



11

Operational Administration Recovery and Security

CERTIFICATION OBJECTIVES

- 11.01 Services and Special Users
- 11.02 Red Hat User Private Group Scheme
- 11.03 anacron and tmpwatch
- 11.04 The **linux rescue** Environment

- 11.05 The Secure Shell Package
- 11.06 Managing Logical Volumes
- ✓ Two-Minute Drill

Q&A Self Test

The themes of this chapter are *security* and *recovery*. These are two critical concepts for the Red Hat exams. This continues the discussion of system security started in the last chapter, with a look at secure ways to run certain network services and a Red Hat Enterprise Linux–specific way of configuring file security.

This chapter includes a description of setting up security with groups. You can set up special groups on an RHEL 3 computer. You can set up a common directory for the users in these groups.

INSIDE THE EXAM

Installation and Configuration

Part of this chapter can help you with the skills you need on the Red Hat Installation and Configuration exams. One of the basic foundations of Linux is that most services have their own special accounts. While the User Private Group scheme isn't explicitly

mentioned in the Red Hat Exam Prep guide, it is part of the prep courses (RH133, RH300) for both RHCT and RHCE exams. The RHCT prep course does also cover **anacron** and **tmpwatch**, so it's possible that you'll see those services on both exams.

Troubleshooting and System Maintenance

When you read this chapter, focus on the requirements associated with the Red Hat Troubleshooting and System Maintenance exams. Perhaps the most important Linux administrative skill is knowing how to rescue a damaged system. RHCTs need to know how to boot from the GRUB menu into different runlevels. RHCEs also need to know how to rescue a system using the first RHEL 3 installation CD. If you're planning to take

the Red Hat exams, learn how to diagnose boot problems. Create a few more of your own, and learn the symptoms. It may help you avoid frustration and solve problems quickly during the RHCE Troubleshooting and System Maintenance exam. Finally, as Logical Volume Management is becoming more important on Red Hat systems, you need to know how to add, remove, and resize logical volumes on the RHCE exam.

Next, you'll learn about two different administrative processes. The **anacron** service can help you make sure that cron jobs that are missed when your RHEL 3 computer is powered down are run. The **tmpwatch** script is a standard cron job that can help you maintain your temporary directories.

You'll also examine one of the most fundamental fears of any systems administrator: what to do when a system will not boot. When the inevitable happens, knowing the right things to look for and having some tricks up your sleeve may possibly help you avoid a potential nightmare and a major loss of service for your users. Understanding these tools is fundamental to getting through troubleshooting scenarios on the Red Hat exams.

Finally, you'll look at how to manage and extend the Logical Volumes that you created in Chapters 2 and 3. With this skill in hand, you can expand full volumes such as the `/home` directory, as needed.

CERTIFICATION OBJECTIVE 11.01

Services and Special Users

Programs that run on Linux are all processes. When Red Hat Enterprise Linux starts on your computer, it first starts a special process known as `init`. The `init` process then starts other basic processes required for a working Linux system, including the shell, the basic user consoles, startup daemons, and more. Because it needs the authority, `init` runs as `root`; in other words, it runs with the privileges of the root user.

Interestingly enough, most other services, especially network daemons, do not run under the root user ID. This is one important way Linux protects your network security.

Suppose you have configured a system to start several network services running under the root user ID. Even if you loaded the latest security patches, the risk is still high. If a cracker stumbles upon your system and is able to break in, he or she can quickly get root access through the service daemon.

To circumvent problems like this, RHEL 3 normally configures services to run under their own user accounts. If a cracker does succeed in breaking into one daemon, the damage is limited because the service is running as a normal, unprivileged user. Alternatively, some services can be run through the `nobody` account. Figure 11-1 shows a typical `/etc/passwd` file. Notice that most common network services have their own user accounts.

4 Chapter 11: Operational Administration Recovery and Security

FIGURE 11-1

/etc/passwd

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

EXERCISE 11-1

Verifying that Services Have Their Own Accounts

In this exercise, you will verify that certain system and network services run with their own accounts. You should try this exercise on a system that is configured to offer various network services. At a shell prompt, issue the following command:

```
# ps aux --headers | less
```

What account is the Web server service (httpd) running under? What account is the xfs service running under?

CERTIFICATION OBJECTIVE 11.02

Red Hat User Private Group Scheme

One major difference between Red Hat Enterprise Linux 3 and non-Red Hat Linux or Unix distributions is how new users are assigned to groups. A Linux group allows its

members to share files. Unfortunately, that also means everyone in the same primary group has access to the home directories of all other group members. Users may not always want to share the files in their home directories with others. For example, if you're setting up an ISP, your users pay for their privacy.

On the other hand, Red Hat Enterprise Linux normally gives each user a unique user ID and group ID in `/etc/passwd`. This is known as the *user private group* scheme. As users get exclusive access to their own groups, they don't have to worry about other users reading the files in their home directories.

Standard and Red Hat Groups

Traditionally, users are assigned to one or more groups such as users in `/etc/group`. For example, you might configure `acctgrp` for the accounting department and `infosys` for the information systems department in your company.

If you have access to one of these other versions of Unix or Linux, check the third and fourth fields in `/etc/passwd`. Many users will have the same fourth field, which represents their *primary* group. Then, when you create a new user, each account receives a unique user ID but shares the same group ID with other users in the `acct` group. Users can still belong to other groups as well.

In RHEL 3, each user gets their own special private group by default. As you probably noticed in Chapter 4, user IDs and group IDs by default start at 500, match, and proceed in ascending order.

By default in RHEL 3, all regular users have a `umask` of 0002. If you are coming from a traditional Unix environment, you may be concerned. With the traditional `user/group` scheme, any member of that user's primary group will automatically have write access to any file that the user creates in his home directory.

This is the advantage behind the *user private group* scheme. Since every user account is the only member in its own private group, having the `umask` set to 002 does not affect file security. This provides advantages for systems such as Internet Service Providers, where you don't want users to have access to each other's files.

Shared Directories

Most people work in groups. They may share files. You can give a group of users access to a specific user's home directory or you can set up a shared directory for a group.

When you configure a shared directory, you can set up a group owner, and then add the users you need to that group, through the `/etc/group` configuration file. When you set the group ID bit (SGID) on this directory, any file created in this directory inherits

6 Chapter 11: Operational Administration Recovery and Security

the group ID. Assuming you have set appropriate permissions, all members of this group can then access files in that the directory.

There are several basic steps required to create a useful shared directory. For example, assume you want to set up a shared directory, `/home/accshared`, for the accountants in your organization. To set this up, you'll want to take the following steps:

1. Create the shared directory:

```
# mkdir /home/accshared
```

2. Create a group with the users in your accounting department. Give it a group ID that doesn't interfere with existing group or user IDs. One way to do this is to add a line such as the following to your `/etc/group` file. You could also create this kind of group using the Red Hat User Manager described in Chapter 4. Note that the name of this new group is `accgrp`.

```
accgrp:x:5000:stewardh,jamiec,davidw,debl,callend,vanessar
```

3. Set up appropriate ownership for the new shared directory. The following command prevents any specific user from taking control of the directory, and assigns group ownership to `accgrp`.

```
# chmod 2770 /home/accshared
```

Any user who is a member of the `accgrp` group can now create files in the `/home/accshared` directory. Any files generated will then be associated with the `accgrp` group ID, and all users listed as members of `accgrp` in the `/etc/group` file will have read, write, and execute access to the `/home/accshared` directory.

What makes this possible are the permissions that you've assigned to the `/home/accshared` directory: `2770`. Let's break this down into its component parts.

The first digit (2) is the *set group ID bit*, also known as the *SGID bit*. When you set the SGID bit for a directory, any files created in that directory automatically have their group ownership set to be that of the group owner of the directory. There are two ways to set the SGID bit for the `/home/accshared` directory:

```
chmod g+s /home/accshared
```

or alternatively:

```
chmod 2770 /home/accshared
```

Setting the SGID bit solves the problem of making sure all files created in a shared directory belong to the correct group—as long as the **umask** is set properly.

The remaining digits are basic knowledge for any experienced Linux or Unix user. The 770 sets read, write, and execute permissions for the user and group that owns the directory. But since the user owner is nobody, the group owner is what counts. In other words, members of the `accgrp` group gain read, write, and execute permissions to files created in this directory.

Otherwise, users who are members of `accgrp` and belong to another primary group would have to remember to use the `chgrp` command on every file they put in `/home/accshared`. While clumsy, that command allows other users in that group to access the file.

EXERCISE 11-2

Controlling Group Ownership with the SGID Bit

In this exercise, you will create new files in a directory where the SGID bit is set.

1. Add users called `test1`, `test2`, and `test3`. Check the `/etc/passwd` and `/etc/group` files to verify that each user's private group was created:

```
# useradd test1; passwd test1
# useradd test2; passwd test2
# useradd test3; passwd test3
```

2. Edit the `/etc/group` file and add a group called `tg1`. Make the `test1` and `test2` accounts a member of this group. You could add the following line to `/etc/group` directly or use the Red Hat User Manager:

```
tg1::9999:test1,test2
```

Before you proceed, make sure the group ID you assign to group `tg1` (in this case, 9999) is not already in use.

3. Create a shared director for the `tg1` group:

```
# mkdir /home/testshared
```

4. Change the user and group ownership of the shared directory:

```
# chown nobody.tg1 /home/testshared
```

5. Log in as `test1` and `test2` separately. Change the directory to the `testshared` directory and try to create a file. Two ways to do so are with the following commands. What happens?

```
$ date >>test.txt
$ touch abcd
```

6. Now as the root user, set group write permissions on the testshared directory.

```
# chmod 770 /home/testshared
```

7. Log in again as user test1, and then try to create a file in the new directory.
So far, so good.

```
$ cd /home/testshared  
$ date >> test.txt  
$ ls -l test.txt
```

8. Now check the ownership on the new file. Do you think other users in the tg1 group can access this file?

```
$ ls -l
```

9. From the root account, set the SGID bit on the directory:

```
# chmod g+s /home/testshared
```

10. Switch back to the test1 account and create another file. Check the ownership on this file. Do you think that user test2 can now access this file?

```
$ date >> testb.txt  
$ ls -l
```

11. Now log in as the test2 account. Go into the /home/testshared directory, create a different file, and use **ls -l** to check permissions and ownership again.

12. Switch to the test3 account and check whether you can or cannot create files in this directory, and whether you can or cannot view the files in this directory.

CERTIFICATION OBJECTIVE 11.03

anacron and tmpwatch

There are two cron jobs which are critical for basic system administration. The anacron RPM package makes sure that cron jobs are run on computers that may be powered down during the night. The tmpwatch RPM configures cron jobs which removes files from various /tmp directories that have not been accessed in a specified number of hours.

exam

watch

*As of this writing, **anacron** is listed as part of the RHCT prep course (RH133). It is therefore an implicit part of the requirements for both Red Hat exams. However, the **anacron** RPM package is not included as part of RHEL 3. There's an open request to add it to RHEL 3; see bug*

*103691 at bugzilla.redhat.com for more information. You'll have to make your own judgment on whether to study **anacron** for your exam. Watch for changes in the Red Hat Exam Prep guide and RH133 syllabus. In my opinion, **anacron** is still an excellent tool for Linux administrators.*

anacron

Some of the cron jobs described in Chapter 5 are essential to the smooth operation of a RHEL 3 computer. And if you add backup jobs to the cron directories, you want to make sure the backups are always executed. For example, if there is a power failure when you schedule a backup job, you may never know that the backup wasn't created—at least until you need the backup that you don't have.

