Red Hat Enterprise Linux 5
Deployment Guide

Deployment, configuration and administration of Red Hat Enterprise Linux 5
Edition 11

Red Hat Customer Content
Services

Red Hat Enterprise Linux 5 Deployment Guide

Deployment, configuration and administration of Red Hat Enterprise Linux 5
Edition 11

Red Hat Customer Content Services

Legal Notice

Abstract

The Deployment Guide documents relevant information regarding the deployment, configuration, and administration of Red Hat Enterprise Linux 5.

Table of Contents

Introduction

Welcome to the *Red Hat Enterprise Linux Deployment Guide*.

The Red Hat Enterprise Linux Deployment Guide contains information on how to customize your Red Hat Enterprise Linux system to fit your needs. If you are looking for a comprehensive, task-oriented guide for configuring and customizing your system, this is the manual for you.

This manual discusses many intermediate topics such as the following:

- Setting up a network interface card (NIC)
- Configuring a Virtual Private Network (VPN)
- Configuring Samba shares
- Managing your software with RPM
- Determining information about your system
- Upgrading your kernel

This manual is divided into the following main categories:

- File systems
- Package management
- Network-related configuration
- System configuration
- System monitoring
- Kernel and Driver Configuration
- Security and Authentication
- Red Hat Training and Certification

This guide assumes you have a basic understanding of your Red Hat Enterprise Linux system. If you need help installing Red Hat Enterprise Linux, refer to the *Red Hat Enterprise Linux Installation Guide*.

1. Document Conventions

In this manual, certain words are represented in different fonts, typefaces, sizes, and weights. This highlighting is systematic; different words are represented in the same style to indicate their inclusion in a specific category. The types of words that are represented this way include the following:

`command`

> Linux commands (and other operating system commands, when used) are represented this way. This style should indicate to you that you can type the word or phrase on the command line and press **`Enter`** to invoke a command. Sometimes a command contains words that would be displayed in a different style on their own (such as file names). In these cases, they are considered to be part of the command, so the entire phrase is displayed as a command. For example:
>
> Use the **`cat testfile`** command to view the contents of a file, named **`testfile`**, in the current working directory.

`file name`

File names, directory names, paths, and RPM package names are represented this way. This style indicates that a particular file or directory exists with that name on your system. Examples:

The **`.bashrc`** file in your home directory contains bash shell definitions and aliases for your own use.

The **`/etc/fstab`** file contains information about different system devices and file systems.

Install the **`webalizer`** RPM if you want to use a Web server log file analysis program.

application

This style indicates that the program is an end-user application (as opposed to system software). For example:

Use **Mozilla** to browse the Web.

`key`

A key on the keyboard is shown in this style. For example:

To use **`Tab`** completion to list particular files in a directory, type **`ls`**, then a character, and finally the **`Tab`** key. Your terminal displays the list of files in the working directory that begin with that character.

`key+combination`

A combination of keystrokes is represented in this way. For example:

The **`Ctrl`**+**`Alt`**+**`Backspace`** key combination exits your graphical session and returns you to the graphical login screen or the console.

`text found on a GUI interface`

A title, word, or phrase found on a GUI interface screen or window is shown in this style. Text shown in this style indicates a particular GUI screen or an element on a GUI screen (such as text associated with a checkbox or field). Example:

Select the **`Require Password`** checkbox if you would like your screensaver to require a password before stopping.

top level of a menu on a GUI screen or window

A word in this style indicates that the word is the top level of a pulldown menu. If you click on the word on the GUI screen, the rest of the menu should appear. For example:

Under **File** on a GNOME terminal, the **New Tab** option allows you to open multiple shell prompts in the same window.

Instructions to type in a sequence of commands from a GUI menu look like the following example:

Go to **Applications** (the main menu on the panel) > **Programming** > **Emacs Text Editor** to start the **Emacs** text editor.

`button on a GUI screen or window`

This style indicates that the text can be found on a clickable button on a GUI screen. For example:

Click on the **`Back`** button to return to the webpage you last viewed.

`computer output`

Text in this style indicates text displayed to a shell prompt such as error messages and responses to commands. For example:

The **`ls`** command displays the contents of a directory. For example:

```
Desktop    about.html    logs     paulwesterberg.png
Mail     backupfiles    mail     reports
```

The output returned in response to the command (in this case, the contents of the directory) is shown in this style.

`prompt`

A prompt, which is a computer's way of signifying that it is ready for you to input something, is shown in this style. Examples:

`$`

`#`

`[stephen@maturin stephen]$`

`leopard login:`

`user input`

Text that the user types, either on the command line or into a text box on a GUI screen, is displayed in this style. In the following example, **`text`** is displayed in this style:

To boot your system into the text based installation program, you must type in the **`text`** command at the **`boot:`** prompt.

<replaceable>

Text used in examples that is meant to be replaced with data provided by the user is displayed in this style. In the following example, *<version-number>* is displayed in this style:

The directory for the kernel source is **`/usr/src/kernels/<version-number>/`**, where *<version-number>* is the version and type of kernel installed on this system.

Additionally, we use several different strategies to draw your attention to certain pieces of information. In order of urgency, these items are marked as a note, tip, important, caution, or warning. For example:

Note

Remember that Linux is case sensitive. In other words, a rose is not a ROSE is not a rOsE.

Note

The directory **`/usr/share/doc/`** contains additional documentation for packages installed on your system.

Important

If you modify the DHCP configuration file, the changes do not take effect until you restart the DHCP daemon.

Warning

Do not perform routine tasks as root — use a regular user account unless you need to use the root account for system administration tasks.

Warning

Be careful to remove only the necessary partitions. Removing other partitions could result in data loss or a corrupted system environment.

2. Send in Your Feedback

If you find an error in the *Red Hat Enterprise Linux Deployment Guide*, or if you have thought of a way to make this manual better, we would like to hear from you! Submit a report in Bugzilla (`http://bugzilla.redhat.com/bugzilla/`) against the component **`Deployment_Guide`**.

If you have a suggestion for improving the documentation, try to be as specific as possible. If you have found an error, include the section number and some of the surrounding text so we can find it easily.

Part I. File Systems

File system refers to the files and directories stored on a computer. A file system can have different formats called *file system types*. These formats determine how the information is stored as files and directories. Some file system types store redundant copies of the data, while some file system types make hard drive access faster. This part discusses the ext3, swap, RAID, and LVM file system types. It also discusses the **`parted`** utility to manage partitions and access control lists (ACLs) to customize file permissions.

Chapter 1. File System Structure

1.1. Why Share a Common Structure?

The file system structure is the most basic level of organization in an operating system. Almost all of the ways an operating system interacts with its users, applications, and security model are dependent upon the way it organizes files on storage devices. Providing a common file system structure ensures users and programs are able to access and write files.

File systems break files down into two logical categories:

- Shareable vs. unshareable files
- Variable vs. static files

Shareable files are those that can be accessed locally and by remote hosts; *unshareable* files are only available locally. *Variable* files, such as documents, can be changed at any time; *static* files, such as binaries, do not change without an action from the system administrator.

The reason for looking at files in this manner is to help correlate the function of the file with the permissions assigned to the directories which hold them. The way in which the operating system and its users interact with a given file determines the directory in which it is placed, whether that directory is mounted with read-only or read/write permissions, and the level of access each user has to that file. The top level of this organization is crucial. Access to the underlying directories can be restricted or security problems could manifest themselves if, from the top level down, it does not adhere to a rigid structure.

1.2. Overview of File System Hierarchy Standard (FHS)

Red Hat Enterprise Linux uses the *Filesystem Hierarchy Standard* (*FHS*) file system structure, which defines the names, locations, and permissions for many file types and directories.

The FHS document is the authoritative reference to any FHS-compliant file system, but the standard leaves many areas undefined or extensible. This section is an overview of the standard and a description of the parts of the file system not covered by the standard.

Compliance with the standard means many things, but the two most important are compatibility with other compliant systems and the ability to mount a **`/usr/`** partition as read-only. This second point is important because the directory contains common executables and should not be changed by users. Also, since the **`/usr/`** directory is mounted as read-only, it can be mounted from the CD-ROM or from another machine via a read-only NFS mount.

1.2.1. FHS Organization

The directories and files noted here are a small subset of those specified by the FHS document. Refer to the latest FHS document for the most complete information.

The complete standard is available online at http://www.pathname.com/fhs/.

1.2.1.1. The `/boot/` Directory

The **`/boot/`** directory contains static files required to boot the system, such as the Linux kernel. These files are essential for the system to boot properly.

Warning

Do not remove the **/boot/** directory. Doing so renders the system unbootable.

1.2.1.2. The /dev/ Directory

The **/dev/** directory contains device nodes that either represent devices that are attached to the system or virtual devices that are provided by the kernel. These device nodes are essential for the system to function properly. The **udev** daemon takes care of creating and removing all these device nodes in **/dev/**.

Devices in the **/dev** directory and subdirectories are either character (providing only a serial stream of input/output) or block (accessible randomly). Character devices include mouse, keyboard, modem while block devices include hard disk, floppy drive etc. If you have GNOME or KDE installed in your system, devices such as external drives or cds are automatically detected when connected (e.g via usb) or inserted (e.g via CD or DVD drive) and a popup window displaying the contents is automatically displayed. Files in the **/dev** directory are essential for the system to function properly.

Table 1.1. Examples of common files in the /dev

File	Description
/dev/hda	The master device on primary IDE channel.
/dev/hdb	The slave device on primary IDE channel.
/dev/tty0	The first virtual console.
/dev/tty1	The second virtual console.
/dev/sda	The first device on primary SCSI or SATA channel.
/dev/lp0	The first parallel port.

1.2.1.3. The /etc/ Directory

The **/etc/** directory is reserved for configuration files that are local to the machine. No binaries are to be placed in **/etc/**. Any binaries that were once located in **/etc/** should be placed into **/sbin/** or **/bin/**.

Examples of directories in **/etc** are the **X11/** and **skel/**:

```
/etc
   |- X11/
   |- skel/
```

The **/etc/X11/** directory is for X Window System configuration files, such as **xorg.conf**. The **/etc/skel/** directory is for "skeleton" user files, which are used to populate a home directory when a user is first created. Applications also store their configuration files in this directory and may reference them when they are executed.

1.2.1.4. The /lib/ Directory

The **/lib/** directory should contain only those libraries needed to execute the binaries in**/bin/** and **/sbin/**. These shared library images are particularly important for booting the system and executing commands within the root file system.

1.2.1.5. The /media/ Directory

The **`/media/`** directory contains subdirectories used as mount points for removable media such as usb storage media, DVDs, CD-ROMs, and Zip disks.

1.2.1.6. The `/mnt/` Directory

The **`/mnt/`** directory is reserved for temporarily mounted file systems, such as NFS file system mounts. For all removable media, please use the **`/media/`** directory. Automatically detected removable media will be mounted in the **`/media`** directory.

Note

The **`/mnt`** directory must not be used by installation programs.

1.2.1.7. The `/opt/` Directory

The **`/opt/`** directory provides storage for most application software packages.

A package placing files in the **`/opt/`** directory creates a directory bearing the same name as the package. This directory, in turn, holds files that otherwise would be scattered throughout the file system, giving the system administrator an easy way to determine the role of each file within a particular package.

For example, if **`sample`** is the name of a particular software package located within the **`/opt/`** directory, then all of its files are placed in directories inside the **`/opt/sample/`** directory, such as **`/opt/sample/bin/`** for binaries and **`/opt/sample/man/`** for manual pages.

Packages that encompass many different sub-packages, data files, extra fonts, clipart etc are also located in the **`/opt/`** directory, giving that large package a way to organize itself. In this way, our **`sample`** package may have different tools that each go in their own sub-directories, such as **`/opt/sample/tool1/`** and **`/opt/sample/tool2/`**, each of which can have their own **`bin/`**, **`man/`**, and other similar directories.

1.2.1.8. The `/proc/` Directory

The **`/proc/`** directory contains special files that either extract information from or send information to the kernel. Examples include system memory, cpu information, hardware configuration etc.

Due to the great variety of data available within **`/proc/`** and the many ways this directory can be used to communicate with the kernel, an entire chapter has been devoted to the subject. For more information, refer to Chapter 5, *The **`proc`** File System*.

1.2.1.9. The `/sbin/` Directory

The **`/sbin/`** directory stores executables used by the root user. The executables in **`/sbin/`** are used at boot time, for system administration and to perform system recovery operations. Of this directory, the FHS says:

> **`/sbin`** contains binaries essential for booting, restoring, recovering, and/or repairing the system in addition to the binaries in **`/bin`**. Programs executed after **`/usr/`** is known to be mounted (when there are no problems) are generally placed into **`/usr/sbin`**. Locally-installed system administration programs should be placed into **`/usr/local/sbin`**.

