



3

After Installation

CERTIFICATION OBJECTIVES

- | | | | |
|------|--------------------------------------|------|--|
| 3.01 | Installation Troubleshooting | 3.06 | Logical Volume Management:
Configuration and Administration |
| 3.02 | Postinstallation Validation | 3.07 | Automated Installation with Kickstart |
| 3.03 | Boot Loaders: GRUB and LILO | ✓ | Two-Minute Drill |
| 3.04 | Partitioning Utilities | Q&A | Self Test |
| 3.05 | RAID Configuration and Data Recovery | | |

In this chapter, you will learn how to manage Linux in advanced installation and configuration scenarios. You'll see how to troubleshoot an installation in progress, as well as how to create automated installation scripts through the Kickstart process. You'll discover how to organize disks with `fdisk`. You'll find out how to manage boot loaders on Linux. You will learn how to implement a Redundant Array of Inexpensive Disks (RAID), as well as a Logical Volume Manager (LVM) disk system.

Remember, half of the RHCE and RHCT exams is based on how well you know the installation process. By the time you finish this chapter, you should be ready to install Linux in an automated fashion from a local boot disk or over a network from an NFS, FTP, or HTTP server.

As you study `fdisk`, RAID, LVM, and the GRUB boot loader, you'll also see how you might troubleshoot problems that can keep you from accessing a partition or even booting Linux completely. In other words, the skills you learn in this chapter can also help you on the Troubleshooting and System Maintenance exam.

INSIDE THE EXAM

Installation Skills

Both the RHCE and RHCT exams require you to install Linux in some configuration. You won't know the specifics until you see your exam. If you have problems during the installation process, the Installation Troubleshooting consoles can be a lifesaver. If you discover a bad network connection or unreadable installation CDs, you may discover that using the troubleshooting consoles can

help you identify the problem to the exam administrator.

It's easiest to configure partitions, to set up RAID arrays, and create LVM groups during the installation process. However, when people get nervous on exams, they may forget details. All is not lost. You don't have to reinstall. You can use commands and configure key files to create new partitions, RAID arrays, and LVM groups.

Troubleshooting Skills

Troubleshooting skills are more difficult. In many cases, you need to know what looks right

with installation files, with boot loaders, and more. Then you can observe the symptoms of a

INSIDE THE EXAM

problem, look at a critical file, say “Aha!” and fix whatever problems you might encounter

during the Troubleshooting and System Maintenance exam.

Using Other Versions of Red Hat

For the purpose of this chapter, you can use Red Hat Linux 9 to test your knowledge of what happens after installation. For what is covered in this chapter, the differences do not matter for the Red Hat exams. However, as described in Chapter 2, I’ve illustrated each of the installation screens for RHEL 3 in Appendix A.

For example, RHEL 3, unlike Fedora Linux, still uses a text-based GRUB configuration menu. It’s the same version of GRUB used in Red Hat Linux 9. However, the kernel included with RHEL 3 no longer requires the `hdc=ide-scsi` command formerly required for most CD/RW drives.

CERTIFICATION OBJECTIVE 3.01

Installation Troubleshooting



There are actually many processes running and many parts to the installation. The system logs everything to an installation log file and separates related information between four of the five virtual console screens supported during the installation.

If your installation is trouble-free, you’ll have a few minutes on your hands during the Installation and Configuration exam. I suggest that you use that time to plan how you’ll configure the services per the requirements of your particular exam. But pay attention to the following sections. If your installation gets stuck, the console screens described can quickly help you diagnose the problem.

The Console Installation Output Screens

There are six consoles available during the installation process. Each console tells a different story. What you see depends slightly on whether you install in text or graphical mode. A network graphical installation is something of a hybrid; it starts in text mode before connecting to the network source and proceeding to the graphical installation.

Text mode starts in the first virtual console. Graphical mode runs in the seventh virtual console (console number 6 is not used). You can switch between virtual consoles using the commands defined in Table 3-1. If you're in text mode, you don't need to use the CTRL key (but it does no harm). As you can see in the table, each console is associated with a function key.

The messages on the third and fourth consoles can scroll by quickly; fortunately they're collected in dedicated files which I'll show you shortly.

Installation bash

You can find a bash shell on the second console. It can help you review what has been installed so far. Check it out for yourself with the CTRL-ALT-F2 command. You'll see the following installation boot prompt during the installation process:

```
-/bin/sh-2.05b#
```

This prompt allows you to run standard bash commands on the system as configured so far. Before Anaconda starts installing packages, you can inspect a number of things at

TABLE 3-1 Installation Virtual Consoles

Command	Console and Function
CTRL-ALT-F1	Text installation display; if you're running in graphical mode, it includes the basic commands to start graphics drivers.
CTRL-ALT-F2	Accesses a bash shell prompt; available after the first few installation steps.
CTRL-ALT-F3	Lists the log of installation messages.
CTRL-ALT-F4	Displays all kernel messages, including detected hardware and drivers.
CTRL-ALT-F5	Installation displays partition formatting; not available until Anaconda formats the actual partitions.
CTRL-ALT-F7	Graphical installation display; active only if you're running the installation program in graphical mode.



this prompt. The installation files from the CD or network source will be mounted on the /mnt/source directory.

While installation proceeds, you'll have a bit of "dead time." You can use this time to start configuring your RHEL 3 system. Just press the CTRL-ALT-F2 command and you'll see a shell. You'll find the standard root directory (/) mounted on the /mnt/sysimage subdirectory during the installation process. You can edit the files of your choice as soon as they're installed.

You can also find a number of interesting files in the /tmp directory; I've described the significant ones in Table 3-2.



If you're installing from a Kickstart configuration file, ks.cfg, you may be able to find it in the /tmp directory in the second virtual console during the installation process. It could be helpful to read this file during the Troubleshooting exam.

Other Consoles

The third console primarily lists detected hardware. If your computer is having problems with something critical such as the CD drive or network card, you'll see it here.

The fourth console tells you more about detected hardware. However, you may need to be a detective to understand these messages. For example, if you see the following message:

```
<6>pcnet32: 1 cards_found.
```

TABLE 3-2 The /tmp Directory Contains Configuration Files During the Installation Process

File from installation /tmp	Description
anaconda.log	Contains a log of installation messages (from the third console).
ks.cfg	If you're installing from a Kickstart configuration file, it's stored in the /tmp directory.
modules.conf	Includes detected hardware that requires a driver module (frequently includes network cards).
netinfo	Contains IP address information for configured network cards.
syslog	Includes a log of kernel messages (from the fourth console).
XF86Config.test	Contains a temporary X Window configuration file.

you might not know there's a problem unless you remember that there are two network cards on this computer. On the fifth console, you can see what happens to your partitions; it lists the output of the **mke2fs** command, which can tell you if there's a problem with your partitions.

While Installing Software

Once Anaconda starts installing software, you'll see the Installing Packages screen, where you can watch as it actually installs Linux on your computer. Once this process starts, press the CTRL-ALT-F2 keys to return to the bash console. Now run the following command:

```
-/bin/sh-2.05b# cd /mnt/sysimage
```

You can now browse around the directory tree as it's being built.

CERTIFICATION OBJECTIVE 3.02

Postinstallation Validation

Red Hat Enterprise Linux 3 creates a number of files when installation is complete. These files essentially document what happened. The basic installation log file, `/root/install.log`, lists the packages that Anaconda installed on your system. The boot information is stored in the `/var/log/dmesg` file. The commands used by Anaconda to install Linux is stored in the `/root/anaconda-ks.cfg` file. This can serve as a template for the Kickstart process, which you can use to install RHEL 3 automatically on different computers. I describe Kickstart in more detail later in this chapter.

The Installation Log File

The installation log file, `/root/install.log`, provides a baseline. After you run Linux for some time, you've probably installed and upgraded a number of additional packages. You can refer back to this file to find the packages installed when Linux was installed on this computer.

dmesg Boot Messages

The `/var/log/dmesg` file contains boot messages duplicated from the console output as seen each time Linux boots. These messages contain hardware information, process initialization, and sequencing information, and more, as shown in Figure 3-1.

Graphical or Text Login Screen

After the installation process is complete, the installation script shuts down and reboots the computer. If you configured a graphical login, Linux starts in runlevel 5, the X Window login screen. Alternatively, if you configured a text login, Linux starts in runlevel 3.



If your X Window server is not configured properly, you can press CTRL-ALT-F2 to go back to a text-based login screen.

It's easy to shift runlevels. The `init x` command, where `x` represents the runlevel, modifies the Linux system accordingly. For example, if you have a text login, the `init 5` command should move Linux to a graphical login screen, assuming you have the appropriate packages such as GNOME and the X Window System installed. Alternatively, the `init 3` command moves Linux to a text login screen.

FIGURE 3-1

dmesg boot
messages

```
OMB HIGHMEM available.
128MB LOWMEM available.
On node 0 totalpages: 32768
zone(0): 4096 pages.
zone(1): 28672 pages.
zone(2): 0 pages.
Kernel command line: ro root=LABEL=/ hdc=ide-scsi
ide_setup: hdc=ide-scsi
Initializing CPU#0
Detected 1193.754 MHz processor.
Console: colour VGA+ 80x25
Calibrating delay loop... 2359.29 BogoMIPS
Memory: 124308k/131072k available (1524k kernel code, 5288k reserved, 1086k data
, 164k init, 0k highmem)
zapping low mappings.
Dentry cache hash table entries: 16384 (order: 5, 131072 bytes)
Inode cache hash table entries: 8192 (order: 4, 65536 bytes)
Mount cache hash table entries: 512 (order: 0, 4096 bytes)
Buffer cache hash table entries: 8192 (order: 3, 32768 bytes)
Page-cache hash table entries: 32768 (order: 5, 131072 bytes)
CPU: L1 I cache: 16K, L1 D cache: 16K
CPU: L2 cache: 512K
Intel machine check architecture supported.
```

CERTIFICATION OBJECTIVE 3.03

Boot Loaders: GRUB and LILO

There are two main boot loaders associated with Red Hat Enterprise Linux: GRUB, the GRand Unified Bootloader, and LILO, the Linux Loader. Red Hat is moving toward making GRUB the only boot loader for its Linux distributions. You can't even install LILO during the Fedora Linux installation process.

There are a couple more clues. LILO is still included in the RHCE and RHCT exam requirements as of this writing. However, LILO is no longer included in the publically available outline for RH300, the main Red Hat prep course for the RHCE exam. But you'll need to come to your own conclusion on whether to study LILO for your exam.



You'll see the terms “bootloader” and “boot loader” used interchangeably. They both refer to software stored on the MBR of a hard disk which can load one or more operating systems. You may also see “Grand Unified Boot Loader” and “Grand Unified Bootloader” both referring to GRUB.

Boot Loader Background

To understand the role of a boot loader, take a step back from Linux. When you boot your computer, the BIOS starts by detecting basic hardware, including your hard drives. Once it's done, it looks for the boot loader on the Master Boot Record of the first available disk. If you're working with an older PC, the BIOS can't find your boot loader unless it's located within the first 1,024 cylinders of the hard disk.

Newer BIOSes overcome this problem with Logical Block Addressing, which is also known as LBA mode. LBA mode reads “logical” values for the cylinder, head, and sector, which allows the BIOS to “see” a larger disk drive.

If you have multiple hard drives, there is one more caveat. If your drives are IDE hard drives, the /boot directory must be on a hard drive attached to the primary IDE controller. If your drives are all SCSI hard drives, the /boot directory must be located on a hard drive with SCSI ID 0 or ID 1. If you have a mix of hard drives, the /boot directory must be located on either the first IDE drive or a SCSI drive with ID 0. In other words, this is not an issue on the Red Hat exams unless the computer that you're tested on has more than two hard drives. And I believe that's less likely, as that would increase the cost of the exam.

GRUB, the GRand Unified Bootloader

Red Hat is moving toward GRUB, the GRand Unified Bootloader as the only way to start its Linux distributions. When you start your computer, your BIOS looks for the /boot directory and finds the GRUB menu, which will look similar to Figure 3-2. If you've configured your computer with multiple operating systems, you can use the GRUB menu to boot any operating system detected during the Linux installation process.

If you need to do something special with GRUB, you can edit the commands. If GRUB is password protected, you'll need to start with the **p** command. Use the **e** command to *temporarily* edit the file. You'll see a number of basic commands that you can use to modify GRUB as shown in Figure 3-3 and Table 3-3. You can use these commands to test different GRUB configurations. They can help you troubleshoot problems with the GRUB configuration file without booting, editing, and rebooting your system. That could possibly help you save time during the Troubleshooting and System Maintenance exam. But once you find the solution, make sure to record the change in the GRUB configuration file, /etc/grub.conf.

You can also use GRUB to boot other operating systems, including various versions of Microsoft Windows. See Chapter 5 for more information, including a detailed line-by-line analysis of a dual-boot /etc/grub.conf file.