Since the **anacron** RPM is not included with RHEL 3, you'll need to get the RPM from a different source, such as the Red Hat Linux 9 or Fedora Linux installation files. If it is included on your Red Hat exam, you should be able to install it from some readily available source, as you won't have access to the Internet during your exam. Therefore, all I'll say for the purposes of this book is that you can download the **anacron** RPM from a source such as ftp.redhat.com or a mirror site such as those specified at www.redhat.com/download/mirror.html.

Once installed, the **anacron** command checks the jobs of your choice, based on the settings in the `/etc/anacrontab` configuration file. The default version of this file is straightforward. The first two lines set the shell and `PATH` for commands in the remainder of the file:

```
SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
```

The following three commands run the cron scripts in the noted directories:

```
1    65    cron.daily      run-parts /etc/cron.daily
7    70    cron.weekly  run-parts /etc/cron.weekly
30   75    cron.monthly  run-parts /etc/cron.monthly
```

These commands are in the following format:

```
period delay job-identifier command
```

In this format, **anacron** checks the jobs in the *job-identifier* directory every *period* (in days). If one or more jobs in this directory have not been run, the *command* is executed after the given *delay* (in minutes).

In other words, if you've installed the anacron RPM, RHEL 3 runs the **anacron** service every time Linux boots on your computer. Take the last command in this list. The **anacron** service checks the commands in the `/etc/cron.monthly` directory. If they have not been run for 30 days, **anacron** waits 75 minutes, and then executes the **run-parts /etc/cron.monthly** command, which executes the scripts in the `/etc/cron.monthly` directory.

tmpwatch

As its name implies, the **tmpwatch** script is normally run on directories such as `/tmp` and `/var/tmp`. The **tmpwatch** script works recursively, so if you specify the top-level directory in a tree, **tmpwatch** will search through the entire directory tree looking for files to remove.

The following **tmpwatch** command deletes all files in the `/tmp` directory that haven't been accessed in the past week (168 hours = 7 days × 24 hours/day):

```
# tmpwatch 168 /tmp
```

While you can run the **tmpwatch** command from the command line, it is often more practical to set it up to be run by the cron daemon on a regular basis. It's run as a script on a daily basis, in the `/etc/cron.daily` directory. By default, RHEL 3 configures the first two commands in **tmpwatch** to delete files in `/tmp` and `/var/tmp` every 10 and 30 days, respectively:

```
/usr/sbin/tmpwatch 240 /tmp
/usr/sbin/tmpwatch 720 /var/tmp
```

The final command deletes formatted manuals, which are cached and stored in various `/var/cache` directories, every 30 days.

```
for d in /var/{cache/man,catman}/{cat?,X11R6/cat?,local/cat?}; do
    if [ -d "$d" ]; then
        /usr/sbin/tmpwatch -f 720 $d
    fi
done
```

EXERCISE 11-3**Clearing an Imaginary /db Directory**

In a bizarre twist of fate, a runaway process has just created 200 temporary files in /db that it did not remove. You could remove them manually, or you can let **tmpwatch** delete all the files that are more than one hour old. Note that this removes all files over an hour old, not just these imaginary files, so this should *not* be done on a production server directory. If you already have a directory named /db, do *not* use it. If necessary, create a separate directory just for the purpose of this exercise.

```
# cp /etc/* /db # copying a large number of files to /db
# ls /db | wc -w # how many files need to be removed
```

Wait at least an hour.

```
# tmpwatch 1 /db
# ls /db # files should be gone
```

Alternatively, you can copy or extract files from an older backup or tar archive to the /db directory, and have **tmpwatch** delete them. If they are all more than seven days old, then use 168 as the waiting period. You could even try various times to see which files are deleted.

CERTIFICATION OBJECTIVE 11.04**The linux rescue Environment**

At some point in your career as a Red Hat Enterprise Linux administrator, maybe even on the Red Hat exams, you're going to be faced with a system that will not boot. It will be up to you to determine the cause of the problem and implement a fix. Sometimes, the problem may be due to hardware failure: the system in question has a bad power supply or has experienced a hard disk crash.

Quite often, however, the failure of a system to boot can be traced back to the actions of a user: you, the system administrator! When you are editing certain system configuration files, typographical errors can render your system unbootable.

Any time you plan to make any substantial modifications to your system or change key configuration files, back them up first. Then, after making changes, you should actually reboot your system rather than assume that it will boot up the next time you need a reboot. It's much better to encounter problems while you can still remember exactly which changes you made. It is even better if you can go back to a working configuration file.

exam

Watch

Know every detail that you can about the *linux rescue* environment for the RHCE exam.

To prepare for boot failures, you should make sure you have a valid boot floppy for your system. But boot floppies can be lost. So it's also important to know how to use the Red Hat installation boot disk or CD to get to the Linux Rescue environment, first discussed in Chapter 2. Refer to that chapter for more information on creating an installation boot disk.

While most of this section applies to the RHCE exam, the RHCT part of the Exam Prep guide suggests that you need to know how to boot Linux into different runlevels, which you can learn about near the end of this part of this chapter.

A mkbootdisk Boot Floppy

When you installed RHEL 3, the last screen may have asked whether you wanted a boot disk. If you answered No to this prompt, you can still create a valid boot floppy for your computer using the **mkbootdisk** command. This command reads the selected kernel images in /boot and the default boot loader, GRUB or LILO, to create a LILO-style boot image on a floppy disk. For example, if the current version of the RHEL 3 kernel is 2.4.21-4.EL, use this command:

```
# mkbootdisk 2.4.21-4.EL
```

You may be able to fix a few problems with this boot disk, such as an accidentally deleted master boot record, by booting from your boot disk.

The **mkbootdisk** command in Red Hat Enterprise Linux 3 actually doesn't work quite as intended. For example, on my computer, this creates a `syslinux.cfg` file on the floppy, with the following two lines at the end of the file:

```
append initrd=initrd.img ro
ro root=/dev/hda2
```

This actually causes a kernel panic. You can find out more in the Red Hat bug database at bugzilla.redhat.com. In this database, bug number 109834 suggests that this is also

a problem on Red Hat Linux 9. But as shown in this bug and in bug 116446, the `syslinux.cfg` file is easily fixed. In this case, I'd combine these two lines into the following:

```
append initrd=initrd.img ro root=/dev/hda2
```

With this fix, the associated boot disk will now work.

If the kernel can't locate the root filesystem, or if the root filesystem is damaged, the Linux kernel will issue a kernel panic with messages similar to the following:

```
Creating root device
Mounting root filesystem
kjournald starting. Commit interval 5 seconds
EXT3-fs: mounted filesystems with ordered data mode.
pivotroot: pivot_root (/sysroot,/sysroot/initrd) failed: 2)
Freeing unused kernel memory: 272k freed
Kernel panic: No init found. Try passing init= option to kernel
```

Although this may look very bad the first time you encounter it, often the problem can easily be fixed from the **linux rescue** environment with a little bit of work. Other problems may also require the use of the **linux rescue** environment, as described in the following section.

Installation Disk Rescue Mode

As discussed in Chapter 3, you can start Linux in rescue mode from the Red Hat Enterprise Linux installation CD or boot disk. When you type **linux rescue** at the installation boot prompt and go through the steps, the installation disks install a compact version of a root filesystem. As this information has to fit on two 1.44MB floppy disks, it includes a minimal set of utilities that will allow you to mount a disk and either repair the problem with the disk or edit the broken files on the disk.

exam

Watch

The RHCE portion of the Red Hat Exam Prep guide explicitly states that you need to know how to boot into linux rescue mode from the first RHEL 3 installation CD.

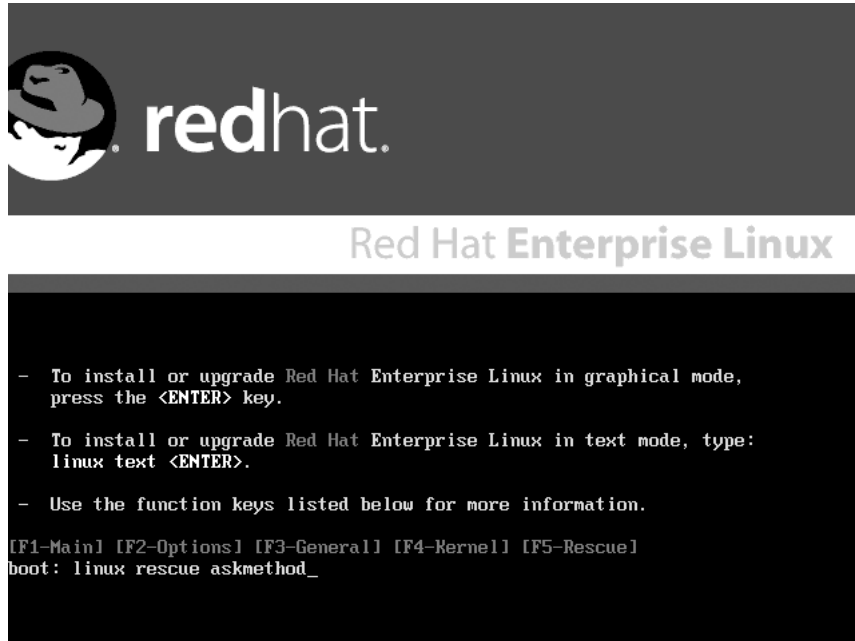
To boot into rescue mode, first boot your system either using your boot floppy or directly with the first installation CD in a bootable CD-ROM drive, as shown in Figure 11-2.

If you've booted from the first RHEL 3 installation CD, you have two options at the boot prompt: you can type **linux rescue** or **linux rescue askmethod** and press ENTER. If you may not need access to the installation RPMs, or

have booted from the RHEL 3 boot floppy, **linux rescue** is sufficient. If you may need the installation RPMs, **linux rescue askmethod** allows you to connect to the network

FIGURE 11-2

Booting into
linux rescue
mode



installation server that you used in Chapter 2—and that may be available to you during the Red Hat exams.



When booting from the RHEL 3 1.44 MB boot floppy, the `linux rescue` command is functionally equivalent to the `linux rescue askmethod` command when booting from the first RHEL 3 installation CD.

When you run the `linux rescue askmethod` command, it's as if rescue mode isn't working; you're taken through the first steps of RHEL 3 installation process in text

exam

Watch

The Red Hat Exam Prep guide suggests that RHCE candidates need to know how to start rescue mode from the first RHEL 3 installation CD. If you need access to the RHEL 3

installation files on a remote network server, run the `linux rescue askmethod` command when you see the prompt shown in Figure 11-2. Otherwise, the `linux rescue` command is sufficient.

mode. You'll need to enter a language, a keyboard type, and the location of the RHEL 3 installation files, as shown in Figure 11-3.

Next, you're asked to configure an IP address, network mask, gateway, and primary (DNS) nameserver for on the local computer. Follow any relevant instructions on your exam carefully. Then as described in Chapter 2, you're asked to point to the network installation server name or IP address, as well as the directory which contains the Red Hat installation files. Once the files associated with the **linux rescue** environment are loaded, you'll see the screen shown in Figure 11-4.

As you can see, you now have three options. I address each option in detail in the following sections:

- **Continue** will search through and mount the available filesystems.
- **Read-Only** performs the same tasks as Continue, except all filesystems that are found are mounted read-only.
- **Skip** does not try to look through the available filesystems. Instead, it proceeds directly to a root shell prompt.

