the `[root]#` prompt. As with other exercises, I'm assuming that you've logged in as the root user.

1. From the command line interface, run the following command to make sure the **autofs** daemon is running:

```
[root]# service autofs start
```

2. Open the `/etc/auto.master` configuration file in a text editor. Modify it if required to include the following line. If there's a pound sign (#) in front of the line, delete it. Save and exit from `/etc/auto.master`.

```
/misc /etc/auto.misc --timeout=60
```

3. Check the `/etc/auto.misc` configuration file in a text editor. Make sure it includes the following line (which should already be there by default). Save and exit from `/etc/auto.misc`.

```
cd      -fstype=iso9660,ro,nosuid,nodev    :/dev/cdrom
```

4. Now restart the **autofs** daemon. (I know, this isn't the most efficient method, but it's a good habit to check the status of a service.)

```
[root]# service autofs restart
```

5. The Automounter service is now active. Insert a CD into the drive on your computer and when you run the following command, you should see the contents of your CD:

```
[root]# ls /misc/cd
```
 6. Run the `ls /misc` command immediately. You should see the `cd` directory in the output.
 7. Wait at least one minute, and repeat the previous command. What do you see?
-

OPTIONAL EXERCISE 4-3

A Floppy Disk and the Automounter

Now that you're more familiar with the Automounter, try using it on a floppy drive. If you don't have a floppy drive, experiment with some of the other commented commands in `/etc/auto.misc`.

CERTIFICATION OBJECTIVE 4.05

The Red Hat Package Manager Command and Utility

One of the major duties of a system administrator is software management. Applications are upgraded. Kernels are patched. Without the right tools, it can be difficult to figure out what software is on a system, what is the latest update, and what applications depend on other software. Worse, you may install a new software package only to find it has overwritten a crucial file from a currently installed package.

The Red Hat Package Manager (RPM) was designed to eliminate these problems. With RPM, software is managed in discrete “packages.” An RPM *package* includes the software with instructions for adding, removing, and upgrading those files. When properly used, the RPM system can back up key configuration files before proceeding

with upgrades and removals. It can also help you identify the currently installed version of any RPM-based application.

What Is a Package?

In the generic sense, an RPM package is a container of files. It includes the group of files associated with a specific program or application, which normally includes binary installation scripts, as well as configuration and documentation files. It also includes instructions on how and where these files should be installed and uninstalled.

An RPM package name usually includes the version, the release, and the architecture for which it was built. For example, the fictional `penguin-3.4.5-26.i386.rpm` package is version 3.4.5, build 26. The “i386” indicates that it is suitable for computers built to the Intel architecture.



Many RPM packages are CPU specific. You can identify the CPU type for your computer in the `/proc/cpuinfo` file. Some RPM packages, with the `noarch` label, can be installed on computers with all types of CPUs.

What Is RPM?

At the heart of this system is the RPM database. Among other things, this database tracks the version and location of each file in each RPM. The RPM database also maintains an MD5 checksum of each file. With the checksum, you can use the **`rpm -V package`** command to see if any file from that RPM package has changed. The RPM database makes adding, removing, and upgrading packages easy, because RPM knows which files to handle, and where to put them.

RPM also manages conflicts between packages. For example, assume you have two different packages that use configuration files with the same name. Call the original configuration file `/etc/someconfig`. You’ve already installed package X. If you then try to install package Y, RPM backs up the original `/etc/someconfig` file before installing package Y.

Using RPM on Remote Packages

With the RPM system, you can even specify package locations just like an Internet address, in Uniform Resource Locator (URL) format. For example, if you want to apply the **`rpm`** command to the `foo.rpm` package on the `/pub` directory of the `ftp.rpmdownloads.com` FTP server, you can install this package with a command such as:

```
# rpm -i ftp://ftp.rpmdownloads.com/pub/foo.rpm
```

Assuming you're connected to the Internet, this particular **rpm** command logs onto the FTP server anonymously and downloads the file.

If the FTP server requires a username and password, you can use the following format: `ftp://username:password@hostname:port/path/to/remote/package/file.rpm`, where *username* and *password* are the username and password you need to log on to this system, and *port*, if required, specifies a nonstandard port used on the remote FTP server. Based on the preceding example, if the username is `mjang` and the password is `Ila451MS`, you could install an RPM directly from a server with the following command:

```
rpm -i ftp://mjang:Ila451MS@ftp.rpmdownloads.com/pub/foo.rpm
```

The key to this system is the **rpm** command. We'll cover the four most common modes associated with this command: query, install, upgrade, and remove. But first, in this age of insecure downloads, you should know how to validate the signature associated with an RPM, as well as verifying the files in a specific package.

Validating a Package Signature

RPM uses two methods of checking the integrity of a package: the MD5 checksum and the GPG signature. You can use the MD5 checksum to verify that the file is intact (no data was lost or corrupted while copying or downloading the file). You can then use the GPG signature to make sure the file is authentic. For example, you can use it to confirm that an RPM file is indeed an official Red Hat RPM. Red Hat provides a GPG public key for its RPM files; you can find it on each installation CD in the RPM-GPG-KEY file or in the `/usr/share/rhn` directory.

To authenticate your RPMs using the GPG system, import the key file using the following command (assuming it's a CD-based keyfile, mounted on the `/mnt/cdrom` directory):

```
# rpm --import /usr/share/rhn/RPM_GPG_KEY
```

Now if you wanted to check the integrity of some RPM such as `pkg-1.2.3-4.noarch.rpm` on a CD, run the following command:

```
# rpm --checksig /mnt/cdrom/RedHat/RPMS/pkg-0.0.0-0.rpm
```

This allows you to verify both the integrity and the authenticity of the RPM.

Verifying One or More Packages

Verifying an installed package compares information about that package with information from the RPM database on your system. The `--verify` switch checks the size, MD5 checksum, permissions, type, owner, and group of each file in the package. Here are a few examples:

- Verify all files. Naturally, this may take a long time on your system.
`# rpm --verify -a`
- Verify all files within a package against an RPM file on a mounted CD-ROM.
`# rpm --verify -p /mnt/cdrom/RedHat/RPMS/inn-2.3.5-5.i386.rpm`
- Verify a file associated with a particular package.
`# rpm --verify --file /bin/ls`

If the files or packages check out, you will see no output. Any output means that a file or package is different from the original. There are eight tests. If there's a change, the output is a string of eight characters, each of which tells you what happened during each test.

If you see a dot (`.`), that test passed. The following example shows `/bin/vi` with an incorrect group ID assignment:

```
# rpm --verify --file /bin/vi
.....G.  /bin/vi
```

Table 4-10 lists the failure codes and their meanings.

TABLE 4-10

rpm --verify
Codes

Failure Code	Meaning
5	MD5 checksum
S	File size
L	Symbolic link
T	File modification time
D	Device
U	User
G	Group
M	Mode

Looking for the Right RPM

Sometimes you need a file or a command, and just don't know what is the correct package to install. Red Hat provides a database that can associate the right RPM package with a specific file. Naturally, this database can be installed from the `rpmdb-redhat-*` RPM. It's installed by default on RHEL 3. It enables you to find the RPM associated with any available file from the Red Hat installation CDs. For example, if you're looking for the package associated with `/etc/passwd`, run the following command:

```
# rpm --redhatprovides /etc/passwd
```

If you're using Red Hat Linux 9 to study for the exam, you'll have to install the `rpmdb-redhat` RPM after installation.

Adding and Removing RPM Packages

The `rpm` command makes it easy to add and remove software packages to your system. It maintains a database regarding the proper way to add, upgrade, and remove packages. This makes it relatively simple to add and remove software with a single command.

Install Mode

The install mode, as its name suggests, is used to install RPM packages on your system. You can install an RPM package with the `-i` option. For example, the following command installs the hypothetical penguin RPM:

```
# rpm -i penguin-3.4.5-26.i386.rpm
```

If the package is available on a remote FTP server, you could install it directly from that server. For example, the following command installs it from a hypothetical `ftp.rpmdownloads.com` server in its `/pub` directory.

```
# rpm -i ftp://ftp.rpmdownloads.com/pub/penguin-3.4.5-26.i386.rpm
```

Before installing the package, RPM performs several checks. First, it makes sure the package you're trying to install isn't already installed—normally, RPM won't let you install a package on top of itself. It also checks to make sure you aren't installing an older version of the package. Next, RPM does a dependency check. Some programs won't work unless others are already installed. In this example, you've just downloaded the latest RPM version of the Penguin utilities, and you now want to install it.

```
# rpm -i penguin-3.4.5-26.i386.rpm
failed dependencies:
iceberg >= 7.1 is needed by penguin-3.26.i386.rpm
```

This error tells you that **rpm** did not install the Penguin package because it requires the iceberg RPM package, version 7.1 or later. You'll have to find and install the iceberg package, and any packages iceberg may require.

Finally, RPM checks to see if it would overwrite any configuration files when it installs a package. RPM tries to make intelligent decisions about what to do in this situation. If RPM chooses to replace an existing configuration file, it gives you a warning like:

```
# rpm -i penguin-3.26.i386.rpm
warning: /etc/someconfig saved as /etc/someconfig.rpm.save
```

It's up to you to look at both files and determine what, if any, modifications need to be made.



If you've already customized a package and upgrade it with rpm, go to the saved configuration file. Use it as a guide to change the settings in the new configuration file. Since you may need to make different changes to the new configuration file, you should test the result in every way that package may be used in a production environment.

Upgrade Mode

The **-U** switch is used to upgrade existing packages. For example, if Penguin utilities, version 3.25, is already installed, the following command:

```
# rpm -U penguin-3.26.i386.rpm
```

upgrades the old version of the package with the new one. In fact, if you've never installed this package before, the **-U** switch works just like **-i**. The package is simply installed for the first time.

Remove Mode

The **rpm -e** command removes a package from your system. But before removing a package, RPM checks a few things first. It does a dependency check to make sure no other packages need what you're trying to remove. If it finds dependent packages, **rpm -e** fails with an error message identifying these packages.

If you have modified any of the configuration files, RPM makes a copy of the file, adds an **.rpm.save** extension to the end of the filename, and then erases the original. Finally, after removing all files from your system and the RPM database, it removes the package name from the database.



Be very careful about which packages you remove from your system. Like many other Linux utilities, RPM may silently let you shoot yourself in the foot. For example, if you were to remove the packages for `/etc/passwd` or the kernel, that would devastate your system.