At a minimum, the following programs should be in **`/sbin/`**:

```
arp, clock,
halt, init,
fsck.*, grub,
ifconfig, mingetty,
mkfs.*, mkswap,
reboot, route,
shutdown, swapoff,
swapon
```

1.2.1.10. The `/srv/` Directory

The **`/srv/`** directory contains site-specific data served by your system running Red Hat Enterprise Linux. This directory gives users the location of data files for a particular service, such as FTP, WWW, or CVS. Data that only pertains to a specific user should go in the **`/home/`** directory.

1.2.1.11. The `/sys/` Directory

The **`/sys/`** directory utilizes the new **`sysfs`** virtual file system specific to the 2.6 kernel. With the increased support for hot plug hardware devices in the 2.6 kernel, the **`/sys/`** directory contains information similarly held in **`/proc/`**, but displays a hierarchical view of specific device information in regards to hot plug devices.

1.2.1.12. The `/usr/` Directory

The **`/usr/`** directory is for files that can be shared across multiple machines. The **`/usr/`** directory is often on its own partition and is mounted read-only. At a minimum, the following directories should be subdirectories of **`/usr/`**:

```
/usr
   |- bin/
   |- etc/
   |- games/
   |- include/
   |- kerberos/
   |- lib/
   |- libexec/
   |- local/
   |- sbin/
   |- share/
   |- src/
   |- tmp -> ../var/tmp/
```

Under the **`/usr/`** directory, the **`bin/`** subdirectory contains executables, **`etc/`** contains system-wide configuration files, **`games`** is for games, **`include/`** contains C header files, **`kerberos/`** contains binaries and other Kerberos-related files, and **`lib/`** contains object files and libraries that are not designed to be directly utilized by users or shell scripts. The **`libexec/`** directory contains small helper programs called by other programs, **`sbin/`** is for system administration binaries (those that do not belong in the **`/sbin/`** directory), **`share/`** contains files that are not architecture-specific, **`src/`** is for source code.

1.2.1.13. The `/usr/local/` Directory

The FHS says:

> The **`/usr/local`** hierarchy is for use by the system administrator when installing software locally. It needs to be safe from being overwritten when the system software is updated. It may be used for programs and data that are shareable among a group of hosts, but not found in **`/usr`**.

The **`/usr/local/`** directory is similar in structure to the **`/usr/`** directory. It has the following subdirectories, which are similar in purpose to those in the **`/usr/`** directory:

```
/usr/local
 |- bin/
 |- etc/
 |- games/
 |- include/
 |- lib/
 |- libexec/
 |- sbin/
 |- share/
 |- src/
```

In Red Hat Enterprise Linux, the intended use for the **`/usr/local/`** directory is slightly different from that specified by the FHS. The FHS says that **`/usr/local/`** should be where software that is to remain safe from system software upgrades is stored. Since software upgrades can be performed safely with *RPM Package Manager* (*RPM*), it is not necessary to protect files by putting them in **`/usr/local/`**. Instead, the **`/usr/local/`** directory is used for software that is local to the machine.

For instance, if the **`/usr/`** directory is mounted as a read-only NFS share from a remote host, it is still possible to install a package or program under the **`/usr/local/`** directory.

1.2.1.14. The `/var/` Directory

Since the FHS requires Linux to mount **`/usr/`** as read-only, any programs that write log files or need **`spool/`** or **`lock/`** directories should write them to the **`/var/`** directory. The FHS states **`/var/`** is for:

> ...variable data files. This includes spool directories and files, administrative and logging data, and transient and temporary files.

Below are some of the directories found within the **`/var/`** directory:

```
/var
   |- account/
   |- arpwatch/
   |- cache/
   |- crash/
   |- db/
   |- empty/
   |- ftp/
   |- gdm/
   |- kerberos/
   |- lib/
   |- local/
   |- lock/
   |- log/
   |- mail -> spool/mail/
   |- mailman/
   |- named/
```

```
     |- nis/
     |- opt/
     |- preserve/
     |- run/
     +- spool/
         |- at/
         |- clientmqueue/
         |- cron/
         |- cups/
         |- exim/
         |- lpd/
         |- mail/
         |- mailman/
         |- mqueue/
         |- news/
         |- postfix/
         |- repackage/
         |- rwho/
         |- samba/
         |- squid/
         |- squirrelmail/
         |- up2date/
         |- uucp
         |- uucppublic/
         |- vbox/
  |- tmp/
  |- tux/
  |- www/
  |- yp/
```

System log files, such as **`messages`** and **`lastlog`**, go in the **`/var/log/`** directory. The **`/var/lib/rpm/`** directory contains RPM system databases. Lock files go in the **`/var/lock/`** directory, usually in directories for the program using the file. The **`/var/spool/`** directory has subdirectories for programs in which data files are stored.

1.3. Special File Locations Under Red Hat Enterprise Linux

Red Hat Enterprise Linux extends the FHS structure slightly to accommodate special files.

Most files pertaining to RPM are kept in the **`/var/lib/rpm/`** directory. For more information on RPM, refer to the chapter Chapter 12, *Package Management with RPM*.

The **`/var/cache/yum/`** directory contains files used by the **Package Updater**, including RPM header information for the system. This location may also be used to temporarily store RPMs downloaded while updating the system. For more information about **Red Hat Network**, refer to Chapter 15, *Registering a System and Managing Subscriptions*.

Another location specific to Red Hat Enterprise Linux is the **`/etc/sysconfig/`** directory. This directory stores a variety of configuration information. Many scripts that run at boot time use the files in this directory. Refer to Chapter 32, *The **`sysconfig`** Directory* for more information about what is within this directory and the role these files play in the boot process.

Chapter 2. Using the `mount` Command

On Linux, UNIX, and similar operating systems, file systems on different partitions and removable devices like CDs, DVDs, or USB flash drives can be attached to a certain point (that is, the *mount point*) in the directory tree, and detached again. To attach or detach a file system, you can use the **`mount`** or **`umount`** command respectively. This chapter describes the basic usage of these commands, and covers some advanced topics such as moving a mount point or creating shared subtrees.

2.1. Listing Currently Mounted File Systems

To display all currently attached file systems, run the **`mount`** command with no additional arguments:

```
mount
```

This command displays the list of known mount points. Each line provides important information about the device name, the file system type, the directory in which it is mounted, and relevant mount options in the following form:

device on *directory* type *type* (*options*)

By default, the output includes various virtual file systems such as **`sysfs`**, **`tmpfs`**, and others. To display only the devices with a certain file system type, supply the **`-t`** option on the command line:

```
mount -t type
```

For a list of common file system types, refer to Table 2.1, “Common File System Types”. For an example on how to use the **`mount`** command to list the mounted file systems, see Example 2.1, “Listing Currently Mounted **`ext3`** File Systems”.

Example 2.1. Listing Currently Mounted `ext3` File Systems

Usually, both **`/`** and **`/boot`** partitions are formatted to use **`ext3`**. To display only the mount points that use this file system, type the following at a shell prompt:

```
~]$ mount -t ext3
/dev/mapper/VolGroup00-LogVol00 on / type ext3 (rw)
/dev/vda1 on /boot type ext3 (rw)
```

2.2. Mounting a File System

To attach a certain file system, use the **`mount`** command in the following form:

```
mount [option…] device directory
```

When the **`mount`** command is run, it reads the content of the **`/etc/fstab`** configuration file to see if the given file system is listed. This file contains a list of device names and the directory in which the selected file systems should be mounted, as well as the file system type and mount options. Because of this, when you are mounting a file system that is specified in this file, you can use one of the following variants of the command:

```
mount [option…] directory
mount [option…] device
```

Note that unless you are logged in as **root**, you must have permissions to mount the file system (see Section 2.2.2, “Specifying the Mount Options”).

2.2.1. Specifying the File System Type

In most cases, **mount** detects the file system automatically. However, there are certain file systems, such as **NFS** (Network File System) or **CIFS** (Common Internet File System), that are not recognized, and need to be specified manually. To specify the file system type, use the **mount** command in the following form:

```
mount -t type device directory
```

Table 2.1, “Common File System Types” provides a list of common file system types that can be used with the **mount** command. For a complete list of all available file system types, consult the relevant manual page as referred to in Section 2.4.1, “Installed Documentation”.

Table 2.1. Common File System Types

Type	Description
ext2	The **ext2** file system.
ext3	The **ext3** file system.
ext4	The **ext4** file system.
iso9660	The **ISO 9660** file system. It is commonly used by optical media, typically CDs.
jfs	The **JFS** file system created by IBM.
nfs	The **NFS** file system. It is commonly used to access files over the network.
nfs4	The **NFSv4** file system. It is commonly used to access files over the network.
ntfs	The **NTFS** file system. It is commonly used on machines that are running the Windows operating system.
udf	The **UDF** file system. It is commonly used by optical media, typically DVDs.
vfat	The **FAT** file system. It is commonly used on machines that are running the Windows operating system, and on certain digital media such as USB flash drives or floppy disks.

See Example 2.2, “Mounting a USB Flash Drive” for an example usage.

Example 2.2. Mounting a USB Flash Drive

Older USB flash drives often use the FAT file system. Assuming that such drive uses the **/dev/sdc1** device and that the **/media/flashdisk/** directory exists, you can mount it to this directory by typing the following at a shell prompt as **root**:

```
~]# mount -t vfat /dev/sdc1 /media/flashdisk
```

2.2.2. Specifying the Mount Options

To specify additional mount options, use the command in the following form:

```
mount -o options
```

When supplying multiple options, do not insert a space after a comma, or **mount** will incorrectly interpret the values following spaces as additional parameters.

Table 2.2, "Common Mount Options" provides a list of common mount options. For a complete list of all available options, consult the relevant manual page as referred to in Section 2.4.1, "Installed Documentation".

Table 2.2. Common Mount Options

Option	Description
async	Allows the asynchronous input/output operations on the file system.
auto	Allows the file system to be mounted automatically using the **mount -a** command.
defaults	Provides an alias for **async,auto,dev,exec,nouser,rw,suid**.
exec	Allows the execution of binary files on the particular file system.
loop	Mounts an image as a loop device.
noauto	Disallows the automatic mount of the file system using the **mount -a** command.
noexec	Disallows the execution of binary files on the particular file system.
nouser	Disallows an ordinary user (that is, other than **root**) to mount and unmount the file system.
remount	Remounts the file system in case it is already mounted.
ro	Mounts the file system for reading only.
rw	Mounts the file system for both reading and writing.
user	Allows an ordinary user (that is, other than **root**) to mount and unmount the file system.

See Example 2.3, "Mounting an ISO Image" for an example usage.

Example 2.3. Mounting an ISO Image

An ISO image (or a disk image in general) can be mounted by using the loop device. Assuming that the ISO image of the Fedora 14 installation disc is present in the current working directory and that the **/media/cdrom/** directory exists, you can mount the image to this directory by running the following command as **root**:

```
~]# mount -o ro,loop Fedora-14-x86_64-Live-Desktop.iso /media/cdrom
```

Note that ISO 9660 is by design a read-only file system.

2.2.3. Sharing Mounts

Occasionally, certain system administration tasks require access to the same file system from more than one place in the directory tree (for example, when preparing a chroot environment). To address such requirements, the **mount** command implements the **--bind** option that provides a means for duplicating certain mounts. Its usage is as follows:

```
mount --bind old_directory new_directory
```

Although the above command allows a user to access the file system from both places, it does not apply on the file systems that are mounted within the original directory. To include these mounts as well, type:

```
mount --rbind old_directory new_directory
```

Additionally, to provide as much flexibility as possible, Red Hat Enterprise Linux 5.10 implements the functionality known as *shared subtrees*. This feature allows you to use the following four mount types:

Shared Mount

A shared mount allows you to create an exact replica of a given mount point. When a shared mount is created, any mount within the original mount point is reflected in it, and vice versa. To create a shared mount, type the following at a shell prompt:

```
mount --make-shared mount_point
```

Alternatively, you can change the mount type for the selected mount point and all mount points under it:

```
mount --make-rshared mount_point
```

See Example 2.4, “Creating a Shared Mount Point” for an example usage.