FIGURE 11-3

Connecting to a network source



FIGURE 11-4

Three choices in the **linux rescue** environment



Standard linux rescue Environment

When you select Continue from the screen shown in Figure 11-4, you're taken through the standard **linux rescue** environment. The rescue files search for your root directory (/) filesystem. If found, your standard root directory (/) is mounted on /mnt/sysimage. All of your other regular filesystems are subdirectories of root; for example, your /etc directory will be found on /mnt/sysimage/etc.

Not all of your filesystems may mount properly. You may see error messages such as:

```
Error mounting filesystem on sdb1: Invalid argument
```

This suggests that at least the filesystem that you would normally mount on /dev/sdb1 isn't working for some reason. If the **linux rescue** environment can mount your root directory (/), you'll see a message noting that your system has been mounted, as shown in Figure 11-5.

Click OK. You should see the following prompt messages:

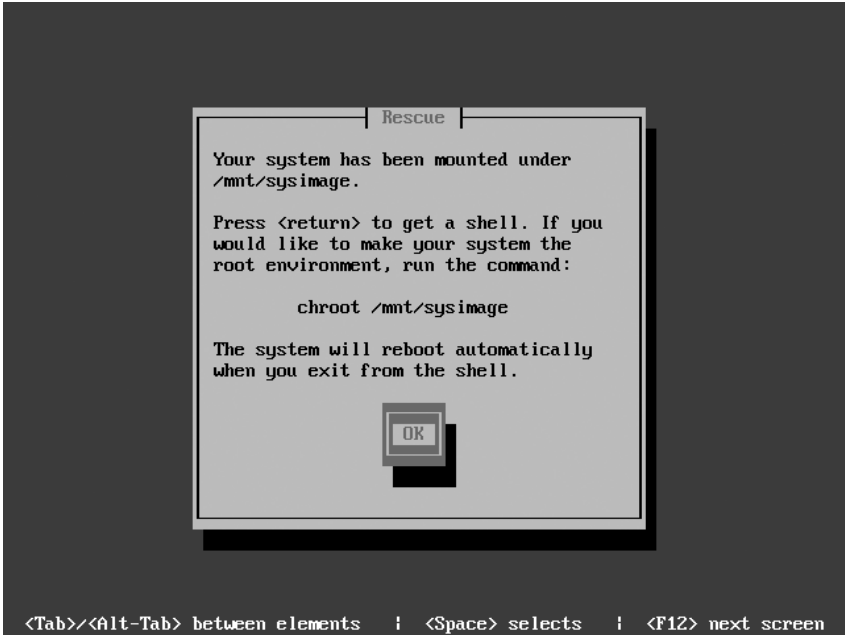
```
Your system is mounted under the /mnt/sysimage directory.
```

```
When finished please exit from the shell and your system will reboot.
```

```
sh-2.05b#
```

FIGURE 11-5

The linux rescue environment has found your root directory (/)



You'll use the `chroot /mnt/sysimage` command shortly. Now you can work on repairing any files or filesystems that might be damaged. First, check for unmounted filesystems. Run a `df` command. The output should look similar to Figure 11-6. Compare the result to the `/mnt/sysimage/etc/fstab` configuration file. If some filesystem is not mounted, it may be configured incorrectly in the `fstab` file. Alternatively, the label associated with a partition may not match the filesystem shown in your `fstab` file. For example, to find the label associated with `/dev/sda1`, run the following command:

```
# e2label /dev/sda1
```

FIGURE 11-6

Labels, filesystems, and partitions

```
sh-2.05b# df
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
rootfs	6120	2468	3302	43%	/
/dev/root.old	6120	2468	3302	43%	/
192.168.30.4:/mnt/inst					
/dev/hda2	6756712	4228736	2184752	66%	/mnt/source
/dev/hda1	3644800	2921256	538396	85%	/mnt/sysimage
/dev/hdd1	101089	15036	80834	16%	/mnt/sysimage/boot
/dev/hdd1	1031800	34328	945060	4%	/mnt/sysimage/home

```
sh-2.05b#
```

which should return the name of a filesystem to be mounted on that partition such as `/boot`.

Sometimes an unmounted filesystem just needs a little cleaning; remember, a command such as the following cleans the `/dev/sdb1` partition.

```
# fsck /dev/sdb1
```

The **fsck** command works only on an unmounted filesystem. For example, if you get a message such as:

```
WARNING!!! Running e2fsck on a mounted filesystem may cause
SEVERE filesystem damage.
```

unmount the subject filesystem such with a command such as **umount /dev/sdb1**. If that doesn't work, restart the rescue process. When you get to the screen shown in Figure 11-4, select Skip and read the "No Mount **linux rescue** Environment" section later in this chapter.

Remember the message in Figure 11-5? It includes an important clue. All you need to do to restore the original filesystem structure is to run the following command:

```
# chroot /mnt/sysimage
```

When you use the rescue disk, your standard root directory (`/`) is actually mounted on the `/mnt/sysimage` directory. This command resets your standard root directory (`/`), so you don't have to go to the `/mnt/sysimage` subdirectory.

When you've made your changes, run the **sync** command, twice, to make sure any changes you've made are written to disk. Type the **exit** command, twice. Linux *should* automatically run the **sync** command again when you exit, making sure any changes are written to disk. Then it stops, allowing you to reboot or restart your computer.



Normally it should not be necessary to run the `sync` command. However, running it several times does make sure that any pending data is actually written to your floppy and hard disks.

Read-Only linux rescue Environment

When you select the Read-Only option shown in Figure 11-4, you'll get the same basic prompt. There is little difference between regular and read-only rescue mode. The rescue system attempts to do everything that it would under regular mode, except all partitions are mounted read-only.

This is appropriate if you have a large number of mounted filesystems; it can help you cull through what is and isn't working with less risk of overwriting key configuration files.

No Mount linux rescue Environment

When you select the Skip option shown in Figure 11-4, the installation files loads a minimal root image from into a RAM disk created by the kernel, and takes you to a root shell prompt (#) as shown:

```
When finished, please exit from the shell and your system will reboot.
-/bin/sh-2.05b# [the actual case is different - mj]
```

At this point, you have access to a basic set of commands. You can mount filesystems, create directories, move files, and edit files using vi. You can apply the **fdisk** and **fsck** commands to various hard disks and partitions. A few other basic commands are also available.

The great difficulty in operating from the rescue environment is that you are working with a minimal version of the Linux operating system. Many of the commands you are used to having at your disposal are not available at this level. If your root partition has not been completely destroyed, you may be able to mount this partition to your temporary root directory in memory and access commands from there.

But you may need a little help identifying the partitions on your system. As I'll show you shortly, the **fdisk -l /dev/hda** command lists the configured partitions on the first IDE hard drive. You can create a new directory such as **/mnt/sysimage**, mount a partition such as **/dev/hda2** on that directory, and check the result with the following commands:

```
# mkdir /mnt/sysimage
# mount /dev/hda2 /mnt/sysimage
# ls /mnt/sysimage
```

If you can verify that you've mounted the standard root directory (/) filesystem on the **/mnt/sysimage** directory, you can run the **chroot /mnt/sysimage** command. You can then have full access to the commands and configuration files available under that mounted partition.



If you mount partitions from your hard drive in rescue mode and then make changes to files on those partitions, remember to use the **sync** command. This writes your files to disk so the information isn't lost if you hit the power button on your computer. Alternatively, a **umount** command applied to any partition also writes data to disk.

Booting into a Different Runlevel

At the boot loader prompt, you can start Linux at a different runlevel. This may be useful for two purposes. If your default runlevel in `/etc/inittab` is 5, your system

normally boots into the GUI. If you're having problems booting into the GUI, you can start RHEL into the standard text mode, runlevel 3.

One other option to help rescue a damaged Linux system is *single-user mode*. This is appropriate if your system can find at least the root filesystem (`/`). Your system may not have problems finding its root partition and starting the boot process, but it may encounter problems

such as damaged configuration files or an inability to boot into one of the higher runlevels. When you boot into single-user mode, your options are similar to the standard **linux rescue** environment where the system has already been mounted and the **chroot /mnt/sysimage** command has been applied.

To boot into a different runlevel, first let us assume that you're using the default RHEL 3 boot loader, GRUB. In that case, press P (lower case) to enter the GRUB password if required. Press A (lower case) to modify the kernel arguments. When you see a line similar to

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

add one of the following commands (shown in bold) to the end of that line:

```
grub append> ro root=LABEL=/ single  
grub append> ro root=LABEL=/ 1  
grub append> ro root=LABEL=/ init=/bin/sh
```

Alternatively, if you're using LILO, the **linux single** command will do nicely. Any of these commands will boot Linux into a minimal runtime environment, and you will receive a bash shell prompt (`bash#`).

Naturally, you can use the same technique to boot into another runlevel; for example, to boot from the GRUB boot loader into runlevel 3, navigate to where you can modify the kernel arguments, and add the following command to the end of the following line:

```
grub append> ro root=LABEL=/ 3
```

exam

Watch

In the current Red Hat Exam Prep guide, it states that RHCTs (and therefore also RHCEs) should be able to boot systems into different runlevels. The method is described in this section.



The terms boot loader and bootloader are used interchangeably. For the purpose of this book, I've used the term "boot loader," as that seems to be the direction of the Red Hat documentation. However, the term "bootloader" is still common even in Red Hat documentation.

When you boot into single-user mode, no password is required to access the system. Running your system in single-user mode is somewhat similar to running a system booted into rescue mode. Many of the commands and utilities you normally use are unavailable. You may have to mount additional drives or partitions and specify the full pathname when running some commands.

When you have corrected the problem, you can reboot the system. Alternatively, you can type the `exit` command to boot into the default runlevel as defined in `/etc/inittab`, probably runlevels 3 or 5.



In single-user mode, any user can change the root password. You do not want people rebooting your computer to go into single-user mode to change your root password. Therefore, it's important to keep your server in a secure location. Alternatively, you can password-protect GRUB to keep anyone with physical access to your computer from booting it in single-user mode.

Troubleshooting the Boot Process

Although there are potentially many things that will cause a system not to boot, they can roughly be categorized as either hardware problems or software and configuration problems. The most common hardware-related problem you will probably encounter is a bad hard disk drive; like all mechanical devices with moving parts, these have a finite lifetime and will eventually fail. Fortunately, the Red Hat exams do not require you to address hardware failures.

Software and configuration problems, however, can be a little more difficult. At first glance, they can look just like regular hardware problems.

In addition to knowing how to mount disk partitions, edit files, and manipulate files, you will need to know how to use several other commands in order to be able to fix problems from rescue mode or single-user mode. The most useful of these are the `df`, `fdisk`, and the `fsck` commands. To diagnose a problem, you need to know how these commands work at least at a rudimentary level.

df

The Linux **df** command was covered briefly in Chapter 3. When you use **df**, you can find mounted directories, the capacity of each partition, and the percentage of each partition that's filled with files. The result shown back in Figure 11-6 illustrates the result in kilobytes. There are a couple of simple variations; the following commands give the result in megabytes and inodes:

```
# df -m
# df -i
```

fdisk

The Linux **fdisk** command was covered briefly in Chapter 3. When you use **fdisk**, you can find the partitions you have available for mounting. For example, the **fdisk -l /dev/hda** command lists available partitions on the first IDE hard disk:

```
# fdisk -l /dev/hda

Disk /dev/hda: 15.0GB, 15020457984 bytes
240 heads, 63 sectors/track, 1940 cylinders
Units = cylinders of 15120 * 512 = 7741440 bytes

