



# I

## RHCE Prerequisites

### CERTIFICATION OBJECTIVES

|      |                                          |      |                              |
|------|------------------------------------------|------|------------------------------|
| I.01 | Basic Linux Knowledge                    | I.09 | Standard Network Services    |
| I.02 | Linux Filesystem Hierarchy and Structure | I.10 | Basic Network Security       |
| I.03 | Basic Commands                           | I.11 | Basic Hardware Knowledge     |
| I.04 | Printing                                 | I.12 | Hardware Compatibility       |
| I.05 | Shells                                   | I.13 | Configuring External Devices |
| I.06 | Basic Security                           | I.14 | Preparing to Install Linux   |
| I.07 | System Administration                    | ✓    | Two-Minute Drill             |
| I.08 | Basic TCP/IP Networking                  | Q&A  | Self Test                    |

**T**he Red Hat exams are an advanced challenge. As both the RHCE and RHCT courses have a number of prerequisites, this book assumes that you know some basics about Linux. This chapter covers the prerequisite topics for Red Hat's RH300 course in a minimum of detail, with references to other books and sources for more information. It also covers the related prerequisites as defined in the Red Hat Exam Prep guide. Unlike in other chapters and other books in this series, the questions include a number of "zingers" that go beyond this chapter's content. That is the only way to see if you have the prerequisite skills necessary for the remaining chapters.

If you're serious about the RHCE or RHCT exams, this chapter should be just a review. In fact, this chapter is far from comprehensive. However, it is okay if you do not feel comfortable with a small number of topics in this chapter. In fact, it's quite natural that many experienced Linux administrators don't use every one of the prerequisite topics in their everyday work. Many candidates are successfully able to fill in the gaps in their knowledge with some self-study and lots of practice.

If you're new to Linux or Unix, this chapter will not be enough for you. It's not possible to detail the commands listed in this chapter, at least in a way that can be understood by newcomers to Linux and other Unix-based operating systems. Such descriptions require several hundred pages in other books, and would take away from the skills that you need for the Red Hat exams. If after reading this chapter, you find a need for more detailed information, please refer to one of the following guides:

- *Red Hat Linux: The Complete Reference, Enterprise Linux & Fedora Edition*, by Richard Petersen and Ibrahim Haddad (McGraw-Hill/Osborne, 2004), provides a detailed step-by-step guide to every part of this operating system. After reading the book you have in your hands, if you want additional exercises in Red Hat Enterprise Linux, this Complete Reference is the book.
- *Hacking Exposed Linux, Second Edition: Linux Security Secrets and Solutions*, by Brian Hatch and James Lee (McGraw-Hill/Osborne, 2003), gives you a detailed look at how you can secure your Linux system and networks in every possible way.
- *Linux Programming: A Beginner's Guide*, by Richard Peterson (McGraw-Hill/Osborne, 2001), takes a fundamental look at the scripts you need to administer

Linux professionally and customize tools such as the GNOME and KDE GUIs for your users.

- *Mastering Red Hat Linux 9*, by Michael Jang (Sybex, 2003), also provides a detailed guide to the operating system that provides the foundation for Red Hat Enterprise Linux 3.

Critical to a Linux administrator is knowledge of one or more text editors to manage the many configuration files on a Linux system. The Linux filesystem hierarchy organizes hardware, drivers, directories, and, of course, files. You need to master a number of basic commands to manage Linux. Printer configuration can be a complex topic. Shell scripts enable you to automate many everyday processes. Security is now a huge issue that Linux can handle better than other operating systems, both locally and on larger networks such as the Internet.

As an administrator, you need a good knowledge of basic system administration commands, TCP/IP configuration requirements, and standard network services. While the RHCE and RHCT exams are by and large not hardware exams, some basic hardware knowledge is a fundamental requirement for any Linux administrator.

This is not a book for beginners to Linux/Unix-type operating systems. Some of what you read in this chapter may be unfamiliar. Use this chapter to create a list of topics that you may need to study further. In some cases, you'll be able to get up to speed with the material in other chapters. But if you have less experience with Linux or another Unix-type operating system, you may want to refer to the aforementioned books.

If you're experienced with other Unix-type operating systems such as Solaris, AIX, or HP-UX, you may need to leave some defaults at the door. When Red Hat developed their Linux distribution, they did a number of things that may not seem completely consistent with Unix standards. When I took RH300, some students with these backgrounds had difficulties with the course and the RHCE exam.

For the purpose of this book, I'll be running most commands as the Linux administrative user, root. Logging in as the root user is normally discouraged unless you're administering a computer. However, since the RHCE and RHCT exams tests your administrative skills, it's appropriate to run commands in this book as the root user.

There are several additional prerequisite skills as defined in the Red Hat Exam Prep guide. They are straightforward, but belong in other chapters. In Chapter 6, I'll show you how to configure an e-mail client and use Mozilla to browse online. In Chapter 7, I'll show you how to use the **lftp** command as a client.

## INSIDE THE EXAM

### Prerequisite Skills

For the RHCE and RHCT exams, the skills outlined in this chapter are generally minimum requirements. For example, while you might not see a question on the vi editor, you will need to know how to use vi on at least the Troubleshooting and System Maintenance exam. While it's not a requirement to know how to pipe the output of **dmesg** to the **less** command, it's a very useful tool that can help you identify problems on the Troubleshooting or Installation portions of either exam.

### Using Other Versions of Red Hat

For the purpose of this chapter, you can use Red Hat Linux 9 or Fedora Linux 1 to test your knowledge of basic commands. There are trivial variations in a few commands. For example, fdisk now interprets the size of a partition slightly differently. (For more information, see the RELEASE-NOTES-i386-en file in the /usr/share/doc/redhat-release-3 directory.)

For those of you with more advanced hardware, the Red Hat exams are based on PCs built with Intel 32-bit CPUs. That means you'll be using the Linux kernel and associated software that has been customized for this CPU.

In future chapters, I will highlight some of the differences between Red Hat Enterprise

But remember, there is more than one way to do most everything in Linux. While it's a good idea to learn all of these "prerequisite" skills, you don't have to know everything in this chapter. In most cases, it's okay if you have other ways to edit or otherwise configure your RHEL 3 system. As there is no longer a multiple choice component to the Red Hat exams, don't worry about the dozens of switches for certain commands. Focus on results, not trivia.

Linux 3 (RHEL 3) and Red Hat Linux 9. This should help students who are using the freely available Red Hat Linux 9 distribution to study for the RHCE and RHCT exams. Generally, you'll be able to update your Red Hat Linux 9 system to the RHEL 3 software. If you're using Fedora Linux 1 or above, the process may be more difficult, as RHEL 3 is based on Red Hat Linux 9.

Fortunately, there is a freely available version of RHEL 3 from the Community Linux group, at [www.caosity.org](http://www.caosity.org). It's available as packages processed from the source code or as a three-CD set of ISO images. It includes all available non-proprietary packages from Red Hat Enterprise Linux 3 server.

**CERTIFICATION OBJECTIVE 1.01****Basic Linux Knowledge**

Linux and Unix are managed through a series of text files. Linux administrators do not normally use graphical editors to manage these configuration files. Editors such as WordPerfect, StarOffice, and yes, even Microsoft Word normally save files in a binary format that Linux can't read. Popular text editors for Linux configuration files include emacs, pico, joe, and vi.

**The Visual Editor**

While emacs may be the most popular text editor in the world of Linux, every administrator needs at least a basic knowledge of vi. While emacs may be more popular and flexible, vi may help you save a broken system. If you ever have to restore a critical configuration file using an emergency boot floppy, vi is probably the only editor that you'll have available.

In reality, RHEL 3 uses an enhanced version of the vi editor, known as vim. It adds color to different types of variables in key configuration files such as /etc/fstab. All regular vi commands work in the vim editor. While you can remove the coloring in the vim editor with the **:nohl** command, this detail does not affect your ability to manage your Linux system.

You need to know how to restore your system from a rescue floppy, which does not have enough room to carry any editor other than vi. However, the Red Hat Exam Prep guide suggests that you need to be able to “use the rescue environment provided by the first installation CD,” which does include other console editors. If you use the joe text editor, it's available when you boot from the rescue floppy.



***This section provides only the briefest of introductions to the vi editor. If you boot in rescue mode and try to start emacs or pico, that starts the joe editor instead.***

You should know how to use the two basic modes of vi: command and insert. When you use vi to open a file, it opens in command mode. Some of the commands start insert mode. Opening a file is easy: just use the **vi filename** command. By default, this starts vi in command mode. An example of vi with the /etc/passwd file is shown in Figure 1-1.

**FIGURE 1-1**

The vi editor

```

root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
news:x:9:13:news:/etc/news:
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:./sbin/nologin
rpm:x:37:37:./var/lib/rpm:/sbin/nologin
vcsa:x:69:69:virtual console memory owner:/dev:/sbin/nologin
nscd:x:28:28:NSCD Daemon:./sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
rpc:x:32:32:Portmapper RPC user:./sbin/nologin
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
nfsnobody:x:65534:65534:Anonymous NFS User:/var/lib/nfs:/sbin/nologin
"/etc/passwd" 37L, 1680C

```

The following is only the briefest of introductions to the vi editor. For more information, there are a number of books available, as well as an extensive manual formatted as a HOWTO available from the Linux Documentation Project at [www.tldp.org](http://www.tldp.org).

## vi Command Mode

In command mode, you can do everything you need to do to a text file except edit it. The options in command mode are broad and varied, and they are the subject of a number of book-length texts. Using vi requires seven critical command skills:

- **Open** To open a file in the vi editor from the command line interface, run the **vi filename** command.
- **Search** Start with a backslash, followed by the search term. Remember, Linux is case sensitive, so if you're searching for "Michael" in `/etc/passwd`, use the `/Michael` (not `/michael`) command.
- **Write** To save your changes, use the **w** command. You can combine commands; for example, `:wq` writes the file and exits vi.
- **Close** To leave vi, use the **:q** command.
- **Abandon** If you want to abandon any changes that you've made, use the **:q!** command.

- **Edit** You can use a number of commands to edit files through vi, such as **x**, which deletes the currently highlighted character, **dw**, which deletes the currently highlighted word, and **dd**, which deletes the current line. Remember, **p** places text from a buffer, and **U** restores text from a previous change.
- **Insert** A number of commands allow you to start insert mode, including **i** to start inserting text at the current position of the editor, and **o** to open up a new line immediately below the current position of the cursor.

## Basic Text Editing

In modern Linux systems, editing files with vi is easy. Just use the normal navigation keys (arrow keys, PAGE UP, and PAGE DOWN), and then one of the basic commands such as **i** or **o** to start vi's insert mode, and type your changes directly into the file.

When you're finished with insert mode, press the ESC key to return to command mode. You can then save your changes, or abandon them and exit vi.

### EXERCISE 1-1

## Using vi to Create a New User

In this exercise, you'll create a new user by editing the `/etc/passwd` file with the vi text editor. While there are other ways to create new Linux users, this exercise helps you verify your skills with vi and at the command line interface.

1. Open a Linux command line interface. Log in as the root user, and type the **vi /etc/passwd** command.
2. Navigate to the last line in the file. As you should already know, there are several ways to do this in command mode, including the DOWN ARROW key, the PAGE DOWN key, the **G** command, or even the **K** key.
3. Make one copy of this line. If you're already comfortable with vi, you should know that you can copy an entire line to the buffer with the **yy** command. This "yanks" the line into buffer. You can then restore, or put that line as many times as desired with the **p** command.
4. Change the username, user ID, group ID, user comment, and home directory for the new user. If you understand the basics of Linux or Unix, you'll understand their locations on each line in the `/etc/passwd` file. For example, in Figure 1-2, this corresponds to `tb`, `501`, `501`, `Tony Blair`, and `/home/tb`. Make sure the username also corresponds to the home directory.

**FIGURE 1-2**

Adding a  
new user in  
/etc/passwd

```
rpm:x:37:37::/var/lib/rpm:/sbin/nologin
vcsa:x:69:69:virtual console memory owner:/dev:/sbin/nologin
nscd:x:28:28:NSCD Daemon:/:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/ssh:/sbin/nologin
rpc:x:32:32:Portmapper RPC user:/:/sbin/nologin
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
nfsnobody:x:65534:65534:Anonymous NFS User:/var/lib/nfs:/sbin/nologin
mailnull:x:47:47::/var/spool/mqueue:/sbin/nologin
smmsp:x:51:51:/var/spool/mqueue:/sbin/nologin
pcap:x:77:77:/var/arpwatch:/sbin/nologin
xfs:x:43:43:X Font Server:/etc/X11/fs:/sbin/nologin
ntp:x:38:38:/etc/ntp:/sbin/nologin
gdm:x:42:42:/var/gdm:/sbin/nologin
desktop:x:80:80:desktop:/var/lib/menu/kde:/sbin/nologin
apache:x:48:48:Apache:/var/www:/sbin/nologin
webalizer:x:67:67:Webalizer:/var/www/usage:/sbin/nologin
squid:x:23:23:/var/spool/squid:/sbin/nologin
postfix:x:89:89:/var/spool/postfix:/sbin/nologin
named:x:25:25:Named:/var/named:/sbin/nologin
netdump:x:34:34:Network Crash Dump user:/var/crash:/bin/bash
mj:x:500:500:/home/mj:/bin/bash
tb:x:501:501:/home/tb:/bin/bash
~
```

5. Return to command mode by pressing the ESC key. Save the file with the **wq** command.
6. As the root user, run the **root newuser** command. Assign the password of your choice to the new user.

## CERTIFICATION OBJECTIVE 1.02

# Linux Filesystem Hierarchy and Structure

Everything in Linux can be reduced to a file. Partitions are associated with *filesystem device nodes* such as `/dev/hda1`. Hardware components are associated with node files such as `/dev/modem`. Detected devices are documented as files in the `/proc` directory. The Filesystem Hierarchy Standard (FHS) is the official way to organize files in Unix and Linux directories. As with the other sections, this introduction provides only the most basic overview of the FHS. More information is available from the official FHS homepage at [www.pathname.com/fhs](http://www.pathname.com/fhs).

## Linux Filesystems and Directories

Several major directories are associated with all modern Unix/Linux operating systems. These directories organize user files, drivers, kernels, logs, programs, utilities, and more into different categories. The standardization of the FHS makes it easier for users of other Unix-based operating systems to understand the basics of Linux.

Every FHS starts with the root directory, also known by its symbol, the single forward slash (/). All of the other directories shown in Table 1-1 are subdirectories of the root directory. Unless they are mounted separately, you can also find their files on the same partition as the root directory.

Mounted directories are often known as volumes, which can span multiple partitions. However, while the root directory (/) is the top-level directory in the FHS, the root user's home directory (/root) is just a subdirectory.



***In Linux, the word “filesystem” has several different meanings. For example, a filesystem can refer to the FHS, an individual partition, or a format such as ext3. A filesystem device node such as /dev/sda1 represents the partition on which you can mount a directory.***

## Media Devices

Several basic types of media are accessible to most PCs, including IDE hard disks, floppy drives, CD/DVD drives, and the various standards of SCSI devices. Other media are accessible through other PC ports, including serial, parallel, USB, and IEEE 1394. You can use Linux to manage all of these types of media.

Most media devices are detected automatically. Linux may require a bit of help for some devices described in Chapter 2. But in the context of the Linux FHS, media devices, like all others, are part of the /dev directory. Typical media devices are described in Table 1-2.