Example 2.4. Creating a Shared Mount Point

There are two places where other file systems are commonly mounted: the **/media** directory for removable media, and the **/mnt** directory for temporarily mounted file systems. By using a shared mount, you can make these two directories share the same content. To do so, as **root**, mark the **/media** directory as “shared”:

```
~]# mount --bind /media /media
~]# mount --make-shared /media
```

Then create its duplicate in **/mnt** by using the following command:

```
~]# mount --bind /media /mnt
```

You can now verify that a mount within **/media** also appears in **/mnt**. For example, if you have non-empty media in your CD-ROM drive and the **/media/cdrom/** directory exists, run the following commands:

```
~]# mount /dev/cdrom /media/cdrom
~]# ls /media/cdrom
EFI  GPL  isolinux  LiveOS
~]# ls /mnt/cdrom
EFI  GPL  isolinux  LiveOS
```

Similarly, you can verify that any file system mounted in the **/mnt** directory is reflected in **/media**. For instance, if you have a non-empty USB flash drive that uses the **/dev/sdc1** device plugged in and the **/mnt/flashdisk/** directory is present, type:

```
~]# mount /dev/sdc1 /mnt/flashdisk
~]# ls /media/flashdisk
en-US  publican.cfg
~]# ls /mnt/flashdisk
en-US  publican.cfg
```

Slave Mount

A slave mount allows you to create a limited duplicate of a given mount point. When a slave mount is created, any mount within the original mount point is reflected in it, but no mount within a slave mount is reflected in its original. To create a slave mount, type the following at a shell prompt:

```
mount --make-slave mount_point
```

Alternatively, you can change the mount type for the selected mount point and all mount points under it:

```
mount --make-rslave mount_point
```

See Example 2.5, “Creating a Slave Mount Point” for an example usage.

Example 2.5. Creating a Slave Mount Point

Imagine you want the content of the **/media** directory to appear in **/mnt** as well, but you do not want any mounts in the **/mnt** directory to be reflected in **/media**. To do so, as **root**, first mark the **/media** directory as “shared”:

```
~]# mount --bind /media /media
~]# mount --make-shared /media
```

Then create its duplicate in **/mnt**, but mark it as “slave”:

```
~]# mount --bind /media /mnt
~]# mount --make-slave /mnt
```

You can now verify that a mount within **/media** also appears in **/mnt**. For example, if you have non-empty media in your CD-ROM drive and the **/media/cdrom/** directory exists, run the following commands:

```
~]# mount /dev/cdrom /media/cdrom
~]# ls /media/cdrom
EFI  GPL  isolinux  LiveOS
~]# ls /mnt/cdrom
EFI  GPL  isolinux  LiveOS
```

You can also verify that file systems mounted in the **/mnt** directory are *not* reflected in **/media**. For instance, if you have a non-empty USB flash drive that uses the **/dev/sdc1** device plugged in and the **/mnt/flashdisk/** directory is present, type: :

```
~]# mount /dev/sdc1 /mnt/flashdisk
~]# ls /media/flashdisk
~]# ls /mnt/flashdisk
en-US  publican.cfg
```

Private Mount

A private mount allows you to create an ordinary mount. When a private mount is created, no subsequent mounts within the original mount point are reflected in it, and no mount within a private mount is reflected in its original. To create a private mount, type the following at a shell prompt:

```
mount --make-private mount_point
```

Alternatively, you can change the mount type for the selected mount point and all mount points under it:

```
mount --make-rprivate mount_point
```

See Example 2.6, “Creating a Private Mount Point” for an example usage.

Example 2.6. Creating a Private Mount Point

Taking into account the scenario in Example 2.4, “Creating a Shared Mount Point”, assume that you have previously created a shared mount point by using the following commands as **root**:

```
~]# mount --bind /media /media
~]# mount --make-shared /media
~]# mount --bind /media /mnt
```

To mark the **/mnt** directory as “private”, type:

```
~]# mount --make-private /mnt
```

You can now verify that none of the mounts within **/media** appears in **/mnt**. For example, if you have non-empty media in your CD-ROM drive and the **/media/cdrom/** directory exists, run the following commands:

```
~]# mount /dev/cdrom /media/cdrom
~]# ls /media/cdrom
EFI  GPL  isolinux  LiveOS
~]# ls /mnt/cdrom
~]#
```

You can also verify that file systems mounted in the **/mnt** directory are not reflected in **/media**. For instance, if you have a non-empty USB flash drive that uses the **/dev/sdc1** device plugged in and the **/mnt/flashdisk/** directory is present, type:

```
~]# mount /dev/sdc1 /mnt/flashdisk
~]# ls /media/flashdisk
~]# ls /mnt/flashdisk
en-US  publican.cfg
```

Unbindable Mount

An unbindable mount allows you to prevent a given mount point from being duplicated whatsoever. To create an unbindable mount, type the following at a shell prompt:

```
mount --make-unbindable mount_point
```

Alternatively, you can change the mount type for the selected mount point and all mount points under it:

```
mount --make-runbindable mount_point
```

See Example 2.7, “Creating an Unbindable Mount Point” for an example usage.

Example 2.7. Creating an Unbindable Mount Point

To prevent the **`/media`** directory from being shared, as **`root`**, type the following at a shell prompt:

```
~]# mount --bind /media /media
~]# mount --make-unbindable /media
```

This way, any subsequent attempt to make a duplicate of this mount will fail with an error:

```
~]# mount --bind /media /mnt
mount: wrong fs type, bad option, bad superblock on /media/,
       missing code page or other error
       In some cases useful info is found in syslog - try
       dmesg | tail  or so
```

2.2.4. Moving a Mount Point

To change the directory in which a file system is mounted, use the following command:

```
mount --move old_directory new_directory
```

See Example 2.8, “Moving an Existing NFS Mount Point” for an example usage.

Example 2.8. Moving an Existing NFS Mount Point

Imagine that you have an NFS storage that contains user directories. Assuming that this storage is already mounted in **`/mnt/userdirs/`**, as **`root`**, you can move this mount point to **`/home`** by using the following command:

```
~]# mount --move /mnt/userdirs /home
```

To verify the mount point has been moved, list the content of both directories:

```
~]# ls /mnt/userdirs
~]# ls /home
jill  joe
```

2.3. Unmounting a File System

To detach a previously mounted file system, use either of the following variants of the **umount** command:

```
umount directory
umount device
```

Note that unless you are logged in as **root**, you must have permissions to unmount the file system (see Section 2.2.2, “Specifying the Mount Options”). See Example 2.9, “Unmounting a CD” for an example usage.

Important

When a file system is in use (for example, when a process is reading a file on this file system), running the **umount** command will fail with an error. To determine which processes are accessing the file system, use the **fuser** command in the following form:

```
fuser -m directory
```

For example, to list the processes that are accessing a file system mounted to the **/media/cdrom/** directory, type:

```
~]$ fuser -m /media/cdrom
/media/cdrom:          1793  2013  2022  2435 10532c 10672c
```

Example 2.9. Unmounting a CD

To unmount a CD that was previously mounted to the **/media/cdrom/** directory, type the following at a shell prompt:

```
~]$ umount /media/cdrom
```

2.4. Additional Resources

The following resources provide an in-depth documentation on the subject.

2.4.1. Installed Documentation

- **man 8 mount** — The manual page for the **mount** command that provides a full documentation on its usage.
- **man 8 umount** — The manual page for the **umount** command that provides a full documentation on its usage.
- **man 5 fstab** — The manual page providing a thorough description of the **/etc/fstab** file format.

2.4.2. Useful Websites

- *Shared subtrees* — An LWN article covering the concept of shared subtrees.
- *sharedsubtree.txt* — Extensive documentation that is shipped with the shared subtrees patches.

Chapter 3. The ext3 File System

The default file system is the journaling *ext3* file system.

3.1. Features of ext3

The ext3 file system is essentially an enhanced version of the ext2 file system. These improvements provide the following advantages:

Availability

After an unexpected power failure or system crash (also called an *unclean system shutdown*), each mounted ext2 file system on the machine must be checked for consistency by the **`e2fsck`** program. This is a time-consuming process that can delay system boot time significantly, especially with large volumes containing a large number of files. During this time, any data on the volumes is unreachable.

The journaling provided by the ext3 file system means that this sort of file system check is no longer necessary after an unclean system shutdown. The only time a consistency check occurs using ext3 is in certain rare hardware failure cases, such as hard drive failures. The time to recover an ext3 file system after an unclean system shutdown does not depend on the size of the file system or the number of files; rather, it depends on the size of the *journal* used to maintain consistency. The default journal size takes about a second to recover, depending on the speed of the hardware.

Data Integrity

The ext3 file system prevents loss of data integrity in the event that an unclean system shutdown occurs. The ext3 file system allows you to choose the type and level of protection that your data receives. By default, the ext3 volumes are configured to keep a high level of data consistency with regard to the state of the file system.

Speed

Despite writing some data more than once, ext3 has a higher throughput in most cases than ext2 because ext3's journaling optimizes hard drive head motion. You can choose from three journaling modes to optimize speed, but doing so means trade-offs in regards to data integrity if the system was to fail.

Easy Transition

It is easy to migrate from ext2 to ext3 and gain the benefits of a robust journaling file system without reformatting. Refer to Section 3.3, “Converting to an ext3 File System” for more on how to perform this task.

The following sections walk you through the steps for creating and tuning ext3 partitions. For ext2 partitions, skip the partitioning and formatting sections below and go directly to Section 3.3, “Converting to an ext3 File System”.

3.2. Creating an ext3 File System

After installation, it is sometimes necessary to create a new ext3 file system. For example, if you add a new disk drive to the system, you may want to partition the drive and use the ext3 file system.

The steps for creating an ext3 file system are as follows:

1. Format the partition with the ext3 file system using **`mkfs`**.
2. Label the partition using **`e2label`**.

3.3. Converting to an ext3 File System

The **`tune2fs`** allows you to convert an **`ext2`** filesystem to **`ext3`**.

Note

Always use the **`e2fsck`** utility to check your filesystem before and after using **`tune2fs`**. A default installation of Red Hat Enterprise Linux uses ext3 for all file systems.

To convert an **`ext2`** filesystem to **`ext3`**, log in as root and type the following command in a terminal:

```
tune2fs -j <block_device>
```

where *<block_device>* contains the ext2 filesystem you wish to convert.

A valid block device could be one of two types of entries:

- A mapped device — A logical volume in a volume group, for example, **`/dev/mapper/VolGroup00-LogVol02`**.
- A static device — A traditional storage volume, for example, **`/dev/hdbX`**, where *hdb* is a storage device name and *X* is the partition number.

Issue the **`df`** command to display mounted file systems.

For the remainder of this section, the sample commands use the following value for the block device:

```
/dev/mapper/VolGroup00-LogVol02
```

You must recreate the initrd image so that it will contain the ext3 kernel module. To create this, run the **`mkinitrd`** program. For information on using the **`mkinitrd`** command, type **`man mkinitrd`**. Also, make sure your GRUB configuration loads the **`initrd`**.

If you fail to make this change, the system still boots, but the file system is mounted as ext2 instead of ext3.

3.4. Reverting to an ext2 File System

If you wish to revert a partition from ext3 to ext2 for any reason, you must first unmount the partition by logging in as root and typing,

```
umount /dev/mapper/VolGroup00-LogVol02
```

Next, change the file system type to ext2 by typing the following command as root:

```
tune2fs -O ^has_journal /dev/mapper/VolGroup00-LogVol02
```

Check the partition for errors by typing the following command as root:

```
e2fsck -y /dev/mapper/VolGroup00-LogVol02
```

Then mount the partition again as ext2 file system by typing:

```
mount -t ext2 /dev/mapper/VolGroup00-LogVol02 /mount/point
```

In the above command, replace */mount/point* with the mount point of the partition.

Next, remove the **`.journal`** file at the root level of the partition by changing to the directory where it is mounted and typing:

```
rm -f .journal
```

You now have an ext2 partition.

If you want to permanently change the partition to ext2, remember to update the **`/etc/fstab`** file.

Chapter 4. The ext4 File System

4.1. Features of ext4

The ext4 file system is a scalable extension of the ext3 file system, which is the default file system of Red Hat Enterprise Linux 5. The ext4 file system can support files and file systems of up to 16 terabytes in size. It also supports an unlimited number of sub-directories (the ext3 file system only supports up to 32,000), though once the link count exceeds 65,000 it resets to 1 and is no longer increased. The following are the most important features of ext4:

Main Features

The ext4 file system uses extents (as opposed to the traditional block mapping scheme used by ext2 and ext3), which improves performance when using large files and reduces metadata overhead for large files. In addition, ext4 also labels unallocated block groups and inode table sections accordingly, which allows them to be skipped during a file system check. This makes for quicker file system checks, which becomes more beneficial as the file system grows in size.

Allocation Features

The ext4 file system features the following allocation schemes:

- Persistent pre-allocation
- Delayed allocation
- Multi-block allocation
- Stripe-aware allocation

Because of delayed allocation and other performance optimizations, ext4's behavior of writing files to disk is different from ext3. In ext4, a program's writes to the file system are not guaranteed to be on-disk unless the program issues an **`fsync()`** call afterwards.

By default, ext3 automatically forces newly created files to disk almost immediately even without **`fsync()`**. This behavior hid bugs in programs that did not use **`fsync()`** to ensure that written data was on-disk. The ext4 file system, on the other hand, often waits several seconds to write out changes to disk, allowing it to combine and reorder writes for better disk performance than ext3.