Adding Updates and Security Fixes

Red Hat Enterprise Linux is constantly being updated. As bugs or security problems are found, they are collected on Red Hat's errata Web page, currently located at www.redhat.com/apps/support/errata/. Additional errata associated with Red Hat Enterprise Linux is available through the Red Hat Network. If you've purchased RHEL 3, it's part of your subscription.



As of this writing, Red Hat has stated that it will discontinue errata for Red Hat Linux 9 in April of 2004. The people associated with the Fedora Legacy Project plan to continue support beyond that date. More information is available at www.fedoralegacy.org.

EXERCISE 4-4

Updating from the Red Hat Errata

The steps in this exercise assume that you're using Red Hat Enterprise Linux 3. If you are using Red Hat Linux 9 or earlier, you may not be able to do this exercise. Red Hat has stated that it will discontinue support for Red Hat Linux 9 after April 2004. However, you may be able to find errata through the Fedora project. If you find errata, here's a good checklist to follow whenever you review the errata page. The actual steps may vary widely for errata from the Fedora project:

1. Go to www.redhat.com/apps/support/errata. Find the Red Hat Errata link for your distribution.
2. The page you see includes links for security, bug fixes, and enhancements. The security alerts are especially important if your system is on a network. Click the security link. You may need to click the link associated with the alert. If you have a subscription to the Red Hat Network for RHEL 3, use your account. You may find additional errata here as well.

3. You may not have installed some of the affected packages. Use the **rpm -qi *packagename*** command to check. If you have installed an affected package on your system, consider the recommended upgrade.
 4. Before replacing an affected package, consider the ramifications. You may need to bring the system down to single-user or perform a reboot, which can affect other users who are connected to your system. If a production computer is affected, you may want to test the changes on another computer first.
 5. If you choose to make an upgrade, back up at least the relevant files on your current system. Upgrades do fail on occasion.
 6. When performing the upgrade, watch for configuration file warnings. If your local configuration files are replaced with new files, you may need to change the new configuration files to reflect your current settings.
 7. Thoroughly test the new package. Make sure you have it configured correctly.
 8. If a package is listed in the errata but not installed on your system, chances are there's no reason to put it on your system now. Read the detailed errata entry for that package carefully, and only install it if needed.
-

Red Hat has included a Network Alert Notification Tool. You can use this to check for revised packages, new packages, errata, and other information. It comes as an entitlement with your purchase of Red Hat Enterprise Linux. There is no reference to this software in the syllabus for the RHCT (RH133) or RHCE (RH300) exams, or in the RHCE Exam Prep guide.

Seeing What Packages Are Installed

Without RPM, you'd need to search around your filesystems to figure out whether a particular software package is installed. RPM makes it easy for you to figure out what RPM packages are installed and get information about those packages.

Running RPM Queries

The **-q** switch is used to query packages. When you use RPM's query mode, you can learn if a specific package is installed or identify the files associated with that package. The **rpm -q *packagename*** command will return the installed version of a specified package. For example, to find the version number of an installed mutt text e-mail reader, run the following command:

```
# rpm -q mutt
lynx-1.4.1-3
```

If you want to see which installed package owns a file, use the `-f` modifier. Here we want to identify the package that owns `/etc/passwd`:

```
# rpm -qf /etc/passwd
setup-2.5.27-1
```

Likewise, if you want to generate a list of files belonging to a certain package, use the `-l` modifier.

```
# rpm -ql setup
/etc/bashrc
/etc/csh.cshrc
/etc/csh.login
/etc/exports
/etc/filesystems
/etc/group
/etc/gshadow
/etc/host.conf
/etc/hosts.allow
/etc/hosts.deny
/etc/inputrc
/etc/motd
/etc/passwd
/etc/printcap
/etc/profile
/etc/profile.d
/etc/protocols
/etc/securetty
/etc/services
/etc/shells
/usr/share/doc/setup-2.5.27
/usr/share/doc/setup-2.5.27/uidgid
/var/log/lastlog
```

One of the most common modifiers to `-q` is `-a`, a query for all installed packages on your system. As configured, my RHEL 3 system includes 700 packages. Here's a truncated output:

```
# rpm -qa
ghostscript-fonts-5.50-9
libmng-1.0.4-3
```

```
libtiff-3.5.7-13
ncurses-5.3-9.3
arts-1.1.3-2.2
...
commons-modeler-1.0-5
sendmail-cf-8.12.10-1
```



It's common to use the `rpm -qa` command as a searchable database. All you need to do is pipe the output with a search term through `grep`. For example, the `rpm -qa | grep kde` command makes it easy to identify all installed packages related to the K Desktop Environment.

For even more information about a specific package, use the `-i` (information) modifier. Table 4-11 lists some of the most important entries from the package output. Run the `rpm -qi packagename` command on the RPM of your choice and see the result for yourself.

The Red Hat Package Management Utility

You can update packages with a graphical tool. The Red Hat Package Management utility includes several advantages. It includes an interface similar to the Anaconda Custom Installation screen. It allows you to install more than one package at a time. It automatically installs any other dependent packages that may be needed.

There are two ways to start the Package Management utility. In the GNOME or KDE desktop, click Main Menu | System Settings | Add/Remove Applications. Alternatively, in a GUI text command window, run the `redhat-config-packages` command. Either action opens the Package Management utility shown in Figure 4-6.

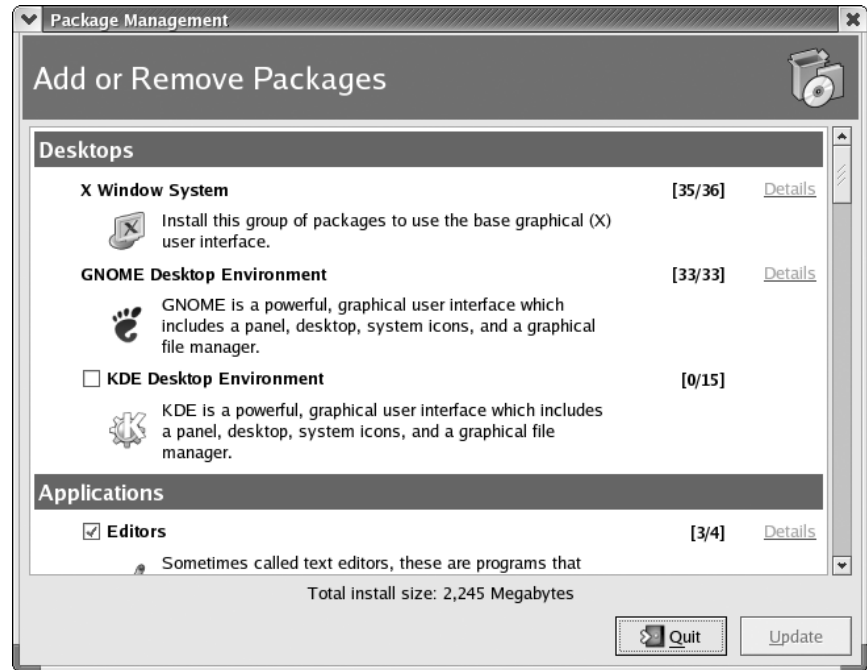
TABLE 4-11

Information from the `rpm -qi packagename` Command

Tag	Description
Name	The name of the package.
Version	The version of the package.
Release	The number of times this package has been released using the same version of the software.
Install Date	When this package was installed on your system.
Group	Your RPM database is divided into groups, which describe the functionality of the software. Every time you install a package, it will be grouped accordingly.
Size	The total size in bytes of all the files in the package.
License	The license under which the original software was released.

FIGURE 4-6

The Package Management utility



You can use it to add the package groups of your choice. The list shown is the same as what you saw during the installation process. Once you've selected a package group, you're allowed to customize the installation by clicking the associated Details link.

There's an important trick with the Package Management utility. With the right switch, you can connect it to a remote installation server, such as the one you created in Chapter 2. For example, if you've mounted the NFS installation source from Chapter 2 to the local `/mnt/inst` directory, you can link the Package Management utility to that source with the following command:

```
# redhat-config-packages --tree=/mnt/inst
```

For more information on mounting remote directories via Samba, see Chapter 8; to do the same via NFS, see Chapter 9

exam

Watch

If you need to install a group of packages during the Red Hat exams, you may not have access to all of the installation CDs. In that case, the

`redhat-config-packages` command can help. With the `-tree` switch, you can use the same network source that you used to install Red Hat Enterprise Linux.

CERTIFICATION OBJECTIVE 4.06

RPM Sources

The RPMs that I've described so far are already in binary format. In other words, they're compiled and ready to install just like "tarballs" or Microsoft-style installation packages. But Linux goes back to the source code. Courtesy of the GPL, the source code is readily available.

In fact, if you're studying for the Red Hat exams using Red Hat Linux 9, the source code may be all you have available for the RHEL packages that you may need for this book. The source code includes the programs and configuration files that you need to set up and install the binary RPMs described earlier.



You can also build RPM sources from a tar archive. However, as of this writing, there are no requirements in the RHCE and RHCT related course outlines related to building source RPMs from a tar archive. And the obsolete version of the RHCE exam prep guide suggests that you only need to be “familiar with the basic elements of” source RPMs.

Creating and Using Custom RPMs

A source RPM is, as the name indicates, a package of source code used to build architecture-specific packages. Properly labeled source RPMs include the “src” identifier as part of the filename, such as:

```
polarbear-2.07-2.src.rpm
```

Binary RPMs are built from source RPMs. The source RPM contains the source code and specifications necessary to create the binary RPM.

Before you continue with this section, install the rpm-build RPM package. For RHEL 3, if you're installing from a network source on the /mnt/inst directory, you'd run the following command:

```
# rpm -Uvh /mnt/inst/RedHat/RPMS/rpm-build-4.2.1-4.2.i386.rpm
```

If you're running Red Hat Linux 9, install the rpm-build package associated with that distribution. You can then use the **rpmbuild** command as described in the following sections.



In RHEL 3, the `rpmbuild` command has superseded the older `rpm` commands associated with building RPM packages from source code.

Installing Source RPMs

Like normal RPMs, a source RPM (SRPM) is installed with the `-i` option. It does not install the package itself. This installs the contents of the SRPM within the `/usr/src/redhat` directory structure, which you can turn into a binary RPM. For the purpose of this section, I've installed the source for the Package Management utility, `redhat-config-packages`, from the freely available RHEL 3 source RPMs. I describe their availability later in this chapter.