## Making Reference to Devices in /dev

Take a look at the files in the /dev directory. Use the `ls -l /dev | more` command. Scroll through the long list for a while. Are you confused yet? Well, there's a method to this madness. Some devices are linked to others, and that actually makes it easier to understand what is connected to what. For example, the virtual device files /dev/mouse and /dev/modem are easier to identify than the true device files. Generally, these devices are

**TABLE 1-1**

Basic Filesystem  
Hierarchy  
Standard  
Directories

| Directory | Description                                                                                                                                                                                                   |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| /         | The root directory, the top-level directory in the FHS. All other directories are subdirectories of root, which is always mounted on some partition.                                                          |
| /bin      | Essential command line utilities. Should not be mounted separately; otherwise, it could be difficult to get to these utilities when using a rescue disk.                                                      |
| /boot     | Includes Linux startup files, including the Linux kernel. The default, 100MB, is usually sufficient for a typical modular kernel and additional kernels that you might install during the RHCE or RHCT exams. |
| /dev      | Hardware and software device drivers for everything from floppy drives to terminals. Do not mount this directory on a separate partition.                                                                     |
| /etc      | Most basic configuration files.                                                                                                                                                                               |
| /home     | Home directories for almost every user.                                                                                                                                                                       |
| /lib      | Program libraries for the kernel and various command line utilities. Do not mount this directory on a separate partition.                                                                                     |
| /mnt      | The mount point for removable media, including floppy drives, CD-ROMs, and Zip disks.                                                                                                                         |
| /opt      | Applications such as the WordPerfect or OpenOffice.org Office suites.                                                                                                                                         |
| /proc     | Currently running kernel-related processes, including device assignments such as IRQ ports, I/O addresses, and DMA channels.                                                                                  |
| /root     | The home directory of the root user.                                                                                                                                                                          |
| /sbin     | System administration commands. Don't mount this directory separately.                                                                                                                                        |
| /tmp      | Temporary files. By default, Red Hat Enterprise Linux deletes all files in this directory periodically.                                                                                                       |
| /usr      | Small programs accessible to all users. Includes many system administration commands and utilities.                                                                                                           |
| /var      | Variable data, including log files and printer spools.                                                                                                                                                        |

automatically linked to the actual device files during Linux installation. For example, if you have a mouse and modem installed, the following commands illustrate possible links between these components and the actual device files:

```
# ls -l /dev/mouse
lrwxrwxrwx 1 root root 5 Apr 18 12:17 /dev/mouse -> psaux
# ls -l /dev/modem
lrwxrwxrwx 1 root root 5 Apr 18 12:17 /dev/modem -> /dev/ttyS0
```

TABLE 1-2

Media Devices

| Media Device                         | Device File                                                                                                            |
|--------------------------------------|------------------------------------------------------------------------------------------------------------------------|
| Floppy drive                         | First floppy (Microsoft A: drive) = /dev/fd0<br>Second floppy (Microsoft B: drive) = /dev/fd1                          |
| IDE hard drive<br>IDE CD/DVD drive   | First IDE drive = /dev/had<br>Second IDE drive = /dev/hdb<br>Third IDE drive = /dev/hdc<br>Fourth IDE drive = /dev/hdd |
| SCSI hard drive<br>SCSI CD/DVD drive | First SCSI drive = /dev/sda<br>Second SCSI drive = /dev/sdb<br>...<br>Twenty-seventh SCSI drive = /dev/sdaa and so on  |
| Parallel port drives                 | First IDE drive = /dev/pd1<br>First tape drive: /dev/pt1                                                               |
| USB drives                           | Varies widely                                                                                                          |
| IEEE 1394 drives                     | IEEE 1394 (a.k.a. FireWire, iLink) is actually a SCSI standard, so these are controlled in Linux as SCSI devices       |

The first output shows that /dev/mouse is linked directly to the PS/2 device driver port, and that /dev/modem is linked directly to the first serial port, which corresponds to COM1 in the Microsoft world.

## Filesystem Formatting and Checking

Three basic tools are available to manage the filesystem on various partitions: fdisk, mkfs, and fsck. They can help you configure partitions as well as create, and then check and repair, different filesystems. As with the rest of this chapter, this section covers only the very basics; for more information, see the man page associated with each respective command tool.

### fdisk

The Linux fdisk utility is a lot more versatile than its Microsoft counterpart. But to open it, you need to know the device file associated with the hard drive that you want to change. Identifying the hard disk device file is covered in Chapter 2. Assuming you want to manage the partitions on the first SCSI hard disk, enter the following command:

```
# fdisk /dev/sda
```

# INSIDE THE EXAM

## Running as Root

Throughout the book, I'm assuming that you're running commands after having logged in as the root user. While it may not be the best practice on the job, it can save you a little bit of time on the RHCT and RHCE exams. For example, if you've logged in as a regular user, you'd start fdisk with the `/sbin/fdisk` command. This applies even if you've taken administrative privileges with the `su` command. (I know, you could take administrative privileges with the root user `PATH` with the `su - root` command, but time is of the essence on these exams.) On the other hand, if you log in as the root user, the default root `$PATH` variable, means all you need to type is **fdisk**.

As you can see in Figure 1-3, the fdisk utility is flexible. Some key fdisk commands are described in Table 1-3.

**FIGURE 1-3**

Linux fdisk  
commands;  
p returns the  
partition table

```
[root@berkeley root]# fdisk /dev/sda
Command (m for help): m
Command action
  a toggle a bootable flag
  b edit bsd disklabel
  c toggle the dos compatibility flag
  d delete a partition
  l list known partition types
  m print this menu
  n add a new partition
  o create a new empty DOS partition table
  p print the partition table
  q quit without saving changes
  s create a new empty Sun disklabel
  t change a partition's system id
  u change display/entry units
  v verify the partition table
  w write table to disk and exit
  x extra functionality (experts only)

Command (m for help): p

Disk /dev/sda: 4294 MB, 4294967296 bytes
255 heads, 63 sectors/track, 522 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1  *           1           13       104391   83  Linux
/dev/sda2             14           474      3702982+  83  Linux
/dev/sda3           475           522      385560    82  Linux swap

Command (m for help):
```

**TABLE 1-3**

Important fdisk Options

| fdisk Command | Description                                                                                                                      |
|---------------|----------------------------------------------------------------------------------------------------------------------------------|
| a             | Allows you to specify the bootable Linux partition (with /boot).                                                                 |
| l             | Lists known partition types; fdisk can create partitions that conform to any of these filesystems.                               |
| n             | Adds a new partition; works only if there is free space on the disk that hasn't already been allocated to an existing partition. |
| q             | Quits without saving any changes.                                                                                                |
| t             | Changes the partition filesystem.                                                                                                |

**mkfs**

To format a Linux partition, apply the **mkfs** command. It allows you to format a partition to a number of different filesystems. To format a typical partition such as /dev/hda2 to the current Red Hat standard, the third extended filesystem, run the following command:

```
# mkfs -t ext3 /dev/hda2
```

The **mkfs** command also serves as a “front-end,” depending on the filesystem format. For example, if you’re formatting a Red Hat standard ext3 filesystem, **mkfs** by itself automatically calls the **mkfs.ext3** command. Therefore, if you’re reformatting an ext3 filesystem, the following command is sufficient:

```
# mkfs /dev/hda2
```



**Be careful with the *mkfs* command. First, back up any data on the subject partition and computer. This command erases all data on the specified partition.**

**fsck**

The **fsck** command is functionally similar to the Microsoft **chkdsk** command. It analyzes the specified filesystem and performs repairs as required. Assume you’re having problems with files in the /var directory, which happens to be mounted on /dev/hda7. If you want to run **fsck**, unmount that filesystem first. In some cases, you may need to go into single-user mode with the **init 1** command before you can unmount a filesystem. To unmount, analyze, then remount the filesystem noted in this section, run the following commands:

```
# umount /var
# fsck -t ext3 /dev/hda7
# mount /dev/hda7 /var
```

The **fsck** command also serves as a “front-end,” depending on the filesystem format. For example, if you’re formatting an ext2 or ext3 filesystem, **fsck** by itself automatically calls the **e2fsck** command (which works for both filesystems). Therefore, if you’re checking an ext3 filesystem, once you unmount it with the **umount** command, the following command is sufficient:

```
# fsck /dev/hda7
```

## Multiple Partitions with One Filesystem

The Logical Volume Manager (LVM) enables you to set up one filesystem on multiple partitions. For example, assume you’re adding more users and are running out of room in your /home directory. You don’t have any unpartitioned space available on your current hard disk.

With the LVM, all you need to do is add another hard disk, configure some partitions, back up /home, and use the LVM tools to combine the new partition and the one used by /home into a volume set. You may need to install the LVM rpm package. Once it is installed, the steps are fairly straightforward, as described in the following exercise:

### EXERCISE 1-2

#### Creating a New LVM Partition

Logical Volume Management (LVM) is new to RHEL 3 and is more important than the prerequisite skills covered in most of this chapter. The latest available Red Hat Exam Prep guide includes LVM requirements on both parts of the RHCE exam. For more information on LVM, see Chapters 3 and 11.

1. Add a new hard disk.
2. Create new partitions. Assign the Linux LVM filesystem to one or more of these partitions. This can be easily done with the Linux **fdisk** utility.
3. Back up /home. Assign the LVM filesystem to that partition.
4. Scan for Linux LVM filesystems with the **vgscan** utility, to create a database for other LVM commands.
5. Create volumes for the set with the **pvcreate /dev/partition** command.

6. Add the desired volumes to a specific volume group with the **vgcreate** *groupname /dev/partition1 /dev/partition2 ...* command.
  7. Now you can create a logical volume. Use the **lvcreate -L xyM -n volname groupname** command, where *xy* is the size of the volume, and *groupname* is the volume group name from the previous step.
  8. Finally, you can format the logical volume with the **mkfs** command for the desired filesystem (usually ext2 or ext3), using the device name returned by the **lvcreate** command.
- 

## Mounting Partitions

The **mount** command can be used to attach local and network partitions to specified directories. Mount points are not fixed; you can mount a CD drive or even a Samba share to any empty directory where you have appropriate permissions.

There are standard mount points based on the FHS. The following commands mount a floppy with the VFAT filesystem, a CD formatted to the ISO 9660 filesystem, and a Zip drive. The devices may be different on your system; if in doubt, look through the startup messages with **dmesg | less**.

```
# mount -t vfat /dev/fd0 /mnt/floppy
# mount -t iso9660 /dev/cdrom /mnt/cdrom
# mount /dev/sdc
```

## exam

### Watch

*This section covers only the most basic of commands that you can use in Linux. It describes only a few of the things that you can do with each command. Unfortunately, a full discussion would require several hundred more pages. Expect to know considerably*

*more about commands for the RHCE and RHCT exams. If you feel a need for a more solid grounding in basic commands, read **Red Hat Linux: The Complete Reference, Enterprise Linux & Fedora Edition**, by Richard Petersen and Ibrahim Haddad.*

## CERTIFICATION OBJECTIVE 1.03

### Basic Commands

Linux was developed as a clone of Unix, which means that it has the same functionality with different source code. And the essence of both operating systems is at the command line. Basic commands for file manipulation and filters are available to help you do more with a file.

### Basic File Operations

Two basic groups of commands are used to manage Linux files. One group helps you get around Linux files and directories. The other group actually does something creative with the files. Remember, in any Linux file operation, you can take advantage of the `HISTORY` (this is capitalized because it's a standard environment variable) of previous commands, as well as the characteristics of command completion, which allow you to use the `TAB` key almost as a wildcard to complete a command or a filename, or give you the options available in terms of the absolute path.

Almost all Linux commands include switches, options that allow you to do more. Few are covered in this chapter. If you're less familiar with any of these commands, use their man pages. Study the switches. Try them out! Only with practice, practice, and more practice can you really understand the power behind some of these commands.

### Basic Navigation

Everything in Linux can be reduced to a file. Directories are special types of files that serve as containers for other files. Drivers are files. As discussed earlier, devices are special types of files. The nodes associated with USB hardware are just files. And so on. To navigate around these files, you need some basic commands to tell you where you are, what is there with you, and how to move around.

**The Tilde (~)** But first, every Linux user has a home directory. You can use the tilde (~) to represent the home directory of any currently logged on user. For example, if your username is `tb`, your home directory is `/home/tb`. If you've logged in as the root user, your home directory is `/root`. Thus, the effect of the `cd ~` command depends on your username. For example, if you've logged in as user `mj`, the `cd ~` command brings you to the `/home/mj` directory. If you've logged in as the root user, this command brings you to the `/root` directory.

**Paths** There are two path concepts you need to know when you work with Linux directories: absolute paths and relative paths. An absolute path describes the complete directory structure based on the top level directory, root (/). A relative path is based on the current directory, also known as the present working directory. Relative paths do not include the slash in front.

The difference between an absolute path and a relative one is important. Especially when you're creating a script, absolute paths are essential. Otherwise, scripts executed from other directories may lead to unintended consequences.

**pwd** In many configurations, you may not know where you are relative to the root (/) directory. The **pwd** command, which is short for present working directory, can tell you, relative to root (/). Once you know where you are, you can know if you need to move to a different directory.

**cd** It's easy to change directories in Linux. Just use **cd** and cite the absolute path of the desired directory. If you use the relative path, just remember that your final destination depends on the present working directory.

**ls** The most basic of commands is to list the files in the current directory. But the Linux **ls** command, with the right switches, can be quite powerful. The right kind of **ls** can tell you everything about a file, such as creation date, last access date, and size. It can help you organize the listing of files in just about any desired order. Important variations on this command include **ls -a** to reveal hidden files, **ls -l** for long listings, and **ls -li** for inode numbers.

## Looking for Files

There are two basic commands for file searches: **find** and **locate**.

**find** The **find** command searches through directories and subdirectories for a desired file. For example, if you wanted to find the directory with the XF86Config GUI configuration file, you could use the following command, which would start the search in the root directory:

```
# find / -name XF86Config
```

But this search on my older laptop computer with a 200 MHz CPU took several minutes. Alternatively, if you know that this file is located in the /etc subdirectory tree, you could start in that directory with the following command:

```
# find /etc -name XF86Config
```

## exam

### Watch

*The first time I started Linux during the RHCE exam, I ran this script. I could then use **locate** to quickly find the files that I needed.*

**locate** If this is all too time consuming, RHEL 3 includes a default database of all files and directories. Searches with the **locate** command are almost instantaneous. And **locate** searches don't require the full filename. The drawback is that the **locate** command database is normally updated only once each day, as documented in the `/etc/cron.daily/slocate.cron` script.

## Getting into the Files

Now that you see how to find and get around different files, it's time to start reading, copying, and moving the files around. Most Linux configuration files are text files. Linux editors are text editors. Linux commands are designed to read text files. If in doubt, you can check the file types in the current directory with the `file *` command.

**cat** The most basic command for reading files is **cat**. The `cat filename` command scrolls the text within the *filename* file. It also works with multiple filenames; it concatenates the filenames that you might list as one continuous output to your screen.

**less and more** Larger files demand a command that can help you scroll through the file text at your leisure. Linux has two of these commands: **more** and **less**. With the `more filename` command, you can scroll through the text of a file, from start to finish, one screen at a time. With the `less filename` command, you can scroll in both directions through the same text with the PAGE UP and PAGE DOWN keys. Both commands support vi-style searches.

**head and tail** The **head** and **tail** commands are separate commands that work in essentially the same way. By default, the `head filename` command looks at the first 10 lines of a file; the `tail filename` command looks at the last 10 lines of a file. You can specify the number of lines shown with the `-nxy` switch. Just remember to avoid the space when specifying the number of lines; for example, the `tail -n15 /etc/passwd` command lists the last 15 lines of the `/etc/passwd` file.