Warning

Unlike ext3, the ext4 file system does not force data to disk on transaction commit. As such, it takes longer for buffered writes to be flushed to disk. As with any file system, use data integrity calls such as **`fsync()`** to ensure that data is written to permanent storage.

Other ext4 Features

The ext4 file system also supports the following:

- *Extended attributes* (**`xattr`**), which allows the system to associate several additional name/value pairs per file.
- *Quota journaling*, which avoids the need for lengthy quota consistency checks after a crash.

Note

The only supported journaling mode in ext4 is **`data=ordered`** (default).

- *Subsecond timestamps*, which allow to specify inode timestamp fields in nanosecond resolution.

4.2. Managing an ext4 File System

In order to manage ext4 file systems on Red Hat Eterprise Linux 5, it is necessary to install the *e4fsprogs* package. You can use the Yum utility to install the package:

```
~]# yum install e4fsprogs
```

The *e4fsprogs* package contains renamed static binaries from the equivalent upstream *e2fsprogs* release. This has been done to ensure stability of the *e2fsprogs* core utilities with all the changes for ext4 included. The most important of these utilities are:

- **`mke4fs`** — A utility used to create an ext4 file system.
- **`mkfs.ext4`** — Another command used to create an ext4 file system.
- **`e4fsck`** — A utility used to repair inconsistencies of an ext4 file system.
- **`tune4fs`** — A utility used to modify ext4 file system attributes.
- **`resize4fs`** — A utility used to resize an ext4 file system.
- **`e4label`** — A utility used to display or modify the label of the ext4 file system.
- **`dumpe4fs`** — A utility used to display the super block and blocks group information for the ext4 file system.
- **`debuge4fs`** — An interactive file system debugger, used to examine ext4 file systems, manually repair corrupted file systems and create test cases for **`e4fsck`**.

The following sections walk you through the steps for creating and tuning ext4 partitions.

4.3. Creating an ext4 File System

After installation, it is sometimes necessary to create a new ext4 file system. For example, if you add a new disk drive to the system, you may want to partition the drive and use the ext4 file system.

The default options are optimal for most usage scenarios but if you need to set your ext4 file system in a specific way, see manual pages for the **`mke4fs`** and **`mkfs.ext4`** commands for available options. Also, you may want to examine and modify the configuration file of **`mke4fs`**, **`/etc/mke4fs.conf`**, if you plan to create ext4 file systems more often.

The steps for creating an ext4 file system are as follows:

1. Format the partition with the ext4 file system using the **`mkfs.ext4`** or **`mke4fs`** command:

   ```
   ~]# mkfs.ext4 block_device
   ```

```
~]# mke4fs -t ext4 block_device
```

where *block_device* is a partition which will contain the ext4 filesystem you wish to create.

2. Label the partition using the **e4label** command.

```
~]# e4label <block_device> new-label
```

3. Create a mount point and mount the new file system to that mount point:

```
~]# mkdir /mount/point
~]# mount block_device /mount/point
```

A valid block device could be one of two types of entries:

- A *mapped device* — A logical volume in a volume group, for example, **/dev/mapper/VolGroup00-LogVol02**.
- A *static device* — A traditional storage volume, for example, ***/dev/hdbX***, where *hdb* is a storage device name and *X* is the partition number.

For striped block devices (for example RAID5 arrays), the stripe geometry can be specified at the time of file system creation. Using proper stripe geometry greatly enhances performance of an ext4 file system.

When creating file systems on lvm or md volumes, **mkfs.ext4** chooses an optimal geometry. This may also be true on some hardware RAIDs which export geometry information to the operating system.

To specify stripe geometry, use the **-E** option of **mkfs.ext4** (that is, extended file system options) with the following sub-options:

stride=*value*

Specifies the RAID chunk size.

stripe-width=*value*

Specifies the number of data disks in a RAID device, or the number of stripe units in the stripe.

For both sub-options, ***value*** must be specified in file system block units. For example, to create a file system with a 64k stride (that is, 16 x 4096) on a 4k-block file system, use the following command:

```
~]# mkfs.ext4 -E stride=16,stripe-width=64 block_device
```

For more information about creating file systems, refer to **man mkfs.ext4**.

4.4. Mounting an ext4 File System

An ext4 file system can be mounted with no extra options, same as any other file system:

```
~]# mount block_device /mount/point
```

The default mount options are optimal for most users. Options, such as **acl**, **noacl**, **data**, **quota**, **noquota**, **user_xattr**, **nouser_xattr**, and many others that were already used with the ext2 and ext3 file systems, are backward compatible and have the same usage and functionality. Also, with the ext4 file system, several new ext4-specific mount options have been added, for example:

barrier / nobarrier

By default, ext4 uses write barriers to ensure file system integrity even when power is lost to a device with write caches enabled. For devices without write caches, or with battery-backed write caches, you disable barriers using the **`nobarrier`** option:

```
~]# mount -o nobarrier block_device /mount/point
```

stripe=*value*

This option allows you to specify the number of file system blocks allocated for a single file operation. For RAID5 this number should be equal the RAID chunk size multiplied by the number of disks.

journal_ioprio=*value*

This option allows you to set priority of I/O operations submitted during a commit operation. The option can have a value from 7 to 0 (0 is the highest priority), and is set to 3 by default, which is slightly higher priority than the default I/O priority.

Default mount options can be also set in the file system superblock using the **`tune4fs`** utility. For example, the following command sets the file system on the **`/dev/mapper/VolGroup00-LogVol02`** device to be mounted by default with debugging disabled and user-specified extended attributes and Posix access control lists enabled:

```
~]# tune4fs -o ^debug,user_xattr,acl /dev/mapper/VolGroup00-LogVol02
```

For more information on this topic, refer to the **`tune4fs`**(8) manual page.

An ext3 file system can also be mounted as ext4 without changing the format, allowing it to be mounted as ext3 again in the future. To do so, run the following command on a block device that contains an ext3 file system:

```
~]# mount -t ext4 block_device /mount/point
```

Doing so will only allow the ext3 file system to use ext4-specific features that do not require a file format conversion. These features include **`delayed allocation`** and **`multi-block allocation`**, and exclude features such as **`extent mapping`**.

Warning

Using the ext4 driver to mount an ext3 file system has not been fully tested on Red Hat Enterprise Linux 5. Therefore, this action is *not supported* because Red Hat cannot guarantee consistent performance and predictable behavior for ext3 file systems in this way.

For more information on mount options for the ext4 file system, see Section 2.2.2, “Specifying the Mount Options” and the **`mount`**(8) manual page.

Note

If you want to enable persistent mounting of the file system, remember to update the **`/etc/fstab`** file accordingly. For example:

```
/dev/mapper/VolGroup00-LogVol02    /test    ext4    defaults    0 0
```

4.5. Resizing an ext4 File System

Before growing an ext4 file system, ensure that the underlying block device is of an appropriate size to hold the file system later. Use the appropriate resizing methods for the affected block device.

When grown, the ext4 filesystem can be mounted. When shrunk, the ext4 file system has to be *unmounted*. You can resize an ext4 file system using the **`resize4fs`** command:

```
~]# resize4fs block_devicenew_size
```

When resizing an ext4 file system, the **`resize2fs`** utility reads the size in units of file system block size, unless a suffix indicating a specific unit is used. The following suffixes indicate specific units:

- **`s`** — 512 byte sectors
- **`K`** — kilobytes
- **`M`** — megabytes
- **`G`** — gigabytes

The **`size`** parameter is optional (and often redundant) when expanding. The **`resize4fs`** automatically expands to fill all available space of the container, usually a logical volume or partition. For more information about resizing an ext4 file system, refer to the **`resize4fs`**(8) manual page.

Chapter 5. The `proc` File System

The Linux kernel has two primary functions: to control access to physical devices on the computer and to schedule when and how processes interact with these devices. The **`/proc/`** directory — also called the **`proc`** file system — contains a hierarchy of special files which represent the current state of the kernel — allowing applications and users to peer into the kernel's view of the system.

Within the **`/proc/`** directory, one can find a wealth of information detailing the system hardware and any processes currently running. In addition, some of the files within the **`/proc/`** directory tree can be manipulated by users and applications to communicate configuration changes to the kernel.

5.1. A Virtual File System

Under Linux, all data are stored as files. Most users are familiar with the two primary types of files: text and binary. But the **`/proc/`** directory contains another type of file called a *virtual file*. It is for this reason that **`/proc/`** is often referred to as a *virtual file system*.

These virtual files have unique qualities. Most of them are listed as zero bytes in size and yet when one is viewed, it can contain a large amount of information. In addition, most of the time and date settings on virtual files reflect the current time and date, indicative of the fact they are constantly updated.

Virtual files such as **`/proc/interrupts`**, **`/proc/meminfo`**, **`/proc/mounts`**, and **`/proc/partitions`** provide an up-to-the-moment glimpse of the system's hardware. Others, like the **`/proc/filesystems`** file and the **`/proc/sys/`** directory provide system configuration information and interfaces.

For organizational purposes, files containing information on a similar topic are grouped into virtual directories and sub-directories. For instance, **`/proc/ide/`** contains information for all physical IDE devices. Likewise, process directories contain information about each running process on the system.

5.1.1. Viewing Virtual Files

By using the **`cat`**, **`more`**, or **`less`** commands on files within the **`/proc/`** directory, users can immediately access enormous amounts of information about the system. For example, to display the type of CPU a computer has, type **`cat /proc/cpuinfo`** to receive output similar to the following:

```
processor : 0
vendor_id : AuthenticAMD
cpu family : 5
model  : 9
model name : AMD-K6(tm) 3D+
Processor stepping : 1 cpu
MHz  : 400.919
cache size : 256 KB
fdiv_bug : no
hlt_bug  : no
f00f_bug : no
coma_bug : no
fpu  : yes
fpu_exception : yes
cpuid level : 1
wp  : yes
flags  : fpu vme de pse tsc msr mce cx8 pge mmx syscall 3dnow k6_mtrr
bogomips : 799.53
```

When viewing different virtual files in the **/proc/** file system, some of the information is easily understandable while some is not human-readable. This is in part why utilities exist to pull data from virtual files and display it in a useful way. Examples of these utilities include **lspci**, **apm**, **free**, and **top**.

Note

Some of the virtual files in the **/proc/** directory are readable only by the root user.

5.1.2. Changing Virtual Files

As a general rule, most virtual files within the **/proc/** directory are read-only. However, some can be used to adjust settings in the kernel. This is especially true for files in the **/proc/sys/** subdirectory.

To change the value of a virtual file, use the **echo** command and a greater than symbol (**>**) to redirect the new value to the file. For example, to change the hostname on the fly, type:

```
echo www.example.com > /proc/sys/kernel/hostname
```

Other files act as binary or Boolean switches. Typing **cat /proc/sys/net/ipv4/ip_forward** returns either a **0** or a **1**. A **0** indicates that the kernel is not forwarding network packets. Using the**echo** command to change the value of the **ip_forward** file to **1** immediately turns packet forwarding on.

Note

Another command used to alter settings in the **/proc/sys/** subdirectory is **/sbin/sysctl**. For more information on this command, refer to Section 5.4, “Using the **sysctl** Command”

For a listing of some of the kernel configuration files available in the **/proc/sys/** subdirectory, refer to Section 5.3.9, “ **/proc/sys/** ”.

5.1.3. Restricting Access to Process Directories

On multi-user systems, it is often useful to secure the process directories stored in **/proc/** so that they can be viewed only by the **root** user. You can restrict the access to these directories with the use of the **hidepid** option.

To change the file system parameters, you can use the **mount** command with the **-o remount** option. As **root**, type:

```
mount -o remount,hidepid=value /proc
```

Here, *value* passed to **hidepid** is one of:

- **0** (*default*) — every user can read all world-readable files stored in a process directory.
- **1** — users can access only their own process directories. This protects the sensitive files like **cmdline**, **sched**, or **status** from access by non-root users. This setting does not affect the actual file permissions.
- **2** — process files are invisible to non-root users. The existence of a process can be learned by other means, but its effective UID and GID is hidden. Hiding these IDs complicates an intruder's task of

gathering information about running processes.

Example 5.1. Restricting access to process directories

To make process files accessible only to the **root** user, type:

```
~]# mount -o remount,hidepid=1 /proc
```

With **hidepid**=**1**, a non-root user cannot access the contents of process directories. An attempt to do so fails with the following message:

```
~]$ ls /proc/1/
ls: /proc/1/: Operation not permitted
```

With **hidepid**=**2** enabled, process directories are made invisible to non-root users:

```
~]$ ls /proc/1/
ls: /proc/1/: No such file or directory
```

Also, you can specify a user group that will have access to process files even when **hidepid** is set to 1 or 2. To do this, use the **gid** option. As **root**, type:

```
mount -o remount,hidepid=value,gid=gid /proc
```

Replace *gid* with the specific group id. For members of selected group, the process files will act as if **hidepid** was set to 0. However, users which are not supposed to monitor the tasks in the whole system should not be added to the group. For more information on managing users and groups see Chapter 37, *Users and Groups*.