   Device   Boot    Start        End    Blocks   Id  System
/dev/hda1   *           1          949    7174408+   b   Win95 FAT32
/dev/hda2             950          963    105840    83   Linux
/dev/hda3             964          1871    6864480    83   Linux
/dev/hda4          1872          1940    521640    f   Win95 Ext'd (LBA)
/dev/hda5          1872          1940    521608+    82   Linux swap
```

Looking at the output from **fdisk**, it's easy to identify the partitions configured with a Linux format, /dev/hda2, /dev/hda3, and /dev/hda5. Given the size of each partition, it is reasonable to conclude that /dev/hda2 is associated with /boot, and /dev/hda3 is associated with root (/).

For simple partitioning schemes, this is easy. It gets far more complicated when you have lots of partitions, as in this next example. You should always have some documentation available that clearly identifies your partition layout within your filesystem:

```
# fdisk -l /dev/hda

Disk /dev/hda: 26.8 GB, 26843545600
255 heads, 63 sectors/track, 3263 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device   Boot    Start        End    Blocks   Id  System
/dev/hda1   *           1          13    104391    83   Linux
/dev/hda2             14         268    2048287+   b   Win95 FAT32
```

/dev/hda3	269	396	1028160	83	Linux
/dev/hda4	397	3263	23029177+	f	Win95 Ext'd (LBA)
/dev/hda5	397	1097	5630751	83	Linux
/dev/hda6	1098	1734	5116671	83	Linux
/dev/hda7	1735	1989	2048256	83	Linux
/dev/hda8	1990	2244	2048256	83	Linux
/dev/hda9	2245	2372	1028218+	83	Linux
/dev/hda10	2373	2499	1020096	82	Linux swap
/dev/hda11	2500	2626	1020096	83	Linux
/dev/hda12	2627	2753	1020096	83	Linux
/dev/hda13	2754	2880	1020096	83	Linux
/dev/hda14	2881	3007	1020096	83	Linux
/dev/hda15	3008	3134	1020096	83	Linux
/dev/hda16	3135	3236	1020096	83	Linux

In this example, it's easy to identify the Linux swap partition. Since /boot partitions are small and normally configured toward the front of a drive, it's reasonable to associate it with /dev/hda1.

However, that is just a guess; some trial and error may be required. For example, after mounting /dev/hda2 on an empty directory, you would want to check the contents of that directory for the typical contents of /boot.

e2label

Based on the previous example, you probably could use a little help to identify the filesystems associated with the other partitions. That's where the **e2label** command can help. When you set up a new filesystem, the associated partition is normally marked with a label. For example, the following command tells you that the /usr filesystem is normally mounted on /dev/hda5.

```
# e2label
Usage: e2label device [newlabel]
# e2label /dev/hda5
/usr
```

dumpe2fs

You can get a lot more information on each partition with the **dumpe2fs** command, as shown in Figure 11-7.

The **dumpe2fs** command not only does the job of **e2label** but also tells you about the format, whether it has a journal, and the block size. Proceed further down this list, and you'll find the locations for backup superblocks, which can help you use **fsck** or **e2fsck** command to check select the appropriate superblock for your Linux partition.

FIGURE 11-7

The **dumpe2fs** command gives a lot of information.

```
dumpe2fs 1.32 (09-Nov-2002)
Filesystem volume name:   /boot
Last mounted on:          <not available>
Filesystem UUID:          ae6d740d-25fb-40d2-b2e7-a5585f8345af
Filesystem magic number:  0xEF53
Filesystem revision #:    1 (dynamic)
Filesystem features:      has_journal filetype needs_recovery sparse_super
Default mount options:    (none)
Filesystem state:         clean
Errors behavior:          Continue
Filesystem OS type:       Linux
Inode count:              26104
Block count:              104391
Reserved block count:     5219
Free blocks:              86052
Free inodes:              26057
First block:              1
Block size:               1024
Fragment size:            1024
Blocks per group:         8192
Fragments per group:      8192
Inodes per group:         2008
Inode blocks per group:   251
--More--
```

on the
 b

fsck is a “front end” for e2fsck, which is used to check partitions formatted to the ext2 and ext3 filesystems.

Filesystem Check—fsck

You should also know how to use the **fsck** command. This command is a front end for most of the filesystem formats available in Linux, such as ext2, ext3, reiserfs, vfat, and more. This command is used to check the filesystem on a partition for consistency. In order to effectively use the **fsck** command, you need to understand something about how filesystems are laid out on disk partitions.

When you format a disk partition under Linux using the **mkfs** command, it sets aside a certain portion of the disk to use for storing *inodes*, which are data structures that contain the actual disk block addresses that point to file data on a disk. The **mkfs** command also stores information about the size of the filesystem, the filesystem label, and the number of inodes in a special location at the start of the partition called the *superblock*. If the superblock is corrupted or destroyed, the remaining information on the disk is unreadable. Because the superblock is so vital to the integrity of the data on a partition, the **mkfs** command makes duplicate copies of the superblock at fixed intervals on the partition, which you can find with the **dumpe2fs** command described earlier.

The **fsck** command checks for, and corrects problems with, filesystem consistency by looking for things such as disk blocks that are marked as free but are actually in use (and vice versa), inodes that don't have a corresponding directory entry, inodes with incorrect link counts, and a number of other problems. The **fsck** command will also fix a corrupted superblock. If **fsck** fails due to a corrupt superblock, you can use the **fsck** command with the **-b** option to specify an alternative superblock. For example, the command:

```
# fsck -b 8193 /dev/hda5
```

tells **fsck** to perform a consistency check on the filesystem on disk partition `/dev/hda5`, using the superblock located at disk block 8193.

exam

Watch

Get to know the key commands and the associated options for checking disks and partitions: *fdisk*, *e2label*, *dumpe2fs*, and *fsck*.

Practice using these commands to check your partitions—on a test computer! (Some of these commands can destroy data.)

Boot Loaders

There are two boot loaders, GRUB and LILO. While you may be more familiar with LILO, Red Hat Enterprise Linux has adapted GRUB as the default boot loader. One of the benefits is that any changes that you make to the GRUB configuration file, `/boot/grub/grub.conf`, need not be written to your hard disk's Master Boot Record (MBR). However, if your MBR has been overwritten by an MS-DOS or Windows NT/2000/2003 boot loader, you can tell your BIOS to look for GRUB with the **grub-install** command. For example, if the `/boot` directory is on the first SCSI hard drive, you would run the following command:

```
# grub-install /dev/sda
```

Alternatively, if you are using LILO, you need to run the **lilo** command whenever you rebuild your Linux kernel or change the disk partition associated with the `/boot` directory. Otherwise, LILO may not be able to find your boot files. In this case, you will have to use the **linux rescue** environment to fix the problem.

In either case, errors to the boot loader configuration file are a common problem that can keep Linux from booting properly.

Missing Files

You may find corruption in some key files or commands such as **mount** or **init**. If you do, one option is to reload the files from the original RPMs. For example, if the **mount** command were to be corrupted or erased, you can reload it from the mount RPM.

When you boot your system into the **linux rescue** environment, using a network source, you have access to those network source files in the `/mnt/source` directory. After your computer boots into the **linux rescue** environment, you'll want to take the following steps:

1. Run the **df** command. You should see how the **linux rescue** environment mounted your partitions. You should also see your network source on the `/mnt/source` directory.
2. Copy the mount RPM from the `/mnt/source` directory. This allows you to reinstall the mount RPM later with files in the correct locations. Use the following command:

```
# cp /mnt/source/RedHat/RPMS/mount-*.rpm /mnt/sysimage/root/
```
3. Run the following **chroot** command to move into the standard directory tree:

```
# chroot /mnt/sysimage
```
4. Install the mount RPM, forcing installation over current files.

```
# rpm -Uvh --force /root/mount-*.rpm
```
5. Check the status of the **mount** command.

```
# rpm -Vf /bin/mount
```

If you see no output, you'll know that there is no longer a problem with the **mount** command. (You can also use this command at the start of the process to see if there is a problem.) You should now be able to run the **exit** command twice to reboot your computer—and at least this problem should be solved.

Places to Look First

Two places where you are likely to make errors that result in an unbootable system are in the boot loader and filesystem configuration files, `/boot/grub/grub.conf` and `/etc/fstab`. In each case, identifying the wrong partition as the root partition (`/`) can lead to a kernel panic. Other configuration errors in `/boot/grub/grub.conf` can also cause a kernel panic when you boot Linux. Whenever you make changes to these files, the only way to test them out is to reboot Linux.

exam

watch

As a Red Hat Enterprise Linux administrator, you will be expected to know how to fix improperly configured

boot files. For this reason, a substantial portion of the exam is devoted to testing your troubleshooting and analysis skills.

The following scenarios and solutions list some possible problems and solutions that you can have during the boot process, and possible associated solutions. It is far from comprehensive. The solutions that I've listed work on my computer, as I've configured it. There may be (and often is) more than one possible cause. **These solutions may not work for you on your computer or on the Red Hat exams. To know what else to try, you need more experience.**

To get the equivalent of more experience, try additional scenarios as proposed in the following Scenario & Solution. Once you're familiar with the **linux rescue** environment, test these scenarios. For the first scenario shown, change the name of the grub.conf file so it can't be loaded. See what it does on your system. Use the **linux rescue** environment to boot into RHEL 3 and use the noted solution to fix your system. Two of the possible error messages are shown in Figure 11-8 and Figure 11-9.

FIGURE 11-8

One possible error message

```
Configuring kernel parameters: [ OK ]
Setting clock (localtime): Thu Jan  8 13:17:09 EST 2004 [ OK ]
Loading default keymap (us): [ OK ]
Setting hostname Enterprise3.example.com: [ OK ]
Initializing USB controller (usb-uhci): [ OK ]
Mounting USB filesystem: [ OK ]
Initializing USB HID interface: [ OK ]
Initializing USB keyboard: [ OK ]
Initializing USB mouse: [ OK ]
Checking root filesystem
fsck.ext2: /:
The superblock could not be read or does not describe a correct ext2
filesystem. If the device is valid and it really contains an ext2
filesystem (and not swap or ufs or something else), then the superblock
is corrupt, and you might try running e2fsck with an alternate superblock:
    e2fsck -b 8193 <device>

Is a directory while trying to open /
[FAILED]

*** An error occurred during the file system check.
*** Dropping you to a shell; the system will reboot
*** when you leave the shell.
Give root password for maintenance
(or type Control-D to continue): _
```

FIGURE 11-9

A second possible error message