## Creating Files

A number of commands are used to create new files. Alternatively, you can let a text editor such as vi create a new file for you.

**cp** The **cp** (copy) command allows you to take the contents of one file and place a copy with the same or different name in the directory of your choice. For example, the **cp file1 file2** command takes the contents of *file1* and saves the contents in *file2*. One of the dangers of **cp** is that it can easily overwrite files in different directories, without prompting you to make sure that's what you really wanted to do.

**mv** While you can't rename a file in Linux, you can move it. The **mv** command essentially puts a different label on a file. For example, the **mv file1 file2** command changes the name of *file1* to *file2*. Unless you're moving the file to a different partition, everything about the file, including the inode number, remains the same.

**ln** You can create a linked file. As discussed earlier, linked files are common with device files such as */dev/modem* and */dev/mouse*. They're also useful to make sure that multiple users have a copy of the same file in their directories. Hard links include a copy of the file. As long as the hard link is made within the same partition, the inode numbers are identical. You could delete a hard-linked file in one directory, and it would still exist in the other directory. For example, the following command creates a hard link from the actual Samba configuration file to *smb.conf* in the local directory:

```
# ln smb.conf /etc/samba/smb.conf
```

On the other hand, a soft link serves as a redirect; when you open up a file created with a soft link, you're directed to the original file. If you delete the original file, the file is lost. While the soft link is still there, it has nowhere to go. The following command is an example of how you can create a soft link:

```
# ln -s smb.conf /etc/samba/smb.conf
```

## File Filters

Linux is rich in commands that can help you filter the contents of a file. There are simple commands to help you search, check, or sort the contents of a file. And there are special files that contain others; these container files are known colloquially as "tarballs," which is an alternative to the Red Hat Package Manager.



**Tarballs are a common way to distribute Linux packages. They are normally distributed in a compressed format, with a .tar.gz or .tgz file extension, consolidated as a package in a single file. In this respect, they are similar to Microsoft-style compressed zip files.**

## sort

You can sort the contents of a file in a number of ways. By default, the **sort** command sorts the contents in alphabetical order depending on the first letter in each line. For example, the **sort /etc/passwd** command would sort all users (including those associated with specific services and such) by username.

## grep and egrep

The **grep** command uses a search term to look through a file. It returns the full line that contains the search term. For example, **grep 'Michael Jang' /etc/passwd** looks for the name of this author in the `/etc/passwd` file.

The **egrep** command is more forgiving; it allows you to use some unusual characters in your search, including `+`, `?`, `|`, `(`, and `)`. While it's possible to set up **grep** to search for these characters with the help of the backslash, the command can be awkward.

**The *locate* command is essentially a specialized version of the *grep* command, which uses the *slocate* command-based database of files on your Linux computer.**

## wc

The **wc** command, short for word count, can return the number of lines, words, and characters in a file. The **wc** options are straightforward; for example, **wc -w filename** returns the number of words in that file.

## sed

The **sed** command, short for stream editor, allows you to search for and change specified words or even text streams in a file. For example, the following command changes the first instance of the word “Windows” with “Linux” in each line of the file `opsys`, and writes the result to the file `newopsys`:

```
# sed 's/Windows/Linux' opsys > newopsys
```

However, this may not be enough. If there's more than one instance of “Windows” in a line in the `opsys` file, it does not change the second instance of that word. But you can fix this by adding a “global” suffix:

```
# sed 's/Windows/Linux/g' opsys > newopsys
```

## awk

The **awk** command, named for its developers (Aho, Weinberger, and Kerrigan), is more of a database management command. It can identify lines with a key word, and read



out the text from a specified column in that line. A common example is with the `/etc/passwd` file. For example, the following command will read out the username of every user with a “Mike” in the comment column:

```
# awk '/Mike/ {print $1}' /etc/passwd
```

## Administrative Commands

You’ll work with a number of administrative commands in this book. But every budding Linux administrator should be familiar with at least two basic administrative commands: **ps** and **who**.

### ps

It’s important to know what’s running on your Linux computer. The **ps** command has a number of critical switches. When trying to diagnose a problem, it’s common to get the fullest possible list of running processes, then look for a specific program. For example, if the Mozilla Web browser were to suddenly crash, you’d want to kill any associated processes. The **ps aux | grep mozilla** command could then help you identify the process(es) that you need to kill.

### who and w

If you want to know what users are currently logged into your system, use the **who** command or the **w** command. This can help you identify the usernames of those who are logged in, their terminal connections, their times of login, and the processes that they are running.



***If you suspect that a username has been compromised, use the **w** command to check currently logged on users. Look at the terminal. If the user is in the office but the terminal indicates a remote dial-in connection, be suspicious. The **w** command can also identify the current process being run by that user.***

## Wildcards

Sometimes you may not know the exact name of the file or the exact search term. That is when a wildcard is handy. The basic wildcards are shown in Table 1-4.



***Wildcards are sometimes known in the Linux world as globbing.***

| TABLE 1-4 |                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Wildcard  | Description                                                                                                                                                                                                                                                                                                                                                                                                       |
| *         | Any number of alphanumeric characters (or no characters at all). For example, the <code>ls ab*</code> command would return the following filenames, assuming they exist in the current directory: <code>ab</code> , <code>abc</code> , <code>abcd</code> .                                                                                                                                                        |
| ?         | One single alphanumeric character: For example, the <code>ls ab?</code> command would return the following filenames, assuming they exist in the current directory: <code>abc</code> , <code>abd</code> , <code>abe</code> .                                                                                                                                                                                      |
| []        | A range of options. For example, the <code>ls ab[123]</code> command would return the following filenames, assuming they exist in the current directory: <code>ab1</code> , <code>ab2</code> , <code>ab3</code> . Alternatively, the <code>ls ab[X-Z]</code> command would return the following filenames, assuming they exist in the current directory: <code>abX</code> , <code>abY</code> , <code>abZ</code> . |

CERTIFICATION OBJECTIVE 1.04

Printing

As of this writing, printers are not always connected or configured during the installation of Red Hat Enterprise Linux. You may have to install printers yourself. The default Red Hat Enterprise Linux 3 print daemon is CUPS, the Common Unix Printing System.

There are two basic ways to configure a printer: first, you can edit the configuration files in the `/etc/cups` directory with a text editor, which can be a difficult process. These files are long, and the language is somewhat obscure, at least on the surface.

Though its support of the Internet Printing Protocol (IPP), CUPS is more suited toward managing printers on a network. CUPS also includes a fairly reliable front-end configuration tool that you can call up in a browser, using TCP/IP port 631.

exam

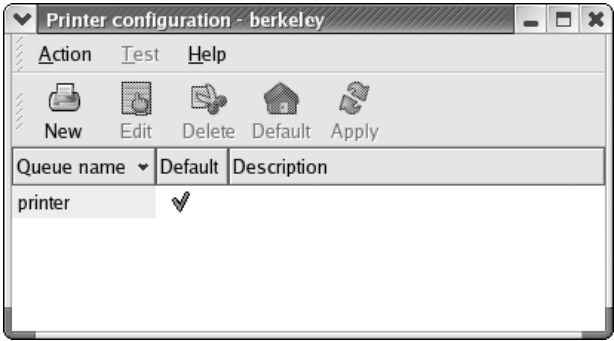
Watch

**Red Hat Enterprise Linux 3 and Fedora Linux do not include any version of the Line Print Daemon (LPD or LPRng), and therefore I believe that coverage of**

**the RHCE or RHCT exams will be limited to CUPS. Nevertheless, LPRng still appears in the RH133 and RH300 course syllabus at the time of this writing.**

**FIGURE 1-4**

The Red Hat Printer Configuration tool



The other method in RHEL 3 is using the Red Hat Printer Configuration tool, which I describe in Chapter 8.

## Adding Printers

The easy way to add a printer is with the Red Hat Printer Configuration tool, which is also known by the command used to start it from a terminal, **redhat-config-printer**. I recommend that you learn to use this GUI tool. Unless you're a CUPS expert, it's a faster way to configure printers on the RHCT or RHCE exams. I show you how to use this utility in Chapter 8.

It's fairly easy to configure printers with this tool; just click New as shown in Figure 1-4 (the computer name is berkeley) and follow the prompts.

## Print Commands

Three basic commands are associated with printing in Linux, as described in Table 1-5.

**TABLE 1-5**

Linux Print Commands

| Command | Description                                                                     |
|---------|---------------------------------------------------------------------------------|
| lpr     | The basic print command. <b>lpr filename</b> prints that file.                  |
| lpq     | Query the print queue for status. <b>lpr -l</b> lists print job numbers.        |
| lprm    | Remove a specific job, usually specified by job number, from the printer queue. |

## CERTIFICATION OBJECTIVE 1.05

### Shells

A *shell* is a user interface. The Linux command shell is the prompt that allows you to interact with your computer with various system commands. With the right file permissions, you can set up commands in scripts to run when you want, even in the middle of the night. Linux shells can process commands in various sequences, depending on how you manage the input and output of each command. The way commands are interpreted is in part determined by variables and parameters associated with each shell. Some of these variables make up the environment that is carried over even if you change from one shell to another.

The default shell in Linux is `bash`, also known as the Bourne Again Shell. A number of other shells are available that are popular with many users. As long as you have installed the appropriate RPMs, users can start any of these shells. As desired, you can change the default shell for individual users in the `/etc/passwd` file.

### Basic Shell Programming

“Real” Linux administrators program their own scripts. They create scripts because they don’t want to sit at their computers all of the time. Scripts can allow Linux to automatically back up directories when nobody is in the office. Scripts can help Linux process databases when few people are using the system.

If you’re not a programmer, don’t worry—this is not as difficult as it sounds. For example, utilities related to the `crontab` command automate the creation of a number of different scripts. The cron system is discussed in more detail in Chapter 6.

If you’re at all familiar with shell commands and programming expressions, you can find some examples of Red Hat Enterprise Linux shell programs in the `/etc/cron.daily` directory.

### Script Execution and Permissions

Any Linux file can be set up as an executable file. Then if the file includes a series of commands that can be interpreted by the shell, the commands in that file are executed. If you want Linux to run a script that you’ve created, you need to assign executable permissions. For additional information on executable files, read the information under the “Basic Security” objective later in this chapter.

## Variables and Parameters

Variables can change. Parameters are set. The bash shell includes a number of standard environment variables. Their default values are shown in the output to the **env** command. One critical variable is the value of **PATH**, which you can check at the command line with the **echo \$PATH** command. The directories listed in **PATH** are automatically searched when you try to run a command. For example, if you want to run the **fdisk** command from the **/sbin** directory, you could do it with the following command:

```
$ /sbin/fdisk
```

However, if the **/sbin** directory were in your **PATH**, you don't need the leading **/sbin** to call out the command; the following would work:

```
$ fdisk
```

You can easily change the **PATH** variable. For example, if you want to add the **/sbin** directory to your **PATH**, just run the following commands:

```
# PATH=$PATH:/sbin
# export PATH
```

The **/sbin** directory is in the default **PATH** for the root user. The most common parameters are the settings associated with Linux configuration files, which are mostly located in the **/etc** directory. For example, the **/etc/resolv.conf** file uses the **nameserver** parameter to represent the DNS servers for your network. This is normally set to the IP address for that DNS server.

### EXERCISE 1-3

---

## Checking the **PATH**

In this exercise, you'll examine the path for a regular and the root user.

1. Log into the Linux command line interface as a regular user. If you're in the GUI, you can get to a command line login with the **CTRL-ALT-F2** command. From the command prompt, run the following command and note the result:

```
$ echo $PATH
```

2. From the regular user command line interface, log in as the superuser. You'll need the root user's password.

```
$ su
Password:
#
```

3. Run the following command again and note the result. Compare it to the result as a regular user. Is there a difference? By default there is no difference on regular Red Hat Linux. There is a slight difference when you repeat this process on Red Hat Enterprise Linux.

```
# echo $PATH
```

4. Log out of Linux. If you followed steps 1, 2, and 3, you'll need to type the **exit** command twice to log out.
5. Now log into Linux as the root user. At the command prompt, run the following command again and note the result:

```
# echo $PATH
```

6. Observe the difference. You'll see more directories in the **PATH** for the root user. Now you can see why many Linux gurus who are doing heavy-duty administrative work log in as the root user. And that is why I also recommend that you log in as the root user during the RHCE or RHCT exams.
- 

## Inherited Environment

It's easy to move from shell to shell. While the default Linux shell is **bash**, many experienced Unix users prefer the Korn shell. Once set, with the **set** command, environment variables stay the same from shell to shell. In contrast, shell variables such as **umask** may change when you move from shell to shell, or even from user to user. For example, **umask** is typically different for regular users and the root user.

## Piping, Input/Output, Error, and Redirection

Linux uses three basic data streams. Data goes in, data comes out, and errors are sent in a different direction. These streams are known as standard input (**stdin**), standard output (**stdout**), and standard error (**stderr**). Normally, input comes from the keyboard and goes out to the screen, while errors are sent to a buffer. Error messages are also sent to the display (as text stream 2). In the following example, *filename* is **stdin** to the **cat** command:

```
# cat filename
```

When you run **cat filename**, the contents of that file are sent to the screen as standard output.

You can redirect each of these streams to or from a file. For example, if you have a program named **database** and a datafile with a lot of data, the contents of that datafile can be sent to the **database** program with a left redirection arrow (**<**). As shown here, datafile is taken as standard input:

```
# database < datafile
```

Standard input can come from the left side of a command as well. For example, if you need to scroll through the boot messages, you can combine the **dmesg** and **less** commands with a pipe:

```
# dmesg | less
```

The output from **dmesg** is redirected as standard input to **less**, which then allows you to scroll through that output as if it were a separate file.

Standard output is just as easy to redirect. For example, the following command uses the right redirection arrow (**>**) to send the standard output of the **ls** command to the file named **filelist**.

```
# ls > filelist
```

You can add standard output to the end of an existing file with a double redirection arrow with a command such as **ls >> filelist**.

If you believe that a particular program is generating errors, redirect the error stream from it with a command like the following:

```
# program 2> err-list
```

## CERTIFICATION OBJECTIVE 1.06

### Basic Security

The basic security of a Linux computer is based on file permissions. Default file permissions are set through the **umask** shell variable. SUID and SGID permissions can give all users access to specific files. Ownership is based on the default user and group IDs of the person who created a file. Managing permissions and ownership involves commands such as **chmod**, **chown**, and **chgrp**.

Users and groups own files. Users and groups have passwords. Security can be enhanced if you configure users and groups in the Shadow Password Suite.

File Permissions

Linux file permissions are straightforward. Take the following output from `ls -l /sbin/fdisk`:

```
-rwxr-xr-x  1 root  root    80236 Sep  1  18:26    /sbin/fdisk
```

The permissions are shown on the left-hand side of the listing. Ten characters are shown. The first character determines whether it's a regular or a special file. The remaining nine characters are grouped in threes, applicable to the file owner (user), the group owner, and everyone else on that Linux system. The letters are straightforward: r=read, w=write, x=execute. These characters are described in Table 1-6.

Key commands that can help you manage the permissions and ownership of a file are **chmod**, **chown**, and **chgrp**. The **chmod** command uses the numeric value of permissions associated with the owner, group, and others. In Linux, permissions are assigned the following numeric values: r=4, w=2, and x=1. For example, if you were crazy enough to give read, write, and execute permissions on `fdisk` to all users, you would run the **chmod 777 /sbin/fdisk** command. The **chown** and **chgrp** commands adjust the user and group owners associated with the cited file.