The `/usr/src/redhat/` Directory Structure

There are five subdirectories under the `/usr/src/redhat` directory, which I describe in Table 4-12.

When you build an SRPM, you will build it within this structure. If you install an SRPM, it is extracted into this structure. The `redhat-config-packages` source RPM that you installed in the previous section should now have a file in the `/usr/src/redhat/SPECS` directory, in the `redhat-config-packages.spec` file.

Changing Compile Options for a Source RPM

While most precompiled RPMs will serve your needs, at times you will want to modify the source code or compile options in the corresponding SRPMs. You can do so in the spec file that you get when you installed the source RPM.

The Spec File To change the compile options in an SRPM, you must understand spec files. The spec file is stored in `/usr/src/redhat/SPECS/packagename.spec`. The spec file controls the way a package is built, and what actions are performed when it is installed or removed from a system. A spec file has ten different sections, as described in Table 4-13.

TABLE 4-12

Build Directories
from RPM
Sources

Directory	Purpose
<code>/usr/src/redhat/SOURCES</code>	Contains the original program source code
<code>/usr/src/redhat/SPECS</code>	Contains spec files, which control the RPM build process
<code>/usr/src/redhat/BUILD</code>	Source code is unpacked and built here
<code>/usr/src/redhat/RPMS</code>	Contains the output binary RPM
<code>/usr/src/redhat/SRPMS</code>	Contains the SRPM created by the build process

TABLE 4-13 Build Directories for Source RPM Files

Section	Description
Preamble	Includes information shown with an <code>rpm -qi</code> command. This normally includes a summary, version, and group. It also includes a list of dependent packages.
Description	A basic package description.
Prep	Includes any preparatory commands required before building the source code, such as unpacking.
Build	Commands to actually compile the spec file and build sources.
Install	Commands to install the software on a system.
Install and uninstall scripts	This section contains scripts that will be run on the end user's system to install or remove the software. RPM can execute a script before the package is installed, after the package is installed, before the package is removed, and after the package is removed.
Verify	Although RPM takes care of most verification tasks, a script can be inserted here for any desired extra checks.
Clean	A script can be specified here to perform any necessary cleanup tasks.
Files	A list of files in the package.
Changelog	A list of revisions.

Several of the sections include commands that can be run as individual shell scripts. You won't see all of these sections in every spec file.

You can change the compile-time options for a package in the build section of the spec file. Here's a sample build section from a different spec file (this is not from the `redhat-config-packages` source RPM):

```
%build
rm -rf $RPM_BUILD_ROOT
mkdir -p $RPM_BUILD_ROOT/usr/bin $RPM_BUILD_ROOT/etc
./configure --prefix=/usr --exec-prefix=/
make CFLAGS="$RPM_OPT_FLAGS" LDFLAGS=-s
```

This section, a shell script, begins with some housekeeping, removing any files that may be left over from a previous build. A directory structure is created for the source files. Then the package is configured and compiled with a “make.”



For a different package, you might modify the make command line to compile other components after LDFLAGS. The compile options from \$RPM_OPT_FLAGS are defaults, set by RPM. Alternatively, you could use this variable to set other compile time options such as a different CPU.

Perhaps the essential reference guide to the RPM system is a book called Maximum RPM. It's available online from Red Hat at www.redhat.com/docs/books/max-rpm.

Building Custom Source and Binary RPMs

By now, you should understand where you should modify an SRPM spec file to change compile time options in the Build section. However, there's much more to building customized RPMs. Once you have modified the spec file, you need to tell RPM to build a new RPM and SRPM.

Starting a Build

You can build an RPM using the **rpmbuild** command, with the build switch, **-b**. By itself, **rpmbuild -b** calls the scripts specified in the Prep, Build, and Install parts of the spec file. Normally, you'll modify the **-b** with **a**, which makes RPM go through the build process, step by step. The RPM build operation is directed at a spec file. For example, the command

```
# rpmbuild -ba redhat-config-packages.spec
```

directs RPM to create binary and source RPMs from this spec file. Alternatively, if you just want the binary RPM, the following command will do the job:

```
# rpmbuild -bb redhat-config-packages.spec
```

Building Red Hat Enterprise RPMs

The SRPMs for RHEL 3 are freely available online from the Red Hat FTP site at ftp.redhat.com or mirror sites specified at www.redhat.com/download/mirror.html. As of this writing, the RHEL 3 SRPMs are located on the `/pub/redhat/linux/enterprise/3/en/os/i386/SRPMS` directory.

If you're studying for the Red Hat exams using Red Hat Linux 9, you don't have to buy RHEL 3. You can build Red Hat Enterprise RPMs from the source code. For

example, we can learn more about the Red Hat Package Management utility from the `redhat-config-packages` source code. To do so, take the following steps:

Navigate to `ftp.redhat.com` (or a mirror site) and navigate to the directory with RHEL 3 SRPMs. Then you can:

1. Download the RHEL 3 version of the package that contains the Package Management utility, `redhat-config-packages-1.2.5-1.src.rpm`.
2. Install the source RPM package. The `rpmbuild` commands described in the previous section do not install the package on your computer; it does set it up in your `/usr/src/redhat` tree.

CERTIFICATION OBJECTIVE 4.07

Network Configuration

The network is where the power of Red Hat Enterprise Linux really comes alive; however, getting there may not be trivial. As in all other things Linux, it's a learning experience. Most critical settings are stored in the `/etc/sysconfig` directory.

In most cases, you'll configure networking when you install RHEL 3 during each exam. However, if you have problems, you may need to diagnose networking problems, especially during the troubleshooting exams.

exam

Watch

Learn the scripts in the `/etc/sysconfig` and `/etc/sysconfig/network` directories. These are crucial to the configuration of Red Hat Enterprise Linux. If you have a configuration to change or repair, it probably involves files in one of these directories. If you have a problem on

the troubleshooting exam, you may find the solution in these files. Red Hat is consolidating a number of its key configuration files in its directories, so expect them to become even more important in the future.

The configuration file that provides the foundation for others in RHEL 3 networking is `/etc/sysconfig/network`. It can contain up to five variables, as described in Table 4-14.

TABLE 4-14

`/etc/sysconfig/
network
Variables`

Variable	Description
NETWORKING	Can be yes or no, to configure or not configure networking.
NISDOMAIN	If you're connected to an NIS network, this should be set to the name of the NIS domain.
HOSTNAME	Sets the hostname of the local computer. If you don't see this variable, it may be set by a DHCP server.
GATEWAY	Sets the IP address for the gateway for your network. If you don't see this variable, it may be set by a DHCP server.
GATEWAYDEV	Sets the network device, such as eth0, that this computer uses to reach a gateway. You won't see this if you have only one network card on your computer.

If you don't see the variable in your `/etc/sysconfig/network` file, the situation does not apply. For example, if you don't see the `GATEWAYDEV` variable, you probably have only one network card on your computer.

The `/etc/sysconfig/network-scripts` Files

We'll start our tour in the `/etc/sysconfig` directory. This is where Red Hat Enterprise Linux stores and retrieves its networking information. With the various Red Hat configuration tools, you don't need to touch these files, but it's good to know they're there. A few representative files are shown in Table 4-15.

TABLE 4-15

`/etc/sysconfig/network-scripts` Files

File in <code>/etc/sysconfig/network-scripts</code>	Description
<code>ifcfg-lo</code>	Configures the loopback device, which is a virtual device that confirms proper installation of TCP/IP.
<code>ifcfg-*</code>	Each installed network adapter, such as eth0, gets its own <code>ifcfg-*</code> script. For example, eth0 gets <code>ifcfg-eth0</code> . This file includes the IP address information required to identify this network adapter on a network.
<code>network-functions</code>	This script contains functions used by other network scripts to bring network interfaces up and down.
<code>ifup-*</code> and <code>ifdown-*</code>	These scripts activate and deactivate their assigned protocols. For example, <code>ifup-ipx</code> brings up the IPX protocol.

TABLE 4-16 Other Network Configuration Commands

Network Script	Description
ifup, ifdown	These scripts start and stop a network card such as eth0. Naturally, ifup eth0 activates eth0; ifdown eth0 deactivates this device.
dhclient	dhclient activates a connection to a DHCP server for your network. The function of a DHCP server is to lease an IP address. Normally activated during the boot process through the ifcfg-* script in the <code>/etc/sysconfig/network-scripts</code> directory. Supersedes scripts from previous versions of Red Hat Linux, including pump and dhcpcd .
ifconfig	The main network interface configuration utility. Can return or set the network parameters on a network device.

There are several closely related commands which can help you manage networking from the command line interface. I describe a few of these commands in Table 4-16.

Setting Up a Network Interface

In this section, I illustrate how you can configure networking on your computer with text commands as well as with the Red Hat GUI Network Configuration utility. Changes are written to various configuration files, including `/etc/sysconfig/network`, and file and related directories. There is a text mode version of this tool, which you can modify your system name, as well as add, remove, and edit network interfaces. While you can start it with the **redhat-config-network-tui** command, it's officially an experimental interface for RHEL 3 and is not nearly as capable as the GUI tool.

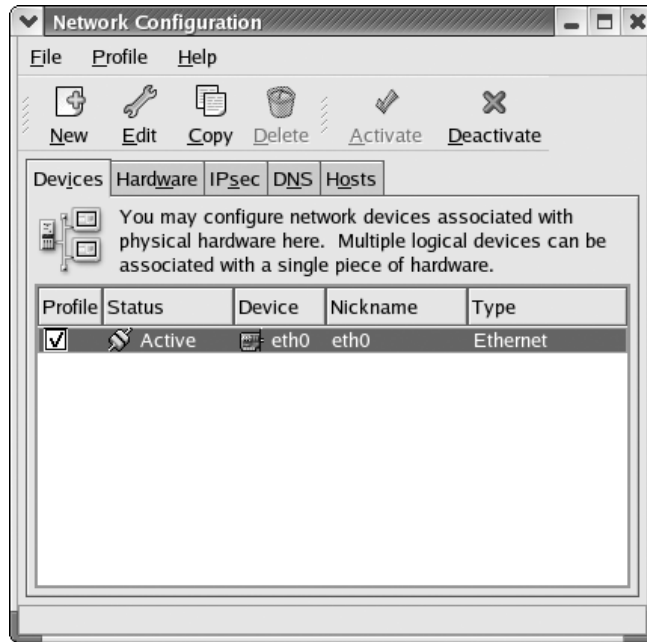


With the redhat-config-* GUI tools, Red Hat has developed a number of utilities that I believe will encourage Microsoft Windows administrators to make the switch to Linux. Unfortunately, I believe the lack of corresponding text-based tools has discouraged some Linux administrators from working with Red Hat.