FIGURE 3-1

The GRand
Unified
Bootloader
(GRUB)

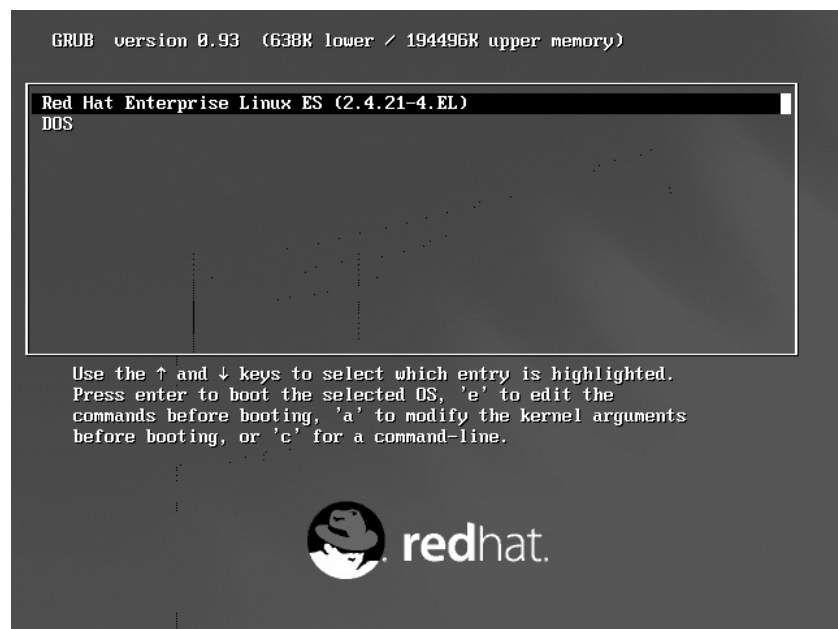
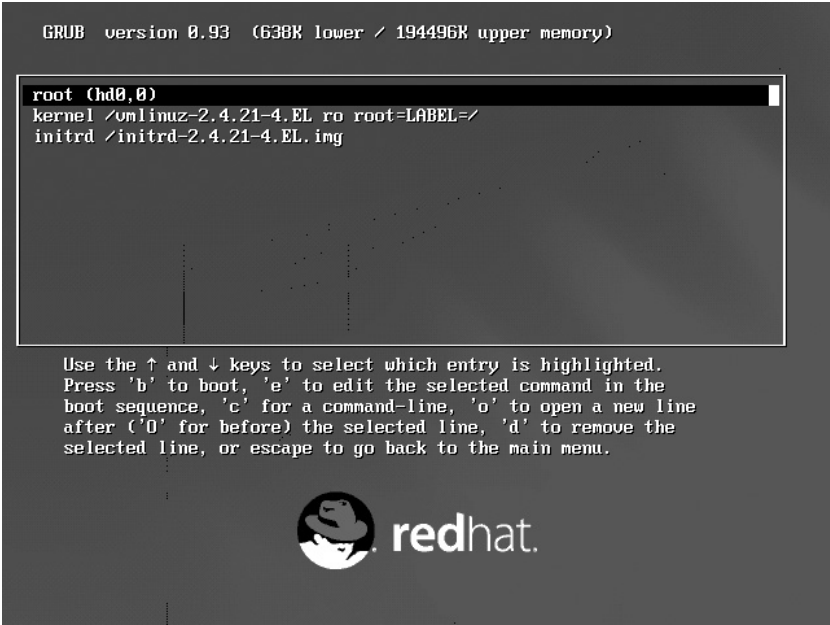


FIGURE 3-2

Details of GRUB



If you’ve previously installed LILO and now want to install GRUB, edit the `/etc/grub.conf` file and then run the `grub-install /dev/xdy` command, where `xdy` represents the hard disk with your `/boot` directory.

The `/etc/grub.conf` file is linked to the actual GRUB configuration file, `/boot/grub/grub.conf`. When you edit `/etc/grub.conf`, changes are automatically reflected in `/boot/grub/grub.conf`.

TABLE 3-3 GRUB Editing Commands

Command	Description
b	Boot the currently listed operating system
d	Delete the current line
e	Edit the current line
o	Create an empty line underneath the current line
O	Create an empty line above the current line

INSIDE THE EXAM

ADD TITLE HERE

If you're troubleshooting GRUB, the GRUB menu can help you experiment with changes quickly. However, if you find the solution in this way, you'll still need to change (and test) the GRUB configuration file, `/etc/grub.conf`.

If you find a problem with GRUB during the Troubleshooting and System Maintenance

exam and are unsure about the solution, testing changes through the GRUB menu could save you time. However, until you record the change in the GRUB configuration file, `/etc/grub.conf`, you won't get credit for the work that you've done.

The LILO Alternative

The more traditional boot loader is LILO, the Linux Loader. It has been deprecated from RHEL 3 as well as Fedora Linux. Since it's no longer a part of the RH300 curriculum, I doubt that you'll see it on the RHCE exam. However, it is still a part of the RHCT prep course (RH133) curriculum as of this writing. You'll have to make your own judgment on whether you want to take the time to study LILO for your Red Hat exam.

The LILO configuration file is `/etc/lilo.conf`. If you edit this file, you need to implement these changes with the `lilo` command. The following is a typical `/etc/lilo.conf` file:



If you've installed GRUB on your system, you can still find a LILO configuration file in `/etc/lilo.conf.anaconda`.

```
prompt
timeout=50
default=linux
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
message=/boot/message
linear

image=/boot/vmlinuz-2.4.21-3
    label=linux
```

12 Chapter 3: After Installation

```
initrd=/boot/initrd-2.4.21-3.img
read-only
append="hdc=ide-scsi root=LABEL=/"
other=/dev/hda1
label=Win98
table=/dev/hda
```

Now here's the line-by-line interpretation of this file:

```
prompt
```

This forces the **lilo boot:** prompt to appear on the console.

```
timeout=50
default=linux
```

This forces LILO to wait five seconds (50 tenths of a second) before moving to the default operating system, labeled linux.

```
boot=/dev/hda
map=/boot/map
```

The system looks for boot and map information on the first IDE hard disk.

```
install=/boot/boot.b
```

This is the secondary boot loader.

```
message=/boot/message
```

This opens a graphical file that you see during the boot process.

```
linear
```

This uses linear addresses, and not standard hard disk geometry, to find the /boot directory.

```
image=/boot/vmlinuz-2.4.21-4
```

This is the actual virtual memory compressed kernel (version 2.4.21, with the -4 representing the Red Hat build number) used for this particular Linux system.

```
label=linux
```

This identifies the “boot option” label that appears if you press TAB at the LILO prompt.

```
initrd=/boot/initrd-2.4.21-4.img
```

This reveals the location of the second stage loader, the RAM disk.

```
read-only
```

During installation, the RAM disk filesystem is started in read-only mode. After the second stage is finished with a few tests, the RAM disk is unloaded from memory, and the real root partition is mounted from the filesystem, normally in read/write mode.

```
append="root=LABEL=/"
```

This adds a command to the kernel. The **root=LABEL=** command points to the label associated with the root directory, which you can find with the **e2label** command described in Chapter 11.

The next section is another boot option—in this case, to boot Windows 98 on the first partition, **/dev/hda1**.

```
other=/dev/hda1
    label=Win98
    table=/dev/hda
```

The table is the location of the partition table to be used—in this case, the first physical IDE hard disk. If you edit this file, you need to run the **lilo** command to write the result to the appropriate boot record, usually the Master Boot Record (MBR) of your hard drive.

GRUB and LILO Parameters

To pass a parameter to GRUB, type the **a** command in the first GRUB menu. This allows you to append the command sent to the kernel. After typing the **a** command, you might see a command line similar to the following:

```
grub append> ro root=LABEL=/
```

You can add the command of your choice to the end of this command. For example, if you add “single” after the end of this line, Linux starts in single-user mode. If you’re having trouble getting Linux to recognize all of the RAM on your computer, try adding **mem=xyzM** (where xyzM represents the amount of memory on your computer) to the end of the line.

LILO works in a similar way. At the LILO boot: prompt, type the parameter after the label name. For example, if you wanted to start your system in rescue mode, you

would type **linux rescue**. If you wanted to start your system in single-user mode, type the following command at the prompt:

```
lilo boot: linux single
```

A wide variety of boot parameters are available; run the **man bootparam** command for more information.

Single-User Mode

Except for a normal boot of Linux, single-user mode is the most commonly used option. This is the system maintenance mode for experienced Linux administrators. It allows you to perform clean backups and restores to any partitions as needed from local hardware. It also allows you to run administration commands, recover or repair password and shadow password files, run filesystem checks, and so forth.

In some cases, to get out of single-user mode you just have to type **exit** and your system will go into multiuser mode. If you have made changes or repairs to any partitions, you should reboot the computer with the **reboot** command. If you've made changes during your exam, you'll want to test those changes with a reboot.

LILLO Errors

The LILLO first stage will also indicate some common and not-so-common problems:

```
(nothing)    did not get to lilo at all
L            first stage loaded and started
LI           second stage loaded from /boot
LILLO        all of lilo is loaded correctly
```

Occasionally, there may be an error due to partition table changes, bad blocks, and so forth. On these rare occurrences, you will only get partial LILLO prompts:

```
LIL          second stage boot loader is started
LIL?         Second stage loaded at an incorrect address
LIL-         the descriptor table is corrupt
```

CERTIFICATION OBJECTIVE 3.04

Partitioning Utilities

It's best to create partitions using Disk Druid during the installation process. This can save you grief as an administrator—and especially during the Red Hat exams. However, mistakes are made. You might forget to create a critical partition during the Installation and Configuration exam. You might need to create a larger /home directory partition for your users. For this purpose, the standard is still the **fdisk** utility, which I'll describe shortly.

Before you use **fdisk** to create or revise partitions, you should check your free space and the partitions that are currently mounted. You can do this with the **df** and **mount** commands. The following example illustrates how the **df** command displays the total, used, and available free space on all currently mounted filesystems.

Note the numbers under the 1k-blocks column. In this case, they add up to about 3GB of allocated space. (Don't include the space associated with /dev/hdc, which refers to a CD drive.) If your hard drive is larger, you may have unallocated space that you can use for another partition.

```
[root@Enterprise root]# df
Filesystem      1k-blocks    Used Available Use% Mounted on
/dev/hda8        932833    502478    382162   57% /
/dev/hda7         23300      2588     19509   12% /boot
/dev/hda1       1052064    914784    137280   87% /dosC
/dev/hda6       1052064    111648    940416   11% /home
none              62828         0      62828  100% /dev/shm
/dev/hdc         556054    556054         0  100% /mnt/cdrom
[root@Enterprise root]#
```

The second command, **mount**, includes the filesystem type. In this case, we see the partition represented by device /dev/hda1 mounted with the VFAT file type on the /DosC directory. It provides direct access to the C: drive of the Windows operating system. I've set up the data shown from the **mount** command in columns for clarity; what you actually see in the RHEL 3 command line is less organized.

```
[root@Enterprise root]# mount
/dev/hda8 on /          type ext3 (rw)
none      on /proc              type proc (rw)
/dev/hda7 on /boot              type ext3 (rw)
/dev/hda1 on /dosC              type vfat (rw)
```

```

/dev/hda6 on /home      type vfat (rw)
none      on /dev/pts    type devpts (rw,mode=0620)
/dev/hdb  on /mnt/cdrom  type iso9660 (ro)
[root@Enterprise root]#

```

on the


As of this writing, Linux can reliably recognize NTFS partitions only in read-only mode. While support for write mode is available through the Linux kernel, it is considered “dangerous.”

One of the benefits is that you can move and copy files between the Linux and DOS partitions using standard Linux commands. You cannot, however, run many Microsoft Windows applications within Linux unless you run a DOS or Windows Emulation package such as WINE (www.winehq.org) or CrossOver Office (www.codeweavers.com).

The fdisk Utility

The **fdisk** utility is universally available and should be one of the first tools you get acquainted with. There are many commands, even an expert mode, but you only need to know a few as discussed here.

Though you can modify the physical disk partition layout using many programs, we will be discussing the Linux implementation of **fdisk**. **FDISK.EXE** from DOS has the same name and is also used for creating partitions, but it doesn't incorporate any Linux-compatible features. It also includes a different interface.

Using fdisk: Starting, Getting Help, and Quitting

The following screen output lists commands that show how to start the **fdisk** program, how to get help, and how to quit the program. The **/dev/hda** drive is associated with the first IDE drive on a regular PC. Your computer may have a different hard drive; you can check the output from the **df** and **mount** commands for clues.