Users, Groups, and umask

Linux, like Unix, is configured with users and groups. Everyone who uses Linux is set up with a username, even if it's just "guest." Take a look at `/etc/passwd`. One version of this file is shown in Figure 1-5.

As you can see, all kinds of usernames are listed in the `/etc/passwd` file. Even a number of Linux services such as mail, news, nfs, and apache have their own usernames. In any case, the `/etc/passwd` file follows a specific format, described in more detail in Chapter 4. For now, note that the only users shown in this file are mj and tb, their user IDs (UID) and group IDs (GID) are 500 and 501, and their home directories match their usernames. The next user gets UID and GID 502, and so on.

TABLE 1-6

Description of File Permissions

| Position | Description                                                          |
|----------|----------------------------------------------------------------------|
| 1        | Type of file; - = regular file, d=directory, b=device, l=linked file |
| 234      | Permissions granted to the owner of the file                         |
| 567      | Permissions granted to the group owner of the file                   |
| 890      | Permissions granted to all other users on the Linux system           |

**FIGURE I-5**

/etc/passwd

```

rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
nfsnobody:x:65534:65534:Anonymous NFS User:/var/lib/nfs:/sbin/nologin
mailnull:x:47:47::/var/spool/mqueue:/sbin/nologin
smmsp:x:51:51::/var/spool/mqueue:/sbin/nologin
pcap:x:77:77::/var/arpwatch:/sbin/nologin
xfs:x:43:43:X Font Server:/etc/X11/fs:/sbin/nologin
ntp:x:38:38::/etc/ntp:/sbin/nologin
gdm:x:42:42:/var/gdm:/sbin/nologin
desktop:x:80:80:desktop:/var/lib/menu/kde:/sbin/nologin
apache:x:48:48:Apache:/var/www:/sbin/nologin
webalizer:x:67:67:Webalizer:/var/www/usage:/sbin/nologin
squid:x:23:23:/var/spool/squid:/sbin/nologin
postfix:x:89:89:/var/spool/postfix:/sbin/nologin
named:x:25:25:Named:/var/named:/sbin/nologin
netdump:x:34:34:Network Crash Dump user:/var/crash:/bin/bash
mj:x:500:500:Michael Jang:/home/mj:/bin/bash
tb:x:501:501:Tony Blair:/home/tb:/bin/bash
~
~
~
~
~
~

```

Users can change their own passwords with the **passwd** command. The root user can change the password of any user. For example, the **passwd mj** command allows the root user to change user mj's password.

## umask

The way **umask** works in Red Hat Enterprise Linux may surprise you, especially if you're coming from a different Unix style environment. You cannot configure **umask** to automatically allow you to create new files with executable permissions. This is a recent change that promotes security; if fewer files have executable permissions, fewer files are available for a cracker to use to run programs to break through your system.



***In the world of Linux, a hacker is a good person who simply wants to create better software. A cracker is someone who wants to break into your system for malicious purposes.***

Every time you create a new file, the default permissions are based on the value of **umask**. In the past, the value of **umask** cancelled out the value of numeric permissions on a file. For example, if the value of **umask** is 000, the default permissions for any file created by that user are  $777 - 000 = 777$ , which corresponds to read, write, and execute permissions for all users.

When you type the **umask** command, you get a four-number output such as 0245. As of this writing, the first number in the **umask** output is always 0 and is not used.

In the future, this first number may be usable to allow for new files that automatically include the SUID or SGID bits.

Also, no matter what the value of **umask**, new files in Red Hat Enterprise Linux can no longer be automatically created with executable permissions. In other words, a **umask** of 0454 leads to identical permissions on new files as a **umask** of 0545. You need to use commands such as **chmod** to specifically set executable permissions on a file.

## SUID and SGID

Permissions can be a risky business. But you need to give all users access to some programs. To set full read, write, and execute permissions for all users on a Linux system can be dangerous. One alternative is setting the SUID and the SGID permission bits for a file. When active, these bits allow you to configure appropriate permissions on the subject file. For example, one common practice is to set the SUID bit for the KPPP Internet Connection Utility so all users can use it to dial in to the Internet. You can set the SUID bit on this utility with the following command:

```
# chmod u+s /usr/sbin/kppp
```

SGID permissions can be useful when you're setting up a special group of users who need to share files on a specific task or project. This process is discussed in more detail in Chapter 11.

## Shadow Passwords

When you look at the default `/etc/passwd` file, you should see an "x" in the second column. Older versions of Linux had an encrypted version of user passwords in this column. As `/etc/passwd` is accessible to all users, a cracker could copy this file and decrypt everyone's password on a Linux computer. This problem led to the development of the Shadow Password Suite.

## Shadow Password Suite

Historically, all that was needed to manage Linux users and groups was the information included in the `/etc/passwd` and `/etc/group` files. These files included passwords and are by default readable by all users.

The Shadow Password Suite was created to provide an additional layer of protection. It is used to encrypt user and group passwords in shadow files (`/etc/shadow` and `/etc/gshadow`) that are readable only by users with root privileges.

The Shadow Password Suite is now enabled by default in Red Hat Enterprise Linux. Standard commands for creating new users and groups automatically set up encrypted passwords in the Shadow Password Suite files. These commands are described in more detail in Chapter 4.

But if you're restoring a system, you may not have access to these special commands. The old way of creating new users and groups is by editing the `/etc/passwd` and `/etc/group` files directly. Four commands allow you to convert passwords to and from the `/etc/shadow` and `/etc/gshadow` files:

- **pwconv** Converts passwords from `/etc/passwd`. This command works even if some of the passwords are already encrypted in `/etc/shadow`.
- **pwunconv** Opposite of **pwconv**.
- **grpconv** Converts passwords from `/etc/group`. This command works even if some of the passwords are already encrypted in `/etc/gshadow`.
- **grpunconv** Opposite of **grpconv**.

## CERTIFICATION OBJECTIVE 1.07

# System Administration

Most system administration tasks require root or superuser privileges. You should already be familiar with a number of basic Linux system administration commands and files. Standard user files are stored in `/etc/skel`. Daemons are processes that run in the background and run various Linux services. `cron` is a special daemon that can run scripts when you want. It's especially useful for setting up backup jobs in the middle of the night. Logging is a key part of monitoring Linux and any services that you choose to run.

## Superuser

Generally in Linux, a system administrator does everything possible as a normal user. It's a good practice to use superuser privileges only when absolutely necessary. But one time where it's appropriate is during the Red Hat exams. Good administrators will return to being normal users when they're done with their tasks. Mistakes as the root user can disable your Linux system.

There are two basic ways to make this work:

- **su** The superuser command, **su**, prompts you for the root password before logging you in with root privileges. A variation, **su -c**, sets up root privileges for one specific command. Many Red Hat GUI utilities are set up to prompt for the root password before they can be started. One more variation, **su - root**, sets up root privileges with the root user PATH. (Remember to have a space on both sides of the dash in this command.)
- **sudo** The **sudo** command allows users listed in `/etc/sudoers` to run administrative commands. You can configure `/etc/sudoers` to set limits on the root privileges given to a specific user.

However, Red Hat Enterprise Linux provides some features that make working as root somewhat safer. For example, logins using the **ftp** and **telnet** commands to remote computers are disabled by default.

## exam

### Watch

*On the RHCE and RHCT exams, time is of the essence. In general, I recommend that you don't bother logging in as a regular user during these exams. It's faster to log in as the root user. You don't*

*have to remember to invoke the **su** or **sudo** commands, and you gain the advantages of a more liberal PATH variable. While you just save a few seconds with each command, that time can add up.*

## /etc/skel

Basic configuration files for individual users are available in the `/etc/skel` directory. This directory includes a number of hidden files. For a full list, run the **ls -a /etc/skel** command. If you want all future users to get specific files in their home directories, include them here.

The next time you create a regular user, check that person's home directory. For example, if you just created a user named `elizabeth`, run the **ls -a /home/elizabeth** command. Compare the results to the previous command on the `/etc/skel` directory.

## Daemons

A *daemon* is a process that runs in the background. It is resident in your computer's RAM and watches for signals before it goes into action. For example, a network daemon such

as `httpd`, the Linux Web server known as Apache, waits for a request from a browser before it actually serves a Web page.

Daemons are often configured to start automatically when you start Linux. This process is documented at various runlevels in the `/etc/rc.d` directory. Alternatively, you can use a tool such as `ntsysv` to identify and manage the daemons that are started at various Linux runlevels. This is discussed in more detail in Chapter 4.

## Network Service Daemons

Networks don't always work. Sometimes you need to restart a network daemon to implement a configuration change. Red Hat Enterprise Linux provides an easy way to control network service daemons through the scripts in `/etc/rc.d/init.d`. This directory includes scripts that can control installed Linux network services (and more) for everything from the Network File System (NFS) to `sendmail`. The actual daemon itself is usually located in the `/sbin` or `/usr/sbin` directory.

With these scripts, it's easy to start, stop, status, reload, or restart a network daemon. This is useful to implement or test changes that you make to a specific configuration file. For example, if you make a change to the Apache Web server configuration file in `/etc/httpd/conf/httpd.conf`, you can implement the change right away with the `/etc/rc.d/init.d/httpd reload` command. Other switches to these scripts allow you to **stop**, **start**, or **status** these services. Network service management is discussed in more detail in Chapter 9.

### exam

#### Watch

**In Red Hat Enterprise Linux, a simpler way to reload or restart a service in the `letc/rc.d/init.d` directory**

**is with the `service` command. For example, to restart the `httpd` service, you could run the `service httpd restart` command.**

## cron

Perhaps the most important daemon is `cron`, which can be used to execute a command or a series of commands in a script, on a schedule. Red Hat Enterprise Linux already includes a series of scripts that are executed by `cron` on committed schedules in the `/etc/cron.hourly`, `/etc/cron.daily`, `/etc/cron.weekly`, and `/etc/cron.monthly` directories.

## System crontab

The easiest way to set up your own cron jobs is through the crontab file, which can be managed through the **crontab** command. Users can edit their own crontab files with the **crontab -e** command; the root user can configure the crontab for a specific user with the **crontab -u username -e** command.

The general format for a crontab file can be found in the `/etc/crontab` script, which is used to run the scripts in the aforementioned schedule-related directories. A typical crontab entry from that file is

```
42 4 1 * * root run-parts /etc/cron.monthly
```

Five schedule fields appear on the left-hand side of each crontab entry: minute, hour, day of month, month, and day of week. The preceding line is executed at 4:42 A.M. on the first of every month, no matter what day of the week it is.

## Backup and Restore

Hard drives include spinning disks and magnetic media. These are mechanical parts. By definition, all mechanical hard drives will eventually fail. If you're administering a Linux system with multiple users, you do not want to have to hear the complaints of people who know that their data is more important than yours, because you'll know that they are right. Configuring backups involves a number of strategic choices that go beyond Linux.

Using full backups, you can back up the entire drive; using incremental backups, you back up just the data that has changed since the last backup. A wide variety of media are available for backups, including tape drives, writable CD/DVDs, and other hard drives in various RAID configurations. You can back up data locally or over a network. Linux includes a number of quality tools for backups.

It's common to back up through a network to a dedicated backup server. Since you're transferring at least substantial portions of a hard drive during a backup, backups can degrade network performance for other users. So it is best to perform backups when few people are using your Linux system, which in most cases is during the middle of the night. For this reason, it's a common practice to automate backups using the previously discussed cron daemon.

## Tape Backups

Using magnetic tape in Linux depends on the `ftape` system, using "tarballs" to group directories into single compressed backup files. Once it is mounted, it's easy to test a tape

drive; just use the **mt -f /dev/tapedevice** command to status, rewind, or eject the tape. If it's a SCSI tape drive, use the **st** command instead.

Unlike when using regular media, you don't mount a tape; you can actually use switches with the **tar** command to write or restore directly from the tape device. Just cite the appropriate */dev/tapedevice* in the command. Just make sure you can also restore from the backup you've made.

## CD Backups

Backups to CDs are made in a similar fashion, using "iso" files instead of tarballs. The **mkisofs -J -r -T -o /tmp/backhome.iso /home** command can consolidate regular users' home directories from */home* onto a single file. You can then record this file onto the CD with a command such as:

```
# cdrecord -v speed=2 dev=0,0,0 /tmp/backhome.iso
```

You can then store the CD and later restore the files from it just by mounting it as you would any regular CD.

## Hard Drive (RAID) Backups

Hard drive-based backups are based on the system known as the Redundant Array of Independent Disks, or RAID, which is covered in more detail in Chapter 5. There are several versions of RAID that can automatically restore data once you've replaced a broken hard disk.

### tar

The **tar** command was originally developed for archiving data to tape drives. However, it's commonly used today for collecting a series of files, especially from a directory. For example, the following command backs up the information from the */home* directory in the *home.tar.gz* file:

```
# tar czvf home.tar.gz /home
```

This is one of the few commands that does not require a dash in front of the switch. This particular command creates (**c**) an archive, compresses (**z**) it, in verbose (**v**) mode, with the filename (**f**) that follows. Alternatively, you can extract (**x**) from that file with the following command:

```
# tar xzvf home.tar.gz /home
```

### **gzip and bzip2**

The **gzip** and **bzip2** commands are similar—they compress and decompress files, using different algorithms. If you wanted to compress a big picture file, you could do so with one of the following commands:

```
# gzip big.jpg
# bzip2 big.jpg
```

It adds a .gz or a .bz2 algorithm. You can uncompress from these files with the **-d** switch:

```
# gzip -d big.jpg.gz
# bzip2 -d big.jpg.bz2
```

## **System Log File Management**

Log files are controlled by the **syslogd** daemon and organized in the **/etc/syslog.conf** file. It is important to use log files to understand the behavior of your Linux system; deviations may be a sign of problems with recently installed service or a security breach. Basic log files are organized in the **/var/log** directory. For more information on system logs, see Chapter 10.

## **CERTIFICATION OBJECTIVE 1.08**

## **Basic TCP/IP Networking**

TCP/IP is a series of protocols organized in layers, known as a protocol suite. It was developed for Unix and eventually adopted as the standard for communication on the Internet. With IP addresses, it can help you organize your network. Then, there are a number of TCP/IP tools and configurations that can help you manage your network.

As with the previous sections in this chapter, the statements here are oversimplifications. So if you find this section overwhelming, read the references cited at the beginning of the chapter. Linux is built for networking, and there is no practical way to pass either the RHCT or the RHCE exams unless you understand networking in some detail.

# IP Numbers and Classes

Every computer that communicates on a network needs its own IP address. Some addresses are assigned permanently to a particular computer; these are known as *static* addresses. Others are leased from a DHCP server for a limited amount of time; these are also known as *dynamic* IP addresses.

Two standards for IP addresses are in use today, IP version 4 (IPv4) and IP version 6 (IPv6). IPv4 addresses have 32 bits and are set up in octets in dotted decimal notation. The range of possible IPv4 addresses is between 0.0.0.0 to 255.255.255.255. While over four billion IP addresses are available, that is not nearly enough for the current Internet.

IPv6 addresses have 128 bits and are set up in hexadecimal notation. An IPv6 address is normally organized in eight groups of four hexadecimal numbers each, and it may look like 4abe:03e2:c132:69fa:0000:0000:c0b8:2148. More than 340,000,000,000,000,000,000,000,000,000,000,000,000,000,000 IPv6 addresses are available.

To ease the transition, specific IPv6 addresses have been assigned for every one of the four billion IPv4 addresses. There are still over  $3.4 \times 10^{38}$  addresses left over. While actual routing on the Internet now commonly uses IPv6, network configuration in Linux is still normally based on IPv4 addresses.