I illustrate how you can use the Network Configuration utility (see Figure 4-7) in the following exercise, and then illustrate how you can monitor and modify your network configuration with various text commands.

FIGURE 4-7

Network
Configuration
utility

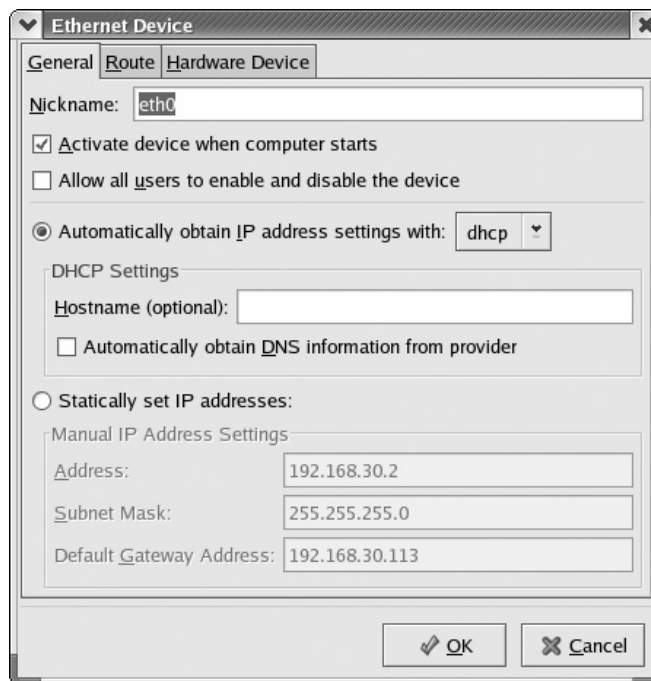


EXERCISE 4-5

Modifying Network Interfaces with redhat config-network

1. You can start the Network Configuration utility in one of two ways in the GUI. You can click Main Menu | System Settings | Network. Alternatively from a GUI terminal, run **redhat-config-network**. This opens the Network Configuration dialog box.
2. Select the Devices tab if required.
3. Select the adapter that you wish to modify, and then select Edit. If it is an Ethernet adapter, you'll see an Ethernet Device dialog box similar to the one shown here.

III 4-2



4. Pay attention to the “Allow all users to enable and disable the device” option. If you activate it, you’ll change the `USERCTL` variable in the `ifcfg-*` script to yes.

Note: Record your current settings for this interface before proceeding.

5. Change the IP value to 192.168.1.11 and the network mask to 255.255.255.0.

Note: If your computer is on the 192.168.1.0 private network, use a different private IP address. It should isolate you from all other hosts on the local network. Test this after step 5 by using the **ping** command to try to connect to other hosts on your network.

6. Click File | Save and wait for the process to complete.
7. At the command prompt, run **ifconfig** to check your new IP settings.
8. Repeat steps 1–3 and then reset the values to your previous settings. Run **ifconfig** again to make sure you’ve restored your original network configuration.

Many values are associated with each network interface. At minimum, each network adapter requires a valid, unique IP address, as well as an appropriate network mask. The Network Configuration utility provides five convenient tabs which you can use to customize each network adapter:

- **Devices** This tab allows you to add a new network adapter or edit a configured adapter. You can revise the name of the adapter, IP address assignments, static routing, and hardware device information. Different devices are configured in the `/etc/sysconfig` files described earlier.
- **Hardware** This tab lets you modify the IRQ port, memory location, I/O address(es), and DMA channel(s) associated with the adapter. Hardware information is documented in different files in the `/proc` directory.



Linux sometimes has trouble recognizing second network adapters; you may need to specify hardware addresses such as the IRQ port.

- **IPsec** This tab supports Virtual Private Network connections.
- **DNS** This tab lets you add the addresses of DNS servers available to network adapters on the given network, which is reflected in `/etc/resolv.conf`.
- **Hosts** This tab allows you to modify the name, alias, and IP address assigned to the specified adapter, which is shown in `/etc/hosts`

In addition, if you run Profile | New, you can create different network configurations, which can be useful for flexible configurations. For example, if you have a laptop computer and a docking port, your configuration may change depending on whether the laptop is connected to the docking port.

If you want to use this tool, some trial and error is appropriate. Make some changes, and check the effect on the files I've described.

ifup/ifdown

For each installed network adapter, there is a corresponding `ifcfg-*` file in `/etc/sysconfig/network-scripts`. You can activate or deactivate that adapter with the **ifup** and **ifdown** commands. Either one of the following commands will activate the `eth0` network adapter:

```
ifup ifcfg-eth0
ifup eth0
```

ifconfig

The **ifconfig** command is used to configure and display network devices. Here is some sample output from this command:

```
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:50:56:40:1E:6A
          inet addr:192.168.30.2  Bcast:192.168.30.255
Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:11253 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1304 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:2092656 (1.9 Mb)  TX bytes:161329 (157.5 Kb)
          Interrupt:10 Base address:0x10a0
```

The preceding command requests configuration data for the first Ethernet device on the system, eth0. If you just specify eth0 (or another device), **ifconfig** displays information about only the specified interface. If you don't specify a device, **ifconfig** shows all network adapters, including the loopback adapter.

The **ifconfig** command can also be used to configure network interfaces. For example, you can assign a new IP address for eth0 with the following command:

```
# ifconfig eth0 207.174.142.142
```

The first parameter, eth0, tells us which interface is being configured. The next argument, 207.174.142.142, indicates the new IP address being assigned to this interface. If we want to make sure our change worked, we issue the **ifconfig** command again to view its current settings.

```
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:50:56:40:1E:6A
          inet addr: 207.174.142.142  Bcast:207.174.142.255
Mask:255.255.255.0
          UP BROADCAST NOTRAILERS RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:11253 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1304 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:2092656 (1.9 Mb)  TX bytes:161329 (157.5 Kb)
          Interrupt:10 Base address:0x10a0
```

Looking at the output of our command, we successfully changed the IP address on the eth0 interface to 207.174.142.142. But this is not enough, as you should realize that the broadcast address doesn't work with this IP address.

With the right switch, the **ifconfig** command can modify a number of other settings for your network adapter. Some of these switches are shown in Table 4-17.

TABLE 4-17 ifconfig Switches

Parameter	Description
up	Activates the specified adapter.
down	Deactivates the specified adapter.
netmask <i>address</i>	Assigns the <i>address</i> subnet mask.
broadcast <i>address</i>	Assigns the <i>address</i> as the broadcast address. Rarely required, since the default broadcast address is standard for most current networks.
metric <i>N</i>	Allows you to set a metric value of <i>N</i> for the routing table associated with the network adapter.
mtu <i>N</i>	Sets the maximum transmission unit as <i>N</i> , in bytes.
-arp	Deactivates the address resolution protocol, which collects network adapter hardware addresses.
promisc	Activates promiscuous mode. This allows the network adapter to read all packets to all hosts on the LAN. Can be used to analyze the network for problems, or to try to crack messages between other users.
-promisc	Deactivates promiscuous mode.
irq <i>port</i>	Assigns a specific IRQ port.
io_addr <i>address</i>	Assigns a specific I/O address.

netstat -r

The **netstat** command is used to display a plethora of network connectivity information. The most commonly used option, **netstat -r**, is used to display local routing tables. Here's a sample **netstat -r** output:

```
# netstat -nr
Kernel routing table
Destination      Gateway          Genmask          Flags MSS Window  irtt Iface
191.72.1.0       *                255.255.255.0    U    40  0        0 eth0
127.0.0.0        *                255.0.0.0        UH   40  0        0 lo
0.0.0.0          191.72.1.1      255.255.255.0    UG   40  0        0 eth0
```

Did you notice we used a **-n** flag? **-n** tells **netstat** to display addresses as IP addresses, instead of as hostnames. This makes it a little easier to see what's going on.

The Destination column lists networks by their IP addresses. The Gateway column indicates gateway addresses. If the destination is on the LAN, no gateway is required, so an asterisk is shown in this column. The Genmask column lists the network mask.

Networks look for a route appropriate to the destination IP address. The IP address is compared against the destination networks, in order. When the IP address is found to be part of one of these networks, it's sent in that direction. If there is a gateway address, it's sent to the computer with that gateway. The Flags column describes how this is done. Flag values are listed in Table 4-18.

arp as a Diagnostic Tool

The Address Resolution Protocol associates the hardware address of a network adapter with an IP address. The **arp** command displays a table of hardware and IP addresses on the local computer. With **arp**, you can detect problems such as duplicate addresses on the network, or you can manually add **arp** entries as required. Here's a sample **arp** command, showing all **arp** entries in the local database:

```
# arp
Address          HWtype  HWaddress          Flags Mask
Iface
192.168.0.121    ether   52:A5:CB:54:52:A2   C
eth0
192.168.0.113    ether   00:A0:C5:E2:49:02   C
eth0
```

If the arp table is empty, you haven't made any connections to other computers on your network. The address column lists known IP addresses, usually on the LAN. The HW Type column shows the hardware type of the adapter, while the HW Address column shows the hardware address of the adapter.

You can use the **-H** option to limit the output from **arp** to a specific hardware type, such as ax25, ether, or pronet. The default is ether, which is short for Ethernet.

The **arp** command can help you with duplicate IP addresses, which can stop a network completely. To remove the offending machine's arp entry from your arp table, use the **-d** option:

```
# arp -d buggy
```

TABLE 4-18

The netstat Flag Indicates the Route

Flag	Description
G	The route uses a gateway.
U	The network adapter (Iface) is up.
H	Only a single host can be reached via this route.
D	This entry was created by an ICMP redirect message.
M	This entry was modified by an ICMP redirect message.

This removes all arp information for the host “bugsy.” To add an arp entry, use the `-s` option:

```
# arp -s buggy 00:00:c0:cf:a1:33
```

This entry will add the host buggy with the given hardware address to the arp table. IP addresses won’t work in this case.

DHCP Clients

You can set up your computer as a DHCP client. If the **redhat-config-network** configuration utility does not work, check the configuration file associated with your network card in the `/etc/sysconfig/network-scripts` directory. You should not need static IP configuration information, and you should see `BOOTPROTO=dhcp`.