As you can see, once you start **fdisk**, it opens its own command line prompt.

```

# fdisk /dev/hda
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): q

Using fdisk: In a Nutshell

In the **fdisk** command line prompt, start with the print the partition table command (**p**). This allows you to review the current entries in the partition table. Assuming you have free space, you then create a new (**n**) partition, either primary (**p**) or logical (**l**). If it doesn't already exist, you can also create an extended partition (**e**) to contain your logical partitions. Remember that there are four primary partitions, which correspond to numbers 1 through 4. One of the primary partitions can be redesignated as an extended partition. The remaining partitions are logical partitions, numbered between 5 and above. The Linux **fdisk** utility won't allow you to create more than 16 partitions on the drive.

When you assign space to a partition, you're assigning a block of cylinders on that hard disk. If you have free space, the **fdisk** default starts the new partition at the first available cylinder. The actual size of the partition depends on disk geometry; do not worry about exact size here.

Using fdisk: Deleting Partitions

In the following example, you will remove the only partition. The sample output screen first starts **fdisk**. Then you print (**p**) the current partition table, delete (**d**) the partition by number (1 in this case), write (**w**) the changes to the disk, and quit (**q**) from the program. Needless to say, *do not perform this action on any partition where you need the data.*

```

# fdisk /dev/hdb
Command (m for help): p
Disk /dev/hdb: 255 heads, 63 sectors, 525 cylinders
Units = cylinders of 16065 * 512 bytes

```

```

Device      Boot      Start          End      Blocks      Id  System
/dev/hdb1   *                1          525     4217031     6   FAT16
Command (m for help): d
Partition number (1-1): 1

```

This is the last chance to change your mind before deleting the current partition. If you want to quit without saving changes, exit from **fdisk** with the **q** command. If you're pleased with the changes that you've made and want to make them permanent, proceed with the **w** command:

```
Command (m for help): w
```

You did it! Now you have an empty hard disk or hard disk area where you can create the partitions you need.

Using fdisk: Creating Partitions

The following screen output sample shows the steps used to create (**n**) the first (/boot) partition, make it bootable (**a**), and then finally write (**w**) the partition information to the disk. (Note: Although you may ask for a 100MB partition, the geometry of the disk may not allow that precise size, as shown in the example.)

```

# fdisk /dev/hdb

Command (m for help): n
Command action
  e   extended
  p   primary partition (1-4)
p
Partition number (1-4):
First cylinder (1-256, default 1): 1
Last cylinder or +size or +sizeM or +sizeK (2-256,def 256): +100M

Command (m for help): a
Partition number (1-4): 1

Command (m for help): p
Disk /dev/hdb: 255 heads, 63 sectors, 256 cylinders
Units = cylinders of 16065 * 512 bytes
   Device Boot      Start          End      Blocks      Id  System
/dev/hdb1   *                1          12        98163     83   Linux

Command (m for help):

```

Repeat the previous commands as appropriate to create any other partitions that you might need. I illustrate one possible group of partitions here:

Command (m for help): p

```
Disk /dev/hdb: 255 heads, 63 sectors, 256 cylinders
Units = cylinders of 16065 * 512 bytes
```

Device	Boot	Start	End	Blocks	Id	System
/dev/hdb1		1	2	16044	83	Linux
/dev/hdb2		3	18	64176	82	swap
/dev/hdb3		19	169	1203300	83	Linux
/dev/hdb4		170	250	649782	5	Extended
/dev/hdb5		170	201	248682	83	Linux
/dev/hdb6		202	257	449232	83	Linux

Command (m for help): w



The number of blocks that you see may vary slightly depending on the size of your hard disk, the number of heads, sectors, and cylinders on that disk, as well as the version of fdisk that you're using.

Using fdisk: A New PC with No Partitions

After installing Linux on a new PC, you'll want to use **fdisk** to configure additional physical disks attached to the system. For example, if it's the first disk attached to the secondary IDE controller, run the **fdisk /dev/hdc** command.

Remember the limitations on partitions. If you need more than four partitions on the new physical disk, configure type Primary for the first three partitions, and then Extended for the rest of the disk as partition 4. You can then creating logical partitions 5–16 within the extended partition.

Using fdisk: Creating a Swap Partition

You need to create a partition before you can reassign it as a swap partition. At the **fdisk** prompt, run the **L** command. You'll see a large number of file types, listed as hex codes. When you create a partition, **fdisk** creates a Linux Native type partition by default. As you can see from the output of the **L** command, the associated hex code is (83).

It's easy to reassign a partition as a swap partition. Run the **p** command. Remember the number of the partition you want to change. Make sure that partition doesn't have data that you want to save.

Now run the **t** command. Type in the number associated with the partition that you want to change. Type in the hex code for the type you want, in this case 82 for a Linux swap partition. For example, I could run the following sequence of commands to set up a new swap partition on the second IDE hard drive. The commands that I type are in bold. The details of what you see depend on the partitions that you may have created. It'll be a 1GB swap space on the first primary partition (/dev/hdb1).

```
# fdisk /dev/hdb
Command (m for help): n
Command action
    e   extended
    p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-10402, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-10402, default 10402): +1000M

Command (m for help): p

Disk /dev/hdb: 5368 MB, 5368709120 bytes
16 heads, 63 sectors/track, 10402 cylinders
Units = cylinders of 1008 * 512 = 516096 bytes

   Device Boot      Start         End      Blocks    Id  System
/dev/hdb1          1         1939     977224+    83  Linux

Command (m for help): t
Selected partition 1
Hex code (type L to list codes): 82
Changed system type of partition 1 to 82 (Linux swap)

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
#
```

The **fdisk** utility doesn't actually write the changes to your hard disk until you run the write (**w**) command. You have a chance to cancel your changes with the quit (**q**) command.

CERTIFICATION OBJECTIVE 3.05

RAID Configuration and Data Recovery

A Redundant Array of Independent Disks (RAID) is a series of disks that can save your data even if there is a catastrophic failure on one of the disks. While some versions of RAID make complete copies of your data, others use the so-called parity bit to allow your computer to rebuild the data on lost disks.

Linux RAID has come a long way. A substantial number of hardware RAID products support Linux, especially from name brand PC manufacturers. Dedicated RAID hardware can ensure the integrity of your data even if there is a catastrophic *physical* failure on one of the disks. Alternatively, you can configure software-based RAID on multiple partitions on the same physical disk. While this can protect you from a failure on a specific hard drive sector, it does not protect your data if there is a failure of the entire physical hard drive.

Depending on your definitions, RAID has nine or ten different levels, which can accommodate different levels of data redundancy. Only three levels of RAID are supported directly by RHEL 3: levels 0, 1, and 5. Hardware RAID uses a RAID controller connected to an array of several hard disks. A driver must be installed to be able to use the controller. Most RAID is hardware based; when properly configured, the failure of one drive in a RAID 1 or RAID 5 array does not destroy the data in the array. Linux, meanwhile, offers a software solution to RAID. Once RAID is configured on a sufficient number of partitions, Linux can use those partitions just as it would any other block device. However, to ensure redundancy, it's up to you in real life to make sure that each partition in a Linux software RAID array is configured on a different physical hard disk.



The RAID md device is a meta device. In other words, it is a composite of two or more other devices such as /dev/hda1 and /dev/hdb1 that might be components of a RAID array.

The following are the basic RAID levels supported on Red Hat Enterprise Linux 3.

RAID 0

This level of RAID makes it faster to read and write to the hard drives. However, RAID 0 provides no data redundancy. It requires at least two hard disks.

Reads and writes to the hard disks are done in parallel, in other words, to two or more hard disks simultaneously. All hard drives in a RAID 0 array are filled equally. But since RAID 0 does not provide data redundancy, a failure of any one of the drives will result in total data loss. RAID 0 is also known as “striping without parity.”

RAID 1

This level of RAID mirrors information to two or more other disks. In other words, the same set of information is written to two different hard disks. If one disk is damaged or removed, you still have all of the data on the other hard disk. The disadvantage of RAID 1 is that data has to be written twice, which can reduce performance. You can come close to maintaining the same level of performance if you also use separate hard disk controllers. That prevents the hard disk controller from becoming a bottleneck.

And it is expensive. To support RAID 1, you need an additional hard disk for every hard disk worth of data. RAID 1 is also known as disk mirroring.

RAID 4

While this level of RAID is not directly supported by current Linux distributions associated with Red Hat, it is still supported by the current Linux kernel. RAID 4 requires three or more disks. As with RAID 0, data reads and writes are done in parallel to all disks. One of the disks maintains the parity information, which can be used to reconstruct the data. Reliability is improved, but since parity information is updated with every write operation, the parity disk can be a bottleneck on the system. RAID 4 is known as disk striping with parity.

RAID 5

Like RAID 4, RAID 5 requires three or more disks. Unlike RAID 4, RAID 5 distributes, or “stripes,” parity information evenly across all the disks. If one disk fails, the data can be reconstructed from the parity data on the remaining disks. RAID does not stop; all data is still available even after a single disk failure. RAID level 5 is the preferred choice in most cases: the performance is good, data integrity is ensured, and only one disk’s worth of space is lost to parity data. RAID 5 is also known as disk striping with parity.



Hardware RAID systems should be “hot-swappable.” In other words, if one disk fails, the administrator can replace the failed disk while the server is still running. The system will then automatically rebuild the data onto the new disk. If you configure different partitions from the same physical disk for a software RAID system, the resulting configuration can easily fail. Alternatively, you may be able to set up “spare disks” on your servers; RAID may automatically rebuild data from a lost hard drive on properly configured spare disks.

RAID in Practice

RAID is associated with a substantial amount of data on a server. It’s not uncommon to have a couple dozen hard disks working together in a RAID array. That much data can be rather valuable.

If continued performance through a hardware failure is important, you can assign additional disks for “failover,” which sets up spare disks for the RAID array. When one disk fails, it is marked as bad. The data is almost immediately reconstructed on the first spare disk, resulting in little or no downtime. The next example demonstrates

INSIDE THE EXAM

Creating RAID Arrays

During the Red Hat Installation and Configuration exam, it’s generally easier to do as much as possible during the installation process. If you’re asked to create a RAID array, it’s easiest to do so with Disk Druid, which only works during installation. You can create RAID arrays once RHEL is installed, but as you’ll see in the following instructions, it is more time consuming, and involves a process that is more difficult to remember.

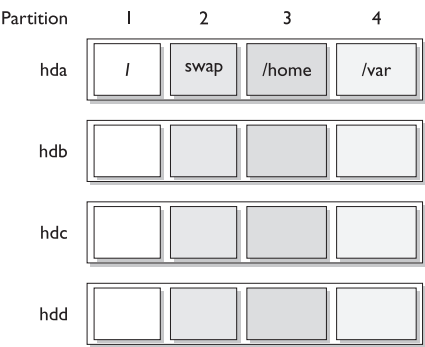
However, if you’re required to create a RAID array during your exam and forget to

create it during the installation process, not all is lost. You can still use the tools I describe in this chapter to create and configure RAID arrays during the exam. And the skills you learn here can serve you well through your career.

The exam may use examples from RAID levels 0, 1, and or 5. However, if the PC that you’re using on the exam includes only one physical hard disk, you may have to configure multiple RAID partitions on the same disk.

this practice in both RAID 1 and RAID 5 arrays. Assuming your server has four physical drives, with the OS loaded on the first, it should look something like this:

III 3-1



All four drives (hda, hdb, hdc, hdd) should be approximately the same size.

This first example shows how to mirror both the /home and the /var directories (RAID 1) on Drive 2 and Drive 3, leaving Drive 4 as a spare.

You need to create nearly identically sized partitions on Drives 2 and 3. In this example, four disks are configured with four partitions of the same size. If you use the Linux fdisk program, use the t command to toggle the drive ID type. You can then set the partition to type fd, which corresponds to the Linux raid autodetect filesystem. You'll get to test it out for yourself shortly in an exercise, as well as a lab at the end of this chapter.

In the partition table of the first drive is /dev/hda3 (currently mounted as /home) and /dev/hda4 (currently mounted as /var). The second drive includes /dev/hdb3 and /dev/hdb4. The third drive is set up with /dev/hdc3 and /dev/hdc4, while the last drive has /dev/hdd3 and /dev/hdd4. All of these partitions have been marked with partition IDs of type 0xFD.



If you need to create a raidtab configuration file during the exam, it may be faster to start with one of the sample raidtab configuration files. There are several available in the following directory: /usr/share/doc/raidtools-1.00.3.

Next, update the configuration file /etc/raidtab. As shown in the following code, you'll see two different RAID 1 arrays (/dev/md0 and /dev/md1):