5.2. Top-level Files within the proc File System

Below is a list of some of the more useful virtual files in the top-level of the **/proc/** directory.

Note

In most cases, the content of the files listed in this section are not the same as those installed on your machine. This is because much of the information is specific to the hardware on which Red Hat Enterprise Linux is running for this documentation effort.

5.2.1. /proc/apm

This file provides information about the state of the *Advanced Power Management (APM)* system and is used by the **apm** command. If a system with no battery is connected to an AC power source, this virtual file would look similar to the following:

```
1.16 1.2 0x07 0x01 0xff 0x80 -1% -1 ?
```

Running the **apm -v** command on such a system results in output similar to the following:

```
APM BIOS 1.2 (kernel driver 1.16ac) AC on-line, no system battery
```

For systems which do not use a battery as a power source, **apm** is able do little more than put the machine in standby mode. The **apm** command is much more useful on laptops. For example, the following output is from the command **cat /proc/apm** on a laptop while plugged into a power outlet:

```
1.16 1.2 0x03 0x01 0x03 0x09 100% -1 ?
```

When the same laptop is unplugged from its power source for a few minutes, the content of the **apm** file changes to something like the following:

```
1.16 1.2 0x03 0x00 0x00 0x01 99% 1792 min
```

The **apm -v** command now yields more useful data, such as the following:

```
APM BIOS 1.2 (kernel driver 1.16) AC off-line, battery status high: 99% (1
day, 5:52)
```

5.2.2. /proc/buddyinfo

This file is used primarily for diagnosing memory fragmentation issues. Using the buddy algorithm, each column represents the number of pages of a certain order (a certain size) that are available at any given time. For example, for zone DMA (direct memory access), there are 90 of 2^(0*PAGE_SIZE) chunks of memory. Similarly, there are 6 of 2^(1*PAGE_SIZE) chunks, and 2 of 2^(2*PAGE_SIZE) chunks of memory available.

The **DMA** row references the first 16 MB on a system, the **HighMem** row references all memory greater than 4 GB on a system, and the **Normal** row references all memory in between.

The following is an example of the output typical of **/proc/buddyinfo**:

```
Node 0, zone      DMA     90      6      2      1      1      ...
Node 0, zone   Normal   1650    310      5      0      0      ...
Node 0, zone  HighMem      2      0      0      1      1      ...
```

5.2.3. /proc/cmdline

This file shows the parameters passed to the kernel at the time it is started. A sample **/proc/cmdline** file looks like the following:

```
ro root=/dev/VolGroup00/LogVol00 rhgb quiet 3
```

This output tells us the following:

ro

The root device is mounted read-only at boot time. The presence of **ro** on the kernel boot line overrides any instances of **rw**.

root=/dev/VolGroup00/LogVol00

This tells us on which disk device or, in this case, on which logical volume, the root filesystem image is located. With our sample **/proc/cmdline** output, the root filesystem image is located on the first logical volume (**`LogVol00`**) of the first LVM volume group (**`VolGroup00`**). On a system not using Logical Volume Management, the root file system might be located on **`/dev/sda1`** or **`/dev/sda2`**, meaning on either the first or second partition of the first SCSI or SATA disk drive, depending on whether we have a separate (preceding) boot or swap partition on that drive.

For more information on LVM used in Red Hat Enterprise Linux, refer to http://www.tldp.org/HOWTO/LVM-HOWTO/index.html.

rhgb

A short lowercase acronym that stands for *Red Hat Graphical Boot*, providing "rhgb" on the kernel command line signals that graphical booting is supported, assuming that **`/etc/inittab`** shows that the default runlevel is set to 5 with a line like this:

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

quiet

Indicates that all verbose kernel messages except those which are extremely serious should be suppressed at boot time.

5.2.4. `/proc/cpuinfo`

This virtual file identifies the type of processor used by your system. The following is an example of the output typical of **`/proc/cpuinfo`**:

```
processor : 0
vendor_id : GenuineIntel
cpu family : 15
model  : 2
model name : Intel(R) Xeon(TM) CPU 2.40GHz
stepping : 7 cpu
MHz  : 2392.371
cache size : 512 KB
physical id : 0
siblings : 2
runqueue : 0
fdiv_bug : no
hlt_bug  : no
f00f_bug : no
coma_bug : no
fpu  : yes
fpu_exception : yes
cpuid level : 2
wp  : yes
flags  : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca  cmov pat
pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm
bogomips : 4771.02
```

- **`processor`** — Provides each processor with an identifying number. On systems that have one processor, only a **`0`** is present.

- **`cpu family`** — Authoritatively identifies the type of processor in the system. For an Intel-based system, place the number in front of "86" to determine the value. This is particularly helpful for those attempting to identify the architecture of an older system such as a 586, 486, or 386. Because some RPM packages are compiled for each of these particular architectures, this value also helps users determine which packages to install.

- **`model name`** — Displays the common name of the processor, including its project name.
- **`cpu MHz`** — Shows the precise speed in megahertz for the processor to the thousandths decimal place.
- **`cache size`** — Displays the amount of level 2 memory cache available to the processor.
- **`siblings`** — Displays the number of sibling CPUs on the same physical CPU for architectures which use hyper-threading.
- **`flags`** — Defines a number of different qualities about the processor, such as the presence of a floating point unit (FPU) and the ability to process MMX instructions.

5.2.5. `/proc/crypto`

This file lists all installed cryptographic ciphers used by the Linux kernel, including additional details for each. A sample **`/proc/crypto`** file looks like the following:

```
name          : sha1
module        : kernel
type          : digest
blocksize     : 64
digestsize    : 20
name          : md5
module        : md5
type          : digest
blocksize     : 64
digestsize    : 16
```

5.2.6. `/proc/devices`

This file displays the various character and block devices currently configured (not including devices whose modules are not loaded). Below is a sample output from this file:

```
Character devices:
  1 mem
  4 /dev/vc/0
  4 tty
  4 ttyS
  5 /dev/tty
  5 /dev/console
  5 /dev/ptmx
  7 vcs
  10 misc
  13 input
  29 fb
  36 netlink
  128 ptm
  136 pts
  180 usb

Block devices:
  1 ramdisk
  3 ide0
```

```
    9 md
    22 ide1
    253 device-mapper
    254 mdp
```

The output from **`/proc/devices`** includes the major number and name of the device, and is broken into two major sections: **`Character devices`** and **`Block devices`**.

Character devices are similar to *block devices*, except for two basic differences:

1. Character devices do not require buffering. Block devices have a buffer available, allowing them to order requests before addressing them. This is important for devices designed to store information — such as hard drives — because the ability to order the information before writing it to the device allows it to be placed in a more efficient order.

2. Character devices send data with no preconfigured size. Block devices can send and receive information in blocks of a size configured per device.

For more information about devices refer to the following installed documentation:

```
/usr/share/doc/kernel-doc-<version>/Documentation/devices.txt
```

5.2.7. `/proc/dma`

This file contains a list of the registered ISA DMA channels in use. A sample **`/proc/dma`** files looks like the following:

```
4: cascade
```

5.2.8. `/proc/execdomains`

This file lists the *execution domains* currently supported by the Linux kernel, along with the range of personalities they support.

```
0-0   Linux           [kernel]
```

Think of execution domains as the "personality" for an operating system. Because other binary formats, such as Solaris, UnixWare, and FreeBSD, can be used with Linux, programmers can change the way the operating system treats system calls from these binaries by changing the personality of the task. Except for the **`PER_LINUX`** execution domain, different personalities can be implemented as dynamically loadable modules.

5.2.9. `/proc/fb`

This file contains a list of frame buffer devices, with the frame buffer device number and the driver that controls it. Typical output of **`/proc/fb`** for systems which contain frame buffer devices looks similar to the following:

```
0 VESA VGA
```

5.2.10. `/proc/filesystems`

This file displays a list of the file system types currently supported by the kernel. Sample output from a generic **`/proc/filesystems`** file looks similar to the following:

```
nodev   sysfs
nodev   rootfs
nodev   bdev
nodev   proc
nodev   sockfs
nodev   binfmt_misc
nodev   usbfs
nodev   usbdevfs
nodev   futexfs
nodev   tmpfs
nodev   pipefs
nodev   eventpollfs
nodev   devpts
 ext2
nodev   ramfs
nodev   hugetlbfs
 iso9660
nodev   mqueue
 ext3
nodev   rpc_pipefs
nodev   autofs
```

The first column signifies whether the file system is mounted on a block device. Those beginning with **`nodev`** are not mounted on a device. The second column lists the names of the file systems supported.

The **`mount`** command cycles through the file systems listed here when one is not specified as an argument.

5.2.11. `/proc/interrupts`

This file records the number of interrupts per IRQ on the x86 architecture. A standard **`/proc/interrupts`** looks similar to the following:

```
  CPU0
  0:   80448940          XT-PIC  timer
  1:     174412          XT-PIC  keyboard
  2:          0          XT-PIC  cascade
  8:          1          XT-PIC  rtc
 10:     410964          XT-PIC  eth0
 12:      60330          XT-PIC  PS/2 Mouse
 14:    1314121          XT-PIC  ide0
 15:    5195422          XT-PIC  ide1
NMI:          0
ERR:          0
```

For a multi-processor machine, this file may look slightly different:

```
    CPU0       CPU1
  0: 1366814704          0          XT-PIC  timer
  1:        128        340    IO-APIC-edge  keyboard
  2:          0          0          XT-PIC  cascade
  8:          0          1    IO-APIC-edge  rtc
 12:       5323       5793    IO-APIC-edge  PS/2 Mouse
 13:          1          0          XT-PIC  fpu
 16:   11184294   15940594   IO-APIC-level  Intel EtherExpress Pro 10/100
Ethernet
 20:    8450043   11120093   IO-APIC-level  megaraid
 30:      10432      10722   IO-APIC-level  aic7xxx
 31:         23         22   IO-APIC-level  aic7xxx
NMI:          0
ERR:          0
```

The first column refers to the IRQ number. Each CPU in the system has its own column and its own number of interrupts per IRQ. The next column reports the type of interrupt, and the last column contains the name of the device that is located at that IRQ.

Each of the types of interrupts seen in this file, which are architecture-specific, mean something different. For x86 machines, the following values are common:

- **`XT-PIC`** — This is the old AT computer interrupts.
- **`IO-APIC-edge`** — The voltage signal on this interrupt transitions from low to high, creating an*edge*, where the interrupt occurs and is only signaled once. This kind of interrupt, as well as the **`IO-APIC-level`** interrupt, are only seen on systems with processors from the 586 family and higher.
- **`IO-APIC-level`** — Generates interrupts when its voltage signal is high until the signal is low again.

5.2.12. `/proc/iomem`

This file shows you the current map of the system's memory for each physical device:

```
00000000-0009fbff : System RAM
0009fc00-0009ffff : reserved
000a0000-000bffff : Video RAM area
000c0000-000c7fff : Video ROM
000f0000-000fffff : System ROM
00100000-07ffffff : System RAM
00100000-00291ba8 : Kernel code
00291ba9-002e09cb : Kernel data
e0000000-e3ffffff : VIA Technologies, Inc. VT82C597 [Apollo VP3] e4000000-
e7ffffff : PCI Bus #01
e4000000-e4003fff : Matrox Graphics, Inc. MGA G200 AGP
e5000000-e57fffff : Matrox Graphics, Inc. MGA G200 AGP
e8000000-e8ffffff : PCI Bus #01
e8000000-e8ffffff : Matrox Graphics, Inc. MGA G200 AGP
ea000000-ea00007f : Digital Equipment Corporation DECchip 21140 [FasterNet]
ea000000-ea00007f : tulip ffff0000-ffffffff : reserved
```

The first column displays the memory registers used by each of the different types of memory. The second column lists the kind of memory located within those registers and displays which memory registers are used by the kernel within the system RAM or, if the network interface card has multiple Ethernet ports, the memory registers assigned for each port.