IPv4 addresses are organized into five different classes, as shown in Table 1-7. The academics among you may note that this is different from the official addresses in each IPv4 class as specified in RFC 1518 from the Internet Engineering Task Force ([www.ietf.org](http://www.ietf.org)). The *assignable* address range includes those IP addresses that can be assigned to a specific computer on a network.

In addition, there are a number of private IP addresses that are not to be assigned to any computer that is directly connected to the Internet. They are associated with network addresses 10.0.0.0, 172.168.0.0, and 192.168.0.0 through 192.168.255.0.

**TABLE 1-7**

IP Address Classes

| Class | Assignable Address Range  | Note                                          |
|-------|---------------------------|-----------------------------------------------|
| A     | 1.1.1.1–126.255.255.254   | Allows networks of up to 16 million computers |
| B     | 128.0.0.1–191.255.255.254 | Allows networks of up to 65,000 computers     |
| C     | 192.0.0.1–223.255.255.254 | Allows networks of up to 254 computers        |
| D     | 224.0.0.1–239.255.255.254 | Reserved for multicasts                       |
| E     | 240.0.0.1–255.255.255.254 | Reserved for experimental use                 |

## IP Addresses Define a Network

Three key IP addresses define a network: the network address, the broadcast address, and the subnet mask. The network address is always the first IP address in a range; the broadcast address is always the last address in the same range. The subnet mask helps your computer define the difference between the two addresses. You can assign IP addresses between the network and broadcast addresses (not including these addresses) to any computer on the network.

As an example, let's define the range of addresses for a private network. Start with the private network address 192.168.122.0. Use the standard subnet mask for a class C network, 255.255.255.0. Based on these two addresses, the broadcast address is 192.168.122.255, and the range of IP addresses that you can assign on that particular network is 192.168.122.1 through 192.168.122.254.

If this is confusing to you in any way, please refer to the IP Sub-Networking Mini-HOWTO of the Linux Documentation Project at [www.tldp.org](http://www.tldp.org).

## Tools and Commands

You have a substantial number of tools available to manage the TCP/IP protocol suite on your Linux computer. Three of the more important commands are **ping**, **ifconfig**, and **netstat**.

### ping

The **ping** command allows you to test connectivity—locally, within your network, and on the Internet. For the purpose of this section, assume your IP address is 192.168.122.43, and the gateway address on your network is 192.168.122.99. If you're having problems connecting to a network, you should use the **ping** command in the following order. The first step is to test the integrity of TCP/IP on your computer:

```
# ping 127.0.0.1
```

Normally, **ping** works continuously on Linux; you'll need to press CTRL-C to stop this command. If you need to see if you're properly connected to your LAN, you should **ping** your own IP address:

```
# ping 192.168.122.43
```

If that works, **ping** the address of another computer on your network. Then start tracing the route to the Internet. **ping** the address for your gateway, in this case, 192.168.122.99.

If possible, **ping** the address of your network's connection to the Internet. And finally, **ping** the address of a computer that you know is *active* on the Internet.

You can substitute host names such as `www.google.com` for an IP address. If the host name doesn't work, there's a problem with the database of host names and IP addresses, more commonly known as a DNS, BIND, or nameserver.

### **ifconfig**

The **ifconfig** command can help you check and configure network adapters. Run the **ifconfig** command by itself to see the detected adapters on your computer. You can also use **ifconfig** to assign IP address or hardware port information as well. For example, if you want to assign IRQ 10 to the second Ethernet adapter, run the following command:

```
# ifconfig eth1 irq 10
```

For more information on **ifconfig**, refer to Chapter 4.

### **netstat**

The **netstat** command is versatile; it can help you see the channels available for network connections, interface statistics, and more. One important version of this command, **netstat -r**, displays routing tables that can tell you if your computer knows where to send a message. More information on this command is available in Chapter 4.

## **Name Resolution**

When I used a static IP address on my high-speed Internet connection, I could sometimes memorize those numbers. But how can anyone memorize the IP addresses of every Web site you need on the Internet? Using four configuration files, Linux can help you translate computer host names to IP addresses.

### **/etc/hosts**

The first database of host names and IP addresses was set up in a static text file, `/etc/hosts`. When there were just a few nodes on the network that eventually turned into the Internet, it was possible to maintain identical `/etc/hosts` files on each computer. Here's a typical line in `/etc/hosts`, which lists the IP address, fully qualified domain name, and alias for one computer connection:

```
192.168.132.32    linux1.mommabears.com  laptop
```

**/etc/resolv.conf**

There are millions of hosts on the Internet. Even if it were possible to collect all domain names and IP addresses into a `/etc/hosts` file, the file would overwhelm every computer. And it would overwhelm every network administrator who would have to make sure that all the `/etc/hosts` files on the Internet match—and get updated every time a new Web site appears. That's why the Domain Name System (DNS) was developed, based on the Berkeley Internet Name Domain (BIND). In `/etc/resolv.conf`, the IP address of each DNS server is listed with a simple line similar to:

```
nameserver 192.168.0.1
```

**/etc/host.conf**

Many networks configure an `/etc/hosts` file for the local network and a DNS server for other networks and/or the Internet. When your computer looks for an IP address, this file determines whether it searches through `/etc/hosts` or DNS first. This is usually a one-line file:

```
order hosts,bind
```

A computer with this line looks through `/etc/hosts` first. If it can't find the computer name that you want in that file, it next looks to the DNS server (bind) for the computer name.

**/etc/nsswitch.conf**

This file relates to the configuration on a network of Linux- and Unix-type computers, which are configured to communicate using the Network File System (NFS). When it is used in concert with the Network Information System (NIS), networks can maintain a single database of usernames and passwords for all NFS-enabled computers on that network.

**CERTIFICATION OBJECTIVE 1.09****Standard Network Services**

Linux is built for networking. The code associated with many standard networking services is integrated into the Linux kernel. A basic understanding of the functionality of standard Linux networking services is essential. Many themes throughout this book

assume that you already understand the purpose of network communication protocols, mail services, host name and IP address management, Web services, and more.

In Red Hat Enterprise Linux, network services are often installed separately. Some include different packages for clients and servers. Some network services are activated through `/etc/xinetd.conf`, which reads activation files in the `/etc/xinetd.d` directory. Others are activated directly with scripts in the `/etc/rc.d/init.d` directory. I briefly examine some key RHEL network services in the following sections.

## Network File System

The first network system on Unix and Linux computers is the Network File System (NFS). Ideally, this leads to a seamless Linux interface; for example, you can set up one `/home` directory for all users on your network on one server. Remember, you need NFS on both server and client computers on your network.

First, make sure NFS support is part of the kernel, as documented in `/proc/filesystems`. If it isn't there, you may need to activate the `nfs` and related modules (`nfsd`, `lockd`, `sunrpc`) in the kernel. Inspect installed modules with the following command:

```
# lsmod | more
```

Make a list of the modules that aren't included in the list. Run a **modprobe** command (for example, **modprobe nfs**) on any missing modules. With a standard Red Hat Enterprise Linux installation, this should add the modules to the `lsmod` list, and then add them to the kernel, as listed in `/proc/filesystems`.

Once you've shared an NFS directory, you can then activate the NFS daemon with the **service nfs start** command. I illustrate an example where you can share the Red Hat Enterprise Linux installation files through NFS in Chapter 2.

Once NFS is configured, you can find shared directories on the server's `/etc/exports` file, and then mount them with a command similar to the following:

```
# mount -t nfs nfsserver:/home /mhome
```

For more information on NFS, see Chapter 9.

## sendmail

There are some who suggest that sendmail is the biggest test for Linux system administrators. The sendmail configuration files, `sendmail.cf` and `submit.cf`, are complex. But it should not be intimidating. With the help of the corresponding `.mc` files, it's easier to define the features you want, the protocols you need, and the way mail is sent and received on your network.

More information on sendmail is available in Chapter 7.

## POP and IMAP

The Post Office Protocol (POP) and the Internet Mail Access Protocol (IMAP) are each a set of rules for delivering e-mail from a server such as sendmail to an e-mail client such as Netscape, elm, or pine. While POP3 is the current standard for e-mail that is sent to clients, IMAP4 is more flexible for users such as those who access their mail using different computers. POP3 and IMAP4 configuration is addressed in Chapter 7.

## File Transfer Protocol (FTP)

Perhaps the most basic file sharing protocol still in common use is the File Transfer Protocol (FTP). It is set up specifically for file transfers; you might already know that file transfers using FTP are generally faster than with any other protocol.

As with NFS and Samba, you need a server and a client. FTP servers can be anonymous, which means they accept connections from anyone, or they can be configured to require a specific username and password. Generally, Linux FTP servers share files from the `/var/ftp` directory. Red Hat Enterprise Linux now comes with the Very Secure FTP daemon (vsFTPD) as the only FTP server.

The original FTP client works from the command line. Most Linux navigational commands work for FTP; just remember that the **get** and **put** commands download and upload specific files. FTP is covered in more detail in Chapter 7.

## Domain Name Service (DNS)

If there were a practical way to list all of the domain names and IP addresses of every Web site on the Internet in a single file, we would not need the Domain Name Service (DNS). The DNS system allows us to set up different parts of this database on different servers around the world. If a DNS server does not have the answer, you can configure it to ask other DNS servers for help. DNS is covered in more detail in Chapter 9.

## Dynamic Host Configuration Protocol (DHCP)

IP version 4 addresses are scarce. The Dynamic Host Configuration Protocol (DHCP) was designed to help ration IP addresses. A DHCP server leases a specific IP address to a computer network card for a limited, but renewable, amount of time. DHCP servers can lease IP addresses on different LANs using the BOOTP protocol. More information on setting up DHCP clients and servers is available in Chapter 9.

## Samba

The network system originally developed for networks with Microsoft and IBM computers is based on the Server Message Block (SMB) format. SMB, also known as Samba is the basis for Microsoft Windows Workgroup and Domain-based network communication. When you install Samba on a Linux computer, you can make it part of one of these Microsoft-style networks. It can share files just like any other member of a workgroup network. It can act as a server. Current versions of Samba can even be configured as a Windows NT-style Primary Domain Controller or a member server on a Windows 2000/XP-based network.

Separate packages are available to set up your Linux computer as a Samba client and as a Samba server. Once shares are configured in `/etc/samba/smb.conf`, other Samba-enabled Linux clients can mount these directories with a command similar to the following:

```
# mount -t smbfs -o username=user //servername/sharename /mountpoint
```

You can also set up the **smbmount** command for this purpose. Samba and the associated configuration tools are discussed extensively in Chapter 8.

## Web Services

Apache is by far the most popular Web server in use on the Internet. It's a standard part of the Red Hat Enterprise Linux server installation. The main configuration file is `/etc/httpd/conf/httpd.conf`. Configuration is based on an extensive array of modules in the `/etc/httpd` directory. Basic HTML files, icons, and CGI applets are installed in the `/var/www` directory. The main Apache log files are part of the `/var/logs/httpd` directory. Daily log files for the largest Web sites can grow into the GB range. Apache is covered in more detail in Chapter 7.

A substantial number of other Web servers are available for Red Hat Enterprise Linux, such as Sun's iPlanet and Zeus's Web server.

## Network Information Service (NIS)

The Network Information Service was formerly known as the “yellow pages,” as it is a centralized database of usernames and passwords on a network with Linux and other Unix-style computers. NIS can be configured as a centralized database for a number of other configuration files in the `/etc` directory. Anything that can standardize the configuration of different computers on a network helps the system administrator. For more information on NIS, see Chapter 10.

**CERTIFICATION OBJECTIVE 1.10**

## Basic Network Security

Network security in Linux has four basic components. Security by computer can help you manage what computers can send messages into and out of your network. Security by port can help you manage the services that others can use to break into your network. Security by address translation can help you hide the computers inside your network. And finally, security by rule can help you manage the type of data allowed into your network in excruciating detail. Red Hat Enterprise Linux includes two different tools to help you configure a firewall on your computer, `lokit` and `redhat-config-securitylevel` (also known as the Red Hat Firewall Configuration tool). Security issues are discussed in more detail in Chapter 10.

### Allowing and Denying

The `/etc/hosts.allow` and `/etc/hosts.deny` files can help you manage what computers are allowed into your network. You can specify computers by name, IP address, network, or domain name in each file. This can help you limit access to a trusted few computers such as those within your company, or it can protect you from computers that you know may pose a problem.

### Port Security

TCP/IP has 65,536 ports, which work sort of like TV channels. If you leave all ports open, you're leaving a lot of options for a cracker who wants to break into your network. With a firewall, you can create a solid barrier and then open only the ports that you need.

### Network Address Translation

Most LAN administrators set up Network Address Translation (NAT) as a matter of course on an IPv4 network. Since IPv4 addresses are scarce, it is typical to use private IP addresses inside a LAN, with a regular IP address only on the gateway computer that is directly connected to an outside network such as the Internet.

For example, when a computer inside a LAN wants access to a Web page, NAT sends the IP address of the gateway to the Internet. Nobody outside the LAN need know the real source of the Web page request.

## iptables

There are two basic services for filtering information in and out of a network, based on the **ipchains** and **iptables** commands. Red Hat has recently implemented **iptables** as the firewall tool of choice in RHEL 3. Once you've configured a firewall and loaded it, the rules are stored in the `/etc/sysconfig/iptables` file.

The **iptables** command has three basic ways to look at a data packet: input, output, or forward. Within these and other parameters, you can set up your firewall with instructions to let the packet pass, let it drop, or direct it someplace else.

**iptables** is covered in more detail in Chapter 10.

### CERTIFICATION OBJECTIVE 1.11

## Basic Hardware Knowledge

While customized Red Hat distributions are available for such diverse platforms as the Alpha, Itanium, and S/390 CPUs, the RHCE and RHCT exams are focused on computers built to the Intel-based 32-bit architecture (or similar 32-bit CPUs such as those built by AMD and National Semiconductor/Cyrix).

The architecture of a personal computer defines the components that it uses as well as the way that they are connected. In other words, the Intel-based architecture describes much more than just the CPU. It includes standards for other hardware such as the hard drive, the network card, the keyboard, the graphics adapter, and more. All software is written for a specific computer architecture, such as the Intel-based 32-bit architecture.

Even when a manufacturer creates a device for the Intel platform, it may not work with Linux. Therefore, it's important to know the basic architecture of an Intel-based computer.

### exam

#### Watch

**While it is important to know how Linux interacts with your hardware, the RHCE and RHCT exams are not hardware exams. As of this writing,**

**while the RH133 and RH300 courses do address hardware issues, there are no hardware components listed in the Red Hat Exam Prep guide.**

## Intel Communications Channels

Three basic channels are used to communicate in an Intel architecture PC: interrupt request (IRQ) ports, input/output (I/O) addresses, and direct memory address (DMA) channels. An IRQ allows a component such as a keyboard or printer to request service from the CPU. An I/O address is a memory storage location for communication between the CPU and different parts of a computer. A DMA channel is used when a device such as a sound card has an independent processor and can bypass the CPU.

With the plug and play features built into RHEL 3, these channels are generally not a problem, but are included because they are on the prerequisite list for the RH300 course, as described in [www.redhat.com/training/rhce/courses/rh300\\_prereq.html](http://www.redhat.com/training/rhce/courses/rh300_prereq.html).

### IRQ Settings

An IRQ is a signal that is sent by a peripheral device (such as a network card, graphics adapter, mouse, modem, or serial port) to the CPU to request processing time. Each device you attach to a computer may need its own IRQ port. Normally, each device needs a dedicated IRQ (except for USB and some PCI devices). The Intel architecture is currently limited to 16 IRQs (0–15), which is often not enough for modern PCs with network cards, modems, hard drives, sound cards, printers, and more.