```
raiddev /dev/md0
raid-level 1

nr-raid-disks 2
```

```

nr-spare-disks 1
persistent-superblock 1
chunk-size 4

device /dev/hdb3
raid-disk 0
device /dev/hdc3
raid-disk 1
device /dev/hdd3
spare-disk 0

raiddev /dev/md1
raid-level 1
nr-raid-disks 3
nr-spare-disks 1
persistent-superblock 1
chunk-size 4
  device /dev/hdb4
raid-disk 0
  device /dev/hdc4
raid-disk 1
  device /dev/hdd4
spare-disk 0

```

Table 3-4 shows what some of the commands are, along with a brief description of what they do.



Take special note that *raid-disks* and *spare-disks* start counting at 0; *nr-raid-disks* and *nr-spare-disks* are the correct number of drives. For example, if *nr-raid-disks* = 3, then the *raid-disks* are 0, 1, and 2.

If you haven't already done so, it's time to format these partitions and convert them to the default ext3 filesystem. The Linux format command is **mkfs**; with the

TABLE 3-4

Commands in
raidtab

Command	Description
nr-raid-disks	Number of RAID disks to use
nr-spare-disks	Number of spare disks to use
persistent-superblock	Required for autodetection
chunk-size	Amount of data to read/write
parity-algorithm	How RAID 5 should use parity

right switch, you can automatically set it up to ext3 with a journal. For example, the following command formats the `/dev/hda4` partition:

```
# mkfs -j /dev/hda4
```

If the partitions in `/etc/raidtab` are new, repeat this command for all of those other partitions.

Partitions from older Linux computers may be formatted to the ext2 filesystem, which is essentially the same as ext3 without a journal. You can add journaling to an older partition with a command such as:

```
# tune2fs -j /dev/hda4
```

When a journal is added to the ext2 filesystem, it upgrades that partition to the ext3 filesystem.



There are advantages to the ext3 journaling filesystem. If your system suffers a sudden power failure, it does not have to check every inode for file data; the information is already available in the journal.

The aforementioned `/etc/raidtab` file includes two RAID devices, `/dev/md0` and `/dev/md1`. To start RAID 1 on those devices, run the following commands:

```
# mkraid -R /dev/md0
# mkraid -R /dev/md1
```

If it works, you'll see the result in the dynamic `/proc/mdstat` file. You can now mount the device and format it with the appropriate `mkfs` command, and finally mount it on the Linux directory of your choice. You can even set it up to be automatically mounted through `/etc/fstab` as described in Chapter 4.



Yes, when you configure a RAID device, you're formatting the same space twice. First, you format the partitions that make up the array. Once you've made RAID devices such as `/dev/md0`, you can then format those devices as if they were new partitions.

For a RAID 5 array on the `/var` partition (in order to preserve mail, print spools, and log files), the `/etc/raidtab` file should be modified as follows:

```
raiddev /dev/md0
raid-level 5
nr-raid-disks 3
```

```

nr-spare-disks 1
persistent-superblock 1
chunksize 32

parity-algorithm right-symmetric
device /dev/hda4
raid-disk 0
device /dev/hdb4
raid-disk 1
device /dev/hdc4
raid-disk 2
device /dev/hdd4
spare-disk 0

```

Now you can run **mkraid /dev/md0** to initialize this RAID 5 device. You can then format and mount this RAID array on the Linux directory of your choice.

Formatting the RAID Array

Now you can run the **mkfs** command to format each RAID array. It's fairly simple; now that you've created arrays such as `/dev/md0` and `/dev/md1`, you can work with them as if they were any other hard drive partition. For example, you can format these arrays to the ext3 filesystems with the following commands:

```

# mkfs -j /dev/md0
# mkfs -j /dev/md1

```

The process is straightforward—for example, if you wanted to mount the `/home/mj` directory on the first RAID array, you'd run the following commands (assume the `/hometmp` directory exists):

```

# cp -r /home/mj /hometmp
# mount /dev/md0 /home/mj
# cp -r /hometmp /home/mj

```

Setting up RAID on a critical set of files such as a `/boot` directory partition is a bit trickier. Because of the importance of this data, manually copy the contents of the `/boot` directory (as well as the boot loader file, `/etc/grub.conf` or `/etc/lilo.conf`) to a different physical drive.

Implementing the RAID Array

But that's not the last step. You may not get full credit for your work on the exam unless the directory gets mounted on the RAID array when you reboot your Linux computer. Based on a standard RHEL 3 `/etc/fstab` configuration file, you might add the following line to that file:

```
LABEL=/home/mj    /home/mj    ext3    defaults    1 2
```

Before this line can work, you'll need to set the label for this directory with the following command:

```
# e2label /dev/md0 /home/mj
```

I describe the `/etc/fstab` file, including the meaning of the data in each these columns, in more detail in Chapter 4.

EXERCISE 3-1

Mirror the /home Partition with Software RAID

Don't do this exercise on a production computer. If you have a computer with Red Hat Enterprise Linux already installed with several different physical hard drives that you can use for testing, that is best. One alternative is to use virtual machine technology such as VMWare, which can allow you to set up these exercises with minimal risk to a production system. You can also set up several IDE and SCSI hard disks on a VMWare machine. When you're ready, use the Linux **fdisk** techniques discussed in Chapter 3 to configure the following two-drive partition scheme:

```
Drive 1:
hda1    256    /
hda2     64    swap
hda3    500    /home
hda4    256    /var

Drive 2:
hdb1   1200    /usr
hdb2     64    swap
hdb3    100    /tmp
hdb4    500    (not allocated)
```

Now with the following steps, you can create a mirror of hda3, which stores the /home directory, to the hdb4 partition. (The partition sizes do not have to be identical.)

If you're making fdisk changes on a production computer, back up the data in the /home partition first. Otherwise, all data on the current /dev/hda3 will be lost.

1. Mark the two partition IDs as type FD using the Linux fdisk utility.

```
# fdisk /dev/hda
Command (m for help) : t
Partition number (1-4)
3
Partition ID (L to list options): FD
Command (m for help) : w
# fdisk /dev/hdb
Command (m for help) : t
Partition number (1-4)
4
Partition ID (L to list options): FD
Command (m for help) : w
```

2. Update the configuration file /etc/raidtab with these lines of code:

```
# vi /etc/raidtab
raiddev /dev/md0
raid-level 1

nr-raid-disks 2
nr-spare-disks 0
persistent-superblock 1
chunk-size 4
device /dev/hda3
raid-disk 0
device /dev/hdb4
raid-disk 1
```

3. Now make the RAID device file md0 and format it this way:

```
# mkraid -R /dev/md0
# mkfs -j /dev/md0
```

4. All that's left is to restore the files to the device and mount it.
5. However, for the exam, you may not get full credit for your work unless your Linux system mounts the directory on the RAID device. Make sure to do so in the /etc/fstab configuration file. Run the **e2label** command as required to

make sure that the LABEL that you add to /etc/fstab is read properly the next time you boot Linux.

CERTIFICATION OBJECTIVE 3.06

Logical Volume Management: Configuration and Administration

Logical Volume Management (LVM) can allow you to manage active partitions. Before LVM, there was no easy way to increase or reduce the size of a partition after Linux is installed.



This isn't quite true; you can use the GNU Parted utility to manage the size of an active partition. However, while the Red Hat curricula does refer to fdisk, there is no reference as of this writing to Parted. For more information on Parted, see www.gnu.org/software/parted.

For example, if you find that you have extra space on the /home directory partition and need more space on your /var directory partition for log files, LVM will let you reallocate the space. Alternatively, if you are managing a server on a growing network, new users will be common. You may come to the point where you need more room on your /home directory partition. With LVM, you can add a new physical disk and allocate its storage capacity to an existing /home directory partition.



While LVM can be an important tool to manage partitions, it does not by itself provide redundancy. Do not use it as a substitute for RAID. However, you can use LVM in concert with a properly configured RAID array.

LVM Concepts

The LVM system organizes hard disks into Logical Volume (LV) groups. Essentially, physical hard disk partitions (or possibly RAID arrays) are set up in a bunch of

INSIDE THE EXAM

Logical Volume Management

One of the critical decisions during the Installation and Configuration exam is whether you install in text or graphical mode. Text mode is faster. However, if you're required to create an LVM group during your exam, you can configure with Disk Druid *only* if you install RHEL in graphical mode.

I can't give you a concrete time savings between graphical and text mode; it depends on the traffic demands (how many other users) and the hardware available during your exam. I can say that when I installed the standard RHEL server configuration in graphical mode, it took five minutes longer than the same process in text mode. If your computer has more than 256MB of RAM (and more than 16MB of video memory), I suspect the difference would decrease.

Alternatively, you can use the techniques I describe in this section to configure LVM groups during the exam. Even if you remember to configure LVM groups during the installation process, read through this section. During the exam, it'll allow you to confirm that the LVM groups were created correctly. And on the job, it'll allow you to create LVM groups for your users in the real world.

If you don't have to configure an LVM group during your Installation and Configuration exam, text mode saves you time. If you are asked to configure an LVM group during your exam, your choice depends on whether you can configure an LVM group more quickly during the graphical installation process or the techniques described in this chapter.

equal-sized chunks known as Physical Extents (PE). As there are several other concepts associated with the LVM system, let's start with some basic definitions:

- **Physical Volume (PV)** is the standard partition that you add to the LVM mix. Normally, a physical volume is a standard primary or logical partition. It can also be a RAID array.
- **Physical Extent (PE)** is a chunk of disk space. Every PV is divided into a number of equal sized PEs. Every PE in a LV group is the same size. Different LV groups can have different sized PEs.
- **Logical Extent (LE)** is also a chunk of disk space. Every LE is mapped to a specific PE.

- **Logical Volume (LV)** is composed of a group of LEs. You can mount a filesystem such as /home and /var on an LV.
- **Volume Group (VG)** is composed of a group of LVs. It is the organizational group for LVM. Most of the commands that you'll use apply to a specific VG.

As of this writing, there are two basic ways to create an LVM Volume Group in Red Hat Enterprise Linux. You can set up an LVM if you install RHEL 3 in graphical mode, using Disk Druid. Alternatively, you can use the basic LVM commands to create and manage Volume Groups after Linux is installed. As LVM is not much good unless you can add and delete specific LVs, you should know the basic LVM commands. But the first step is to create a PV.

Creating a Physical Volume

The first step to create an LVM is to start with a physical disk. If you have a freshly installed hard disk, you can set up a PV on the entire disk. For example, if that hard disk is attached as the third IDE hard disk (/dev/hdc), and you haven't configured partitions on the drive, you'd run the following command:

```
# pvcreate /dev/hdc
```

Alternatively, you can set up a new PV on a properly formatted partition. For example, assume that you've added a new partition, /dev/hdc2. You could then use **fdisk** or a similar tool to set it to the Linux LVM partition type. In fdisk, this corresponds to partition type 8e. The sequence of commands would look similar to the following:

```
# fdisk /dev/hdc
Command (m for help) : t
Partition number (1-4)
2
Partition ID (L to list options): 8e
Command (m for help) : w
```

Once your partition is ready, you can run the following command to create a new PV on that partition (/dev/hdc2):

```
# pvcreate /dev/hdc2
```

Creating a Volume Group

Once you have two or more PVs, you can create a Volume Group (VG). You can substitute the name of your choice for *volume group*.

```
# vgcreate volume group /dev/hdc2 /dev/hdd2
```

You can add more room to any VG. When you create a new partition that you want to add to the VG, just extend it with the following command:

```
# vgextend volume group /dev/sda1
```

Creating a Logical Volume

However, a new VG doesn't help you unless you can mount a filesystem on it. So you need to create a logical volume (LV), for this purpose. The following command creates an LV. You can add as many chunks of disk space (also known as Physical Extents or PEs) as you need.