5.2.13. `/proc/ioports`

The output of **`/proc/ioports`** provides a list of currently registered port regions used for input or output communication with a device. This file can be quite long. The following is a partial listing:

```
0000-001f : dma1
0020-003f : pic1
0040-005f : timer
0060-006f : keyboard
0070-007f : rtc
0080-008f : dma page reg
00a0-00bf : pic2
00c0-00df : dma2
00f0-00ff : fpu
0170-0177 : ide1
01f0-01f7 : ide0
02f8-02ff : serial(auto)
0376-0376 : ide1
03c0-03df : vga+
03f6-03f6 : ide0
03f8-03ff : serial(auto)
0cf8-0cff : PCI conf1
d000-dfff : PCI Bus #01
e000-e00f : VIA Technologies, Inc. Bus Master IDE
e000-e007 : ide0
e008-e00f : ide1
e800-e87f : Digital Equipment Corporation DECchip 21140 [FasterNet]
e800-e87f : tulip
```

The first column gives the I/O port address range reserved for the device listed in the second column.

5.2.14. `/proc/kcore`

This file represents the physical memory of the system and is stored in the core file format. Unlike most **`/proc/`** files, **`kcore`** displays a size. This value is given in bytes and is equal to the size of the physical memory (RAM) used plus 4 KB.

The contents of this file are designed to be examined by a debugger, such as **`gdb`**, and is not human readable.

Warning

Do not view the **`/proc/kcore`** virtual file. The contents of the file scramble text output on the terminal. If this file is accidentally viewed, press **`Ctrl`**+**`C`** to stop the process and then type **`reset`** to bring back the command line prompt.

5.2.15. `/proc/kmsg`

This file is used to hold messages generated by the kernel. These messages are then picked up by other programs, such as **`/sbin/klogd`** or **`/bin/dmesg`**.

5.2.16. `/proc/loadavg`

This file provides a look at the load average in regard to both the CPU and IO over time, as well as additional data used by **`uptime`** and other commands. A sample **`/proc/loadavg`** file looks similar to the following:

```
0.20 0.18 0.12 1/80 11206
```

The first three columns measure CPU and IO utilization of the last one, five, and 15 minute periods. The fourth column shows the number of currently running processes and the total number of processes. The last column displays the last process ID used.

In addition, load average also refers to the number of processes ready to run (i.e. in the run queue, waiting for a CPU share.

5.2.17. `/proc/locks`

This file displays the files currently locked by the kernel. The contents of this file contain internal kernel debugging data and can vary tremendously, depending on the use of the system. A sample **`/proc/locks`** file for a lightly loaded system looks similar to the following:

```
1: POSIX  ADVISORY  WRITE 3568 fd:00:2531452 0 EOF
2: FLOCK  ADVISORY  WRITE 3517 fd:00:2531448 0 EOF
3: POSIX  ADVISORY  WRITE 3452 fd:00:2531442 0 EOF
4: POSIX  ADVISORY  WRITE 3443 fd:00:2531440 0 EOF
5: POSIX  ADVISORY  WRITE 3326 fd:00:2531430 0 EOF
6: POSIX  ADVISORY  WRITE 3175 fd:00:2531425 0 EOF
7: POSIX  ADVISORY  WRITE 3056 fd:00:2548663 0 EOF
```

Each lock has its own line which starts with a unique number. The second column refers to the class of lock used, with **`FLOCK`** signifying the older-style UNIX file locks from a **`flock`** system call and **`POSIX`** representing the newer POSIX locks from the **`lockf`** system call.

The third column can have two values: **`ADVISORY`** or **`MANDATORY`**. **`ADVISORY`** means that the lock does not prevent other people from accessing the data; it only prevents other attempts to lock it. **`MANDATORY`** means that no other access to the data is permitted while the lock is held. The fourth column reveals whether the lock is allowing the holder **`READ`** or **`WRITE`** access to the file. The fifth column shows the ID of the process holding the lock. The sixth column shows the ID of the file being locked, in the format of ***`MAJOR-DEVICE`*****`:`*****`MINOR-DEVICE`*****`:`*****`INODE-NUMBER`*** . The seventh and eighth column shows the start and end of the file's locked region.

5.2.18. `/proc/mdstat`

This file contains the current information for multiple-disk, RAID configurations. If the system does not contain such a configuration, then **`/proc/mdstat`** looks similar to the following:

```
Personalities :  read_ahead not set unused devices: <none>
```

This file remains in the same state as seen above unless a software RAID or **`md`** device is present. In that case, view **`/proc/mdstat`** to find the current status of **`md`*****`X`*** RAID devices.

The **`/proc/mdstat`** file below shows a system with its **`md0`** configured as a RAID 1 device, while it is currently re-syncing the disks:

```
Personalities : [linear] [raid1] read_ahead 1024 sectors
md0: active raid1 sda2[1] sdb2[0] 9940 blocks [2/2] [UU] resync=1%
finish=12.3min algorithm 2 [3/3] [UUU]
unused devices: <none>
```

5.2.19. `/proc/meminfo`

This is one of the more commonly used files in the **`/proc/`** directory, as it reports a large amount of valuable information about the systems RAM usage.

The following sample **`/proc/meminfo`** virtual file is from a system with 256 MB of RAM and 512 MB of swap space:

```
MemTotal:       255908 kB
MemFree:         69936 kB
Buffers:         15812 kB
Cached:         115124 kB
SwapCached:          0 kB
Active:          92700 kB
Inactive:        63792 kB
HighTotal:           0 kB
HighFree:            0 kB
LowTotal:       255908 kB
LowFree:         69936 kB
SwapTotal:      524280 kB
SwapFree:       524280 kB
Dirty:               4 kB
Writeback:           0 kB
Mapped:          42236 kB
Slab:            25912 kB
Committed_AS:   118680 kB
PageTables:       1236 kB
VmallocTotal:  3874808 kB
VmallocUsed:      1416 kB
VmallocChunk:  3872908 kB
HugePages_Total:     0
HugePages_Free:      0
Hugepagesize:     4096 kB
```

Much of the information here is used by the **`free`**, **`top`**, and **`ps`** commands. In fact, the output of the **`free`** command is similar in appearance to the contents and structure of **`/proc/meminfo`**. But by looking directly at **`/proc/meminfo`**, more details are revealed:

- **`MemTotal`** — Total amount of physical RAM, in kilobytes.
- **`MemFree`** — The amount of physical RAM, in kilobytes, left unused by the system.
- **`Buffers`** — The amount of physical RAM, in kilobytes, used for file buffers.
- **`Cached`** — The amount of physical RAM, in kilobytes, used as cache memory.
- **`SwapCached`** — The amount of swap, in kilobytes, used as cache memory.
- **`Active`** — The total amount of buffer or page cache memory, in kilobytes, that is in active use. This is memory that has been recently used and is usually not reclaimed for other purposes.
- **`Inactive`** — The total amount of buffer or page cache memory, in kilobytes, that are free and available. This is memory that has not been recently used and can be reclaimed for other purposes.
- **`HighTotal`** and **`HighFree`** — The total and free amount of memory, in kilobytes, that is not directly mapped into kernel space. The **`HighTotal`** value can vary based on the type of kernel used.

- **`LowTotal`** and **`LowFree`** — The total and free amount of memory, in kilobytes, that is directly mapped into kernel space. The **`LowTotal`** value can vary based on the type of kernel used.
- **`SwapTotal`** — The total amount of swap available, in kilobytes.
- **`SwapFree`** — The total amount of swap free, in kilobytes.
- **`Dirty`** — The total amount of memory, in kilobytes, waiting to be written back to the disk.
- **`Writeback`** — The total amount of memory, in kilobytes, actively being written back to the disk.
- **`Mapped`** — The total amount of memory, in kilobytes, which have been used to map devices, files, or libraries using the **`mmap`** command.
- **`Slab`** — The total amount of memory, in kilobytes, used by the kernel to cache data structures for its own use.
- **`Committed_AS`** — The total amount of memory, in kilobytes, estimated to complete the workload. This value represents the worst case scenario value, and also includes swap memory.
- **`PageTables`** — The total amount of memory, in kilobytes, dedicated to the lowest page table level.
- **`VMallocTotal`** — The total amount of memory, in kilobytes, of total allocated virtual address space.
- **`VMallocUsed`** — The total amount of memory, in kilobytes, of used virtual address space.
- **`VMallocChunk`** — The largest contiguous block of memory, in kilobytes, of available virtual address space.
- **`HugePages_Total`** — The total number of hugepages for the system. The number is derived by dividing **`Hugepagesize`** by the megabytes set aside for hugepages specified in **`/proc/sys/vm/hugetlb_pool`**. *This statistic only appears on the x86, Itanium, and AMD64 architectures.*
- **`HugePages_Free`** — The total number of hugepages available for the system. *This statistic only appears on the x86, Itanium, and AMD64 architectures.*
- **`Hugepagesize`** — The size for each hugepages unit in kilobytes. By default, the value is 4096 KB on uniprocessor kernels for 32 bit architectures. For SMP, hugemem kernels, and AMD64, the default is 2048 KB. For Itanium architectures, the default is 262144 KB. *This statistic only appears on the x86, Itanium, and AMD64 architectures.*

5.2.20. `/proc/misc`

This file lists miscellaneous drivers registered on the miscellaneous major device, which is device number 10:

```
63 device-mapper 175 agpgart 135 rtc 134 apm_bios
```

The first column is the minor number of each device, while the second column shows the driver in use.

5.2.21. `/proc/modules`

This file displays a list of all modules loaded into the kernel. Its contents vary based on the configuration and use of your system, but it should be organized in a similar manner to this sample **`/proc/modules`** file output:

Note

This example has been reformatted into a readable format. Most of this information can also be viewed via the **`/sbin/lsmod`** command.

```
nfs      170109  0 -          Live 0x129b0000
lockd    51593   1 nfs,       Live 0x128b0000
nls_utf8 1729    0 -          Live 0x12830000
vfat     12097   0 -          Live 0x12823000
fat      38881   1 vfat,      Live 0x1287b000
autofs4  20293   2 -          Live 0x1284f000
sunrpc   140453  3 nfs,lockd, Live 0x12954000
3c59x    33257   0 -          Live 0x12871000
uhci_hcd 28377   0 -          Live 0x12869000
md5      3777    1 -          Live 0x1282c000
ipv6     211845 16 -          Live 0x128de000
ext3     92585   2 -          Live 0x12886000
jbd      65625   1 ext3,      Live 0x12857000
dm_mod   46677   3 -          Live 0x12833000
```

The first column contains the name of the module.

The second column refers to the memory size of the module, in bytes.

The third column lists how many instances of the module are currently loaded. A value of zero represents an unloaded module.

The fourth column states if the module depends upon another module to be present in order to function, and lists those other modules.

The fifth column lists what load state the module is in: **`Live`**, **`Loading`**, or **`Unloading`** are the only possible values.

The sixth column lists the current kernel memory offset for the loaded module. This information can be useful for debugging purposes, or for profiling tools such as **`oprofile`**.

5.2.22. `/proc/mounts`

This file provides a list of all mounts in use by the system:

```
rootfs / rootfs rw 0 0
/proc /proc proc rw,nodiratime 0 0 none
/dev ramfs rw 0 0
/dev/mapper/VolGroup00-LogVol00 / ext3 rw 0 0
none /dev ramfs rw 0 0
/proc /proc proc rw,nodiratime 0 0
/sys /sys sysfs rw 0 0
none /dev/pts devpts rw 0 0
usbdevfs /proc/bus/usb usbdevfs rw 0 0
/dev/hda1 /boot ext3 rw 0 0
none /dev/shm tmpfs rw 0 0
none /proc/sys/fs/binfmt_misc binfmt_misc rw 0 0
sunrpc /var/lib/nfs/rpc_pipefs rpc_pipefs rw 0 0
```

The output found here is similar to the contents of **`/etc/mtab`**, except that **`/proc/mount`** is more up-to-date.

The first column specifies the device that is mounted, the second column reveals the mount point, and the third column tells the file system type, and the fourth column tells you if it is mounted read-only (**`ro`**) or read-write (**`rw`**). The fifth and sixth columns are dummy values designed to match the format used in **`/etc/mtab`**.

5.2.23. `/proc/mtrr`

This file refers to the current Memory Type Range Registers (MTRRs) in use with the system. If the system architecture supports MTRRs, then the **`/proc/mtrr`** file may look similar to the following:

```
reg00: base=0x00000000 (   0MB), size= 256MB: write-back, count=1
reg01: base=0xe8000000 (3712MB), size=  32MB: write-combining, count=1
```

MTRRs are used with the Intel P6 family of processors (Pentium II and higher) and control processor access to memory ranges. When using a video card on a PCI or AGP bus, a properly configured **`/proc/mtrr`** file can increase performance more than 150%.