If you run out of IRQs, some PCI devices can share IRQs. USB devices can share IRQs. This support is available in most PCs manufactured after the year 2000.



***If you're having a problem with your USB ports or PCI cards, check your BIOS first. Many BIOS menus include options that enable PCI sharing and support USB connections.***

### Planning the IRQ Layout: Standard IRQs

IRQs are a precious commodity on a PC. IRQ conflicts are common when you're connecting a lot of devices. If your printer doesn't work after you've connected a second network card, it can help to know the standard IRQ for printers. You can then assign a different IRQ to that network card. If you don't have any free IRQs to assign to that network card, you may be able to sacrifice a component that uses a standard IRQ. For example, if you always connect to a server remotely, that server PC may not need a keyboard. If you can boot a computer with a CD-ROM, you may not need a floppy drive.

Some IRQs are essential to the operation of a PC and just can't be changed. These are reserved by the motherboard to control devices such as the hard disk controller and the real-time clock. Do not use these interrupts for other devices or there will be conflicts! Other IRQs are normally assigned to common devices such as a floppy disk

and a printer. In Linux, you can check `/proc/interrupts` to see which interrupts are being used and which are free for new devices.

### Input/Output Addresses

Every computer device requires an *input/output (I/O) address*. It's a place where data can wait in line for service from your CPU. I/O addresses are listed in hexadecimal notation, where the numbers are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, and f. Some typical I/O addresses include those for the basic serial ports, known in the Microsoft world as COM1, COM2, COM3, and COM4. These ports normally use the following I/O addresses: 03f8, 02f8, 03e8, and 02e8.

You can find a list of assigned I/O addresses in your `/proc/iports` file.

### Direct Memory Addresses

A *direct memory address (DMA)* is normally used to transfer information directly between devices, bypassing the CPU. Many components don't need a CPU. For example, many sound cards include their own processor. This allows your PC to set up a DMA channel between a hard drive and a sound card to process and play any music files that you may have stored.

While DMA channels bypass the CPU, devices that use DMA are still configured with IRQ ports. There are eight standard DMA channels (0–7); DMA 4 is reserved and cannot be used by any device.

You can find a list of assigned DMA addresses in your `/proc/dma` file.

## RAM Requirements

Red Hat nominally requires that you install Red Hat Enterprise Linux on a computer with at least 256MB of RAM. While I've installed RHEL 3 on computers with less RAM, 256MB is the minimum that is required if you've purchased the appropriate version of RHEL 3 and want contracted tech support from Red Hat.

The maximum amount of memory your system will use is the sum of all of the memory requirements of every program that you will ever run at once. That's hard to compute. Therefore, you should buy as much memory as you can afford. Extra RAM is usually cost effective when compared to the time you would spend trying to tune an underpowered system. Limitations are few; on Red Hat Enterprise Linux 3 Advanced Server, you can use up to 64GB of RAM.

If you're installing RHEL 3 on a computer with between 16GB and 64GB of RAM, you'll need to use the "hugemem" kernel, which I describe in a bit more detail in Chapter 5.



***If you're setting up Linux as a server, RAM requirements increase with the number of users who may need to log in simultaneously. The same may be true if you're running a large number of programs or have memory-intensive data such as that required by a database.***

## Hard Drive Options

Before your computer can load Linux, the BIOS has to recognize the active primary partition on the hard drive. This partition should include the Linux boot files. The BIOS can then set up and initialize that hard drive, and then load Linux boot files from that active primary partition. You should know the following about hard drives and Linux:

- The standard Intel architecture PC is configured to manage up to four IDE (Integrated Drive Electronics) hard drives.
- Depending on the SCSI (Small Computer Systems Interface) hardware that you have, you can attach up to 31 different SCSI hard drives.
- While you can use as many IDE or SCSI drives as your hardware can handle, you need to install the Linux boot files from the /boot directory on one of the first two hard drives. If Linux is installed on a later drive, you'll need a boot floppy.
- Although you can install Linux on USB (Universal Serial Bus) or IEEE 1394 (Institute of Electrical and Electronics Engineers standard 1394, also known as FireWire or iLink) hard drives, as of this writing, you can't load Linux boot files directly from these drives. However, it is possible to set up a boot floppy to start Linux from these drives.

## CERTIFICATION OBJECTIVE 1.12

## Hardware Compatibility

Now it's time to explore in detail the hardware that Red Hat Enterprise Linux can handle. While some manufacturers now include their own Linux hardware drivers, most Linux hardware support comes from third parties. Fortunately, there is a vast community of Linux users, many of whom produce drivers for Linux and distribute them freely on the Internet. If a certain piece of hardware is popular, you can be certain that Linux support

for that piece of hardware will pop up somewhere on the Internet and will be incorporated into various Linux distributions, including Red Hat Enterprise Linux.

Be careful when purchasing a new computer to use with Linux. Though Linux has come a long way the last few years, and you should have little problem installing it on most modern PCs, you shouldn't assume Linux will run on *any* PC, especially if the PC in question is a state-of-the-art laptop computer. Laptops are often designed with proprietary configurations that work with Linux only after some reverse engineering.

Fortunately, there are a number of major manufacturers who support RHEL 3 out of the box for their workstations and servers. Check the Red Hat Hardware Compatibility List, as discussed in the section of the same name, for the latest information.

Other kinds of hardware, such as “winmodems” and “winprinters,” are designed to use Microsoft Windows driver libraries. Integrated hardware (such as video chips that share system RAM) and parallel port devices can also be problematic. While Linux drivers exist for many of these devices, do your research.

Linux runs very well on lower-end computers. This is one of Linux's strong points over other operating systems, such as Microsoft's Windows XP. Linux runs fine on 64MB of RAM, although more is always better, especially if you want to run any graphical applications.



***While it is important to know how to select and configure hardware components to get to a smoothly running Linux computer, the RHCE exam is not a hardware exam.***

## Linux Hardware Documentation

There are many resources available to help you select the best hardware for Linux. Thousands of Linux gurus are available online in areas such as mailing lists and newsgroups. Perhaps the best places to look are the Linux Documentation Project (LDP) or the Red Hat Hardware Compatibility List. The LDP is a global effort to produce reliable documentation for all aspects of the Linux operating system, including hardware compatibility.

### Linux Hardware HOWTO

The Linux Hardware HOWTO is a document listing most of the hardware components supported by Linux. It's updated irregularly with added hardware support, so it is a relatively up-to-date source of information. As of this writing, various LDP HOWTOs are supplied on the documentation CD-ROM in text format and in various languages, in the /HOWTOS directory. The official up-to-date list can be found at the LDP [www.tldp.org](http://www.tldp.org).

The Red Hat Hardware Compatibility List

The Red Hat Hardware Compatibility List is different from the one you'll find in the Linux Hardware HOWTO. It specifies name brand hardware that has been tested with RHEL and Red Hat Linux (versions 9 and below). If you've purchased RHEL 3, Red Hat will provide some level of installation support for any certified or compatible hardware. Some hardware that has been tested by Red Hat has specifically been found not to work with Red Hat Linux or RHEL and is therefore not supported. Red Hat hasn't tested all PC hardware; as a courtesy, they also include a list of hardware that others have tested with Red Hat Enterprise Linux, as "Community Knowledge" hardware. These four categories of hardware are described in Table 1-8.

Like the Linux Hardware HOWTO, the Red Hat Hardware Compatibility List draws upon the efforts of volunteers. If you want to check if any of the "latest" hardware (such as USB) will run on your Linux system, it's probably best to consult the Red Hat support site first, then maybe LDP's Linux Hardware HOWTO. However, if you want the option of being able to contact Red Hat for support, you should stay within the "supported" list of the Red Hat Hardware Compatibility List.

Check the documentation for your hardware. Find a component such as a modem or a network card. Cross-check this component against the Red Hat and LDP hardware compatibility lists (HCLs). Find the Red Hat lists by starting at [www.redhat.com](http://www.redhat.com). Find their HCL in their support area. Find the LDP Hardware HOWTO by starting at [www.tldp.org](http://www.tldp.org). Find this list in the LDP section on HOWTOs. Compare the results. While in most cases the results are identical, it's good to know how to search through both sources just in case.

As part of this process, find a component listed on one or both of these HCLs as incompatible with Linux. Do a search on your favorite search engine or the newsgroups based on the name and model of the product. Don't forget to include "linux" in your list of search terms. You might be pleasantly surprised. As of this writing, a searchable newsgroup database is available at [groups.google.com](http://groups.google.com).

| TABLE 1-8                                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |        |             |           |                                                                                |            |                                                 |               |                                                                        |                     |                                                                                                     |
|-------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|-------------|-----------|--------------------------------------------------------------------------------|------------|-------------------------------------------------|---------------|------------------------------------------------------------------------|---------------------|-----------------------------------------------------------------------------------------------------|
| Red Hat Hardware Compatibility Categories | <table><tr><th>Status</th><th>Description</th></tr><tr><td>Certified</td><td>Approved by Red Hat, Inc., through the Red Hat Hardware Certification Program.</td></tr><tr><td>Compatible</td><td>Reviewed by Red Hat, and known to be supported.</td></tr><tr><td>Not Supported</td><td>Reviewed by Red Hat, and known not to work with Red Hat Linux or RHEL.</td></tr><tr><td>Community Knowledge</td><td>Untested by Red Hat; others have reported some degree of compatibility with Red Hat Linux and RHEL.</td></tr></table> | Status | Description | Certified | Approved by Red Hat, Inc., through the Red Hat Hardware Certification Program. | Compatible | Reviewed by Red Hat, and known to be supported. | Not Supported | Reviewed by Red Hat, and known not to work with Red Hat Linux or RHEL. | Community Knowledge | Untested by Red Hat; others have reported some degree of compatibility with Red Hat Linux and RHEL. |
| Status                                    | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |        |             |           |                                                                                |            |                                                 |               |                                                                        |                     |                                                                                                     |
| Certified                                 | Approved by Red Hat, Inc., through the Red Hat Hardware Certification Program.                                                                                                                                                                                                                                                                                                                                                                                                                                                   |        |             |           |                                                                                |            |                                                 |               |                                                                        |                     |                                                                                                     |
| Compatible                                | Reviewed by Red Hat, and known to be supported.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |        |             |           |                                                                                |            |                                                 |               |                                                                        |                     |                                                                                                     |
| Not Supported                             | Reviewed by Red Hat, and known not to work with Red Hat Linux or RHEL.                                                                                                                                                                                                                                                                                                                                                                                                                                                           |        |             |           |                                                                                |            |                                                 |               |                                                                        |                     |                                                                                                     |
| Community Knowledge                       | Untested by Red Hat; others have reported some degree of compatibility with Red Hat Linux and RHEL.                                                                                                                                                                                                                                                                                                                                                                                                                              |        |             |           |                                                                                |            |                                                 |               |                                                                        |                     |                                                                                                     |

## CPU and SMP Support

Red Hat Enterprise Linux for Intel supports computers with Intel and compatible processors. It is “Itanium-ready,” which means that it will be able to support this 64-bit Intel CPU when it is finally released.

Linux is commonly used as a server operating system. Many server applications can take advantage of the flexibility provided by multiple CPUs. This is known as symmetric multiprocessing (SMP) support. Linux began supporting multiple CPUs with the release of the 2.4 kernel back in 2001.

on the  
UoB

***Some of the developers hope to increase the SMP limit to 128 CPUs. If you’re running Linux on a SMP computer, keep up to date with the latest kernel developments at [www.kernel.org](http://www.kernel.org).***

### exam

Watch

***Red Hat Enterprise Linux was released with Linux Kernel version 2.4.22. While it includes a number of features, or “backports,” from Linux Kernel 2.6, I don’t***

***believe that you’ll need to install the 2.6 kernel for the RHCE or RHCT exams. As always, monitor the Red Hat syllabus and requirements for the latest information.***

## Plug and Play

*Plug and play (PnP)* refers to the capability of an operating system to automatically allocate hardware ports or addresses to specific devices such as hard drives, sound cards, or modems. Linux’s ability to work with plug and play devices is somewhat less than perfect. For example, if you have the right network modules installed with the kernel, Linux may be able to automatically detect and install the drivers for a new network card in a PCMCIA slot. However, Linux does not automatically detect all printers; in this case, you may need to use the techniques discussed in Chapter 8 to install the appropriate print driver.

A plug and play system has three parts: the BIOS, the device, and the operating system. Unless all three work perfectly, problems can arise with plug and play. The BIOS has to allow the operating system to find the devices on your computer. Plug and play devices have to accept port and channel assignments from the operating system. And a plug and play operating system is constantly searching each connection for new hardware. Red Hat developed the kudzu utility to look for and configure any hardware changes when you boot Linux.

## Plug and Play Support in Linux

The unfortunate truth is that Linux doesn't handle plug and play as well as we may want. The main problem lies with plug and play support for devices that run on an ISA bus. ISA is a legacy technology from older IBM PCs, created without plug and play in mind, so support for it is complex.

PCI hardware is a different story. As Linux loads, device drivers can easily find PCI devices. However, conflicts may still arise with ISA devices. While Linux works well with many USB and IEEE 1394 devices, support for the latest USB 2.0 and IEEE 1394 hardware is still officially "experimental" as of this writing.

## Plug and Play Conflicts

The Linux plug and play subsystem may have problems with the newest computer devices or some very old ones. If you're having problems with the newest computer equipment, various Web sites are dedicated to offering help. For example, [www.linmodems.org](http://www.linmodems.org) can help you configure many so-called "winmodems," and [www.linux-usb.org](http://www.linux-usb.org) can help you configure the latest USB equipment on Linux.

Many hardware conflicts with relatively old equipment are fairly simple to eliminate. There are three possible areas of conflict:

- A physical hardware jumper is conflicting with another card.
- Your ISA plug and play cards are not properly configured.
- You are out of IRQs or other resources to add to your new device.

You can use the `/proc` files to check the currently used IRQ ports, I/O addresses, and DMA channels. For example, to check the occupied IRQs, the following command lists the devices that *are* loaded by the kernel:

```
# cat /proc/interrupts
```

If there is a conflict, the device is not loaded. You can quickly scan over the left side to see what interrupts are available. To get a list of used I/O addresses and DMA channels, issue the following commands:

```
# cat /proc/ioports
# cat /proc/dma
```

The kernel included with Red Hat Enterprise Linux 3 and above should keep plug and play configuration problems to a minimum. When problems arise, two or more devices are probably trying to use the same IRQ, I/O, and/or DMA. In that case, one or both devices may not be loaded. It may take a little detective work to identify the troubled hardware; conflicts may prevent it from being listed in one of the associated `/proc`

directory files. Then select one of the devices, and change its IRQ, I/O, and/or DMA to a free location.

This is usually a two-step process: first, change the settings on the card itself through physical jumpers or a diagnostic disk, as described in the next section. If Linux doesn't detect your changes, use the appropriate configuration utility, such as **redhat-config-mouse**, **ifconfig**, **modprobe**, or **redhat-config-network**, to change the settings on your device.

Generally, Linux should not have problems with PCI plug and play cards or USB devices. Linux should recognize them and set them up with appropriate IRQ ports, I/O addresses, and DMA channels. If you cannot see what your PCI cards are set to, you can type **cat /proc/pci**. If a PCI card that you're concerned about does not show up here, you may be out of IRQs. If you run out of IRQs, you may want to look into alternatives such as IEEE 1394 (also known as FireWire or iLink) or USB devices.