```
Booting 'Red Hat Enterprise Linux ES (2.4.21-4.EL)'
root (hd0,1)
Filesystem type is ext2fs, partition type 0x83
kernel /vmlinuz-2.4.21-4.EL ro root=LABEL=/
Error 15: File not found
Press any key to continue..._
```

Summary of the linux rescue Environment

If you have a problem during the boot process, get as much information as you can. Use the experience that you have to recognize or diagnose the problem. Then boot into your

system using a different method, confirm and then fix the problem (naturally, that's the hard part).

The easiest way to boot into an unbootable system is with a customized boot disk.

Alternatively, you may be able to boot into your system at a different runlevel such as 1 or 3. If that is not available or appropriate to your problem, you'll also need to know how to use the **linux rescue** environment to rescue a system, using the following basic steps:

exam

Watch

Whenever you're working in rescue mode or single-user mode, always remember to run the `sync` command to save changes to your drives before halting or rebooting your system.

1. Boot using a Red Hat Enterprise Linux installation floppy or the first installation CD.
2. Know the location of your installation files, such as from a CD or over a network. You are taken to single-user mode.
3. At the rescue shell prompt, use **fdisk -l diskdevice** to identify your partitions.
4. If filesystem problems are suspected or indicated, run **fsck** on the afflicted partitions.
5. If the problem is with a configuration file:
 - a. Create (a) temporary mount point(s), if necessary.
 - b. Mount the appropriate partition(s), if necessary.
 - c. Use the vi editor to fix the problem in the broken file(s).
6. Sync your changes to the drive.
7. Type the **exit** command as needed to restart the system.

SCENARIO & SOLUTION

When you boot RHEL 3, you see a grub> prompt in place of the standard GRUB boot menu.	You may have a problem that prevents the boot loader from reading the GRUB configuration file, grub.conf. The file may be missing or corrupt.
When you boot your computer, you see the following message: "Missing operating system"	Your Master Boot Record (MBR) has been erased, and you'll need to go into the Linux rescue environment and run grub-install /dev/hda (or /dev/sda) to reload GRUB on the MBR.
During the boot process, you see the following message: "Cannot open file '/proc/mounts' for reading (no such file or directory)"	You may have a problem with a corrupt mount command. You'll need to reload it from the mount RPM.
You see the following prompt: init-2.05b#	Check the current directory tree. If you see the standard directories, your init command may be corrupt. Try reloading it from the SysVinit RPM.
You see the following message: "INIT: No inittab file found"	This is straightforward—there is something wrong with your /etc/inittab file; it may be erased. You'll want to go into the linux rescue environment and restore it from backup or the initscripts RPM.
You see a message such as what's shown in Figure 11-8	You may have a problem with the integrity of the /etc/fstab file. Start the linux rescue environment, check the integrity of /etc/fstab. If there is still a problem, run the steps described earlier to check the superblock and more. (For more experience, try including additional errors in /etc/fstab.)
You see a message similar to what's shown in Figure 11-9. Take careful note of the last file cited in the message.	RHEL 3 has encountered some problem when reading the grub.conf configuration file. Start the Linux rescue environment and check this file. Alternatively, check the files noted in grub.conf referenced in the /boot directory. For example, you can create a new initrd file with the mkinitrd command. (For more experience, try introducing other errors in the /boot directory.)
When you boot RHEL 3 into runlevel 5, you don't see a graphical login screen; a text login screen flashes periodically.	You may have some problem with the X Font Server. Boot into a different runlevel that does not require the X server, such as 1, 2, or 3. Check the xfs service script to see that it's set to run in runlevel 5. If yes, check for related error messages.
You see a Welcome to Kudzu screen during the boot process; afterwards, it stops after detecting a network card.	This is symptomatic of a problem with one of the boot scripts, such as /etc/inittab or /etc/rc.d/rc.sysinit. Boot into the linux rescue environment and check the files associated with the boot process.

EXERCISE 11-4**Performing an Emergency Boot Procedure**

To do this exercise, you should have a test system at your disposal. Any data on the computer where you do this procedure is at risk. In this exercise, you will “break” your system by purposely misconfiguring a file and then reboot into the **linux rescue** environment to fix the problem. You’ll be configuring a boot option in your GRUB menu which makes Linux look for boot files in the wrong partition.

1. Install the mkbootdisk RPM from a network source if required:

```
# rpm -q mkbootdisk
# rpm -Uvh /mnt/inst/RedHat/RPMS/mkbootdisk*
```

2. Even if you already have a boot disk, make one. Insert a floppy into the disk drive and type the following (the ``uname -r`` command switch includes the version number of the current kernel):

```
# mkbootdisk `uname -r`
```

3. Edit the file `/boot/grub/etc/grub.conf` and make a copy of your boot stanza. Title this stanza `badboot`. Change the location of the root device to point to an invalid partition. For example, if your original `grub.conf` looks like this:

```
default=0
timeout=10
splashimage=(hd0,0)/grub/splash.xpm.gz
title Red Hat Enterprise Linux ES (2.4.21-4.EL)
    root=(hd0,0)
    kernel /vmlinuz-2.4.21-4.EL ro root=LABEL=/
    initrd /initrd-2.4.21-4.EL.img
```

your new version should look like this:

```
default=0
timeout=10
splashimage=(hd0,0)/grub/splash.xpm.gz
title Red Hat Enterprise Linux ES (2.4.21-4.EL)
    root=(hd0,0)
    kernel /vmlinuz-2.4.21-4.EL ro root=LABEL=/
    initrd /initrd-2.4.21-4.EL.img

title badboot
    root=(hd0,1)
    kernel /vmlinuz-2.4.21-4.EL ro root=LABEL=/
    initrd /initrd-2.4.21-4.EL.img
```

4. Reboot your system. In the GRUB menu, select badboot. GRUB will return a “File not found” message.
5. Since you left a valid boot stanza, your system isn’t really broken. To fix the problem, however, you’re going to boot into rescue mode. Insert your first RHEL 3 installation CD, and reboot the system. At the prompt, type **linux rescue askmethod**.
6. Proceed through the first steps of the Red Hat Enterprise Linux installation process.
7. When you see the Rescue menu, select Skip. None of your partitions will be mounted.
8. Although you know the source of the problem, once you boot into rescue mode, you should familiarize yourself with some of the repair utilities:

```

-/bin/sh-2.05b# fdisk -l
    Device Boot      Start         End      Blocks   Id  System
/dev/hda1   *           1           13       104391   83   Linux
/dev/hda2             14          474      3702982+   83   Linux
/dev/hda3          475          522       385560   82   Linux swap

```

The output of the following command will vary.

```

-/bin/sh-2.05b# fsck -y /dev/hda1
fsck 1.32 (09-Nov-2002)
WARNING: couldn't open /etc/fstab: No such file or directory
e2fsck 1.32 (09-Nov-2002)
/boot: clean, 47/26104 files, 18339/104391 blocks

```

9. Create (a) temporary mount point(s) for your /boot and root directory (/) partitions, and mount those partitions (if they are not already mounted). If the output from **fdisk -l** is different for you, revise the mounted devices accordingly.

```

-/bin/sh-2.05b# mkdir /tmpmnt
-/bin/sh-2.05b# mkdir /tmpmnt/boot
-/bin/sh-2.05b# mount /dev/hda1 /tmpmnt/boot

```

10. Edit the bad stanza in grub.conf and fix the problems:

```

-/bin/sh-2.05b# vi /tmpmnt/boot/grub/grub.conf

```

11. Your new version should look like this:

```

title badboot
    root=(hd0,0)
    kernel /vmlinuz-2.4.21-4.EL ro root=LABEL=/
    initrd /initrd-2.4.21-4.EL.img

```

12. Save your changes to the `grub.conf` file.
 13. Unmount any mounted partitions and **sync** your changes:

```
# umount /dev/hda1
# sync
```
 14. Remove any boot media from your disk drives. Type **exit** to unmount all drives and restart the system. You should now be able to boot from the `badboot` stanza.
-

CERTIFICATION OBJECTIVE 11.05

The Secure Shell Package

Red Hat Enterprise Linux 3 installs the Secure Shell (SSH) packages by default, using the `openssh-server`, `openssh-clients`, and `openssh` RPMs. The Secure Shell and Secure Copy programs, **ssh** and **scp**, are replacements for the **rsh**, **telnet**, and **rcp** programs. They encrypt communication between different computers. The secure daemon, `sshd`, listens for all inbound traffic on port 22. The SSH configuration files are located in the `/etc/ssh` directory.

The Secure Shell daemon works because it encrypts messages. RHEL 3 incorporates SSH version 2, which includes an enhanced key exchange algorithm.

Basic Encrypted Communication

Basic encryption in computer networking normally requires a private key and a public key. You keep the private key and send the public key to others. When they want to send data to you through SSH, their messages are encrypted with the public key. Your computer can descramble the message with the private key.

Encryption keys are based on random numbers. The numbers are so large (typically 512 bits or more), the chance that someone will break into your system, at least with a PC, is quite small in the foreseeable future. Private and public encryption keys are based on a matched set of these random numbers.

Private Keys

Your private key (essentially a file with your special number) must be secure. When you enable an application, it can attach the key to your messages. Anything you send—say,

from your e-mail account—can then be digitally signed and encrypted. The public key is added to the end as part of your signature. Only the recipient will be able to decrypt the message.

Public Keys

Your public key value is just that, publicly available. A central authority such as Verisign, Globalsign, or Thawte provides public access to public keys they have created. If they generate a private key for you, they'll keep a secure copy on their system. You can just attach your public key to the e-mail, or the end users can publicly retrieve it from the Web site associated with the central authority.

The example shown in Figure 11-10 lists the directories and files associated with SSH usage as well as a public key that has been added to your “keyring.”

This key is like a password used to encrypt your data. Imagine trying to remember the 1024-bit number expressed in hexadecimal value as shown below. That is why the applications save this value for you, on a “public keyring.” You can add as many public keys from other users, sites, and services as you wish.

```
3081 8902 8181 00D4 596E 01DE A012 3CAD 51B7
7835 05A4 DEFC C70B 4382 A733 5D62 A51B B9D6
29EA 860B EC2B 7AB8 2E96 3A4C 71A2 D087 11D0
E149 4DD5 1E20 8382 FA58 C7DA D9B0 3865 FF6E
88C7 B672 51F5 5094 3B35 D8AA BC68 BBEB BFE3
9063 AE75 8B57 09F9 DCF8 FFA4 E32C A17F 82E9
7A4C 0E10 E62D 8A97 0845 007B 169A 0676 E7CF
5713 1423 96E0 8E6C 9502 0301 0001
```

Your private key is similar, *but you must keep it private*, or this whole system fails. Keeping it private means no one should have access to your PC. If your PC is public, secure your system with a passphrase (password), as shown in Figure 11-11, in order to use this key.

FIGURE 11-10

A public key

```
[root@Enterprise3 root]# ls .ssh/
id_dsa id_dsa.pub identity identity.pub known_hosts
[root@Enterprise3 root]# cat .ssh/id_dsa.pub
ssh-dss AAAAB3NzaC1kc3MAAACBAOC9nj2/VnRoQ1MoYzElULkaX3cH2YB1QB/H4TYsgXjxKyisv6eZ
/lujSoA3vUfQixeJ8Vf67+sswRo0zDIP1qV6LD2tgueyN8s4YTOXx68qo0JfppabVuHl/Jw+021d+YDH
oXNJAL4xWqCTe8mc2Fee1NVa3tasqQJ4+CA5WJkPAAAFQCcSbdGnx06EkoYqkLiXzCgNWqjbQAAAAIBL
xSPrcX9yr6g6hUDiBf5iSGuXvjVCen47IRgukebFCnoWrDaD+5BfSz9ep7ompgZZGiZqsiEZTHJKRzu+
pZ0zJGblWKvnrrzQz60ZCkk4L8cQbfAs8xgVcpLGM5wjVX2dtwcFIAPIfsdq/mXT0dHfehvo+EMxvL3GS
04cifu6gqAAAAIEAhhbGY2XZ1cdglutzz6GB50IeChErEYsH2frd1TOGiXg5fm2W3XBTkQPuNBHsmIXD4
pnn0612xQqj8KMxojMQwd+9w2dMO/cNV+OZPt0/iwMK/q0BBiPksZUr24Ij/hL4HuaQxmgnKCTD2KT9
qYrg2MfjhjcqGIjoRlzlul/xrfs= root@Enterprise3.example.com
[root@Enterprise3 root]#
```

FIGURE 11-11

Generating
encryption keys