```
# lvcreate -l number_of_PEs volume group -n logvol
```

This creates a device named */dev/volume group/logvol*. You can format this device as if it were a regular disk partition, and then mount the directory of your choice on your new logical volume.

But this isn't useful if you don't know how much space is associated with each PE. You could use trial and error, using the **df** command to check the size of the volume after you've mounted a directory on it. Alternatively, you can use the **-L** switch to set a size in MB. For example, the following command creates an LV named *flex* of 200MB:

```
# lvcreate -L 200M volume group -n flex
```

Using a Logical Volume

But that's not the last step. You may not get full credit for your work on the exam unless the directory gets mounted on the LVM group when you reboot your Linux computer. Based on a standard RHEL 3 */etc/fstab* configuration file, you might add the following line to that file:

```
LABEL=/home/mj    /home/mj    ext3    defaults    1 2
```

Before this line can work, you'll need to set the label for this directory with the following command:

```
# e2label /dev/volumegroup/logvol /home/mj
```

I describe the `/etc/fstab` file, including the meaning of the data in each these columns, in more detail in Chapter 4.

CERTIFICATION OBJECTIVE 3.07

Automated Installation with Kickstart

Kickstart is Red Hat's solution for an automated installation of Red Hat. All of the questions asked during setup can be automatically supplied with one text file. You can easily set up nearly identical systems very quickly. Kickstart files are very useful for quick deployment and distribution of Linux systems.

INSIDE THE EXAM

Kickstart and the Red Hat exams

For the Troubleshooting and System Maintenance exam, it's possible that the exam proctor might configure your computer using a customized Kickstart configuration file. The file might be local or it might be stored on the server. Understanding Kickstart is a very useful skill that can help you install Linux on a number of different computers simultaneously. You can start the process and walk away. The options are rich and varied. The Red Hat Exam Prep guide suggests that you need to

configure "hands-free installation using Kickstart."

In addition, as of this writing, there is a bug with respect to Kickstart installations that rely on a DHCP server (Red Hat Bugzilla #90774). I therefore doubt that you'll see or have to configure a Kickstart DHCP server on either exam, at least until this bug is closed. You can check the bug status for yourself at bugzilla.redhat.com.

Kickstart Configuration Concepts

There are two methods for creating the required Kickstart configuration file:

- Use the `anaconda-ks.cfg` file from the root user's home directory, `/root`.
- Use the graphical Kickstart Configurator, accessible through the **ksconfig** command or in the GUI, click Main Menu | System Tools | Kickstart.

The first option lets you use the Kickstart template file created for your computer by Anaconda, the Red Hat Enterprise Linux installation program. The second option, the Kickstart Configurator, is discussed in detail later in this chapter.

If you're installing RHEL 3 on a number of different computers, the `anaconda-ks.cfg` file is handy. You can install RHEL 3 the way you want on one computer. You can then use the `anaconda-ks.cfg` file from that computer as a template to install RHEL on the other identical computers on your network. If the computers aren't identical, you can customize each `anaconda-ks.cfg` file as required, for elements such as a different hard disk size, host name, and so on.

Configuring a Kickstart Floppy

Once the Kickstart file is configured, the easiest way to use it is through the RHEL installation boot floppy. To do so, follow these basic steps:

1. Configure and edit the `anaconda-ks.cfg` file as desired. I'll describe this process in more detail shortly.
2. Insert and mount a 1.44MB installation floppy. If you need to create one, I describe the process in Chapter 2. You can then mount the floppy with the `mount /mnt/floppy` command.



The `mount /mnt/floppy` command is not a complete command; however, it works in RHEL because of the settings in `/etc/fstab` as described in Chapter 4.

3. Copy the Kickstart file to the floppy with the following command:


```
# cp /root/anaconda-ks.cfg /mnt/floppy/ks.cfg
```
4. Check the `ks.cfg` file on the floppy with the `vi /mnt/floppy/ks.cfg` command. If you don't see your entire `ks.cfg` file, you may not have enough room on your installation boot floppy. You can make room by deleting the `snake.msg` file from the floppy. If you don't need the description of options during the

installation process, you can also delete the `general.msg`, `options.msg`, `param.msg`, and `rescue.msg` files without affecting the functionality of the installation boot floppy.

5. You should now be ready to try out the installation boot floppy on a different computer. You'll get to try this again shortly in an exercise.

At the time of this writing, boot disks still bring up the LILO boot: prompt. Use LILO's boot options to initiate the Kickstart file. To boot, and then perform the Kickstart installation from a floppy, type

```
boot: linux ks=floppy
```

As long as you have a DHCP server on your network, you may be able to use the same Red Hat Linux installation boot floppy. Just boot each new Linux computer from that floppy, type in the **linux ks=floppy** command at the boot prompt, give the computer a moment to read the `ks.cfg` file, and insert the same floppy in the next new Linux computer.



During the troubleshooting exam, your exam proctor may set up a computer for you using a Kickstart floppy or CD. If it's a floppy, it's quite possible that you'll also need a driver floppy, or first installation CD, which your proctor should provide.

Configuring a Kickstart Server

Alternatively, you can configure a Kickstart configuration file on the DHCP/BOOTP server for your network. If you want to put the file on a DHCP/BOOTP server, open the `/etc/dhcpd.conf` configuration file on that server. Specify the Kickstart file. For example, if you've stored the appropriate Kickstart file in the `/usr/install` directory on the DHCP server, add the **filename** `"/usr/install/"` command to `dhcpd.conf`. That prompts Anaconda to look on the server in the `/usr/install` directory for a `client_ip-kickstart` file.

Alternatively, if the Kickstart file is stored on a different server, add the **next-server** *servername* option to the `dhcpd.conf` configuration file. In this case, the *servername* is the name of the computer with the Kickstart file.

Starting the Installation with a Kickstart File

No matter where you choose to put the Kickstart file, you can boot with a floppy or a specialized CD. You can put the Kickstart file on the Red Hat boot floppy, or with a

bit more work, on a Red Hat boot ISO. Just mount the floppy and copy it as the `ks.cfg` file from wherever you've saved your Kickstart configuration file.

If you're booting from a Red Hat installation floppy, enter the following command at the boot: prompt:

```
boot: linux ks=floppy
```

If you're booting from the Red Hat installation CD-ROM, you can still refer to a Kickstart configuration file on a floppy disk with the following command:

```
boot: linux ks=hd:fd0:/ks.cfg
```

This assumes the Kickstart configuration file is called `ks.cfg` and is located on the first floppy disk on your PC. Alternatively, you can refer to the Kickstart configuration file on a hard disk with this command:

```
boot: linux ks=hd:hda2:/home/mj/ks.cfg
```

This assumes the Kickstart configuration file is called `ks.cfg` and is located on the second partition of the first IDE drive in the `/home/mj` directory. The syntax of this command certainly looks strange; it's been updated for Red Hat Linux 9 and RHEL 3.

You don't need to get a Kickstart file from a DHCP server. To boot from a specific NFS or HTTP server on the network, say with an IP address of 192.168.17.18, from the `/kicks/ks.cfg` file, type one of the following commands:

```
boot: linux ks=nfs:192.168.17.18:/kicks/ks.cfg
boot: linux ks=http:192.168.17.18:/kicks/ks.cfg
```

However, even if you've specified a static IP address in `ks.cfg`, this installation looks for IP address information from a DHCP server. If not found, Anaconda continues with a standard installation, not using the Kickstart file.

Sample Kickstart File

I've based this section on the `anaconda-ks.cfg` file created when I installed my RHEL 3 Server computer. I've added a number of comments. While you're welcome to use it as a sample file, be sure to customize it for your hardware and network. This section just scratches the surface on what you can do with a Kickstart file; your version of this file may vary.

While most of the options are self-explanatory, I've interspersed my explanation of each command within the file. This file illustrates just a small portion of available

commands. For more information on each command (and options) in this file, read the RHEL 3 Configuration Guide, which is available online at www.redhat.com/docs/manuals/enterprise.

Follow these ground rules and guidelines to use when setting up a Kickstart file:

- Do *not* change the order of the options.
- You do not need to use all the options.
- If you leave out a required option, the user will be prompted for the answer.
- For upgrades, you *must* have at least the following options defined:
 - Language
 - Installation method
 - Device specification
 - Keyboard setup
 - The upgrade keyword
 - Boot loader (GRUB or LILO) configuration



If you leave out an option, you will be prompted to complete it. This is an easy way to see if your Kickstart file is properly configured. But as some Kickstart options change the partitions on your hard drive, even testing this file can delete all of the data on your computer. So make sure to have a test computer available to test your Kickstart configuration file.

Here is the code from my Kickstart file. The first line is a comment that tells me that this file was created during the installation process.

```
# Kickstart file automatically generated by anaconda.
```

The first command is simple; it starts the installation process. It defaults to the first available local media, in this case, the first RHEL installation CD or floppy.

```
install
```

However, if you want to specify an installation, you could add the source of your choice on the next line. It could be *cdrom*, or if you're installing from an NFS server, you can specify it as follows. If you have the name of the server, you can substitute it for the IP address.

```
nfs --server=192.168.30.4 --dir=/mnt/inst
```

You can also configure a connection to an FTP or HTTP server by substituting one of the commands shown here. The directories I specify are based on the FTP and HTTP installation servers created in Chapter 2:

```
url --url http://192.168.30.4/inst
or
url --url ftp://192.168.30.4/pub/inst
```

The **lang** command specifies the language to use during the installation process. It matters if the installation stops due to a missing command in this file. The **langsupport** command specifies the language to install with the operating system. The keyboard command is self-explanatory, as it specifies the keyboard to configure on this computer.

```
lang en_US.UTF-8
langsupport --default en_US.UTF-8 en_US.UTF-8
keyboard us
```

The **mouse** command specifies the pointing device associated with this computer. This particular command specifies a generic PS/2 mouse, connected to `/dev/psaux`, configured to emulate a middle mouse button.

```
mouse genericps/2 --device psaux --emulthree
```

The **xconfig** command specifies the graphical configuration for this system. The switches are nearly self-explanatory. The **--card** switch specifies the graphics card driver. The **--videoram** switch specifies the amount of RAM on the video card. The **--hsync** and **--vsync** switches specify horizontal sync and vertical refresh rates, respectively. The **--resolution** switch sets the default resolution for the monitor, with a given color **--depth** in bits. The **--defaultdesktop** switch is normally **gnome** or **kde**.

Due to typesetting limitations, we're showing the following code on *three* lines. Please note that all options for the **xconfig** command *must* be on *one* line. Line wrapping, if the options exceed the space in your editor, is acceptable.

```
xconfig --card "VESA driver (generic)" --videoram 16384
--hsync 31.5-37.9 --vsync 40-70 --resolution 800x600
--depth 24 --defaultdesktop gnome
```

The **network** command is a lot more straightforward if you have a DHCP server for your network; it would read **network --device eth0 --bootproto dhcp**. This particular

line configures static IP address information, with the specified network mask (**--netmask**), gateway address (**--gateway**), DNS servers (**--nameserver**), and computer name (**--hostname**).

Due to typesetting limitations, we're showing the following code on *three* lines. Please note that all options for the **network** command *must* be on *one* line. Line wrapping, if the options exceed the space in your editor, is acceptable.

```
network --device eth0 --bootproto static --ip 192.168.30.149
--netmask 255.255.255.0 --gateway 192.168.30.113
--nameserver 207.217.120.83,207.217.126.81 --hostname
Enterprise3
```

You need to specify the root user password during the installation process. This line specifies the password in encrypted format. You don't have to encrypt the root password in this line. If needed, you can copy this password from `/etc/shadow`.

```
rootpw --iscrypted $1$5UrLfXTk$CsCW0nQytrUuvycuLT317/
```

As for security, in RHEL 3, the firewall is either **--enabled** or **--disabled**. By default, the **authconfig** command sets up the Shadow Password Suite (**--enablesshadow**) and md5 encryption (**--enablemd5**).

```
firewall --enabled
authconfig --enablesshadow --enablemd5
```

The **timezone** command is associated with a long list of time zones. If you've installed the squirrelmail RPM, you can find a full list in `/usr/share/squirrelmail/locale/timezones.cfg`. If you want to set the hardware clock to Greenwich Mean Time, you can add the **--utc** switch.



Linux uses the UTC acronym for Greenwich Mean Time. UTC is the French acronym for this standard, which does not refer to a city in the United Kingdom.

```
timezone America/New_York
```

The default boot loader is GRUB, which is installed unless you add the **--useLilo** switch. Naturally, it should normally be installed on the Master Boot Record (MBR) of a hard drive. You can include an **--append** switch to specify commands for the kernel.

```
bootloader --location=mbr
```

By default, the commands that partition your hard drives are commented out of the `anaconda-ks.cfg` file. These options are a bit more complex, so I discuss them in more detail in the next section. If the commands work for you, don't forget to delete the comment character (`#`) to activate these commands.

```
# The following is the partition information you requested
# Note that any partitions you deleted are not expressed
# here so unless you clear all partitions first, this is
# not guaranteed to work

# clearpart --all --drives=hda
# part /boot --fstype ext3 --size=100 --ondisk=hda
# part / --fstype ext3 --size=1024 --grow --ondisk=hda
# part swap --size=256 --grow --maxsize=512 --ondisk=hda
```

The following is a list of package groups that are installed through this Kickstart configuration file. These names corresponds to the names that you can find in the `comps.xml` file described in Chapter 2.

```
%packages
@ system-tools
@ base-x
@ web-server
@ smb-server
@ printing
@ text-internet
@ mail-server
@ legacy network server
@ dns-server
@ server-cfg
@ dialup
@ sound-and-video
@ admin-tools
@ news-server
@ gnome-desktop
@ ftp-server
@ network-server
@ kernel-development
@ graphical-internet
@ compat-arch-support
```

After the package groups are installed, you can specify post installation commands (`%post`) in the Kickstart installation file.

```
%post
```

Kickstart Partitioning

The partitioning options in the Kickstart file are a bit complex. They can give you full control of the partitioning options, too. You can clear all partitions with **clearpart --all**, or just clear any Linux-type partitions with **clearpart --linux**, or just add to the end of the current partitions. You can create partitions on more than one drive, but you need to identify each device specifically.

To add Linux partitions, use the **part** command with the following syntax:

```
part mount dir --size size [--grow] [--maxsize size]
```

The *size* is in megabytes. You can use the **--grow** option to allow the partition to expand and fill all remaining disk space (or share it with any other partitions marked “grow” on the same disk). This will not expand on the fly, but rather, when all fixed-size partitions are added, these “growable” partitions will use the rest of the space.

If you specify multiple partitions with the **--grow** option, their space will be divided evenly. You can also specify a **--maxsize**, which will allow the partition to grow only to the size specified in megabytes.

Therefore, the following commands from my Kickstart configuration file (with comment characters removed) perform the following tasks:

This **clearpart** command removes all of the partitions on the first IDE hard drive.