## ACPI and APM

Closely related to plug and play are the latest computer power management standards, known as Advanced Configuration and Power Interface (ACPI) and Advanced Power Management (APM). Both are efforts to manage PC power consumption. As such, they are important tools to extend the lifetime of battery-operated devices such as laptop computers.

Microsoft has driven developments in both areas toward computers that can be easily suspended and reactivated from a minimum power state. Red Hat has incorporated some ACPI features from the 2.6 kernel in RHEL.

If you have problems with an ACPI computer and, you can deactivate ACPI support with the **acpi=off** command to the kernel during the Red Hat Enterprise Linux boot or installation process.

## CERTIFICATION OBJECTIVE 1.13

### Configuring External Devices

You can install many devices externally to your computer. These devices are sometimes known as peripherals. Generally, peripheral devices fall into five categories: serial, parallel, USB, IEEE 1394, and PC Cards. A device attached to a serial port, such as a mouse or a modem, uses the device associated with that port. Devices attached to parallel, USB, or IEEE 1394 ports normally use their own device files. PC Cards are a special case normally associated with laptop computers.

While Linux normally recognizes basic devices attached to serial or USB ports, such as a mouse during installation, configuring other devices may take additional work.

## Serial Ports

In many cases, configuring a device for a serial port is as simple as linking to the driver of the associated port. For example, if you have an external modem connected to the only serial port on your computer, the Linux plug and play subsystem may have already linked the device for that port with the device for your modem. Run the **ls -l /dev/modem** command. If it shows something like the following output, you know that Linux has already linked your modem driver with the second serial port:

```
lrwxrwxrwx 1 root root 10 Apr 4 11:28 /dev/modem -> /dev/ttyS1
```

Otherwise, you can use the **ln** command to create a link to the appropriate port. If you have a serial mouse, you should find the same type of link from /dev/mouse.

## Parallel Ports

Configuring devices attached to a parallel port can be more complex. For example, Linux doesn't always recognize plug and play printers that are attached to a parallel port such as /dev/lp0. Further configuration with tools such as the CUPS Web-based tool or the Red Hat Printer Configuration tool may be required. You can find the device associated with your CUPS printer in the /etc/cups/printers.conf file.

If you're connecting an external hard drive to a parallel port, you'll want to install the paride module and the module associated with your device, whether it is a hard drive, a tape drive, or a CD-ROM. Similar steps are required for other parallel port devices. Detailed information on configuring parallel port devices is available from the Linux Parallel Port Web site at [www.torque.net/linux-pp.html](http://www.torque.net/linux-pp.html).

## USB

Linux support for USB is growing with the evolution of the latest kernels. While the latest versions of Red Hat Enterprise Linux supports USB hot-swapping, support for the higher-speed USB 2.0 standard is still "experimental," as are Linux drivers for many USB devices. For the latest information, see the Linux USB Web site at [www.linux-usb.org](http://www.linux-usb.org). You may be able to download your driver and install it using the techniques discussed in Chapter 4.

## IEEE 1394

The Institute of Electrical and Electronics Engineers (IEEE) has developed the IEEE 1394 specifications for very high speed data transfer applications, such as digital movies. Equipment designed to these standards is often known by its trade names: FireWire and iLink. The current status is similar to USB; in other words, some IEEE 1394 equipment works with Linux, and development continues. For the latest information, see the Linux IEEE 1394 Web site at [linux1394.sourceforge.net/hcl.php](http://linux1394.sourceforge.net/hcl.php).

## PC Card (PCMCIA)

Linux has one package called Card Services that deals exclusively with PC cards. This package includes all the kernel modules you'll need to manage PCMCIA cards and a set of drivers for specific cards. The package also includes a daemon that handles hot-swapping for most PC cards.

While development of the Card Services package is ongoing, there is often a period where there is no support for the proprietary configurations especially common on laptops. For this reason, the latest laptop is often not the best choice for a Linux installation. However, support for Linux on most name brand laptops is now common even when the laptop is first released. In fact, several companies sell laptops with Linux installed.

### Supported PCMCIA Controllers

According to the Linux PCMCIA Information page at [pcmcia-cs.sourceforge.net](http://pcmcia-cs.sourceforge.net), Linux now supports all common PCMCIA controllers. If you have a problem with a specific PCMCIA card, focus on finding a driver for the card itself. A current list of supported PCMCIA controllers can be found on the Hardware HOWTO.

### Supported Cards

The Card Services package comes bundled with a file named SUPPORTED.CARDS. You can probably find it on your Linux computer with the **locate SUPPORTED.CARDS** command. Also, you can check the PCMCIA HOWTO or the Red Hat Hardware Compatibility List for supported cards. Alternatively, the Linux PCMCIA Information Page may also be helpful.



*During your career as a computer professional, there will be times you'll be asked to research a specific product or technology. To get an idea of how hard or easy this can be, call a major computer retailer or manufacturer and inquire about their latest laptop. Ask them if it supports Linux. What kind of answer do you get? Ask them if they have any earlier models that will. Do you believe the answers you receive are reliable? Check out the company's Web page, if you can, and find out if they provide any information about the product on the Internet. Doing this kind of research can be very trying, with or without success. Before deciding what kind of hardware you want to install Linux on, you should have a good understanding of what will and will not work. Start early and build a good base of reliable references you can use to find out new computer information. Web sites, such as the Linux Documentation Project, as well as magazines like Linux Journal, Linux Format (UK), Sys Admin Magazine, and Linux Magazine, will help you stay informed.*

## CERTIFICATION OBJECTIVE 1.14

### Preparing to Install Linux

Installing Linux on most Intel-based computers is pretty straightforward. In many cases, most installation proceeds without problems. Generally if you are installing Linux on one modern computer, it should be okay to just install Linux without worrying too much about your hardware.

However, if you have problems, you'll save yourself a lot of time and frustration by knowing exactly what hardware you have. Before you start installing Red Hat Enterprise Linux, it's helpful to be familiar with the following components of your system:

- **Drives** Check to see if you are using SCSI or IDE drives. You should know the manufacturer, model number, and capacity of the drive. In addition, if it's a SCSI drive, make sure you know its SCSI ID number. (As of this writing, if you're installing Red Hat Enterprise Linux on a VMWare machine, emulated SCSI drives don't work; you'll have to configure an emulated IDE hard drive.)
- **Hard drive controller** Know the manufacturer and model number of the drive controller. If this data is hard to find, at least try to find the chipset of the controller. If it's an IDE controller, the documentation is associated with

the computer motherboard. If it's a SCSI controller, see the documentation associated with that controller.

- **CD-ROM** For most standard SCSI or IDE CD-ROMs, the standard drivers should work without problems. However, if you are using a CD-ROM with a proprietary interface, you should know the manufacturer, as well as the model of the drive and controller card.
- **Mouse** You should know the type of mouse that you have—such as PS/2, serial, or USB. If your mouse uses a serial port, it helps if you know which port. For example, if you're converting a computer that's running Microsoft Windows, a serial mouse is associated with a serial port, typically COM1, COM2, COM3, or COM4. The corresponding Linux device files are `/dev/ttyS0`, `/dev/ttyS1`, `/dev/ttyS2`, and `/dev/ttyS3`. And the number of buttons on a mouse may not be obvious; if you have a two-button mouse with a scrolling wheel that you can click, you actually have a three-button mouse.
- **Graphics card** If you will be running the Linux graphical user interface (GUI), also known as X or X11, you will need the manufacturer, the model number, the chipset, and the amount of video memory. If it's a fairly common graphics card and you can't find the chipset or memory, you should be able to select a generic or older version of the card from the X installation database.
- **Sound, video, and game adapters** If you want to set up sound on your system, you should know the manufacturer and model number of the sound card. If plug and play doesn't work for your sound card, you'll also need the default IRQ port, I/O address, and DMA channel(s). Especially on laptops, this information may be stored in your BIOS.
- **Network adapters** If you are going to network your Linux system, you should know the manufacturer and model number of the network adapter. If plug and play doesn't work for your network adapter, you should find its default IRQ port and I/O address.
- **Monitor** If you will be running the Linux GUI, based on the Linux implementation of the X Window System ([www.xfree86.org](http://www.xfree86.org)), you will need the manufacturer, model number, available resolutions, and refresh frequencies of the monitor.



***Be especially careful with older monitors or laptop displays. Exceeding the frequency refresh capabilities of such monitors could easily overload the display system. Replacing a laptop display is not a pleasant exercise!***

Not all hardware will work with Linux. After you've collected information about your system, you should consult the Red Hat Hardware Compatibility List (HCL) or LDP Hardware HOWTO to determine if your components are compatible with the current version of Red Hat Enterprise Linux.

## **CERTIFICATION SUMMARY**

The RHCE and RHCT exams are not for beginners. This chapter covers the prerequisites for the RHCE exam and thus the elementary skills that you need for the remainder of this book. If the explanations in this chapter are too brief, you may need to refer to sources such as those I cite at the beginning of this chapter. While these exams are based on RHEL 3, you can use Red Hat Linux 9 to study for these exams, with the tips I provide throughout this book.

This chapter provides an overview of many Linux fundamentals. While the RHCE and RHCT hands-on exams may not explicitly test the skills you learn in this chapter, you need to know many of these fundamentals to solve the problems presented on those exams. However, you may see some questions related to this chapter on the multiple-choice portion of the RHCE exam.

Before you start planning your Linux installation, you need a basic degree of knowledge of PC hardware, specifically the Intel-based architecture. A basic understanding of IRQ ports, I/O addresses, DMA channels, and hard drive systems can help you plan how Linux manages and connects every component in your PC.

But not all hardware is supported by Linux. You should now have enough information to find the hardware that fits your needs. Alternatively, you now know about the resources that help you determine what other hardware you need that also works with Linux. Planning your Linux installation makes it easier to handle a wide variety of hardware.



## TWO-MINUTE DRILL

Here are some of the key points from the certification objectives in Chapter 1.

### Basic Linux Knowledge

- ☐ Linux is managed through a series of text configuration files.
- ☐ Even though Red Hat Enterprise Linux now has a rescue CD with text editors such as emacs, you need to know how to restore a system from a rescue floppy, which normally includes just the vi editor. Therefore, you need to know how to use vi.

### Linux Filesystem Hierarchy and Structure

- ☐ Linux directories are organized to the Filesystem Hierarchy Standard (FHS).
- ☐ In the FHS, devices such as mice and hard drives are grouped in the /dev directory. Some /dev files have logical names such as mouse and modem and are linked to the actual device files.
- ☐ FHS partitions can be managed and formatted with the **fdisk**, **fsck**, and **mkfs** commands.
- ☐ The Logical Volume Manager allows you to consolidate multiple partitions in one filesystem, on one directory.
- ☐ Once configured, Linux directories can be mounted on a partition through /etc/fstab or directly with the **mount** command.

### Basic Commands

- ☐ Linux administrators need to know how to use the command line interface.
- ☐ Basic commands allow you to navigate, find the files that you need, read file contents, create new files, and more.
- ☐ File filters allow you to search through the files themselves for specific citations or other file characteristics.
- ☐ Administrative commands allow you to manage Linux in a number of ways, including running processes and logged in users.

## Printing

- ☐ The default Red Hat Enterprise Linux print system is CUPS.
- ☐ You can configure printers by directly editing the files in the `/etc/cups` directory, or by running the Red Hat Printer Configuration tool, **redhat-config-printer**.

## Shells

- ☐ Command lines are based on a shell.
- ☐ With the right permissions, you can set up shell programs in executable scripts.
- ☐ The way a shell works depends on the settings in its variables and parameters. Some variables and parameters are grouped in the inherited environment, which maintains settings from shell to shell.
- ☐ With `stdin`, `stdout`, and `stderr`, you can manage different data streams.

## Basic Security

- ☐ Basic security within Linux is based on file permissions, users, groups, and **umask**.
- ☐ The SUID and SGID bits allow you to share owner-level permissions with different users and groups.
- ☐ Shadow passwords hide user authentication data. The Shadow Password Suite protects user and group passwords in files that should be accessible only to the root user.

## System Administration

- ☐ While it's normally best to log in as a regular user, it's faster to log in as the root user for the RHCE and RHCT exams.
- ☐ Standard files for new users are kept in `/etc/skel`.
- ☐ Daemons are processes that run in the background.
- ☐ Network service can be controlled through scripts in the `/etc/rc.d/init.d` directory.
- ☐ The cron daemon helps you schedule different jobs, including backup and restore jobs, which should be done when network use is at a minimum.
- ☐ When you have problems, system log files, as organized by `/etc/syslog.conf`, provide important clues to the causes.

### Basic TCP/IP Networking

- ☐ Most of the work in TCP/IP networking is with configuring IP addresses.
- ☐ There are three different sets of private IPv4 addresses suitable for setting up TCP/IP on a LAN.
- ☐ Tools such as **ping**, **ifconfig**, and **netstat** can help you diagnose problems on that LAN.
- ☐ Name resolution configuration files determine how your computer finds the right IP address.

### Standard Network Services

- ☐ There are a number of standard network services. They include NFS, sendmail, POP, IMAP, FTP, DNS, DHCP, Samba, Apache, and NIS.
- ☐ Each of these services, when installed, can be configured to start and stop through the scripts located in the `/etc/rc.d/init.d` or `/etc/xinetd.d` directories.

### Basic Network Security

- ☐ Basic network security settings can depend on allowing or denying access to different computers by their IP addresses or by the desired TCP/IP port.
- ☐ Computers behind a firewall can be protected through Network Address Translation or various **iptables** commands.

### Basic Hardware Knowledge

- ☐ The focus of the RHCE exam is on computers built with an Intel-based architecture.
- ☐ An Intel-architecture PC has three basic communications channels: IRQ ports, I/O addresses, and DMA channels.
- ☐ The latest version of Red Hat Enterprise Linux as certified requires at least 256MB of RAM.
- ☐ You can set up Linux on IDE, SCSI, USB, or IEEE 1394 hard drives. However, the BIOS of a PC can load Linux boot files only from the first two IDE or SCSI hard drives or a boot floppy.

## Hardware Compatibility

- ☐ Linux has come a long way the last few years, and you should have little problem installing it on most modern PCs.
- ☐ You may not be able to install Linux on every PC; there are occasional problems on newer laptop computers.
- ☐ The best places to look for compatible hardware are the Hardware HOWTO of the Linux Documentation Project or the Red Hat Hardware Compatibility List.
- ☐ Red Hat Enterprise Linux has a very capable plug and play service that can configure most current hardware.
- ☐ Closely related to plug and play are the ACPI and APM power management standards.

## Configuring External Devices

- ☐ There are five basic categories of external devices: serial, parallel, USB, IEEE 1394, and PCMCIA.
- ☐ Serial port devices are usually linked to specific device files. For example, `/dev/modem` is often linked directly to a specific serial device file.
- ☐ Parallel port device configuration can be more complex. For example, a separate configuration utility is required to recognize devices such as printers.
- ☐ While Linux supports USB and IEEE 1394, support for many specific USB and IEEE 1394 devices is still in the works.
- ☐ Linux supports PCMCIA cards, also known as PC Cards, through the Card Services package, which includes drivers for the PCMCIA adapter and individual cards.

## Preparing to Install Linux

- ☐ Installing on most Intel-based computers is pretty straightforward, but you can save yourself much time and frustration by knowing exactly what hardware you have.
- ☐ It can help to know the make and model number for each of the following components: hard drive controllers, network adapters, graphics cards, and sound adapters.
- ☐ If possible, also find the resolution and horizontal and vertical refresh rates of your monitor.