```
[root@Enterprise3 root]# ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/root/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_dsa.
Your public key has been saved in /root/.ssh/id_dsa.pub.
The key fingerprint is:
ec:0d:48:0c:5b:38:6d:a0:31:c1:f2:c7:fb:b5:d3:e5 root@Enterprise3.example.com
[root@Enterprise3 root]# ssh-keygen -t rsa1
Generating public/private rsa1 key pair.
Enter file in which to save the key (/root/.ssh/identity):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/identity.
Your public key has been saved in /root/.ssh/identity.pub.
The key fingerprint is:
94:f7:2d:10:48:6b:c3:07:b2:1c:42:e7:11:30:81:c3 root@Enterprise3.example.com
[root@Enterprise3 root]#
```

How to Generate Your Keys

There are a few SSH-oriented utilities you need to know about:

- **sshd** The daemon service; this must be running for inbound Secure Shell client requests.
- **ssh-agent** A program to hold private keys used for RSA authentication. The idea is that the **ssh-agent** command is started in the beginning of an X session or a login session, and all other windows or programs are started as clients to the **ssh-agent** program.
- **ssh-add** Adds RSA identities to the authentication agent, **ssh-agent**.
- **ssh** The Secure Shell command, **ssh**, is a secure way to log in to a remote machine, similar to Telnet or rlogin. To make this work, you need a private key on the server and a public key on the client. Take the public key file, **identity.pub** or **id_dsa.pub**, created later in this section. Copy it to the client. Place it in the home directory of an authorized user, in the **~/.ssh/authorized_keys** or **~/.ssh/authorized_keys2** file.
- **ssh-keygen** A utility that will create your keys for you. The **ssh-keygen -t keytype** command will create the keys you desire. The *keytype* can be **dsa** (Digital Secure Algorithm) or **rsa1** (RSA Security). The commands work as shown in Figure 11-11.

All you need to do is transfer the public key, with the .pub extension, to an authorized user. It's important to add a passphrase to protect that digital signature. In the worst case, a cracker could use this file to effectively steal your identity.

Why Use SSH?

The Internet is a public network. If you're connected to the Internet, anyone in the world could conceivably access your computer through this public network. All that is needed is Internet access from an anonymous location. In other words, a skilled cracker may be able to capture your passwords from a computer in a public library.

In contrast, private networks are used for security applications. Merchants who dial into a central server to check authorized credit card numbers are connecting to a private network. Access to such private networks can be expensive.

Unix and the network that became the Internet started in an educational setting, where there is a premium on the free exchange of information. While the resulting openness of the Internet is good, it can present security challenges. The original Unix tools developed for networks were not designed with security in mind.

These tools include **telnet**, **ftp**, and the “r” (remote) commands (**rlogin**, **rcp**, **rsh**). These utilities pass all information, including login names and passwords, across the network in clear text format. Anyone with a simple protocol analyzer such as Ethereal can find your password in this way. The Ethereal output shown in Figure 11-12 highlights one of the letters in a Telnet password on my private LAN (it's an “e”). The other Telnet packets contain the other letters and or numbers of the password.

This is the Kerberos version of the Telnet server discussed in Chapter 7. While it's more secure than the older Telnet server, you can still find the clear text password using a tool such as Ethereal.

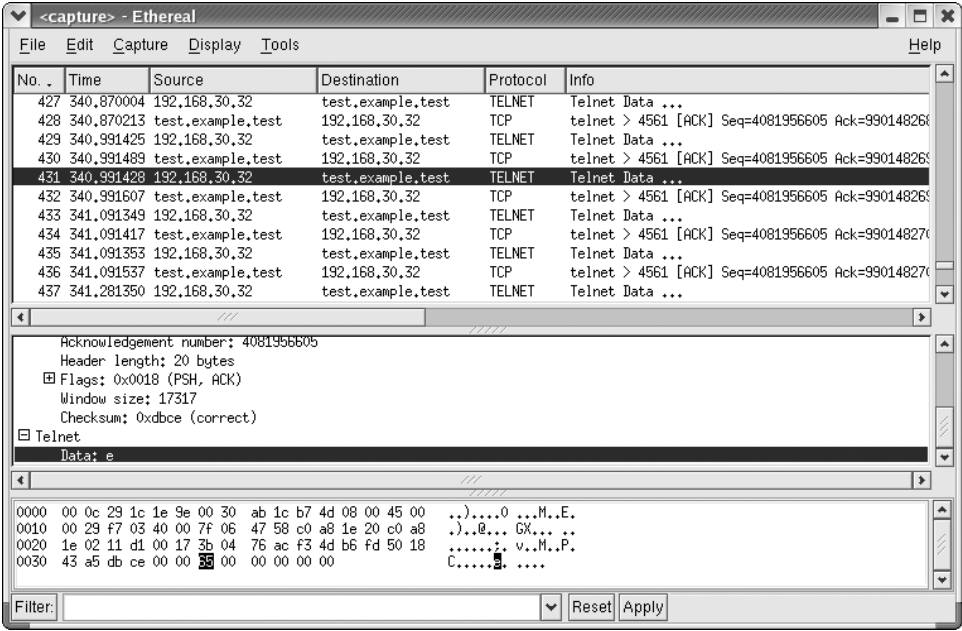


We do not in any way endorse the cracking of passwords. However, as a system administrator, you do need to know your vulnerabilities. If you can trace clear text passwords on your own network, you are at risk.

The Secure Shell utilities were an answer to this problem, using high encryption standards. The Secure Shell tools replaced their insecure brethren and provided full encryption of all data between the hosts that is very hard to break, even by the “brute force” method.

FIGURE 11-12

It's easy to decipher a clear text password.



Brute Force Decryption

All computer data, including passwords, can be broken down into bits and bytes—in other words, 1s and 0s. If you can see the pattern of bits and bytes, you can decipher the password.

By convention, every character is associated with 7 of the 8 bits in a byte. Now assume you have 8 characters in a password. This makes for a 56-bit password. All you need to do to find the password is to compare it with every possible combination of bits and bytes. In other words, one of the bit-based values between 0 and 2^{56} (72,057,594,037,927,936) represents your password.

While this may seem like a large number, it is in fact trivial for a higher-end PC, which could decipher such a password in a matter of hours. This suggests that the older standard for secure HTTP Web pages, 40-bit encryption, is not very secure. On the other hand, the higher-level 128-bit-encrypted browsers are a lot more secure. It would take more than a few years for the latest PCs to break through such encryption.

By default, the `ssh-keygen` command creates encryption keys with 1024 bits. Breaking into such systems is beyond the capabilities of the latest PCs using brute force methods.

Lost Passphrase on Private Keys

Unfortunately, if you lose the passphrase on your private encryption key, there is no way to recover. You would need to start over, which means you would need to create a new set of keys and send your new public key to all concerned for them to replace your old public key.

PGP (Pretty Good Privacy)

E-mail has become a standard fixture in everyday business; e-mail security is a significant issue. One of the current standards for managing e-mail security is known as PGP (Pretty Good Privacy), developed by Phil Zimmerman. Like SSH, PGP provides a private- and public-key system for e-mail clients. The public key is usually attached to the end of the data, as shown in the next example.

The Linux implementation of PGP is known as GPG, the Gnu Privacy Guard. GPG is an implementation of the OpenPGP standard. Both standards are now common ways people and companies protect their correspondence.

```
*****
To: Dreg (SD444466)
From: The SANS NewsBites service
*****
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1
... (text removed)

Please feel free to share this with interested parties via e-mail (not
on bulletin boards). For a free subscription, (and for free posters)
e-mail sans@sans.org with the subject: Subscribe NewsBites
...
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1.0.4 (BSD/OS)
Comment: For info see http://www.gnupg.org

iD8DBQE63beV+LUG5KFpTkYRAu/WAJ0fUwoQFUOETTd+wAbe1L784S3PDwCfULr0
DXDk20qZotKDLmfjLz1Gty4=
=K1Av
-----END PGP SIGNATURE-----
```

Notice in the previous listing that the BSD/OS (Berkeley Software Distribution/Operating System) is also using GPG, which is also known as GnuPG.

Validating RPMs with GPG

If you download an RPM from the Internet and would like to verify that it is an official unadulterated Red Hat RPM, GPG can help. All you need is the Red Hat Enterprise Linux public key. It's available from at least three different locations:

- Each Red Hat Enterprise Linux installation CD, in the main directory, in the RPM-GPG-KEY file
- On a standard Red Hat Enterprise Linux 3 installation, in `/usr/share/rhn/RPM-GPG-KEY`
- Online from `www.redhat.com`

First, you need to import the key. For example, if you want to import the key from the Red Hat installation CD, mounted on `/mnt/cdrom`, run the following command:

```
# rpm --import /mnt/cdrom/RPM-GPG-KEY
```

Now you should find the RPM-enabled key in the `/var/lib/rpm/Pubkeys` file. Next, you can verify the package's PGP-encrypted checksum using the `rpm -vK packagename` command. For example, if you wanted to verify the openoffice RPMs on a downloaded installation CD, run the following command:

```
# rpm -vK /mnt/cdrom/RedHat/RPMS/setup-*
```



Older versions of Red Hat Linux (and RHEL 2.1) used the `gpg --import keyfile` command to download the Red Hat public GPG key. This is no longer possible with the advent of RPM version 4.1.

CERTIFICATION OBJECTIVE 11.06

Managing Logical Volumes

Red Hat often waits a while before including software in its Linux distributions. I believe this practice promotes the interests of stability. This practice extends to Logical Volume Management (LVM). As a RHCE, you need to know how to add, remove, and resize Logical Volumes (LV).

You learned to create logical volumes during the installation process in Chapter 2, and after RHEL 3 is installed in Chapter 3. If you've created and mounted any Logical Volumes on your RHEL 3 computer, they should be readily available in the `/etc/fstab`

configuration file. For example, I have a fairly generic LV for the /home directory on a different RHEL 3 computer, and it shows as the following command line in /etc/fstab:

```
/dev/Volume00/LogVol100 /home ext3 defaults 1 2
```

exam

Watch

In the Red Hat Exam

Prep guide, Red Hat has recently added a requirement to add, remove, and resize LVs to the Troubleshooting and System Maintenance portion of the RHCE exam.

Alternatively, you should be able to find the same basic information with the **vgscan** command. You can verify configured volume groups (VG) with the **vgdisplay** command. For example, Figure 11-13 illustrates the configuration of logical volume Volume00.

Before Logical Volumes are useful, you need to know how to add another LV. For example, if you've added more users, and they need more

room than you have on the /home directory, you may need to add more LV for other filesystems, or resize the current /home directory Logical Volume.



Linux can't read /boot files if they're installed on a Logical Volume.

If you haven't added an LV before, you'll need to set up the basic files and the kernel LVM module, **lvm-mod**. To do so, run the **vgscan** command. This creates the /etc/lvmtab configuration file and adds the **lvm-mod** module. The next time you boot Linux, the /etc/rc.d/rc.sysinit file finds your /etc/lvmtab file and automatically loads the **lvm-mod** module.

FIGURE 11-13

Configuration
of a Logical
Volume (LV)