In any case, if you have a working DHCP server on your network, you can connect your computer to it with the **dhclient** command.



*Red Hat has used different commands to support DHCP clients in the past, including **dhcpcd** and **pump**.*

CERTIFICATION OBJECTIVE 4.08

The Basic Boot Process

Understanding how your system boots and shuts down will help you immensely as a Red Hat system administrator. Red Hat Enterprise Linux uses a boot process called System V init, which means that after the kernel is loaded, it starts a program called init, which then starts everything else. To understand the process better, let’s go through the steps Red Hat Enterprise Linux takes to boot itself up to a usable system.

Once you understand the basic boot process, Red Hat Enterprise Linux provides two utilities that assist the system administrator in configuring and maintaining

the commands that make up that startup and shutdown process. The **redhat-config-services** utility provides a GUI interface, while **chkconfig** provides a command line interface to manage services at different runlevels. Before we begin, it’s important to understand Red Hat runlevels before reading about the boot process.

exam

Watch

The Red Hat course associated with the RHCT, RH133, explicitly requires that you understand what happens during the Linux boot process.

Runlevels

There are six basic runlevels in Red Hat Enterprise Linux, as defined in `/etc/inittab`. Each runlevel is associated with a level of functionality. For example, in single-user mode, also known as runlevel 1, only one user is allowed to connect to that Linux system. X11 mode, also known as runlevel 5, starts Linux into a GUI login screen. The Red Hat definitions for System V init runlevels are shown in Table 4-19.

Making each runlevel work is the province of a substantial number of scripts. Each script can start or stop fundamental Linux processes such as printing (`cupsd`), scheduling (`crond`), Apache (`httpd`), Samba (`smbd`), and more. The starting and stopping of the right scripts becomes part of the boot process.

It should go without saying that if you set your `initdefault` to 0, your system will shut down when Linux tries to boot. Likewise, if you set the `initdefault` to 6, Linux will enter a continuous reboot cycle.

exam

Watch

For the Troubleshooting and System Maintenance exam, it can be useful to back up and then modify critical configuration files such as `/etc/inittab`. But remember to do this on a test computer; if you can't solve the problem, you may lose the data on that computer. Before you proceed, learn the rescue mode techniques described in Chapter 11.

The Boot Process

When you start your computer, the BIOS checks your system, and looks for a bootloader such as GRUB. It finds and installs the Linux kernel. Then Linux can start installing the services you have configured.

TABLE 4-19	
Red Hat Runlevels	Runlevel
	Description
	0
	Halt
	1
	Single-user mode, for maintenance (backups/restores) and repairs
	2
	Multiuser, without networking
	3
	Multiuser, with networking
	4
	Unused
	5
	X11, defaults to a GUI login screen. Logins bring the user to a GUI desktop.
	6
	Reboot (never set <code>initdefault</code> in <code>/etc/inittab</code> to this value!)

The kernel always starts by calling `init`. The `init` process in turn runs `/etc/rc.d/rc.sysinit`, which performs a number of tasks, including network configuration, keyboard maps, partition mounts, and hostnames. The `init` process then determines which runlevel it should be in by looking at the `initdefault` entry in `/etc/inittab`. A runlevel is defined as a group of activities. For example, the entry:

```
id:5:initdefault:
```

shows a default starting point in runlevel 5. Next, `init` starts the appropriate scripts associated with runlevel 5, from the `/etc/rc.d` directory. That directory includes the following subdirectories:

```
init.d
rc0.d
rc1.d
rc2.d
rc3.d
rc4.d
rc5.d
rc6.d
```

If the default runlevel is 5, `init` will look in `/etc/rc.d/rc5.d` and run each “kill” and “start” script it finds in that directory. A kill script is any file or symbolically linked file with a name that begins with a “K.” Likewise, start scripts start with “S.” If you run an `ls -l` command in this directory, you’ll see only symbolic links to the actual scripts in `/etc/rc.d/init.d`. Observe current examples of kill and start scripts at runlevel 5 in Figure 4-8.

What’s going on here? System V `init` knows to go to the directory associated with a particular runlevel. Once there, `init` runs the scripts in that directory that start with a K and then the scripts starting with an S. The K scripts stop processes that

FIGURE 4-8

Sample kill and start scripts in runlevel 5

```
[root@Enterprise3 rc5.d]# \ls K*
K05innnd      K20netdump-server  K34dhcrelay    K40smartd      K50vsftpd
K05saslauthd  K20nfs             K34yppasswdd   K45named       K70aep1000
K10psacct     K20rwhod           K35dhcpd       K50netdump     K70bcm5820
K15dc_client  K20spamassassin    K35smb         K50snmpd       K74ypserv
K15dc_server  K24irda            K35vncserver   K50snmptrapd   K74ypxfrd
K15httpd      K25squid           K35winbind     K50tux
[root@Enterprise3 rc5.d]# \ls S*
S00microcode_ctl S12syslog      S25netfs      S58ntpd      S97rhnsd
S05kudzu         S13irqbalance S26apmd       S59hpoj      S99local
S08arptables_jf  S13portmap    S28autofs     S80sendmail  S99mdmonitor
S08ip6tables     S14nfslock    S55cups       S85gpm
S08iptables     S17keytable   S55sshd       S90crond
S09isdn          S20random     S56rawdevices S90xfs
S10network       S24pcmcia     S56xinetd     S95atd
[root@Enterprise3 rc5.d]#
```

aren't supposed to operate in that runlevel. The S scripts start the processes associated with that runlevel. Within each category, scripts are run in numeric order; for example, K20nfs is run before K50tux, which is run before S10network.

Red Hat uses six different runlevels: 0, 1, 2, 3, 5, and 6. (Runlevel 4 is unused as of this writing.) Symbolic links allows the collection of all init scripts in one directory, /etc/init.d. Any changes to a start or kill script can be made in that directory.

You can run a start script yourself, with some key switches. For example, you can run the `smbd` (Samba) and `sshd` (secure shell daemon) scripts with the following options:

```
# /etc/rc.d/init.d/smb
Usage: /etc/rc.d/init.d/smb {start|stop|restart|reload|status|condrestart}
# service smb      # service is a shortcut to the management scripts
Usage: /etc/rc.d/init.d/smb {start|stop|restart|reload|status|condrestart}
# service sshd
Usage: sshd {start|stop|restart|reload|condrestart|status}
#
```

For example, the `/etc/rc.d/init.d/smb restart` command stops and starts Samba. See from the code how you can substitute the `service` command for `/etc/rc.d/init.d` in this command. It's one more timesaver that you can use on the Red Hat exams.

This is used by the scripts at each runlevel. In other words, if there is a K script for the `smbd` daemon, init runs `/etc/rc.d/init.d/smb stop`. And naturally, an S script for the `ssh` daemon runs `/etc/rc.d/init.d/sshd start`.

exam

watch

Make sure you go through the `/etc/rc.d` hierarchy as well as the `/etc/inittab` and `/etc/rc.d/rc.sysinit` files, and understand what's happening along the way. This is the key to understanding what's happening during the boot process.

CERTIFICATION OBJECTIVE 4.09

Virtual Consoles

As Red Hat Enterprise Linux is a multiterminal operating system, you can log into Linux, even with the same user ID, several times. This is made possible through the use of virtual consoles.

It's easy to open up a new virtual console. Just use the appropriate ALT-function key combination. For example, pressing the ALT key and F2 at the same time brings

you to the second virtual console. You can switch between adjacent virtual consoles by pressing ALT-RIGHT ARROW or ALT-LEFT ARROW. For example, to move from virtual console 2 to virtual console 3, press ALT-RIGHT ARROW.

/etc/inittab

Virtual consoles are configured in the init configuration file, `/etc/inittab`. By default, Red Hat Enterprise Linux is configured with six virtual consoles. You can add up to six more virtual consoles in `/etc/inittab`. Here are the default `/etc/inittab` entries for the first six virtual consoles:

```
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6
```

Virtual consoles really bring the multiuser capabilities of Linux to life. You can be viewing a man page on one console, compiling a program in another, and editing a document in yet a third virtual console. Other users who are connected through a network can do the same thing at the same time.

The Graphical Console

The GUI is, in one way, just another console. By default, six virtual consoles are configured with Linux, so the GUI is next in line, at console 7. To switch from the GUI to a regular virtual console, press CTRL-ALT-Fx, where x represents one of the other virtual consoles.

CERTIFICATION OBJECTIVE 4.10

Other Configuration Tools

There are a number of other key configuration files in the `/etc/sysconfig` directory. You can configure them with a text editor, with text commands, or in many cases, with a Red Hat graphical tool.

Time is of the essence on the Red Hat exams. It's fastest if you know how to configure these systems directly using text commands or by directly editing the key configuration file. However, if you forget how to manage one or two configuration commands or files, the Red Hat graphical tools can be a lifesaver.

I only address those systems that I believe are relevant on the Red Hat exams which are not already covered in other chapters.

Other /etc/sysconfig Files

Let's return to the /etc/sysconfig directory and discuss some of the non-network configuration files. I describe some of these files in Table 4-20. I cover only those files which might remotely be of interest on the Red Hat exams. I cover some of these files in other chapters. Some of these files can be edited directly; others can be configured with other Red Hat tools discussed in the following section.

TABLE 4-20 Key Non-network /etc/sysconfig Files

File in the /etc/sysconfig Directory	Description
clock	Contains defaults for the system clock, including time zone, UTC, and ARC (Alpha CPU-based) settings. If UTC=true, the BIOS is set to Greenwich Mean Time.
firstboot	If RUN_FIRSTBOOT=YES, then you can start the First Boot process with the firstboot command, if you're in runlevel 5.
gpm	Points to the mouse device, usually /dev/mouse.
grub	Lists the hard disk with your /boot drive, assuming you're using the GRUB boot loader.
harddisks	Supports settings for special hard disks.
hwconf	Lists peripherals detected by kudzu. Do not edit this file!
i18n	Sets the default language.
iptables	Includes the iptables firewall commands run when you boot Linux.
irda	Controls infrared devices
keyboard	Contains keyboard configuration data: KEYBOARDTYPE, usually "pc" and KEYTABLE, usually "us".