Most of the time, this value is properly configured by default. More information on manually configuring this file can be found locally at the following location:

```
/usr/share/doc/kernel-doc-<version>/Documentation/mtrr.txt
```

5.2.24. `/proc/partitions`

This file contains partition block allocation information. A sampling of this file from a basic system looks similar to the following:

```
major minor  #blocks  name
  3     0   19531250 hda
  3     1     104391 hda1
  3     2   19422585 hda2
253     0   22708224 dm-0
253     1     524288 dm-1
```

Most of the information here is of little importance to the user, except for the following columns:

- **`major`** — The major number of the device with this partition. The major number in the **`/proc/partitions`**, (**`3`**), corresponds with the block device **`ide0`**, in **`/proc/devices`**.
- **`minor`** — The minor number of the device with this partition. This serves to separate the partitions into different physical devices and relates to the number at the end of the name of the partition.
- **`#blocks`** — Lists the number of physical disk blocks contained in a particular partition.
- **`name`** — The name of the partition.

5.2.25. `/proc/pci`

This file contains a full listing of every PCI device on the system. Depending on the number of PCI devices, **`/proc/pci`** can be rather long. A sampling of this file from a basic system looks similar to the following:

```
Bus  0, device 0, function 0: Host bridge: Intel Corporation 440BX/ZX -
```

```
82443BX/ZX Host bridge (rev 3). Master Capable. Latency=64. Prefetchable 32
bit memory at 0xe4000000 [0xe7ffffff].
Bus  0, device 1, function 0: PCI bridge: Intel Corporation 440BX/ZX -
82443BX/ZX AGP bridge (rev 3).   Master Capable. Latency=64. Min Gnt=128.
Bus  0, device 4, function 0: ISA bridge: Intel Corporation 82371AB PIIX4
ISA (rev 2).
Bus  0, device 4, function 1: IDE interface: Intel Corporation 82371AB PIIX4
IDE (rev 1). Master Capable. Latency=32. I/O at 0xd800 [0xd80f].
Bus  0, device 4, function 2: USB Controller: Intel Corporation 82371AB
PIIX4 USB (rev 1). IRQ 5. Master Capable. Latency=32. I/O at 0xd400 [0xd41f].
Bus  0, device 4, function 3: Bridge: Intel Corporation 82371AB PIIX4 ACPI
(rev 2). IRQ 9.
Bus  0, device 9, function 0: Ethernet controller: Lite-On Communications
Inc LNE100TX (rev 33). IRQ 5. Master Capable. Latency=32. I/O at 0xd000
[0xd0ff].
Bus  0, device 12, function  0: VGA compatible controller: S3 Inc. ViRGE/DX
or /GX (rev 1). IRQ 11. Master Capable. Latency=32. Min Gnt=4.Max Lat=255.
```

This output shows a list of all PCI devices, sorted in the order of bus, device, and function. Beyond providing the name and version of the device, this list also gives detailed IRQ information so an administrator can quickly look for conflicts.

Note

To get a more readable version of this information, type:

```
lspci -vb
```

5.2.26. `/proc/slabinfo`

This file gives full information about memory usage on the *slab* level. Linux kernels greater than version 2.2 use *slab pools* to manage memory above the page level. Commonly used objects have their own slab pools.

Instead of parsing the highly verbose **`/proc/slabinfo`** file manually, the **`/usr/bin/slabtop`** program displays kernel slab cache information in real time. This program allows for custom configurations, including column sorting and screen refreshing.

A sample screen shot of **`/usr/bin/slabtop`** usually looks like the following example:

```
Active / Total Objects (% used)    : 133629 / 147300 (90.7%)
Active / Total Slabs (% used)      : 11492 / 11493 (100.0%)
Active / Total Caches (% used)     : 77 / 121 (63.6%)
Active / Total Size (% used)       : 41739.83K / 44081.89K (94.7%)
Minimum / Average / Maximum Object : 0.01K / 0.30K / 128.00K
OBJS   ACTIVE USE      OBJ   SIZE     SLABS OBJ/SLAB CACHE SIZE NAME
44814  43159  96%    0.62K   7469      6      29876K ext3_inode_cache
36900  34614  93%    0.05K    492     75       1968K buffer_head
35213  33124  94%    0.16K   1531     23       6124K dentry_cache
7364   6463   87%   0.27K    526      14       2104K radix_tree_node
2585   1781   68%   0.08K     55      47        220K vm_area_struct
2263   2116   93%   0.12K     73      31        292K size-128
1904   1125   59%   0.03K     16      119         64K size-32
```

```
 1666    768  46%    0.03K     14      119        56K anon_vma
 1512   1482  98%    0.44K    168        9       672K inode_cache
 1464   1040  71%    0.06K     24       61        96K size-64
 1320    820  62%    0.19K     66       20       264K filp
 678    587  86%    0.02K      3      226        12K dm_io
 678    587  86%    0.02K      3      226        12K dm_tio
 576    574  99%    0.47K     72        8       288K proc_inode_cache
 528    514  97%    0.50K     66        8       264K size-512
 492    372  75%    0.09K     12       41        48K bio
 465    314  67%    0.25K     31       15       124K size-256
 452    331  73%    0.02K      2      226         8K biovec-1
 420    420 100%    0.19K     21       20        84K skbuff_head_cache
 305    256  83%    0.06K      5       61        20K biovec-4
 290      4   1%    0.01K      1      290         4K revoke_table
 264    264 100%    4.00K    264        1      1056K size-4096
 260    256  98%    0.19K     13       20        52K biovec-16
 260    256  98%    0.75K     52        5       208K biovec-64
```

Some of the more commonly used statistics in **`/proc/slabinfo`** that are included into **`/usr/bin/slabtop`** include:

- **`OBJS`** — The total number of objects (memory blocks), including those in use (allocated), and some spares not in use.
- **`ACTIVE`** — The number of objects (memory blocks) that are in use (allocated).
- **`USE`** — Percentage of total objects that are active. ((ACTIVE/OBJS)(100))
- **`OBJ SIZE`** — The size of the objects.
- **`SLABS`** — The total number of slabs.
- **`OBJ/SLAB`** — The number of objects that fit into a slab.
- **`CACHE SIZE`** — The cache size of the slab.
- **`NAME`** — The name of the slab.

For more information on the **`/usr/bin/slabtop`** program, refer to the **`slabtop`** man page.

5.2.27. `/proc/stat`

This file keeps track of a variety of different statistics about the system since it was last restarted. The contents of **`/proc/stat`**, which can be quite long, usually begins like the following example:

```
cpu  259246 7001 60190 34250993 137517 772 0
cpu0 259246 7001 60190 34250993 137517 772 0
intr 354133732 347209999 2272 0 4 4 0 0 3 1 1249247 0 0 80143 0 422626
5169433
ctxt 12547729
btime 1093631447
processes 130523
procs_running 1
procs_blocked 0
preempt 5651840
cpu  209841 1554 21720 118519346 72939 154 27168
cpu0 42536 798 4841 14790880 14778 124 3117
```

```
cpu1 24184 569 3875 14794524 30209 29 3130
cpu2 28616 11 2182 14818198 4020 1 3493
cpu3 35350 6 2942 14811519 3045 0 3659
cpu4 18209 135 2263 14820076 12465 0 3373
cpu5 20795 35 1866 14825701 4508 0 3615
cpu6 21607 0 2201 14827053 2325 0 3334
cpu7 18544 0 1550 14831395 1589 0 3447
intr 15239682 14857833 6 0 6 6 0 5 0 1 0 0 0 29 0 2 0 0 0 0 0 0 0 94982 0
286812
ctxt 4209609
btime 1078711415
processes 21905
procs_running 1
procs_blocked 0
```

Some of the more commonly used statistics include:

- **`cpu`** — Measures the number of *jiffies* (1/100 of a second for x86 systems) that the system has been in user mode, user mode with low priority (nice), system mode, idle task, I/O wait, IRQ (hardirq), and softirq respectively. The IRQ (hardirq) is the direct response to a hardware event. The IRQ takes minimal work for queuing the "heavy" work up for the softirq to execute. The softirq runs at a lower priority than the IRQ and therefore may be interrupted more frequently. The total for all CPUs is given at the top, while each individual CPU is listed below with its own statistics. The following example is a 4-way Intel Pentium Xeon configuration with multi-threading enabled, therefore showing four physical processors and four virtual processors totaling eight processors.
- **`page`** — The number of memory pages the system has written in and out to disk.
- **`swap`** — The number of swap pages the system has brought in and out.
- **`intr`** — The number of interrupts the system has experienced.
- **`btime`** — The boot time, measured in the number of seconds since January 1, 1970, otherwise known as the *epoch*.

5.2.28. `/proc/swaps`

This file measures swap space and its utilization. For a system with only one swap partition, the output of **`/proc/swaps`** may look similar to the following:

```
Filename                          Type        Size      Used     Priority
/dev/mapper/VolGroup00-LogVol01   partition   524280    0        -1
```

While some of this information can be found in other files in the **`/proc/`** directory, **`/proc/swaps`** provides a snapshot of every swap file name, the type of swap space, the total size, and the amount of space in use (in kilobytes). The priority column is useful when multiple swap files are in use. The lower the priority, the more likely the swap file is to be used.

5.2.29. `/proc/sysrq-trigger`

Using the **`echo`** command to write to this file, a remote root user can execute most System Request Key commands remotely as if at the local terminal. To **`echo`** values to this file, the **`/proc/sys/kernel/sysrq`** must be set to a value other than **`0`**. For more information about the System Request Key, refer to Section 5.3.9.3, “ `/proc/sys/kernel/` ”.

Although it is possible to write to this file, it cannot be read, even by the root user.

5.2.30. `/proc/uptime`

This file contains information detailing how long the system has been on since its last restart. The output of **`/proc/uptime`** is quite minimal:

```
350735.47 234388.90
```

The first number is the total number of seconds the system has been up. The second number is how much of that time the machine has spent idle, in seconds.

5.2.31. `/proc/version`

This file specifies the version of the Linux kernel and **`gcc`** in use, as well as the version of Red Hat Enterprise Linux installed on the system:

```
Linux version 2.6.8-1.523 (user@foo.redhat.com) (gcc version 3.4.1 20040714
\  (Red Hat Enterprise Linux 3.4.1-7)) #1 Mon Aug 16 13:27:03 EDT 2004
```

This information is used for a variety of purposes, including the version data presented when a user logs in.

5.3. Directories within `/proc/`

Common groups of information concerning the kernel are grouped into directories and subdirectories within the **`/proc/`** directory.

5.3.1. Process Directories

Every **`/proc/`** directory contains a number of directories with numerical names. A listing of them may be similar to the following:

```
dr-xr-xr-x    3 root     root            0 Feb 13 01:28 1
dr-xr-xr-x    3 root     root            0 Feb 13 01:28 1010
dr-xr-xr-x    3 xfs      xfs             0 Feb 13 01:28 1087
dr-xr-xr-x    3 daemon   daemon          0 Feb 13 01:28 1123
dr-xr-xr-x    3 root     root            0 Feb 13 01:28 11307
dr-xr-xr-x    3 apache   apache          0 Feb 13 01:28 13660
dr-xr-xr-x    3 rpc      rpc             0 Feb 13 01:28 637
dr-xr-xr-x    3 rpcuser  rpcuser         0 Feb 13 01:28 666
```

These directories are called *process directories*, as they are named after a program's process ID and contain information specific to that process. The owner and group of each process directory is set to the user running the process. When the process is terminated, its **`/proc/`** process directory vanishes.

Each process directory contains the following files:

- **`cmdline`** — Contains the command issued when starting the process.
- **`cwd`** — A symbolic link to the current working directory for the process.
- **`environ`** — A list of the environment variables for the process. The environment variable is given in all upper-case characters, and the value is in lower-case characters.

- **exe** — A symbolic link to the executable of this process.
- **fd** — A directory containing all of the file descriptors for a particular process. These are given in numbered links:

```
total 0
lrwx------    1 root     root           64 May  8 11:31 0 -> /dev/null
lrwx------    1 root     root           64 May  8 11:31 1 -> /dev/null
lrwx------    1 root     root           64 May  8 11:31 2 -> /dev/null
lrwx------    1 root     root           64 May  8 11:31 3 -> /dev/ptmx
lrwx------    1 root     root           64 May  8 11:31 4 -> socket:
[7774817]
lrwx------    1 root     root           64 May  8 11:31 5 -> /dev/ptmx
lrwx------    1 root     root           64 May  8 11:31 6 -> socket:
[7774829]
lrwx------    1 root     root           64 May  8 11:31 7 -> /dev/ptmx
```

- **maps** — A list of memory maps to the various executables and library files associated with this process. This file can be rather long, depending upon the complexity of the process, but sample output from the **sshd** process begins like the following:

```
08048000-08086000 r-xp 00000000 03:03 391479     /usr/sbin/sshd
08086000-08088000 rw-p 0003e000 03:03 391479 /usr/sbin/sshd
08088000-08095000 rwxp 00000000 00:00 0
40000000-40013000 r-xp 0000000 03:03 293205 /lib/ld-2.2.5.so
40013000-40014000 rw-p 00013000 03:03 293205 /lib/ld-2.2.5.so
40031000-40038000 r-xp 00000000 03:03 293282 /lib/libpam.so.0.75
40038000-40039000 rw-p 00006000 03:03 293282 /lib/libpam.so.0.75
40039000-4003a000 rw-p 00000000 00:00 0
4003a000-4003c000 r-xp 00000000 03:03 293218 /lib/libdl-2.2.5.so
4003c000-4003d000 rw-p 00001000 03:03 293218 /lib/libdl-2.2.5.so
```

- **mem** — The memory held by the process. This file cannot be read by the user.
- **root** — A link to the root directory of the process.
- **stat** — The status of the process.
- **statm** — The status of the memory in use by the process. Below is a sample **/proc/statm** file:

```
263 210 210 5 0 205 0
```

  The seven columns relate to different memory statistics for the process. From left to right, they report the following aspects of the memory used:

  - Total program size, in kilobytes.
  - Size of memory portions, in kilobytes.
  - Number of pages that are shared.
  - Number of pages that are code.
  - Number of pages of data/stack.
  - Number of library pages.