## 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. Don't focus exclusively on these questions. There is no longer any multiple choice questions on the Red Hat exams. These questions exclusively test your understanding of the chapter. While the topics in this chapter are "prerequisites," it is okay if you have another way of performing a task. Getting results, not memorizing trivia, is what counts on the Red Hat exams.

Choose all correct answers for each question.

### Basic Linux Knowledge

1. If you're editing the `/etc/inittab` file, which of the following `vi` commands would you use to copy the currently highlighted line?
  - A. `p`
  - B. `c`
  - C. `yy`
  - D. `cw`

### Linux Filesystem Hierarchy and Structure

2. Which of the following commands would you use to mount an MS-DOS floppy disk?
  - A. `mount -t /dev/fd0 /mnt/floppy`
  - B. `mount -t ext2 /dev/fd0 /mnt/floppy`
  - C. `mount -t ext3 /dev/fd0 /mnt/floppy`
  - D. `mount -t vfat /dev/fd0 /mnt/floppy`

### Basic Commands

3. Which of the following commands returns the actual number of times user `mj` is logged into your Linux computer?
  - A. `wc -l`
  - B. `who | grep mj`
  - C. `who | wc -l`
  - D. `who | grep mj | wc -l`

## Printing

4. You're maintaining a large queue of print jobs on your network, and you need some job numbers to make sure the engineers get highest priority on the printer. Which of the following commands lists print job numbers?
- A. `lpr -l`
  - B. `lpq -l`
  - C. `lprm -l`
  - D. `lpd`

## Shells

5. Which of the following commands would you use to add the `/usr/sbin` directory to your `PATH`?
- A. `$PATH=$PATH:/usr/sbin`
  - B. `$PATH=PATH:/usr/sbin`
  - C. `PATH=$PATH:/sbin`
  - D. `PATH=$PATH:/usr/sbin`

## Basic Security

6. When you run the `umask` command, you see the following result: 0000. The next time you create a file, what will be the permissions?
- A. `drwxrwxrwx`
  - B. `-----`
  - C. `-rwxr-xr-x`
  - D. `-rw-rw-rw-`

## System Administration

7. Based on the following line from a user's crontab file, when will the Berkeleylives program be run?
- ```
0 1 2 3 * Berkeleylives
```
- A. At 1:23 A.M. every day
  - B. At 1:00 A.M. on March 2
  - C. At 1:00 A.M. on February 3
  - D. At 2:00 A.M. on March 2

### Basic TCP/IP Networking

8. Which of the following sets of numbers, in order, correspond to an appropriate network address, subnet mask, and broadcast address?
- A. 192.168.14.255, 255.255.255.0 192.168.14.0
  - B. 192.168.14.0, 255.255.255.0 192.168.14.255
  - C. 255.255.255.0 192.168.14.255, 192.168.14.0
  - D. 192.168.14.0, 192.168.14.255, 255.255.255.0

### Standard Network Services

9. Which of the following services works to connect Linux to a Microsoft Windows-based network?
- A. NFS
  - B. SMB
  - C. DNS
  - D. Windows for Workgroups

### Basic Network Security

10. Which of the following commands are associated with a Red Hat Enterprise Linux firewall configuration utility?
- A. lokwall
  - B. lokkit
  - C. redhat-config-securitylevel
  - D. firewall-lokkit

### Basic Hardware Knowledge

11. You've checked your `/proc/interrupts` file and find that you don't have any leftover IRQ ports. Nevertheless, you want to install another printer and network card. Which of the following actions would let you keep your current devices?
- A. You need to make some hard decisions on what devices you need to remove from your computer before installing anything new.
  - B. Use the free PCI slots or USB connectors in your PC for new devices. The PCI system allows all PCI and USB devices to share a single IRQ.
  - C. Look through `/proc/ioports` and `/proc/dma`. Find free I/O addresses and DMA channels for your new devices. Then IRQ conflicts are not a problem.
  - D. Just install the new devices. The Linux plug and play system can make sure that extra devices share the appropriate IRQ ports.

## Hardware Compatibility

12. When you look through the Red Hat Hardware Compatibility List, you find that a number of devices in your computer are listed as “community knowledge.” What should you do about these devices before installing Linux?
- A. Replace those devices with hardware that you know is compatible.
  - B. Examine the LDP Hardware HOWTO.
  - C. Check the Web sites of the manufacturers of each community knowledge device.
  - D. Look at the documentation for each device, and remove any winmodems from your PC.

## Configuring External Devices

13. How would you know if your serial mouse is properly attached to a serial port?
- A. Run the `ls -l /dev/mouse` command. You should see a file link to the appropriate serial port.
  - B. Check the physical connection. If the connection is not solid, Linux may not be receiving signals from your mouse.
  - C. Run the `ls -l /dev/ttys0` command. You should see a file link to your serial mouse.
  - D. Run the `redhat-config-mouse` utility and make sure you have a serial mouse.

## Preparing to Install Linux

14. What kind of information should you collect about your PC's video system if you want to install Linux with a graphical user interface?
- A. Model and manufacturer of the graphics card
  - B. Horizontal and vertical refresh rates of the monitor
  - C. Video memory
  - D. Maximum monitor resolution

# LAB QUESTIONS

## Lab 1

You have 18 computers on a LAN behind a firewall. Diagram your computers on a sheet of paper. Connect them together in a “star” configuration. Assign a private IP address to each computer. Take one computer and draw a second connection to the Internet.

While this is a fairly simple exercise, Linux is built for networking. To understand what you can do with Red Hat Enterprise Linux, you need to think in terms of the role of your computer on a network.

## Lab 2

In the next two labs, we'll be experimenting with the `/etc/inittab` file. So before you begin, back it up to a file such as `/etc/inittab.bak`, or back up a copy to your home directory.

1. Use the vi editor to open the `/etc/inittab` file in your computer.
2. Take a look at your `id` variable. If it's set to 3, change it to 5, and visa versa.
3. Reboot your computer and see what happens.
4. Restore your original `/etc/inittab` file.

## Lab 3

In this lab, we'll experiment a bit more with the `/etc/inittab` configuration file.

1. If you haven't already done so, create a backup for `/etc/inittab`.
2. Press CTRL-ALT-F2. You should see a virtual console text login screen.
3. Return to the original text console with CTRL-ALT-F1 or the GUI console with CTRL-ALT-F7.
4. In the `/etc/inittab` file, identify the lines related to the virtual login consoles.
5. Try experimenting with these lines with the `mingetty` commands. Add a comment character (`#`) in front of the second line with the `mingetty` command.
6. Run the `init q` command to make Linux reread this file.
7. Try pressing CTRL-ALT-F2 again. What happens?
8. Restore your original `/etc/inittab` configuration file.

## SELF TEST ANSWERS

### Basic Linux Knowledge

1. ☒ **C**. The **yy** command copies the entire line associated with the current location of the cursor. You can then use the **p** command to insert that line into the file.  
☒ **A** is incorrect, since the **p** command only takes data from the buffer. **B** is incorrect, since there is no **c** command. **D** is incorrect, since it places only one word into the buffer.

### Linux Filesystem Hierarchy and Structure

2. ☒ **D**. Current MS-DOS floppy disks are usually formatted to VFAT, which supports long filenames.  
☒ **A** is not correct, since you need to specify a file type with the **-t** switch. **B** and **C** are not correct, since ext2 and ext3 are unusable file types for an MS-DOS disk.

### Basic Commands

3. ☒ **D**. While this level of piping isn't covered in this chapter, this should be a straightforward question if you're sufficiently familiar with basic command line tools. The **who** command returns every active login of every user. Piping the result returns just the lines associated with the logins of user **mj**. Piping that result to **wc -l** returns the actual number of lines.  
☒ **A** is not correct, as the **wc** command needs a file or other input to read first. **B** is not correct, as it returns the lines associated with the logins of user **mj**. While you could count the number of lines, that does not address the requirements of the question. **C** is not correct, as it would return the number of times all users are logged into this system.

### Printing

4. ☒ **B**. The **lpq -l** command checks print queues. If you get an error message from this command, you may need to install a printer first.  
☒ **A** is incorrect, as **lpr** is a print command, and its **-l** switch tells **lpr** to expect a binary file. **C** is incorrect, as **lprm** is for removing print jobs. **D** is incorrect, as **lpd** is the line print daemon. While this must be running before you can check a print queue, **lpd** does not itself check print queues.

### Shells

5. ☒ **D**. The variable is **PATH**. When you input **\$PATH**, the value of that variable, in this case, the directories in your path, are substituted in this equation.

- ☒ **A** and **B** are not correct, since `$PATH` is not itself a variable. **C** is not correct, since `/sbin` is the wrong directory.

## Basic Security

- 6. ☒ **D**. The effect of `umask` has changed. Even if you try to set it to allow execute permissions, Red Hat won't let you do this anymore. You'll need to set execute permissions on each file after creation.
- ☒ **A** is not correct, since the file is not necessarily a directory, and execute permissions are no longer set up by default. **B** is not correct, as this would correspond to a `umask` value of 0666 or 0777. **C** is not correct, as execute permissions are no longer set up by default.

## System Administration

- 7. ☒ **B**. This is based on the convention for the first five entries in a crontab line: minute, hour, day of month, month, and day of week.
- ☒ **A**, **C**, and **D** are incorrect, as they are readings of the cited crontab line that don't correspond to the convention.

## Basic TCP/IP Networking

- 8. ☒ **B**. By convention, a network with a 192.168.14.0 address with a 255.255.255.0 subnet mask uses a 192.168.14.255 broadcast address.
- ☒ **A** is not correct, as there cannot be a network that starts with a broadcast address. **C** is not correct, as there cannot be a network that starts with a subnet mask. **D** is not correct, as 255.255.255.0 is not a qualified broadcast address.

## Standard Network Services

- 9. ☒ **B**. The Server Message Block (SMB) file system, also known as Samba, is the standard way to connect Linux as a member of a Microsoft Windows– or IBM OS/2–based network.
- ☒ **A** is not correct. While it is possible to set up “Services for Unix” on some Microsoft Windows computers, that would no longer be a Microsoft Windows–based network. **C** is not correct, since the Domain Name System has nothing to do with protocols necessary to connect operating systems. **D** is not correct, since Windows for Workgroups is not an available service in Linux.

## Basic Network Security

- 10. ☒ **B** and **C** are both current Red Hat Enterprise Linux firewall configuration utilities.
- ☒ **A** and **D** are not valid Red Hat Enterprise Linux commands.

## Basic Hardware Knowledge

11. ☒ **B.** The PCI system really does make sure that installed PCI devices share IRQ ports, as needed. This works as well for any USB or IEEE 1394 devices on your system.
- ☒ **A** is incorrect because it is possible for PCI devices to share IRQ ports. In fact, USB devices can also share an IRQ port. **C** is incorrect because all devices need CPU service. **D** is incorrect because no plug and play system by itself can compensate for a lack of available IRQs.

## Hardware Compatibility

12. ☒ **B** and **C.** Red Hat Enterprise Linux community knowledge hardware lists devices that have not been tested by Red Hat. But others in the Linux community have tested such hardware, and the results are often documented in the LDP's Hardware HOWTO. Many device manufacturers now include any special installation instructions that you may need to install their devices on Linux.
- ☒ **A** and **D** are both incorrect. Most devices are compatible with Linux. It would be a waste to remove hardware from your PC that Red Hat Enterprise Linux would recognize without any problems. While winmodems are a special case, some winmodems can be made to work with Red Hat Enterprise Linux.

## Configuring External Devices

13. ☒ **A.** As with modems, the `/dev/mouse` file is linked to the port used by your mouse.
- ☒ **B** is incorrect since the question addresses device filenames in Linux, not any hardware issue. **C** is incorrect, since you don't know which serial port is attached to your mouse. **D** is incorrect, since you shouldn't have to reconfigure your mouse just to find the serial port to which it is attached.

## Preparing to Install Linux

14. ☒ **A, B, C, D.** Most Linux installation programs allow you specify the model and manufacturer of the graphics card. This information is correlated as part of the Linux installation database to provide information on other needed settings, including chipset and video memory. You do want to make sure Linux does not exceed the horizontal or vertical refresh capabilities of the monitor, to minimize the risk of damage. The video memory allows you to verify what the Red Hat Enterprise Linux installation program reads from your system. If you exceed the resolution capabilities of the monitor, the graphics may degrade.
- ☒ There are no incorrect answer choices.

# LAB ANSWERS

## Lab 1

There are many ways to configure the IP addresses on a LAN. But it is generally best to do it by setting up a network from one of the private IP address ranges. When you configure networking on your LAN, pay particular attention to the computer that also has a connection to the Internet. The IP address of its connection to your network will be the gateway address for every other computer on your LAN. It's also the logical location for any firewall that you may wish to configure.

## Lab 2

When you troubleshoot a Red Hat Enterprise Linux computer, one of the things you'll be working through are critical configuration files. One key file in the boot process is `/etc/inittab`. One thing that I can do in this book is to illustrate the behavior of potential problems. The more problems that you're familiar with, the easier it is to troubleshoot or debug a problem during the RHCT and RHCE exams. However, there is often more than one way to solve a problem. I present one method. You may be able to find others.

To go through this lab, I'd take the following steps:

1. Log in as the root user. You can do this from either the GUI or the text login interface.
2. Run the `cp /etc/inittab /root/inittab` command. This backs up the subject configuration file in the root user's home directory.
3. Open the subject file with the `vi /etc/inittab` command.
4. Scroll down to until you see the following line:  

```
id:3:initdefault
```
5. The number after the `id` command identifies your starting runlevel. If it's 3, Linux starts in text mode; if it's 5, Linux starts in the GUI.
6. Change this number from 3 to 5 (or 5 to 3).
7. Save your changes and exit from the vi editor with the `:wq` command.
8. Reboot your computer with the `reboot` command.
9. Linux should now start in your new runlevel (3 or 5).
10. Restore your original settings in `/etc/inittab`. You can do this by opening `/etc/inittab` with the vi editor. Alternatively, you can copy your backup from the `/root` directory with the `cp /root/inittab /etc/inittab` command.

## Lab 3

In this lab, we experiment with deactivating a specific virtual console. By default, six virtual text login consoles are configured in the `/etc/inittab` configuration file. In this lab, we deactivate the second of the six consoles.

1. Log in as the root user. You can do this from either the GUI or the text login interface. If you're in the GUI, open a text console. Right-click on the desktop and click New Terminal in the pop-up menu.
2. Run the **cp /etc/inittab /root/inittab** command. This backs up the subject configuration file in the root user's home directory.
3. Open the subject file with the **vi /etc/inittab** command.
4. Scroll down until you see the following line:

```
2:2345:respawn:/sbin/mingetty tty2
```

5. Press CTRL-ALT-F2. This should start a text login interface. You should be able to log in at the prompt with your username and password.
6. If you logged into the GUI, press CTRL-ALT-F7 to return to the GUI. If you logged into the text interface, press CTRL-ALT-F1 to return to your original screen.
7. Turn the **subject** command in `/etc/inittab` into a comment. Add a comment character in front of the line as shown:

```
#2:2345:respawn:/sbin/mingetty tty2
```
8. Close and save this change to `/etc/inittab` with the **:wq** command.
9. Make Linux reread `/etc/inittab` with the **init q** command.
10. Press CTRL-ALT-F2. This should start a text login interface. Try logging in again. You'll see that it's not possible. Now you can see how adding a comment character to the right line in `/etc/inittab` deactivates the second virtual console.
11. If you logged into the GUI, press CTRL-ALT-F7 to return to the GUI. If you logged into the text interface, press CTRL-ALT-F1 to return to your original screen.
12. Restore your original settings in the `/etc/inittab` file.