TABLE 4-20 Key Non-network /etc/sysconfig Files (*continued*)

File in the /etc/sysconfig Directory	Description
mouse	Contains mouse configuration data: FULLNAME describes the mouse type. MOUSETYPE lists the generic model such as IMPS2, msbm, or Microsoft. If XEMU3=yes, you have a two-button mouse set to emulate a third button. If WHEEL=yes, your mouse has a scroll wheel.
pcmcia	Contains PCMCIA configuration data. If PCMCIA=yes, Linux loads PCMCIA modules on boot.

GUI Configuration Utilities

It's best to know how to configure RHEL 3 by hand. It's the best way to control everything on your Linux system. It's faster on the Red Hat exams, where time is of the essence. There are a number of good GUI configuration tools available; almost all of them are "front-ends" which edit text configuration files, which you could edit directly.

However, there's a lot to learn about Linux. Learning how to edit *all* key Linux configuration files can be more than some can handle. While you should learn how to edit these files by hand, you may not have time. You may get nervous during the Red Hat exams and forget details. In these cases, the Red Hat GUI tools can be a lifesaver.

I cover a number of Red Hat GUI tools in this book. I summarize some other tools in RHEL 3 in this section. I cover only those tools which I believe can help you on the Red Hat exams in this section.



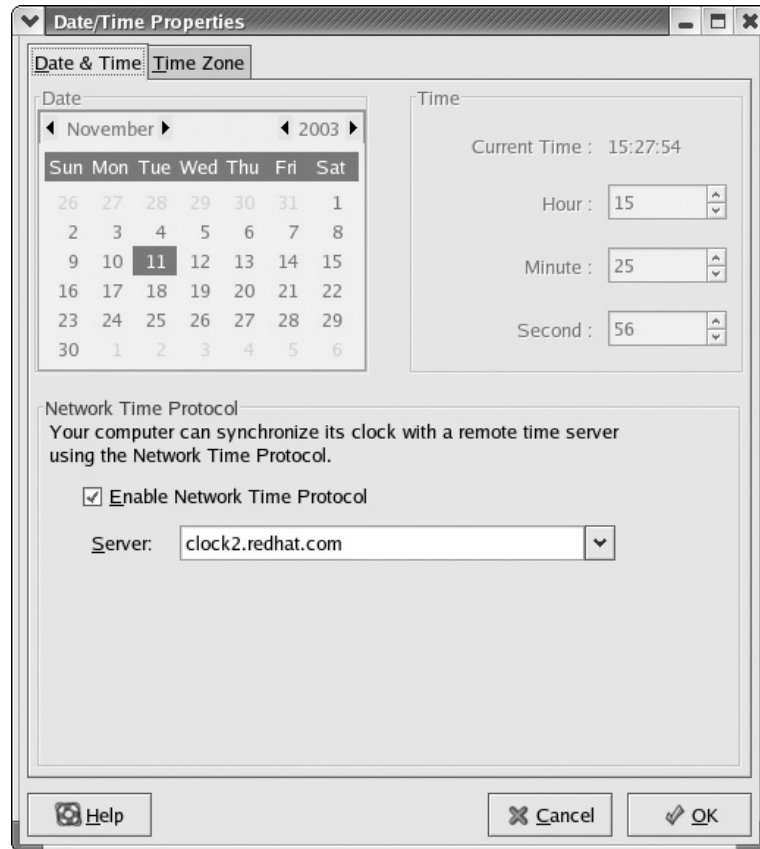
The text mode setup tool is a front-end to a number of other tools you can view from the text console: authentication, firewall, mouse, network, printer, service, and time zone configuration. Unfortunately, they have not been updated for RHEL 3, and I therefore do not cover them in this book.

redhat-config-date

With the Date/Time Properties configuration tool, you can set the date, time, time zone, and NTP server for your system. You can start it in one of three ways in the GUI: run the **redhat-config-date** or **redhat-config-time** commands, or click Main Menu | System Settings | Date and Time. This opens the Date/Time Properties window shown in Figure 4-9.

FIGURE 4-9

The Date/Time Properties tool



redhat-config-keyboard

The Keyboard configuration tool allows you to reselect the keyboard associated with your system. You can start it in one of two ways in the GUI: run the **redhat-config-keyboard** command or click Main Menu | System Settings | Keyboard. The options are the same as you saw during the installation process. Results are recorded in `/etc/sysconfig/keyboard`.

redhat-config-mouse

The Mouse Configuration tool allows you to reselect the pointing device associated with your system. You can start it in one of two ways in the GUI: run the **redhat-config-mouse** command or click Main Menu | System Settings | Mouse. The options are the same as you saw during the installation process. Results are recorded in `/etc/sysconfig/mouse`.

redhat-config-services

The Service Configuration tool shown in Figure 4-10 allows you to select the services that are to be activated in runlevels 3 (text login) and runlevel 5 (GUI login). You can start it in one of two ways in the GUI: run the **redhat-config-services** command or click Main Menu | System Settings | Server Settings | Services.

But it's faster to make these changes at the command line. The **chkconfig** command gives you a simple way to maintain different runlevels within the `/etc/rc.d` directory structure. With **chkconfig**, you can add, remove, and change services, list startup information, and check the state of a particular service. For example, you can check the runlevels where the sendmail service is set to start with the following command:

```
# chkconfig --list sendmail
sendmail 0:off 1:off 2:on 3:on 4:on 5:on 6:off
```

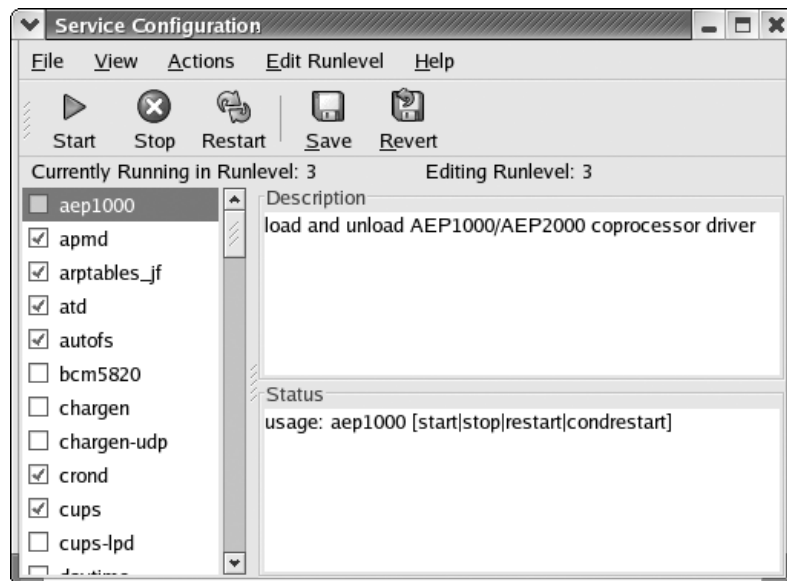
which indicates that sendmail is configured to start in runlevels 2, 3, 4, and 5. If you want to turn the sendmail service off for runlevel 4, execute the following command:

```
# chkconfig --level 4 sendmail off
```

Now sendmail is configured to run only on runlevels 2, 3, and 5. To turn it back on, you run the same command, substituting **on** for **off**. With **chkconfig**, you can also add

FIGURE 4-10

The Service Configuration utility



or delete services with the `--add` and `--del` switches. Installing a service sets up the appropriate links within the `/etc/rc.d` directory hierarchy. Uninstalling that service removes the associated links from the same hierarchy.

exam

Watch

When you configure or repair a service, use `chkconfig` (or a related utility such as `redhat-config-services`) to make

sure that the service is activated at the appropriate runlevels. Otherwise, you may not get credit for your work.

CERTIFICATION SUMMARY

This chapter covered basic configuration and administration of a Red Hat Enterprise Linux system. We learned the steps necessary to create a basic user, how to populate a user's home directory with the templates in `/etc/skel`, and where window manager configuration files are located. We also covered the different types of filesystems Linux uses, discussed how to mount them, and described what mount options to use with them.

This chapter covered the management of RPM packages. We learned the steps necessary to validate a package signature, how to add, remove, and upgrade packages, and how to add updates. We also talked about verifying packages and how to see what package a file belongs to. We finished the topic with a discussion on installing SRPMs and building RPMs from SRPMs and tar archives.

We addressed the configuration files in the `/etc/sysconfig` hierarchy. Some are important for networking; others are important for basic parameters such as the system clock, mouse, and keyboard. We also discussed a number of related networking commands, including `ifup`, `ifdown`, `ifconfig`, `netstat`, `arp`, and `dhclient`.

We concluded the chapter by describing some of the Red Hat GUI tools that you can use if you forget how to configure some key configuration files during the Red Hat exams.



TWO-MINUTE DRILL

Here are some of the key points from the certification objectives in Chapter 4.

Adding, Deleting, and Modifying User Accounts

- ☐ After installation, your system may have only a single login account: root. For most installations, you'll want to create more accounts, even if only for everyday use.
- ☐ Accounts can be added by directly editing `/etc/passwd` or with the **useradd** command. The advantage of **useradd** is that it automatically adds the new home directory as well as configuration files from `/etc/skel`.
- ☐ Accounts can be added with the Red Hat User Manager utility. You can also use this utility or related commands such as **chage** and **usermod** to further configure the account with parameters such as a password lifetime or a time limit on the account.
- ☐ Discourage the use of shared accounts, where several people use a single account. Shared accounts are almost always unnecessary, and they are easily compromised.
- ☐ If you'll be using the Network File System (NFS), it can help to make user accounts with the same UID across systems. The Network Information System (NIS) can serve the same purpose by establishing one database for all systems on your network.

The Basic User Environment

- ☐ Each user on your system has an environment when logged on to the system.
- ☐ The home directory for each login account is the initial directory in which users are placed when they first log on. They start with files from `/etc/skel`
- ☐ Window manager configuration files are often stored in `/etc/X11/windowmanager`, where *windowmanager* is the name of the window manager. KDE window manager files are stored in each user's home directory.

Red Hat Filesystem Basics

- ☐ Linux filesystems can be loosely defined as regular and journaling filesystems. While there are other filesystems available, this describes the essential difference between the older ext2 and the current default ext3 filesystems.

Filesystem Management and the Automounter

- ☐ If you have the kernel source RPMs installed, you can review supported filesystems.
- ☐ A number of mount options are available for `/etc/fstab`. The **defaults** option sets up a partition as `rw` (read/write), `suid` (superuser ID files allowed), `dev` (device files read), `exec` (binaries can be run), `auto` (automatic mounting), `nouser` (mountable only by root), and `async` (data is read asynchronously).