```
[root@Enterprise3a root]# vgdisplay Volume00
--- Volume group ---
VG Name                Volume00
VG Access               read/write
VG Status               available/resizable
VG #                    0
MAX LV                  256
Cur LV                  1
Open LV                  1
MAX LV Size             255.99 GB
Max PV                  256
Cur PV                  2
Act PV                  2
VG Size                 1.71 GB
PE Size                 4 MB
Total PE                438
Alloc PE / Size         275 / 1.07 GB
Free PE / Size          163 / 652 MB
VG UUID                 nSvJ9E-76ZC-hAOC-wFgj-j31I-ZUCh-zC06pc

[root@Enterprise3a root]#
```

Adding Another Logical Volume

Adding another LV is a straightforward process. I've described it in some detail in Chapter 3, so I only summarize the steps here. For example, if you've just added a fourth SCSI hard drive, it's known as device `/dev/sdd`. If you need more Logical Volumes for the `/tmp` directory, you'd follow these basic steps:

1. Add the new hard drive.
2. Configure the new hard drive with partitions using a command tool such as **fdisk**. Make sure the hex code for the LVM partition(s) corresponds to the Linux LVM format. It's code 8e within **fdisk**. Alternatively, you can dedicate all space on the new hard drive as a Physical Volume (PV) with the **pvcreate /dev/sdd** command.
3. If you've created separate partitions, you can dedicate the space of a specific partition to a PV. If you don't already have an empty logical volume, you'll need to create more than one. For example, for the first partition `/dev/sdd1`, you can do this with the following command:

```
# pvcreate /dev/sdd1
```

4. Next, you'll want to create a Volume Group (VG) from two or more empty, properly configured partitions (or drives). One way to do this, assuming you have an empty `/dev/sdc3` partition, is with the following command:

```
# vgcreate Volume01 /dev/sdc3 /dev/sdd1
```

5. Before proceeding, you should inspect the VG with the **vgdisplay** command. I've illustrated an example back in Figure 11-13.
6. You should now be able to add another LV with the **lvcreate** command. For example, the following command takes 20 Physical Extents (PE) for the new LV, `LogVol01`:

```
# lvcreate -l 20 Volume01 -n LogVol01
```

7. You've added a new logical volume. Naturally, you'll need to format and mount a directory on this LV before you can use it. For the example shown, you would use the following commands:

```
# mkfs -j /dev/Volume01/LogVol01
# mount -t ext3 /dev/Volume01/LogVol01 /tmp
```

Removing Logical Volumes

Removing an existing LV requires a straightforward command. The basic command is **lvremove**. If you've created an LV in the previous section and want to remove it, the basic steps are simple:

1. Save any data in directories that are mounted on the LV.
2. Unmount any directories associated with the LV. Based on the example in the previous section, you would use the following command:

```
# umount /dev/Volume01/LogVol01
```
3. Apply the **lvremove** command to the LV with a command such as:

```
# lvremove /dev/Volume01/LogVol01
```
4. You should now have the PEs from this LV free for use in other LVs.

Resizing Logical Volumes

If you have an existing LV, you can add a newly created PV to extend the space available on your system. All it takes is appropriate use of the **vgextend** and **lvextend** commands. For example, if you want to add PEs to the VG associated with the aforementioned `/home` directory, you could take the following basic steps:

1. Back up any data existing on the `/home` directory.
2. Unmount the `/home` directory from the current logical volume.
3. Extend the VG to include the new hard drive or partitions that you've created. For example, if you wanted to add `/dev/sdd1` to the `/home` VG, you would run the following command:

```
# vgextend Volume00 /dev/sdd1
```
4. Make sure the new partitions are included in the VG with the following **vgdisplay** command:

```
# vgdisplay Volume00
```
5. Extend the current LV to include the space you need. For example, if you wanted to extend the LV to 2000MB, you'd run the following command:

```
# lvextend -L2000M /dev/Volume00/LogVol00
```

The **lvextend** command can help you configure LVs in KB, MB, GB, or even TB. For example, you could get the same result with the following command:

```
# lvextend -L2G /dev/Volume00/LogVol00
```

6. Reformat and remount the LV, using commands described earlier:

```
# mkfs -j /dev/Volume00/LogVol100
# mount -t ext3 /dev/Volume00/LogVol100 /home
```

CERTIFICATION SUMMARY

As you've seen in the past two chapters, there are many facets to system security. Making sure that network services run under nonprivileged accounts helps minimize the risks associated with outside attack.

By default, Red Hat Enterprise Linux assigns unique user and group ID numbers to each new user. This is known as the User Private Group scheme. This scheme allows you to configure special groups for a specific set of users. The users in the group can be configured with read and write privileges in a dedicated directory, courtesy of the SGID bit.

Two key cron jobs are *anacron* and *tmpwatch*. The *anacron* system, once installed from an outside source, automatically runs jobs that did not run while a RHEL 3 computer was turned off. The *tmpwatch* system is used to clear files from specified directories on a regular basis.

One of the most valuable skills you can have as a RHEL 3 systems administrator is knowing how to rescue a system. Some configuration problems can prevent a system from booting. Others can keep you from logging in. Key tools to manage configuration problems include: a boot disk from the **mkbootdisk** command, the **linux rescue** environment from a Red Hat installation floppy or CD, and single-user mode at the GRUB or LILO prompt. The **linux rescue** environment loads basic utilities and attempts to remount your directories on */mnt/sysimage*. Key tools for managing problems include **fdisk**, **fsck**, **e2label**, and **dumpe2fs**.

Any network that is connected to an insecure network such as the Internet is vulnerable. The OpenSSH server can help you set up encrypted communication between computers. Encryption uses private and public keys. Whether you're using SSH or GPG, the default 1024-bit encryption scheme provides a "reasonable" level of security.

Finally, you need to know how to add, remove, and resize Logical Volumes (LV). Once you've configured partitions properly, you can add an LV with appropriate use of the **pvccreate**, **vgcreate**, and **lvcreate** commands. You can remove an LV with the **lvremove** command. You can resize an LV with the help of the **vgextend** and **lvextend** commands.



TWO-MINUTE DRILL

Here are some of the key points from the certification objectives in Chapter 11.

Services and Special Users

- ☐ When Red Hat Enterprise Linux starts on your computer, it triggers a special process known as `init`, which then opens the other basic processes required for a working Linux system.
- ☐ Many services have their own user ID, as shown in `/etc/passwd`; alternatively, some services can run under the `nobody` account.
- ☐ Service user IDs are not for real users.

Red Hat User Private Group Scheme

- ☐ Red Hat's user private group scheme configures users with their own unique user and group ID numbers.
- ☐ With appropriate SGID permissions, you can configure a shared directory for a specific group of users.
- ☐ Setting the SGID bit ensures that all files created in a shared directory belong to the correct group.
- ☐ Setting the SGID bit is easy; use `chown` to set `nobody` as the user owner, and the name of the group as the group owner. Then run the `chmod 2770` command on the shared directory.

anacron and tmpwatch

- ☐ The `anacron` service includes a special cron job which runs jobs that were supposed to be run while a computer is powered down.
- ☐ The `tmpwatch` command (`/usr/sbin/tmpwatch`) is used to remove files that have not been accessed in a specified number of hours.

The linux rescue Environment

- ☐ Rescue disks aren't always available. Be prepared to use the `linux rescue` environment from a Red Hat installation CD.
- ☐ One good boot disk customized for your system can be created with the `mkbootdisk` command.

- ☐ When you start the **linux rescue** environment, be able to access the Red Hat installation files.
- ☐ The **linux rescue** environment may be able to mount your filesystems on `/mnt/sysimage`. It will provide a minimum set of commands such as **fdisk** and **fsck**, and utilities such as **vi**.
- ☐ If your filesystems are properly mounted, you can access documents such as man pages normally after a **chroot /mnt/sysimage** command.
- ☐ Linux may not be able to mount all of your filesystems.
- ☐ Some mount points fail because of bad labels. Check the label on a specific partition with the **e2label** command.
- ☐ The **fsck** command is used to check the filesystem on a partition for consistency.
- ☐ The **dumpe2fs** command can provide a lot of information about each partition, including the label, the journal, and the block size.
- ☐ One alternative to the **linux rescue** environment is single-user mode. You can even boot from GRUB into other runlevels such as 3 and 5.
- ☐ Common problem files that can prevent Linux from booting include `/boot/grub/grub.conf` and `/etc/fstab`.

The Secure Shell Package

- ☐ The OpenSSH command utilities—**sshd**, **ssh**, **ssh-keygen**, **ssh-add**, and **ssh-agent**—provide secure remote services over any network connections.
- ☐ Encryption is based on private and public keys.
- ☐ You can keep your private key secure with a passphrase.
- ☐ Public keys are shared with others, so that they can communicate with you through SSH.
- ☐ As it is easy to decipher traffic, even passwords, from **telnet**, **ftp**, and the “r” commands, it is best to use SSH on any publicly accessible network.
- ☐ PGP, Pretty Good Privacy, is the way much e-mail is encrypted today.
- ☐ The Linux implementation of PGP is GPG.
- ☐ You can validate the signature of any RPM using the **rpm -vK packagename** command.

Managing Logical Volumes

- ☐ To add an LV, you need to know how to use the **pvccreate**, **vgcreate**, and **lvcreate** commands.
- ☐ To remove an LV, you need to know how to use the **lvremove** command.
- ☐ To resize an LV, you need to know how to use the **vgextend** and **lvextend** commands.



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.

Services and Special Users

1. You are setting up a Red Hat Enterprise Linux system and are adding several new services. What can you do to make sure your system is more secure from outside attack?
 - a. Create individual accounts for each service.
 - b. Pick a really secure password for the root account.
 - c. Run services under their own accounts, or alternatively, the nobody account.
 - d. Make sure the system is stored in a secure room.

Red Hat User Private Group Scheme

2. You've just issued the following command: `chown nobody.developgrp /home/developer`. How would you set the SGID bit on this directory?
 - a. `chmod 2775 /home/developer`
 - b. `chgrp 2775 /home/developer`
 - c. `chmod 775 /home/developer`
 - d. `chmod g+s /home/developer`

anacron and tmpwatch

3. What are the normal uses of the `tmpwatch` command script?
 - a. To monitor the system for break-in attempts
 - b. To clean up unused user account directories
 - c. To scan system-wide temporary directories and clean up old temporary files
 - d. To monitor the `/tmp` directory for the appearance of certain files

The linux rescue Environment

4. The junior system administrator at your site has just come to you to report a suspected bad hard drive on the system he was working on. Whenever he tries to boot the system, he gets a kernel panic with a message saying the root partition cannot be found. What is the most likely cause?
 - a. The hard drive has crashed.
 - b. The I/O bus is going bad.
 - c. Intermittent RAM problems are masquerading as disk problems.
 - d. The junior system administrator was modifying a system configuration file and has managed to configure the system so that it will not boot.
5. How can you boot a damaged Linux system in order to perform repairs?
 - a. Boot from your system's custom boot floppy.
 - b. Boot into the **linux rescue** environment.
 - c. Boot into single-user mode. To do so, add the word **single** to the end of the kernel command line in GRUB.
 - d. Boot into runlevel 4.
6. You are a consultant and are helping a client who has managed to render his system unbootable. You have booted into rescue mode, but the client doesn't have any documentation on the partition layout on his disk drive. What can you do?
 - a. Use the **fdisk -l** command to display the partition table for installed drives.
 - b. Reinstall Linux.
 - c. Use the **fsck** command and look for the superblock.
 - d. Use the **fdisk** command in interactive mode.
7. You are trying to boot a system and keep receiving a message about a corrupted partition. You manage to boot into rescue mode. From this point, what might you do to fix the problem?
 - a. Use **fdisk** and delete the partition, then add it back.
 - b. Use the **fdisk -l** command.
 - c. Run **grub-install** to rebuild the boot block on the BIOS.
 - d. Run the command **fsck -b 8193**.
8. Where are some likely places for configuration errors that can prevent your system from booting? (Choose all that apply.)
 - a. `/boot/grub/grub.conf`
 - b. `/etc/fstab`
 - c. `/etc/passwd`
 - d. `/boot`

The Secure Shell Package

9. When you set up a set of secure RSA keys for Secure Shell access, what should you do?
- Run `ssh-keygen -t dsa1`, and then enter a passphrase for your public key.
 - Run `ssh-keygen -t rsa`, and then enter a passphrase for your private key.
 - Run `ssh-keygen -t dsa1`, and then enter a passphrase for your private key.
 - Run `ssh-keygen -t rsa1`, and then enter a passphrase for your private key.

Managing Logical Volumes

10. If you already have added PEs to your VG, and want to add room to your LV with your `/var` directory, what command would you use to increase the size of that LV? Assume that you've backed up any essential data from `/var`, and the volume names stated in the answers are correct.
- `vgextend Volume00 /dev/hdd1`
 - `lvcreate -L2000M /dev/Volume00/LogVol00`
 - `lvextend -L2000M /dev/Volume00/LogVol00`
 - `vgcreate Volume00 /dev/hdc1 /dev/hdd1`

LAB QUESTIONS

Lab 1

In this lab, you'll create a private directory for a group of engineers designing some galleys. You'll want to create a group named `galley` for the engineers named `mike`, `rick`, `terri`, and `maryam`. They'll want to share files in the `/home/galley` directory. What do you need to do?

Lab 2

In this lab, you'll configure the `tmpwatch` script in `/etc/cron.daily` to delete files from the `/var/log/httpd` directory on a periodic basis. You're doing this because the data that comes through your Web server is overwhelming your system. Unless you delete older files from this directory on a periodic basis, the data will crowd out the space needed by your users. Presumably, you're doing this until your new hard disk array is ready in a few weeks.

Lab 3

For this exercise, use a test computer. Do not use a production computer. Do not use a computer where any data might be important to you. If something goes wrong, and you are unable to restore from a backup, you may need to reinstall Linux. This exercise assumes that you're using the default Red Hat Enterprise Linux boot loader, GRUB.

Navigate to the `/boot` directory. Change the name of the `initrd-versionnumber.img` file. Make sure it's something easy to remember such as `initrd-versionnumber.bak`. Reboot Linux. As GRUB goes through the boot sequence, it will probably stop when it can't find your Initial RAM Disk (`initrd`) file, similar to what is shown in Figure 11-14.

Now that your boot loader isn't working, what do you do? Can you try to start Linux in single-user mode?

Lab 4

Your company bought another competitor on the opposite coast recently, just as the new corporate application was being deployed everywhere, so you sent the app to them, too. They use a Unix host for this application on their network. You need to be able to connect to this host for maintenance purposes on the new system-wide application you deployed. Both networks have Internet access.

Lab 5

In this lab, you'll create new PEs, and use them to increase the size of a configured LV. You're doing this for the LV used by the `/var` directory. Because of the increasing demands of your Web site, you need more room for the `/var` directory for your Web site data. Assume your `/etc/fstab` configuration file includes the following line:

```
/dev/Volume00/LogVol100 /var ext3 defaults 1 2
```

You've created PEs from the `/dev/sde` and `/dev/sdf` hard drives, and have just added another SCSI hard drive, `/dev/sdg`. Assume you've backed up the data that you need from the `/var` directory.

FIGURE 11-14

A boot failure

```
Booting 'Red Hat Enterprise Linux ES (2.4.21-4.EL)'
root (hd0,0)
Filesystem type is ext2fs, partition type 0x83
kernel /vmlinuz-2.4.21-4.EL ro root=LABEL=/
[Linux-bzImage, setup=0x1400, size=0x12c9d11]
initrd /initrd-2.4.21-4.EL.img

Error 15: File not found

Press any key to continue..._
```

SELF TEST ANSWERS

Services and Special Users

1. ☒ **A** and **C**. You should run network services under their own accounts or the nobody account. If a cracker does find a security hole, the damage will be limited to that service.
☒ **B** and **D** are incorrect. **B** is a good idea, but most outside crackers get root account privileges through security holes they find in programs or services. **D** is always a good idea but does nothing to protect your system from attacks from the outside, such as from the Internet.

Red Hat User Private Group Scheme

2. ☒ **A** and **D**. Both commands will set the SGID bit. Answer **D** is sometimes more appropriate, as it does not affect the permissions for the owner or others.
☒ **B** is incorrect, as **chgrp** does not modify SGID bits, and **C** is incorrect because it does not include the proper value for the SGID bit (2xyz).

anacron and tmpwatch

3. ☒ **C**. The **tmpwatch** command is usually run periodically by the cron daemon. It recursively searches through temporary directories and removes files that have not been accessed for a fixed amount of time.
☒ **A**, **B**, and **D** are incorrect. While **tmpwatch** does not monitor break-in attempts, you may be able to do this by creating /var/log/tmp using the system logging service (**A**). While it is a good idea to clean up unused account directories (**B**), you may not want to delete directories of people who are just on vacation. The **tmpwatch** command utility (**D**) does not monitor any directory for the appearance of specific files.

The linux rescue Environment

4. ☒ **D**. In a situation like this, the cause is most likely human error.
☒ **A**, **B**, and **C** are incorrect. If you have a kernel panic, then you know that your PC has checked its hardware and run through its initial BIOS checks. Therefore, you know that the hard disk is not bad (**A**), the I/O bus still works (**B**), and RAM was detected (**C**).
5. ☒ **A**, **B**, and **C** are correct procedures to boot a damaged Linux system in order to perform repairs.
☒ **D** is incorrect. By default, Red Hat Enterprise Linux does not use runlevel 4.

6. ☒ **A** is a good starting point, which will allow you to determine what partitions are available. Alternatively, **D** will give you some clues as to partition sizes. In addition, you could then try the **e2label** command; if you're fortunate, you'll see the labels associated with actual mounted partitions such as **/boot** or **/home**. One other approach is to just mount each partition, look at the file contents, and deduce what filesystem it represents.
- ☒ **B** and **C** are incorrect. Reinstallation is a last resort (**B**), would take too much time, and isn't allowed on either Red Hat troubleshooting exams. The **fsck** command might tell you the name of the partition (**C**), but you may end up waiting a long time.
7. ☒ **D**. Try running the **fsck** command with an alternative superblock, such as 8193, 16385, and so on. You can use the **dumpe2fs** command to help determine the appropriate superblock size.
- ☒ **A** might fix the problem but would have the unfortunate side effect of deleting all the data on the partition. **B** would just list the configured partitions, and **C** does not do anything to fix your existing partitions.
8. ☒ **A**, **B**, and **D**. Any typo in the **/boot/grub/grub.conf** or **/etc/fstab** files can make a good system unbootable. Any time you make changes that affect the files in **/boot**, you should make sure **grub.conf** reflects these changes.
- ☒ **C** is incorrect. The **/etc/passwd** file contains user authentication information and does not affect how Linux boots on your computer.

The Secure Shell Package

9. ☒ **D**. RSA keys are configured by the **ssh-keygen -t rsa1** command. A passphrase can be used to protect your private key.
- ☒ **A**, **B**, and **D** are incorrect. As of this writing, there are no **dsa1** key options for the **ssh-keygen** command; even if there are, it would work with Digital Signature Algorithm keys (**A**, **C**). The **rsa** option is currently obsolete (**D**).

Managing Logical Volumes

10. ☒ **C**. The **lvextend** command extends the scope of the LV to the size specified with the **-L** switch, in this case, 2000MB.
- ☒ **A**, **B**, and **D** are incorrect. The **vgextend** command adds more PEs from a newly configured partition to a current VG. It may be required before you increase the size of the LV, but does not address the question. The **lvcreate** command creates a new LV. It does not increase the size of an LV, as stated in the question. The **vgcreate** command creates a new VG, which is not directly related to the question.

LAB ANSWERS

Lab 1

This is a straightforward process, using the following basic steps:

1. Create accounts for mike, rick, terri, and maryam if required. You can use the **useradd** command, edit the `/etc/passwd` file directly, or work through the Red Hat User Manager.
2. Set up a group for these users. Configure a group ID outside the range of your regular users with a line such as:

```
galley:10000:mike,rick,terri,maryam
```

3. Create the `/home/galley` directory. Give it proper ownership and permissions with the following commands:

```
# mkdir /home/galley
# chown nobody.galley
# chmod 2770 /home/galley
```

Lab 2

In this lab, you'll want to set up a command in the `tmpwatch` script in the `/etc/cron.daily` directory, which deletes files that haven't been accessed for a certain number of hours. The commands in the default version of this script can help guide you in this process. For example, if you wanted to delete files that haven't been accessed in more than 30 calendar days, you just need to translate this into the equivalent number of hours. Thus, all you need to do is add the following command to the noted script:

```
/usr/sbin/tmpwatch 720 /var/log/httpd
```

Lab 3

As you practice learning about Linux for the RHCE exam, it's important to know how GRUB works. By default, it requires an initial RAM disk file, `initrd-versionnumber.img`. If GRUB can't find this file, it'll give you the error shown in Figure 11-14. Since your computer does not boot, you'll need to boot with a rescue disk before you can fix the `initrd` file. Remember to make sure that the filename matches the name shown in `/boot/grub/grub.conf` *exactly*.

You can repeat this process with the `vmlinuz` file or the `root` directive in `grub.conf`. Make sure to have backups of key files so you can restore your original configuration. When you repeat this process, what

happens after you select a kernel from the GRUB menu? Do you see a different error? Is it associated with a different file?

Understanding these answers can help you learn to use GRUB messages to better diagnose specific problems with Linux.

Lab 4

If you need access now, and both systems are connected to the Internet, you can set up SSH for secure communications. If the other network does not already have it installed, have them download it from the Internet, install it, and then create an account for you.

The basic steps which are outlined here may vary with the version of Unix used on the other network.

Get the OpenSSH utility source from the Net and put it into a specific directory. Since RPMs are not yet made for Unix, you'll need to unpack a "tarball." You can then unpackage the files in the tarball and use the files in the resulting directory to compile and configure a Secure Shell server. Once it is configured, set up private and public keys.

If you don't need immediate access, you could, alternatively, configure a computer with Linux and a Secure Shell server. Send the computer to the administrator of the remote Unix network. Have them add it to their network, and you can check the problem from your site securely. The application is running on the Linux computer that you sent. (Alternatively, you can even set up OpenSSH on Microsoft Windows, as described earlier in this chapter.)

Lab 5

If you've just added the new hard drive, you'll need to set up partitions or use the entire hard drive for PEs. Based on the premises of the lab, you have the entire SCSC /dev/sdg hard drive available, so you can just allocate this entire hard drive as PEs with the following command:

```
# pvcreate /dev/sdg
```

The next step is to extend the VG, Volume00, to include the newly configured PEs. You can do so with the following command:

```
# vgextend Volume00 /dev/sdg
```

With the additional PEs at your disposal, you can increase the size of the LV allocated to the /var directory. For example, if you wanted to increase the size to 2GB, you could run the following command:

```
# lvextend -L2G /dev/Volume00/LogVol100
```