```
clearpart --all --drives=hda
```

This first **part** command configures the `/boot` directory to the ext3 filesystem with 100MB of space on the first IDE hard drive.

```
part /boot --fstype ext3 --size=100 --ondisk=hda
```

This second **part** command configures the root directory (`/`) to the ext3 filesystem with at least 1024MB of space on the first IDE hard drive. If more room is available, this partition expands to take the available space.

```
part / --fstype ext3 --size=1024 --grow --ondisk=hda
```

This third **part** command configures a swap partition to be at least 256MB, which corresponds to the amount of RAM available on this computer. If space is available on the hard drive, it’s allowed to grow up to 512MB.

```
part swap --size=256 --grow --maxsize=512 --ondisk=hda
```

EXERCISE 3-2**Creating a Sample Kickstart File from a Running System for a Second Similar System Installation**

In this exercise, you will use the `anaconda-ks.cfg` file to duplicate the installation from one computer to another with identical hardware. This exercise simply installs all the exact same packages with the same partition configuration on the second computer. Assume that both computers use DHCP to set up their IP addresses. Add one `/home` partition of 2000MB with a 512MB swap partition. (This means you must have at least 2500MB of free space available. If you have more space, let it remain unused for now.) You want to install all the same packages as your current installation, so you do not need to make any package changes to the default Anaconda Kickstart file in the `/root` directory.

If you do not have multiple computers for this exercise, one alternative as discussed earlier is VMWare.

1. Review the `/root/anaconda-ks.cfg` file. Copy it to `ks.cfg`.
2. Create the boot disk and add this `ks.cfg` file to it:

```
dd if=/mnt/cdrom/images/bootdisk.img of=/dev/fd0
mount /dev/fd0 /mnt/floppy
cp ks.cfg /mnt/floppy
```
3. Prepare the second computer so that it has the same disk configuration, the same C: drive size if it was present, and same amount of unused and unpartitioned space as the first computer. Reboot the second computer with the Linux boot file with the Kickstart file in the floppy drive and the first binary CD in the CD-ROM.
4. At the Red Hat Installation menu boot prompt, enter the following startup command:

```
boot: linux ks=floppy
```

You should now see the system installation creating the same setup as the first system. Depending on how you installed Linux on the original computer, you may need a network connection to the same installation source or the other installation CDs.

OPTIONAL EXERCISE 3-3

Modify the Packages to Be Installed

Edit the `ks.cfg` file on the floppy and remove all the game packages you can find. Alternatively, set up an interactive installation so that you can test and observe the result on another computer.

The Red Hat exams test your skills on Red Hat Enterprise Linux. While many Unix skills are transferable to the exam, there are also many features unique to RHEL. While you can configure anything in Linux from the command line interface, Red Hat has created a number of administrative tools that you can use only in the GUI.

Based on the nature of the published Red Hat Exam Prep guide, you'll almost certainly be installing a GUI during the installation portion of the exam. When you install the GUI, you'll also have access to the tools developed by Red Hat. You can also have access to the command line interface in the GUI. You can do so with a right-click, which starts a pop-up menu where you can start the New Terminal command. Alternatively, you can click on the Main Menu button (in the lower-left corner), click System Tools | Terminal to open the same command line interface, also known as GNOME terminal.



In the KDE desktop, the default command line terminal is konsole, which is slightly different from GNOME terminal.

INSIDE THE EXAM

For Administrators from Other Unix-Style Distributions

Linux is not the only operating system developed in the style of Unix. There are a number of other Unix-style operating systems that are important in the computing world. These include but are not limited to Solaris,

HP-UX, AIX, IRIX, or even SCO UNIX. A substantial number of candidates for the RHCE learned their skills on these other operating systems.

You can then start the Red Hat tool of your choice from the command line; most of these utilities start with `redhat-config-*`. To see a full list, type **redhat-** at the command line and press the TAB key twice. Alternatively, you can start a number of these tools from the Main Menu button, under the System Settings and System Tools menus.

If you're confident in your skills editing the Linux text configuration files, don't bother with the GUI tools. It's faster to edit Linux text configuration files directly in any Unix-style operating system. However, if you get nervous on exams or are uncertain of your skills where a GUI tool is available, learn the appropriate Red Hat GUI tools. They can be a lifesaver during the exam.

The Kickstart Configurator

Now that you understand the basics of what goes into a Kickstart file, it's time to solidify your understanding through the graphical Kickstart Configurator. When you experiment with this GUI tool, you can learn more about what happens in the Kickstart configuration file.

To start the Kickstart Configurator, go into the Linux GUI. Open a command line interface shell. Start the Kickstart Configurator with the **redhat-config-kickstart** command. Alternatively, click Main Menu | System Tools | Kickstart. The following sections should look familiar, as they are closely related to the RHEL installation menus.

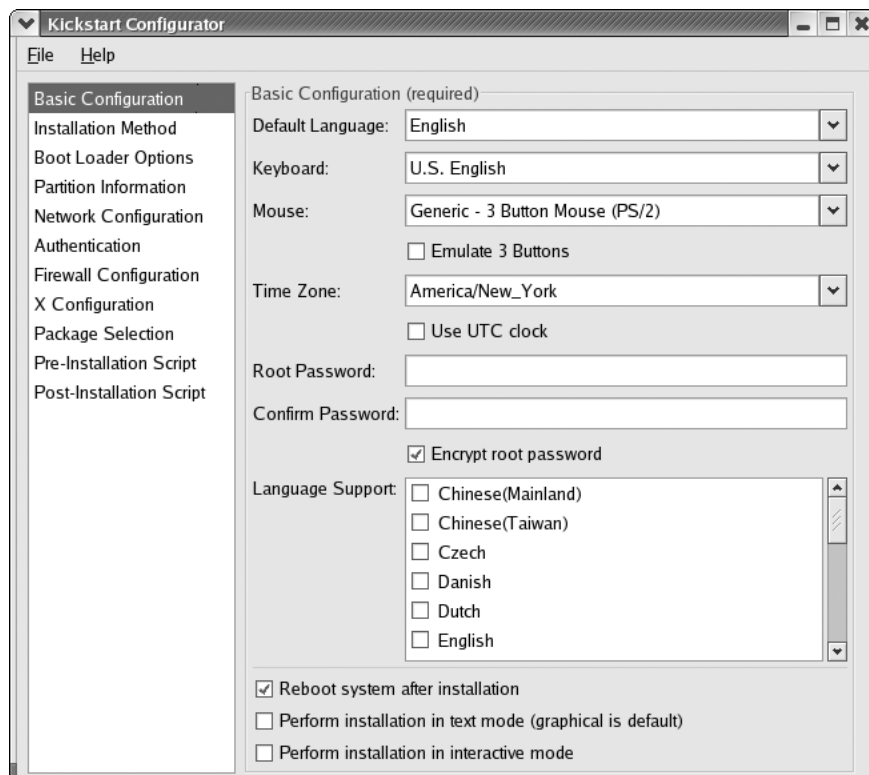
The first screen in Figure 3-4 illustrates a number of basic installation steps. If you've installed RHEL or Red Hat Linux before, all of these steps should look familiar.

As you can see, there are a number of other options in the left pane, each associated with different Kickstart commands. To learn more about Kickstart, experiment with some of these settings. Use the File | Save command to save these settings in the filename of your choice, which you can then review in a text editor. Alternatively, you can run the File | Preview command to see the effect of different settings on the Kickstart file.

The following sections provide a brief overview of each option shown in the left pane. I believe it's unlikely that you'll have to know Kickstart in this level of detail on the Red Hat exams. However, an understanding of the Kickstart Configurator can help you understand the installation process, which is important on the installation part of each exam.

FIGURE 3-4

The Kickstart Configurator



Basic Configuration

In the Basic Configuration screen, you can assign settings for the following components:

- Default Language for the installation; also set as the default language for the operating system
- Keyboard
- Mouse
- Time Zone, which supports computers where the hardware clock is set to Greenwich Mean Time (also known as UTC)
- Root Password, which can be encrypted in the Kickstart file
- Language Support to be installed with the operating system

- Reboot; adds the reboot command to the end of the Kickstart file
- Perform Installation In Text Mode supports automated installation in text mode

Installation Method

The Installation Method options are straightforward. You're either installing Linux for the first time or upgrading a previous installation. The installation method and your entries are based on the location of the installation files. For example, if you select an NFS installation method, the Kickstart Configurator prompts you for the name or IP address of the NFS Server, and the shared directory with the RHEL installation files.

You can set up your Kickstart file to install RHEL 3 from a CD, a local hard drive partition, or one of the standard network servers: NFS, HTTP, or FTP.

Boot Loader Options

The next section lists Boot Loader Options. The two boot loaders available to Linux are GRUB and LILO. Unlike for Fedora Linux, LILO is still available as an option. With GRUB, you can set a password, which can provide an additional level of security during the boot process.

Linux boot loaders are normally installed on the MBR. If you're dual-booting with Windows NT/2000/XP with GRUB, you can set up the Windows boot loader (or an alternate boot loader such as Partition Magic or System Commander) to point to GRUB on the first sector of the Linux partition with the /boot directory. Kernel parameters allow you to pass commands to the kernel through the GRUB configuration file as described earlier in this chapter.

Partition Information

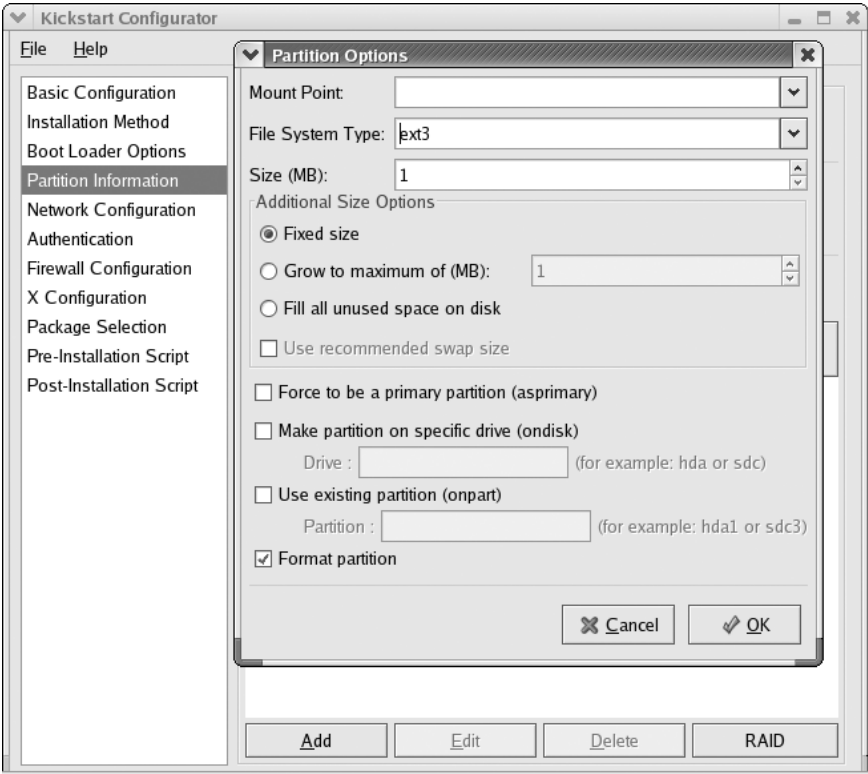
The Partition Information options determine how this installation configures the hard disks on the affected computers. While it supports the configuration of standard and RAID partitions, it does not yet support the configuration of LVM groups. The Clear Master Boot Record option allows you to wipe the MBR from an older hard disk that might have a problem there; it sets up the **zerombr yes** command in the Kickstart file.

Don't use the zerombr yes option if you want to keep an alternate on the MBR such as the NT Boot Loader.

You can remove partitions depending on whether they've been created to a Linux filesystem. If you're using a new hard drive, you'll want to initialize the disk label as well. Click the Add button to open the Partition Options menu. As you can see in Figure 3-5, this corresponds to the Anaconda Add Partition menu shown back in Chapter 2, Figure 2-13.

FIGURE 3-5

Using the Kickstart Configurator to set up partitions



Network Configuration

The Network Configuration section enables you to set up IP addressing on the network cards on a target computer. You can customize static IP addressing for a specific computer, or you can configure the use of a DHCP server. You can also make Kickstart look for a BOOTP server, which is a specially configured DHCP server on a remote network.

Authentication

The Authentication section lets you set up two forms of security for user passwords: Shadow Passwords, which encrypt user passwords in the /etc/shadow file, and MD5 encryption. This section also allows you to set up authentication information for various protocols:

- **NIS** Network Information Service for one login database on a network with Unix and Linux computers on a network.
- **LDAP** The Lightweight Directory Assistance Protocol for certain types of databases such as directories.
- **Kerberos 5** The MIT system for strong cryptography to authenticate users on a network.
- **Hesiod** Associated with Kerberos 5.
- **SMB** Samba allows configuration of your Linux computer on a Microsoft Windows-based network.
- **Name Switch Cache** Associated with NIS for looking up passwords and groups.

Firewall Configuration

The Firewall Configuration section allows you to configure a default firewall for the subject computer. Generally, you'll want a firewall only for those computers that are connected to outside networks such as the Internet.

X Configuration

The X Configuration section allows you to configure the Linux GUI. While there is a lot of debate on the superiority of GUI- or text-based administrative tools, text-based tools are more stable. For this reason (and more), many Linux administrators don't even install a GUI. However, if you're installing Linux on a series of computers, it's likely that most of the users won't be administrators.

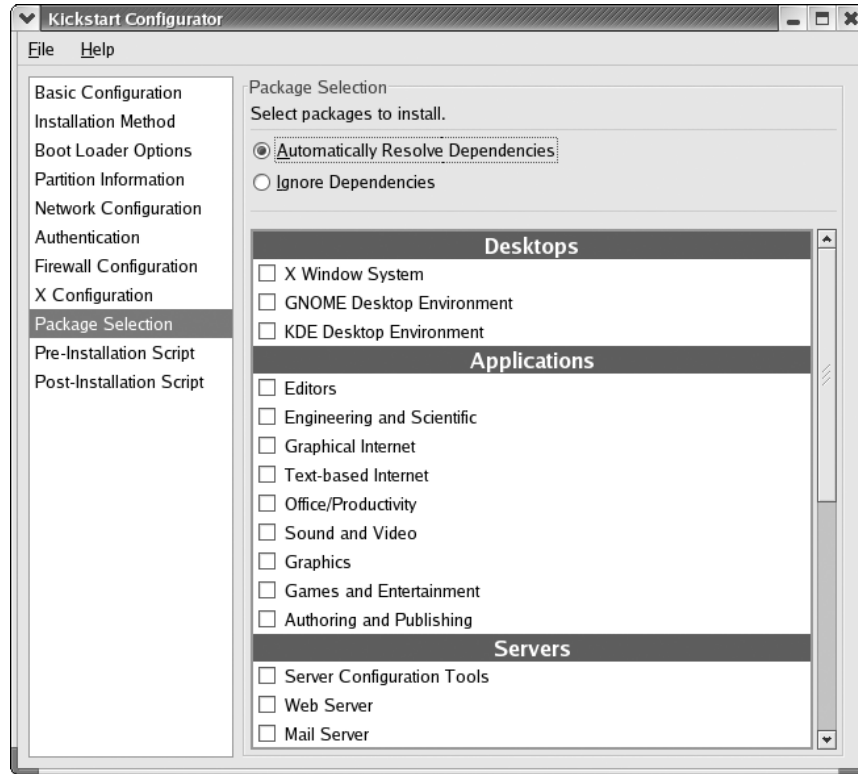
In this section, there are three tabs. Under the General tab, you can set a default color depth and resolution. Under the Video Card and Monitor tabs, you can set Linux to probe your hardware or specify the hardware from a list. If you want to do something fancy such as specifying multiple resolutions, you'll need to modify the `ks.cfg` file directly.

Package Selection

The Package Selection section shown in Figure 3-6 allows you to choose the package groups that are installed through this Kickstart file. It's a good idea to check the Automatically Resolve Dependencies option; if you miss a package or a package group, Anaconda can find the critical package and install it with your list.

FIGURE 3-6

Configuring
package groups
with Kickstart



Installation Scripts

You can add preinstallation and postinstallation scripts to the Kickstart file. Postinstallation scripts are more common, and they can help you configure other parts of a Linux operating system in a common way. For example, if you wanted to install a directory with employee benefits information, you could add a postinstallation script that adds the appropriate `cp` commands to copy files from a network server.

CERTIFICATION SUMMARY

The Linux installation is extremely flexible. You can troubleshoot the installation process with several different consoles. Some provide useful messages; one console provides a bash shell prompt where you can inspect the current detailed status of the installation.

After Linux is installed, you can find out what happened. The `/var/log/dmesg` file helps you figure out what hardware was detected. The `/root/install.log` file lists the packages that were installed.

The default boot loader is GRUB. The GRUB configuration file, `/etc/grub.conf`, is highly customizable. You can use it to boot multiple operating systems. You can add commands to the boot sequence. You can customize GRUB to test different configurations. Red Hat is moving away from LILO as an alternative boot loader.

It's most efficient to configure partitions with Disk Druid during the installation process. However, if you forget something, you can use `fdisk` to help configure the partitions, RAID arrays, and LVM groups that you need on the exam and on production computers.

You can set up RAID level devices to mirror and stripe your drives. RAID levels 1 and 0 require that you have two or more drives, while RAID 4 and 5 require three or more. You can also set up linear mode to combine multiple physical hard disks into a single volume.

You can set up LVM groups to increase your flexibility. LVM allows you to set up partitions with the sizes you need and readjust them at will. While it's best to configure LVM groups using the graphical version of Disk Druid, you can do so with appropriately configured partitions and commands.

You can automate your entire installation of Red Hat Linux with the Kickstart system. You have two ways to set up the Kickstart configuration file: from the `anaconda-ks.cfg` file based on your local installation or through the GUI Kickstart Configurator. In this way, you can automate the installation of Linux from a local source such as a CD-ROM, or through an NFS or HTTP network connection.



TWO-MINUTE DRILL

The following are some of the key points from the certification objectives in Chapter 3.

Installation Troubleshooting

- ☐ When you start in graphical mode, you're taken to the seventh virtual console, which you can access with the CTRL-ALT-F7 command. A bash shell is on the second virtual console, an installation message log is on the third, kernel messages are on the fourth, and the output of format commands on each filesystem is displayed. If you want to see other screens, press the following key sequences at any time:
 - ☐ CTRL-ALT-F1 Installation display; if you're running the installation process in text mode, this is the main display.
 - ☐ CTRL-ALT-F2 The bash shell gives you access to limited system information.
 - ☐ CTRL-ALT-F3 The Installation message log is displayed.
 - ☐ CTRL-ALT-F4 Displays all kernel messages.
 - ☐ CTRL-ALT-F5 The installation displays partition formatting.

Postinstallation Validation

- ☐ The file `/var/log/dmesg` contains boot messages duplicated from the console output, as seen each time you boot Linux.
- ☐ The `/root/install.log` file contains a list of the installed packages.
- ☐ The installation program reboots your computer when it's done. If you configured a graphical login, Linux starts in runlevel 5. Alternatively, if you configured a text login, Linux starts in runlevel 3.

Boot Loaders: GRUB and LILO

- ☐ The original Intel motherboard design provided a mechanism to start most any operating system. It would load a boot program from the Master Boot Record of the first available disk.
- ☐ The new default Red Hat boot loader is GRUB, the GRand Unified Boot Loader.
- ☐ You can edit GRUB directly during the boot process or pass commands to the kernel. For example, at the GRUB screen, if you add the word **single** to the end of the kernel argument, Linux starts in single-user mode.

- ☐ Single-user mode is the most commonly used option. This is the system maintenance mode for experienced Linux administrators.

Partitioning Utilities

- ☐ The **fdisk** utility is universally available and should be one of the first tools you learn as a Linux administrator.
- ☐ Disk Druid is the alternative to **fdisk** that is available only during the Red Hat installation process.

RAID Configuration and Data Recovery

- ☐ Software-based RAID can be set up on any number of physical partitions. Hardware-based RAID requires separate physical drives for each partition.
- ☐ You can set up software-based RAID through the Red Hat Linux installation program.
- ☐ RAID has several different levels. Only three are currently supported by Red Hat Linux: levels 0, 1, and 5. Only RAID levels 1 and 5 offer true data protection against the failure of a single hard drive.
- ☐ RAID 0 and 1 require two or more drives, while RAID 5 requires three or more drives.

Logical Volume Management: Configuration and Administration

- ☐ Logical Volume Management allows you to manage the effective size of your active partitions.
- ☐ Once properly configured through **fdisk**, you can collect LVM chunks known as Logical Extents into Logical Volumes.
- ☐ Once Logical Volumes are collected into a Volume Group, it functions as a partition that you can resize as needed.

Automated Installation with Kickstart

- ☐ Kickstart is Red Hat's solution for an automated simultaneous installation of Red Hat Enterprise Linux on several computers.
- ☐ Kickstart installations can be configured to take installation files from a CD-ROM or an NFS, FTP, or HTTP server.
- ☐ There are two ways to create a Kickstart file: from the configuration from when you installed Linux as documented in the `/root/anaconda-ks.cfg` file, or from what you can configure with the GUI Kickstart Configurator.

SELF TEST

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

Installation Troubleshooting

1. While the installation is in progress, which of the following key combinations displays kernel messages?
 - A. CTRL-ALT-F1
 - B. CTRL-ALT-F2
 - C. CTRL-ALT-F3
 - D. CTRL-ALT-F4

Postinstallation Validation

2. Which of the following commands displays the messages that scrolled on your screen when you last booted Linux?
 - A. `cat /var/messages`
 - B. `cat /var/log/dmesg`
 - C. `cat /var/log/messages`
 - D. `/sbin/messages`

Boot Loaders: GRUB and LILO

3. In the GRand Unified Bootloader, what should you do if you wanted to start Linux in single-user mode for a quick repair?
 - A. Enter **linux single** at the boot prompt.
 - B. Run the **a** command and then add **single** at the end of the line that appears.
 - C. Get a LILO boot disk and then run **linux sum** at the boot prompt.
 - D. Boot Linux and edit the `/etc/grub.conf` file to automatically start Linux in single-user mode, then reboot.

4. When you see the **root=/dev/sda7** command in either GRUB or LILO, what does it mean?
- A. The root directory, /, is located on the third logical partition of the first SCSI hard disk.
 - B. The /root directory is located on the third logical partition of the first SCSI hard disk.
 - C. If it's GRUB, this represents the root directory, /. If it's LILO, this represents the /root directory. In either case, it is located on the third logical partition of the first SCSI hard disk.
 - D. It is an error in either boot loader.

Partitioning Utilities

5. You are told to check the Web server drive table after installation. There are eight partitions. Your MIS manager asks how that can be. Her Microsoft Windows computer can include only one primary and one extended partition. How many IDE primary partitions can you configure using Red Hat Enterprise Linux?
- A. 4 primary, 1 of which is an extended partition, making for 16 total partitions
 - B. 3 primary, 2 of which are extended, making for a total of 12 partitions
 - C. 12 primary, with 1 an extended partition
 - D. 16 extended partitions
6. Once you're in the **fdisk** utility, which command gives you the current partition table?
- A. m
 - B. p
 - C. x
 - D. w

RAID Configuration and Data Recovery

7. You have a mirrored RAID system with three drives. The first two are mirrored and the third is supposed to be a spare. When you look at the **/etc/raidtab** file, you see that it says **spare-disks 0**. What does this entry tell you?
- A. The raidtab entry is set up correctly.
 - B. Spare-disks support is turned off. To turn it on, change spare-disks to 1.
 - C. Currently no spare disks are loaded.
 - D. The mirror failed and had to use the spare disk.

8. Which of the following versions of RAID does not provide data redundancy?
- A. RAID 0
 - B. RAID 1
 - C. RAID 4
 - D. RAID 5

Logical Volume Management: Configuration and Administration

9. Which of the following can be an LVM Physical Volume?
- A. A Volume Group
 - B. A Logical Volume
 - C. A Logical Extent
 - D. A partition

Automated Installation with Kickstart

10. You're using the Kickstart Configurator to create a ks.cfg file for several computers. Which of the following lines allows for a **growable** /var partition with no limit on growth save the capacity of the hard disk?
- A. `part /var --size 1000 --grow yes --maxsize no`
 - B. `part /var --size 1000 --maxsize 0`
 - C. `part /var --size 1000 --grow --maxsize 100000`
 - D. `part /var --size 1000 --grow`

LAB QUESTIONS

Lab 1

You just got hold of ten new PCs for the Human Resources department from a name brand PC manufacturer and you want to install Red Hat Enterprise Linux on each computer. You want to install Linux on all of them with an optimized set of packages, and you want to do it quickly.

Each of these computers has a standard 3Com network card that you know Linux has support for because you ordered the computers that way. They also each have one big 10GB disk that already contains Windows 98. You do not have time to install Red Hat Enterprise Linux 3 Workstation on each computer manually. What should you do?

Assume you need to configure each computer with static IP addresses.

Lab 2

This lab requires that you have one of the network installation servers described in Chapter 2 and a test computer where you're willing to reinstall RHEL 3. **This exercise erases any operating system that you currently have on a computer! Do not run this lab unless you can afford to lose all files and are willing to reinstall RHEL 3 on this computer.** You're going to move one of the critical files out of the installation source directory.

You could perform this lab on a new VMWare machine. Alternatively, you could run this on a computer with a standard configuration, where you can set up an automated reinstallation with a Kickstart configuration file. You've probably just seen this process in the answer to Lab 1.

1. Log into the remote network server with your RHEL 3 installation files.
2. Look at your comps.xml configuration file. It's stored in the RedHat/base/ subdirectory of the installation source. Open it in a text editor or a text reader such as the **less** command.
3. Browse through the list of packages in XML format. Pay attention to the packages in the Base and Core package groups.
4. Pick a RPM package from one of these groups. If you can't think of one, I've selected the filesystem-2.2.1-3.i386.rpm file.
5. Move that file to the home directory to some other directory on that network installation server. Make sure to use the **mv** command; the **cp** command leaves a copy on the installation source.

If you have valuable data on this computer, STOP HERE! The following steps will delete the information at least on the Linux partitions on your computer—and possibly on the entire computer.

6. Return to your test computer, and proceed through the RHEL 3 installation process.
7. At some point, the RHEL 3 installation process will miss the RPM that you moved and stop with a message to that effect.
8. Press CTRL-ALT-F2. You should see a bash prompt. Look in the /mnt/sysimage directory. What is the directory tree in this directory?
9. Navigate to the /mnt/sysimage/root directory. Look at the files in this directory. Which file tells you about the RPMs that have been installed?

- 10.** Run the **chroot /mnt/sysimage** command. If it doesn't work, your installation stopped before RHEL 3 had a chance to install the bash shell. With the RPM cited, this is the case.
- 11.** Press CTRL-ALT-F3. Take a look at the messages. You should see the partitions you specified, mounted in some way. Where are they mounted?
- 12.** Press CTRL-ALT-F4. Take a look at the messages. You should see various kernel messages. What types of messages do you see?
- 13.** Press CTRL-ALT-F5. Take a look at the messages. You should see various partition messages. What commands do you see in this screen?
- 14.** On the network server, restore the RPM that you moved.
- 15.** Depending on whether you're installing in text or graphical mode, return to your installation screen with the CTRL-ALT-F1 or CTRL-ALT-F7 command.
- 16.** Proceed with the installation of RHEL 3.

Lab 3

What would you do differently from Lab 2, assuming that you have configured a DHCP server for your network?

Lab 4

You want to set up a server for these new PCs. You want to configure that computer with the Red Hat Enterprise Linux 3 Server operating system. That server includes five separate SCSI physical hard drives of 10GB each. You'll install RHEL 3 on the first hard drive. How would you configure a RAID 5 array on the remaining hard drives? Assume you'll need 5GB from each of the 10GB hard drives.

Lab 5

On that server, you want to configure LVM in the remaining space. How would you set up the Physical Volumes (PVs)? How do you create a Volume Group (VG) from the PVs? Once you've created a VG, how would you set up a Logical Volume? Where would you mount a directory such as /home/flexuser?

SELF TEST ANSWERS

Installation Troubleshooting

1. ☒ **D.** The CTRL-ALT-F4 key combination accesses the screen with kernel messages during the Red Hat Linux installation process.
☒ **A, B, and C** are all incorrect, as they access the default text installation display, the bash shell, and the installation message log, respectively.

Postinstallation Validation

2. ☒ **B.** The `/var/log/dmesg` file contains boot messages.
☒ **A, C, and D** are all incorrect. The `/var/log/messages` file is significant as the location for service startup and shutdown messages.

Boot Loaders: GRUB and LILO

3. ☒ **B.** The `a` command in GRUB accesses the information passed to the kernel. The **single** command makes the kernel access single-user mode.
☒ **A, C, and D** are not correct. There is no boot prompt in GRUB. The **linux sum** command has no meaning at the LILO boot: prompt. While answer **D** could work, it is at best inefficient.
4. ☒ **A.** In both LILO and GRUB, the `root=/dev/sda7` command indicates that the root directory, `/`, is located on the third logical partition of the first SCSI hard disk.
☒ **B, C, and D** are not correct. In either boot loader, `root` represents the top-level `/` directory, not the home directory of the root user (`/root`). It is not an error in either boot loader.

Partitioning Utilities

5. ☒ **A.** 4 total, 3 that are called primary and 1 that is a primary partition that is configured as an extended partition. Extended partitions can contain up to 12 logical partitions. $12 + 4 = 16$.
☒ **B, C, and D** are all incorrect configurations. The limitation described in the question applies to the Microsoft version of the FDISK.EXE utility.
6. ☒ **B.** The `p` command “prints” the current partition table to the screen.
☒ **A, C, and D** are all incorrect. The `m` command lists other available commands within `fdisk`. The `x` command exits from `fdisk`. The `w` command writes your changes to the partition table.

RAID Configuration and Data Recovery

7. ☒ A. The first spare-disks variable is number 0. Since we have one spare disk, the raidtab entry is set up correctly.
☒ B, C, and D are incorrect. There is no way to know from this file the status of the RAID disks, nor the current loading, as this is a configuration file, not a log of the running service.
8. ☒ A and B. RAID 0 does not include any parity data or any other way to recover data from a failed disk. RAID 1 includes an identical copy of data on two different disks. No parity data is required in this version of RAID.
☒ C and D are incorrect. Both RAID 4 and RAID 5 use parity bits. If one drive on either of these systems is lost, the data from that drive can be rebuilt from the parity bits.

Logical Volume Management: Configuration and Administration

9. ☒ D. An LVM Physical Volume is the physical partition or hard disk where you've applied the `pvccreate` command to set up Physical Extents.
☒ A, B, and C are incorrect. A Logical Extent is a chunk of disk space that maps to a Physical Extent. A Logical Volume is a group of Logical Extents. A Volume Group is composed of a group of Logical Volumes.

Automated Installation with Kickstart

10. ☒ D. The `--grow` switch alone is enough to accommodate a growable partition. If there is no `--maxsize` switch, that implies that you do not want to limit the size of this partition.
☒ A, B, and C are incorrect. Neither the `--grow` nor the `--maxsize` switch is associated with a yes or a no, so A is incorrect. A `--maxsize` switch alone without `--grow` is not meaningful, so B is incorrect. Answer C includes a limit on the size of the `/var` partition.

LAB ANSWERS

Lab 1

The solution is simple. The details are long. Each of the following steps are high level; for the nitty-gritty details of what you need to do, refer back to discussions in this chapter (as well as Chapter 2). If you're using these instructions to restore a RHEL 3 installation from Part 1, minimize the changes that you need to make to this file.

1. Set up the RHEL 3 Workstation installation files on a network source.

2. Install RHEL 3 Workstation over the network on one computer.
3. Edit the `/root/anaconda-ks.cfg` file as needed.
4. Modify this file with the **clearpart --all** command to wipe out all the old partitions, including the Windows 98 partitions. (This assumes you've backed up any needed data files first.)
5. Activate the command lines related to creating partitions. They are commented out by default in the `anaconda-ks.cfg` file.
6. In the Kickstart configuration file, make sure that the line after the **install** command points through the network to the source for the RHEL 3 Workstation installation files. If you're connecting through NFS, this starts with the **nfs** command; if you're connecting through FTP or HTTP, this starts with the **url** command.
7. Use this Kickstart file as a configuration file template for installing Linux on each of your computers.
8. Create a different file for each computer; you'll need to set a different IP address on the same subnet for each file.
9. Set up a different installation floppy for each computer. Each installation floppy will include the customized Kickstart file. You'll probably also need one or more driver floppies. Alternatively, you can set this up based on the `boot.iso` file. All of these disks can be created from images on your first RHEL 3 installation CD. This process is described in Chapter 2.
10. You can then set up a network installation on each computer. You'll probably also need the RHEL 3 network driver floppy described in Chapter 2. You can use the same driver floppy on each computer.
11. Now insert the appropriate installation floppy in each of your workstations. When prompted, add the driver floppy to load the appropriate network drivers.
12. Your installation should now proceed automatically on each workstation.
13. Save any installation boot and driver disks or CDs that you created in this lab for Part 2.

Lab 2

This lab should be fairly self-explanatory. It provides a specific problem that stops the installation of RHEL 3. It gives you an opportunity to examine the other consoles available during the installation process. Remember, you should not run this lab on any computer where you have valuable data. I've added some commentary which may clarify the steps you need to take in this lab.

The `comps.xml` file is located in the installation source. If you're using the same NFS installation source that I described in Chapter 2, it's on the `/mnt/inst` directory. Thus, the `comps.xml` file is located in the `/mnt/inst/RedHat/base` directory. If you already have an installed copy of RHEL 3 on an Intel 32-bit computer, you can find another copy of this file in the `/usr/share/comps/i386` directory.

The format of the `comps.xml` file might be confusing to anyone not familiar with the XML language. However, it's not as complex as it seems. For example, you can find the filesystem RPM in this file in the following format:

```
<packagereq type="mandatory">filesystem</packagereq>
```

When you browse higher on this file, you'll see that the filesystem RPM is part of the Core package group.

Once the installation stops at the problem you created, you can browse through the different consoles. The second console includes the bash prompt, even though bash commands aren't installed yet. You should also recognize the standard root directory (`/`) tree in the `/mnt/sysimage` directory.

In the third virtual console, you'll see messages that partitions such as `/tmp/hda1` or `/tmp/sda1` are mounted. For example, a partition mounted on the root directory is mounted on the `/mnt/sysimage` directory during the installation process. The partitions that you've created through Disk Druid are mounted on a temporary basis until installation is complete.

The kernel messages that you see in the fourth virtual console show hardware and filesystem messages. The fifth virtual console lists the messages you see when the filesystem is formatted.

Lab 3

The basic steps are the same. However, if the ten computers are connected to a network with a DHCP server, your task is easier. Modify the aforementioned Kickstart template file. Assuming it's the first Ethernet adapter on each computer, the key command will be **`network --device eth0 --bootproto dhcp`**.

Since you're getting network information from a DHCP server, you can use the same installation floppy for each of the workstations. You'll still need the driver floppy. Now insert the appropriate installation floppy in each of your workstations. When prompted, add the driver floppy to load the appropriate network drivers. You should be on your way!

Alternatively, you can create a customized boot CD based on the `boot.iso` file on the first RHEL 3 installation CD. Add the `ks.cfg` file when you burn your CD from `boot.iso`.

Lab 4

Now you're configuring a RAID 5 array for your workstations on a RHEL 3 server. With the given configuration, RHEL 3 will be installed on the first hard disk. The remaining four hard disks are available for other purposes. You'll use 5GB of each of those SCSI hard disks to create RAID 5 arrays. It's easiest to do so during the installation process. However, you can still do so after RHEL 3 is installed.

As they are SCSI hard disks, Linux represents them as `/dev/sdb`, `/dev/sdc`, `/dev/sdd`, and `/dev/sde`. You can divide the 5GB on each of these hard disks into near equal parts (RAID does not require identical components in its arrays.). If you created 5GB partitions on each hard disk, you could set up a RAID 5 array with the first 3 partitions: `/dev/sdb1`, `/dev/sdc1`, and `/dev/sdd1`. You could then set up the partition on the final drive, `/dev/sde1`, as a spare. (While you could set up all four partitions as a RAID 5 array, it's useful to have a spare. If one drive fails, the RAID 5 software automatically begins rebuilding data on the spare drive.)

You can create a RAID array after RHEL 3 is installed, using the following basic steps. For detailed steps, refer back to the chapter.

1. Use the **fdisk** utility to create and size partitions from available space on the SCSI hard disks.
2. With the **t** switch, set these partitions as usable by RAID by setting them to the Linux RAID autodetect filesystem.
3. Format each partition, set up the array in `/etc/raidtab`, and use the **mkraid -R** command to create the RAID device.
4. Format the RAID device to the Linux `ext3` filesystem.
5. Save any data from the directory that you want to mount on the RAID array.
6. Mount the directory of your choice to that filesystem, and finally, implement the change in `/etc/fstab`.

Lab 5

You can create LVM groups after RHEL 3 is installed, using the following basic steps. For detailed steps, refer back to the chapter.

1. Use **fdisk** to create partitions from the remaining available space on each of the last four SCSI drives. Make sure to set each partition to the Linux LVM filesystem.
2. Initialize each of these partitions with the **pvcreate** command.

3. Set up all of these partitions together as a Volume Group (VG) with the **vgcreate** command.
4. Now set up a Logical Volume from that VG with the **lvcreate** command. You can set it to a fixed number of Physical Extents or a fixed amount of space.
5. Take the directory of your choice. Back up any data currently on that directory.
6. Mount the directory on the LV. Copy the data from backup.
7. Finally, set up the mount in the `/etc/fstab` configuration file.