Using the Red Hat Package Manager Command and Utility

- ☐ The RPM database tracks where each file in a package is located, its version, and much more.
- ☐ Verifying an installed package compares information about that package with information from the RPM database on your system or the original package.
- ☐ The install mode of RPM, as its name suggests, is used to install RPM packages on your system.
- ☐ The upgrade mode of RPM will replace the old version of the package with the new one.
- ☐ The **rpm -e** command (erase) removes a package from your system.
- ☐ Using the **rpm** command query mode (**-q**), you can determine which packages are installed on your system or what file belongs to a particular package.
- ☐ You can use the Red Hat Package Management utility to install a group of packages from sources such as your installation CD. With the right **redhat-config-packages** command, you can even point to a network source.

RPM Sources

- ☐ Source RPMs are, as the name indicates, contain the source code used to build architecture-specific packages.
- ☐ The spec file is stored in `/usr/src/redhat/SPECS/packagename.spec`. It controls the way a package is built, and what actions are performed when it is installed or removed from a system.
- ☐ Run **rpmbuild -ba *packagename* spec** to build your RPM and SRPM.

Network Configuration

- ☐ Key network configuration files are in the `/etc/sysconfig` directory: the `network` file, and the `networking` and `network-scripts` subdirectories.
- ☐ You can start the Network Configuration utility with the **redhat-config-network** command.
- ☐ To manage network settings on each interface, use **dhclient** (dhcp/bootp client management), **ifup**, and **ifdown**.
- ☐ The **ifconfig** command is used to configure and display network devices.
- ☐ Use **ifup eth0** and **ifdown eth0** to activate and deactivate the `eth0` interface.
- ☐ The **netstat** command is used to display a plethora of network connectivity information.
- ☐ The **arp** command is used to view or modify the local hardware address database.

The Basic Boot Process

- ☐ Red Hat uses a boot process called System V init, which means that after the kernel is loaded, it starts a program called `init`, which then starts `/etc/rc.sysinit`.
- ☐ Study the `/etc/rc.d` hierarchy and the `/etc/inittab` and `/etc/rc.d/rc.sysinit` files. This is the key to understanding what's happening during the boot process.
- ☐ The **chkconfig** command gives you a simple way to maintain the `/etc/rc.d` directory structure. The **redhat-config-services** command provides a GUI tool for the same purpose.

Virtual Consoles

- ☐ Because Red Hat Enterprise Linux is a multiterminal operating system, it allows you to have more than one login session on the system console at a time using virtual consoles.
- ☐ By default, six virtual consoles are started when you boot Linux.
- ☐ You can switch between virtual consoles by pressing ALT-F1 to ALT-F6. By default, you start on the first virtual (ALT-F1) console screen.
- ☐ You can get to console mode from your X Window System session with CTRL-ALT-(F1 to F6). Return to the X Window System with ALT-F7.

Other Configuration Tools

- ☐ There are a number of non-network configuration files in the `/etc/sysconfig` directory.
- ☐ You can edit many of these files directly, or use GUI tools, which you can start with commands such as **redhat-config-date**, **redhat-config-keyboard**, **redhat-config-mouse**, and **redhat-config-services**.



SELF TEST

The following questions will help you measure your understanding of the material presented in this chapter. Read all the choices carefully, as there may be more than one correct answer. Choose all correct answers for each question. Don't focus exclusively on these questions. There is no longer any multiple choice questions on the Red Hat exams. These questions test your understanding of the chapter. Getting results, not memorizing trivia, is what counts on the Red Hat exams.

Adding, Deleting, and Modifying User Accounts

1. When deleting a user account using **userdel**, which command switch deletes that user's home directory?
 - A. `-d`
 - B. `-r`
 - C. `-h`
 - D. `-a`

The Basic User Environment

2. Where are individual users' configuration files for the KDE window manager stored?
 - A. `/usr/lib/X11/.kde`
 - B. `~/.kde`
 - C. `/etc/X11/.kde`
 - D. `/etc/skel/.kde`

Red Hat Filesystem Basics

3. Which of the following is an advantage of a journaling filesystem such as ext3?
 - A. It supports the use of VFAT for floppy disks.
 - B. It collects filesystem information in `/etc/filesystems`.
 - C. It allows you to read CDs.
 - D. It minimizes the risk if power is lost.

Filesystem Management and the Automounter

4. To change the mount options for a local filesystem, which of the following files would you edit?
- A. `/etc/filesystems`
 - B. `/etc/fstab`
 - C. `/etc/group`
 - D. `/etc/mnttab`

Using the Red Hat Package Manager Command and Utility

5. Which of the following commands correctly installs the package `penguin-3.26.i386.rpm`?
- A. `rpm -l penguin-3.26.i386.rpm`
 - B. `rpm -i penguin`
 - C. `rpm -i penguin-3.26.i386.rpm`
 - D. `rpm --install penguin.rpm`

RPM Sources

6. Assume you have the `rpm-build` RPM installed. When you install a source RPMs, where will you find key files?
- A. `/usr/lib/rpm`
 - B. `/usr/src/rpm`
 - C. `/usr/src/redhat`
 - D. `/usr/src/redhat/rpm`

Network Configuration

7. What information can you find in the `/etc/sysconfig/network` file?
- A. Your computer's hostname
 - B. The devices used for your network connections
 - C. Chat scripts for PPP and SLIP connections
 - D. The status of the network

The Basic Boot Process

8. Upon boot, the kernel starts init. What does init start next?
- A. `/etc/rc.d/init.d`
 - B. `/etc/inittab`
 - C. `/etc/rc.d/initdefault`
 - D. `/etc/rc.d/rc.sysinit`

Virtual Consoles

9. To switch to the fourth virtual console from another text console, which keys would you press?
- A. ALT-4
 - B. ALT-F4
 - C. CTRL-4
 - D. CTRL-F4

Other Configuration Tools

10. What types of services can you configure with files in the `/etc/sysconfig` directory?
- A. All types of services. These files supplement, but do not replace any other configuration files.
 - B. Only network services.
 - C. Only critical services related to your hardware configuration.
 - D. All runlevels.

LAB QUESTIONS

The Red Hat exams are unique based on their reliance on labs and hands-on demonstrations. With these questions, you're practicing the skills you need on both Red Hat exams.

Lab I

In this exercise, you are going to experiment with two ways to manage services at different runlevels: the `chkconfig` command and the `redhat-config-services` utility, also known as the Service Configuration utility. The commands in this lab don't start or stop scripts immediately; just the next time you move your Linux system into runlevel 3.

1. Open the GUI. From a text command line interface, run the **redhat-config-services &** command. This allows you to use the same terminal window for other commands. Alternatively, click Main Menu | System Settings | Server Settings | Services.
2. The Service Configuration utility is a graphical tool for controlling the services that Linux starts and stops at each runlevel.
3. These next steps assume that the `nfs` service is already running and installed. If not, pick another service to add and remove (it does not matter which as long as you restore the original condition when you're done).
4. At the command line, run the **ls /etc/rc3.d/*nfs** command to find the priority number.
5. Remove the `nfs` service from runlevel 3 using the Service Configuration utility.
6. Switch back to the command line window and run the **chkconfig --list nfs** command to see if it has been deactivated in runlevel 3.
7. Restore the `nfs` start script with the following command:


```
# chkconfig --level 3 nfs on
```
8. Switch back to the console window and run the **chkconfig --list nfs** command to verify that it is back.
9. Return to the Service Configuration utility, and click View | Refresh Service List to confirm that `nfs` is back on in runlevel 3.
10. Although the Red Hat Service Configuration utility provides a nice graphical interface, the **chkconfig** command is faster and more reliable, especially since X is not always available in an emergency or through remote login.

Lab 2

While the Red Hat User Manager provides a convenient interface to add new users, it's important for the exam to know the basic command line tools. Specifically, you can use the **useradd**, **usermod**, and **userdel** commands to add, modify, and delete user accounts.

1. Use the man pages for **useradd** (or just type **useradd** with no arguments for a simple help summary) to find the switches you need to add the following account with each of these attributes:

```
login: brianr
name: brian rite
UID: 5010
```

```
GID: nobody
shell: /bin/bash
```

2. Change the passwd for brianr to RvRg49().
3. Open a different virtual terminal. Log in as brianr. What files are present in this new account? Exit from this login.
4. Remove the brianr account using the **userdel** command. Is the brianr home directory gone? What option would have done this for you?

Lab 3

In this lab, you'll configure the Automounter on your computer on an NFS connection. You'll need a second computer with Linux or Unix installed, and a shared NFS directory. You can use the shared NFS installation source created in Chapter 2 or any other shared NFS directory described in Chapter 9. A virtual machine such as a VMWare computer qualifies as a second computer.

1. Back up your current `/etc/auto.master` and `/etc/auto.misc` configuration files.
2. Open the `/etc/auto.master` configuration file in the text editor of your choice. Add or activate the command which applies the Automounter to the `/misc` directory.
3. Open the `/etc/auto.misc` configuration file. Use the example shown in this file to create an NFS entry, which points to the shared NFS directory on the second computer. For the purpose of this lab, set the name of the directory to test.
4. Restart the autofs server.
5. Try your connection. Run the following command:


```
# ls /misc/test
```
6. You should see the contents of the shared NFS directory. Run the following command. What do you see?


```
# ls /misc
```
7. Wait a while, at least the *timeout* specified in the `/etc/auto.master` configuration file.
8. Run the `ls /misc` command again. What happens?
9. Once you're satisfied with the result, restore the files you backed up in step 1.

Lab 4

In this lab, you'll observe what happens when you avoid mounting the `/boot` filesystem. This of course assumes that you have a computer configured with `/boot` mounted on a separate partition.

1. Back up your `/etc/fstab` configuration file to something you can easily remember, such as `/etc/bak.fstab`.
2. Run the `df` command and make a note of the mounted filesystems, especially the partition associated with `/boot`.
3. Open `/etc/fstab` in the text editor of your choice. You should have a line associated with the `/boot` directory. If you do not, *stop*, as you cannot continue with this lab. **Trying to continue with this lab under these circumstances will probably render your system unbootable.**
4. Comment out the line with the `/boot` directory. Add a `#` in front of that line. For example, your `/etc/fstab` `/boot` directory line might read something like:

```
# LABEL=/boot          /boot          ext3          defaults 1 2
```
5. Save `/etc/fstab` and reboot Linux.
6. Observe the results. Does Linux start normally?
7. Look at your boot directory. Are there any files missing? How did Linux boot without benefit of a `vmlinuz` kernel or a `mkinitrd` RAM disk file?
8. Run the `df` command and make a note of the mounted filesystems.
9. Open `/etc/fstab` again. Remove the comment field, the `#`, in front of the `/boot` directory line. Save `/etc/fstab`. This should restore your `/etc/fstab` to the original configuration.
10. Mount the `/boot` directory with the `mount /boot` command.
11. Open your GRUB configuration file, `/boot/grub/grub.conf` in your text editor.
12. Observe the line associated with the `root` variable. This should correspond to the default partition for the `/boot` directory. Note that GRUB does not require that a partition is mounted to access a file.
13. Reboot your system. Run the `df` command and compare the list of mounted filesystems to step 2. If they are the same, stop here. Otherwise, restore your `/etc/fstab` from the backup file, `/etc/bak.fstab`.

Lab 5

In this lab, you'll install a new package group with the **redhat-config-packages** command, also known as the Package Management utility. You'll need a remote network source with the Red Hat installation files. It can be the same network source that you used in Chapter 2 to install RHEL 3.

1. In the Red Hat GUI, open a command line interface.
2. Click Main Menu | System Settings | Add/Remove Applications to open the Package Management utility.
3. Try adding the software of your choice. When you're prompted for a CD, click Cancel and exit from the utility.
4. Mount the network installation source that you created in Chapter 2. For the purpose of this lab, create a `/mnt/test` directory, and then use that as the mount point.
5. Run the following command. If it doesn't start the Package Management utility, there may be a problem with the mount or the network source. See Chapter 2 for more information.

```
# redhat-config-packages --tree=/mnt/test
```

6. Try adding the software of your choice. Now see if you're prompted for a CD.

SELF TEST ANSWERS

Adding, Deleting, and Modifying User Accounts

1. ☒ B. The `userdel -r username` command deletes the user as well as all of the files in the `/home/username` directory.
☒ A, C, and D are not real switches for the `userdel` command.

The Basic User Environment

2. ☒ B. The tilde (`~`) is the generic term for each user's home directory. When users are created, these files are transferred from `/etc/skel` to the right home directory.
☒ A and C are not valid locations for KDE configuration files. While the generic KDE configuration files are stored in `/etc/skel`, they do not belong to any particular user (D).

Red Hat Filesystem Basics

3. ☒ D. Journaling filesystems can speed recovery from a crash, which includes what can happen when power to your computer is lost.
☒ A, B, and C are incorrect. The `ext3` filesystem is unrelated to VFAT (A). The `/etc/filesystems` file lists currently supported filesystem formats. While that includes `ext3`, filesystems are not collected by `ext3` (B). CDs are formatted to the `ISO9660` filesystem, not `ext3` (D).

Filesystem Management and the Automounter

4. ☒ B. Information regarding filesystems, including mount options, is stored in `/etc/fstab`.
☒ A, C, and D are incorrect. The `/etc/filesystems` file lists currently supported filesystem formats (A). Group information and passwords are documented in `/etc/group` (C). `/etc/mnttab` (D) is an invalid filename.

Using the Red Hat Package Manager Command and Utility

5. ☒ C, `rpm -i penguin-3.26.i386.rpm`. When installing a package, the `-i` option is used, followed by the name of the RPM file.
☒ D is almost correct. It has an incomplete filename for the RPM, but it would otherwise work. B also does not use the full filename. An asterisk at the end would make it work (as in `penguin*`), but this would install every package starting with the name `penguin`, so it may install more than you want. The `-I` option is invalid with `rpm` (A).

RPM Sources

6. ☒ C. When installing SRPMs, they are, by default, extracted into the `/usr/src/redhat` directory structure.
☒ A, B, and D are nonexistent files and directories.

Network Configuration

7. ☒ A. If networking is enabled, `/etc/sysconfig/network` activates networking, contains the host name, and possibly NIS domain name information as well.
☒ This file does not identify the network device, nor will it contain status information or any chat scripts (B, C, and D).

The Basic Boot Process

8. ☒ D. `init` runs `/etc/rc.d/rc.sysinit`, which performs a number of tasks, including configuring the network and setting up keymapping, swapping, and the hostname.
☒ A is a directory. B, `/etc/inittab`, is the `init` configuration file, which sets the runlevel at startup. However, `/etc/inittab` is not a script. While C, `/etc/rc.d/initdefault` is not a real filename, `initdefault` is a variable within `/etc/inittab` that determines the default runlevel of your system.

Virtual Consoles

9. ☒ B. Press ALT-F4. Each virtual console has an ALT-function key associated with it to move to that virtual console.
☒ A, C, and D normally do not do anything when you are at a regular virtual console.

Other Configuration Tools

10. ☒ A. You can configure Red Hat services of all types in the `/etc/sysconfig` directory.
☒ B, C, and D are all incorrect. Files in `/etc/sysconfig` are not limited to network services or hardware configuration. It has nothing to do with configuring runlevels.

LAB ANSWERS

Lab 1

1. To open a command line interface in the Red Hat GUI, right-click on the desktop. In the pop-up menu that appears, click New Terminal.
2. In the new terminal, open the Service Configuration utility. Run the applicable graphical tool in the background so you can still use this terminal window with the following command:

```
# redhat-config-services &
```

3. Run the `ls /etc/rc3.d` command. Look for the `nfs` start script and record the current order number, 60. If you see only a kill script for `nfs`, it is not active in runlevel 3. If you don't see it at all, you need to install the `nfs-utils` RPM.
4. Deactivate the `nfs` service from level 3. It's easy to do so in the Service Configuration utility. But you have to remember to save your changes with the `File | Save Changes` command.
5. Switch back to the console window and run **chkconfig** to see if it has been deactivated in runlevel 3. If it has, you should see the following result:

```
# chkconfig --list nfs
autoofs  0:off  1:off  2:off  3:off  4:on   5:on   6:off
```

6. Now run the **chkconfig** command to reactivate the `nfs` service. Return to the Service Configuration utility. Add the `nfs` service back in with the following command:

```
# chkconfig --level 3 nfs on
```

7. Now run the following **chkconfig** command to verify that `nfs` is active again in runlevel 3:

```
# chkconfig --list nfs
autoofs  0:off  1:off  2:off  3:on   4:on   5:on   6:off
```

This should show that the `nfs` is started in runlevel 3. You practice using the **chkconfig** command. One way is to redo this lab. Use the service of your choice. Make sure what you see in the GUI Service Configuration utility matches what you see. After each change with the **chkconfig** command, run the `View | Refresh Service List` command to make sure the GUI tool reflects your change.

Lab 2

1. To add the account, enter:

```
# useradd brianr -u 5010 -g nobody -c 'brian rite' -s /bin/bash
```

2. Change the password:

```
# passwd brianr
```

3. Open a new virtual console. For example, to open the second virtual console, press ALT-F1. If you're in the GUI, also press the CTRL key.

4. Log into the new account:

5. What files are there? Include all the hidden files to see the skeleton files copied over from the /etc/skel directory.

```
$ ls -a
```

6. Remove the brianr account using the **userdel** command. Is the brianr home directory gone? What option would have done this for you?

```
# userdel brianr          # leaves the home directory
# userdel -r brianr       # also removes home directory
```

7. Rerun these tasks in the Red Hat User Manager.

Lab 3

Configuring the Automounter on a shared NFS directory is easier than it looks. Before you begin, make sure that you can mount the shared NFS directory from the remote computer. If there's a problem, resolve those first before beginning this lab. Refer to Chapter 2 on creating an NFS Installation Server or Chapter 9 on NFS for more information. If there's no problem with a source on an NFS server with an IP address of 192.168.30.4, you should be able to mount it locally. For example, you can mount a shared remote NFS /mnt/inst directory on an existing empty local /mnt/test directory as follows:

```
# mount -t nfs 192.168.30.4:/mnt/inst /mnt/test
```

Configuring the Automounter is easier than it looks. But first, as with any of these experiments, it's important to back up any files that you're about to edit. In this case, those are the configuration files which govern the autofs daemon, /etc/auto.master and /etc/auto.misc.

One of the fortunate things about the RHEL 3 version of these files is that they contain tips and example commands that you can use and learn from as you work with the Automounter. Use them to your advantage. You can activate the standard commented command in the /etc/auto.master file to activate the Automounter on the /misc directory:

```
/misc      /etc/auto.misc      --timeout=60
```

This command sets a timeout of 60 seconds, depending on details specified in /etc/auto.misc. Activate this command (remove the #), and save your changes to /etc/auto.master.

Open `/etc/auto.misc` in your text editor. The example shown is easy to follow:

```
#linux    -ro,soft,intr    ftp.example.org:/pub/linux
```

Let's assume you're using your NFS installation server from Chapter 2, and that server is at IP address 192.168.30.4. In that case, you'd add the following command to this file:

```
test      -ro,soft,intr    192.168.30.4:/mnt/inst
```

Now you can restart the autofs server; the quickest way is with the following command:

```
# service autofs restart
```

Now when you test the result, you should be able to see the contents of your shared NFS directory.

```
# ls /misc/test
```

You can test the result in a different way. Before the timeout in `/etc/auto.master` expires, the following command should reveal the test subdirectory:

```
# ls /misc
test
```

After the timeout is complete, rerun the `ls /misc` command again. The test subdirectory should no longer be there, which tells you that the timeout specified in `/etc/auto.master` has expired. Please, retry this lab with other shared NFS directories. Remember to restore the original `/etc/auto.master` and `/etc/auto.misc` files when you're done.

Lab 4

Remember, you should not try this exercise if you don't have a separate line for the `/boot` filesystem in `/etc/fstab`. And this assumes that GRUB is your default bootloader.

This should be a self-explanatory exercise. Essentially, you're just making a minor modification to the `/etc/fstab` file. If you do it correctly, Linux should boot normally, even if it does not mount the `/boot` filesystem. The key is within the GRUB configuration file, as the `root` variable in `/boot/grub/grub.conf` doesn't require a mounted filesystem to read your `vmlinuz` kernel or your `mlinix` initial RAM disk files.

Lab 5

The point of this lab is to see how you can use the Package Management utility with a network installation source. You'll be able to install any additional package groups that you need. That can be much more efficient than using the `rpm` command to install the dozens of RPMs associated with some package groups.

The steps I've described work only if your connection to the NFS server works, as described in Chapter 2 as well as Lab 3.