  - Number of dirty pages.

- **`status`** — The status of the process in a more readable form than **`stat`** or **`statm`**. Sample output for **`sshd`** looks similar to the following:

```
Name: sshd
State: S (sleeping)
Tgid: 797
Pid: 797
PPid: 1
TracerPid: 0
Uid: 0 0 0 0
Gid: 0 0 0 0
FDSize: 32
Groups:
VmSize:     3072 kB
VmLck:        0 kB
VmRSS:      840 kB
VmData:      104 kB
VmStk:       12 kB
VmExe:      300 kB
VmLib:     2528 kB
SigPnd: 0000000000000000
SigBlk: 0000000000000000
SigIgn: 8000000000001000
SigCgt: 0000000000014005
CapInh: 0000000000000000
CapPrm: 00000000fffffeff
CapEff: 00000000fffffeff
```

  The information in this output includes the process name and ID, the state (such as **`S (sleeping)`** or **`R (running)`**), user/group ID running the process, and detailed data regarding memory usage.

5.3.1.1. `/proc/self/`

The **`/proc/self/`** directory is a link to the currently running process. This allows a process to look at itself without having to know its process ID.

Within a shell environment, a listing of the **`/proc/self/`** directory produces the same contents as listing the process directory for that process.

5.3.2. `/proc/bus/`

This directory contains information specific to the various buses available on the system. For example, on a standard system containing PCI and USB buses, current data on each of these buses is available within a subdirectory within **`/proc/bus/`** by the same name, such as **`/proc/bus/pci/`**.

The subdirectories and files available within **`/proc/bus/`** vary depending on the devices connected to the system. However, each bus type has at least one directory. Within these bus directories are normally at least one subdirectory with a numerical name, such as **`001`**, which contain binary files.

For example, the **`/proc/bus/usb/`** subdirectory contains files that track the various devices on any USB buses, as well as the drivers required for them. The following is a sample listing of a **`/proc/bus/usb/`** directory:

```
total 0 dr-xr-xr-x    1 root     root            0 May  3 16:25 001
-r--r--r--    1 root     root            0 May  3 16:25 devices
-r--r--r--    1 root     root            0 May  3 16:25 drivers
```

The **`/proc/bus/usb/001/`** directory contains all devices on the first USB bus and the **`devices`** file identifies the USB root hub on the motherboard.

The following is a example of a **`/proc/bus/usb/devices`** file:

```
T:  Bus=01 Lev=00 Prnt=00 Port=00 Cnt=00 Dev#=  1 Spd=12  MxCh= 2
B:  Alloc=  0/900 us ( 0%), #Int=  0, #Iso=  0
D:  Ver= 1.00 Cls=09(hub  ) Sub=00 Prot=00 MxPS= 8 #Cfgs=  1
P:  Vendor=0000 ProdID=0000 Rev= 0.00
S:  Product=USB UHCI Root Hub
S:  SerialNumber=d400
C:* #Ifs= 1 Cfg#= 1 Atr=40 MxPwr=  0mA
I:  If#= 0 Alt= 0 #EPs= 1 Cls=09(hub  ) Sub=00 Prot=00 Driver=hub
E:  Ad=81(I) Atr=03(Int.) MxPS=   8 Ivl=255ms
```

5.3.3. `/proc/driver/`

This directory contains information for specific drivers in use by the kernel.

A common file found here is **`rtc`** which provides output from the driver for the system's *Real Time Clock (RTC)*, the device that keeps the time while the system is switched off. Sample output from **`/proc/driver/rtc`** looks like the following:

```
rtc_time        : 16:21:00
rtc_date        : 2004-08-31
rtc_epoch       : 1900
alarm           : 21:16:27
DST_enable      : no
BCD             : yes
24hr            : yes
square_wave     : no
alarm_IRQ       : no
update_IRQ      : no
periodic_IRQ    : no
periodic_freq   : 1024
batt_status     : okay
```

For more information about the RTC, refer to the following installed documentation:

`/usr/share/doc/kernel-doc-<`*`version`*`>/Documentation/rtc.txt`.

5.3.4. `/proc/fs`

This directory shows which file systems are exported. If running an NFS server, typing **`cat /proc/fs/nfsd/exports`** displays the file systems being shared and the permissions granted for those file systems. For more on file system sharing with NFS, refer to Chapter 21, *Network File System (NFS)*.

5.3.5. `/proc/ide/`

This directory contains information about IDE devices on the system. Each IDE channel is represented as a separate directory, such as **`/proc/ide/ide0`** and **`/proc/ide/ide1`**. In addition, a **`drivers`** file is available, providing the version number of the various drivers used on the IDE channels:

```
ide-floppy version 0.99.
newide ide-cdrom version 4.61
ide-disk version 1.18
```

Many chipsets also provide a file in this directory with additional data concerning the drives connected through the channels. For example, a generic Intel PIIX4 Ultra 33 chipset produces the **`/proc/ide/piix`** file which reveals whether DMA or UDMA is enabled for the devices on the IDE channels:

```
Intel PIIX4 Ultra 33 Chipset.
------------- Primary Channel ---------------- Secondary Channel ---------
----
  enabled                          enabled

------------- drive0 --------- drive1 -------- drive0 ---------- drive1 --
----
DMA enabled:    yes              no              yes               no
UDMA enabled:   yes              no              no                no
UDMA enabled:   2                X               X                 X
UDMA DMA PIO
```

Navigating into the directory for an IDE channel, such as **`ide0`**, provides additional information. The **`channel`** file provides the channel number, while the **`model`** identifies the bus type for the channel (such as **`pci`**).

5.3.5.1. Device Directories

Within each IDE channel directory is a device directory. The name of the device directory corresponds to the drive letter in the **`/dev/`** directory. For instance, the first IDE drive on **`ide0`** would be **`hda`**.

Note

There is a symbolic link to each of these device directories in the **`/proc/ide/`** directory.

Each device directory contains a collection of information and statistics. The contents of these directories vary according to the type of device connected. Some of the more useful files common to many devices include:

- **`cache`** — The device cache.
- **`capacity`** — The capacity of the device, in 512 byte blocks.
- **`driver`** — The driver and version used to control the device.
- **`geometry`** — The physical and logical geometry of the device.
- **`media`** — The type of device, such as a **`disk`**.
- **`model`** — The model name or number of the device.
- **`settings`** — A collection of current device parameters. This file usually contains quite a bit of useful,

technical information. A sample **settings** file for a standard IDE hard disk looks similar to the following:

```
name                value          min          max          mode
----                -----          ---          ---          ----
acoustic            0              0            254          rw
address             0              0            2            rw
bios_cyl            38752          0            65535        rw
bios_head           16             0            255          rw
bios_sect           63             0            63           rw
bswap               0              0            1            r
current_speed       68             0            70           rw
failures            0              0            65535        rw
init_speed          68             0            70           rw
io_32bit            0              0            3            rw
keepsettings        0              0            1            rw
lun                 0              0            7            rw
max_failures        1              0            65535        rw
multcount           16             0            16           rw
nice1               1              0            1            rw
nowerr              0              0            1            rw
number              0              0            3            rw
pio_mode            write-only     0            255          w
unmaskirq           0              0            1            rw
using_dma           1              0            1            rw
wcache              1              0            1            rw
```

5.3.6. `/proc/irq/`

This directory is used to set IRQ to CPU affinity, which allows the system to connect a particular IRQ to only one CPU. Alternatively, it can exclude a CPU from handling any IRQs.

Each IRQ has its own directory, allowing for the individual configuration of each IRQ. The **/proc/irq/prof_cpu_mask** file is a bitmask that contains the default values for the **smp_affinity** file in the IRQ directory. The values in **smp_affinity** specify which CPUs handle that particular IRQ.

For more information about the **/proc/irq/** directory, refer to the following installed documentation:

```
/usr/share/doc/kernel-doc-<version>/Documentation/filesystems/proc.txt
```

5.3.7. `/proc/net/`

This directory provides a comprehensive look at various networking parameters and statistics. Each directory and virtual file within this directory describes aspects of the system's network configuration. Below is a partial list of the **/proc/net/** directory:

- **arp** — Lists the kernel's ARP table. This file is particularly useful for connecting a hardware address to an IP address on a system.
- **atm/** directory — The files within this directory contain *Asynchronous Transfer Mode (ATM)* settings and statistics. This directory is primarily used with ATM networking and ADSL cards.
- **dev** — Lists the various network devices configured on the system, complete with transmit and receive statistics. This file displays the number of bytes each interface has sent and received, the number of packets inbound and outbound, the number of errors seen, the number of packets dropped, and more.

- **`dev_mcast`** — Lists Layer2 multicast groups on which each device is listening.
- **`igmp`** — Lists the IP multicast addresses which this system joined.
- **`ip_conntrack`** — Lists tracked network connections for machines that are forwarding IP connections.
- **`ip_tables_names`** — Lists the types of **`iptables`** in use. This file is only present if **`iptables`** is active on the system and contains one or more of the following values: **`filter`**, **`mangle`**, or **`nat`**.
- **`ip_mr_cache`** — Lists the multicast routing cache.
- **`ip_mr_vif`** — Lists multicast virtual interfaces.
- **`netstat`** — Contains a broad yet detailed collection of networking statistics, including TCP timeouts, SYN cookies sent and received, and much more.
- **`psched`** — Lists global packet scheduler parameters.
- **`raw`** — Lists raw device statistics.
- **`route`** — Lists the kernel's routing table.
- **`rt_cache`** — Contains the current routing cache.
- **`snmp`** — List of Simple Network Management Protocol (SNMP) data for various networking protocols in use.
- **`sockstat`** — Provides socket statistics.
- **`tcp`** — Contains detailed TCP socket information.
- **`tr_rif`** — Lists the token ring RIF routing table.
- **`udp`** — Contains detailed UDP socket information.
- **`unix`** — Lists UNIX domain sockets currently in use.
- **`wireless`** — Lists wireless interface data.

5.3.8. `/proc/scsi/`

This directory is analogous to the **`/proc/ide/`** directory, but it is for connected SCSI devices.

The primary file in this directory is **`/proc/scsi/scsi`**, which contains a list of every recognized SCSI device. From this listing, the type of device, as well as the model name, vendor, SCSI channel and ID data is available.

For example, if a system contains a SCSI CD-ROM, a tape drive, a hard drive, and a RAID controller, this file looks similar to the following:

```
Attached devices:
Host: scsi1
Channel: 00
Id: 05
Lun: 00
Vendor: NEC
Model: CD-ROM DRIVE:466
Rev: 1.06
```

```
Type:   CD-ROM
ANSI SCSI revision: 02
Host: scsi1
Channel: 00
Id: 06
Lun: 00
Vendor: ARCHIVE
Model: Python 04106-XXX
Rev: 7350
Type:   Sequential-Access
ANSI SCSI revision: 02
Host: scsi2
Channel: 00
Id: 06
Lun: 00
Vendor: DELL
Model: 1x6 U2W SCSI BP
Rev: 5.35
Type:   Processor
ANSI SCSI revision: 02
Host: scsi2
Channel: 02
Id: 00
Lun: 00
Vendor: MegaRAID
Model: LD0 RAID5 34556R
Rev: 1.01
Type:   Direct-Access
ANSI SCSI revision: 02
```

Each SCSI driver used by the system has its own directory within **`/proc/scsi/`**, which contains files specific to each SCSI controller using that driver. From the previous example, **`aic7xxx/`** and **`megaraid/`** directories are present, since two drivers are in use. The files in each of the directories typically contain an I/O address range, IRQ information, and statistics for the SCSI controller using that driver. Each controller can report a different type and amount of information. The Adaptec AIC-7880 Ultra SCSI host adapter's file in this example system produces the following output:

```
Adaptec AIC7xxx driver version: 5.1.20/3.2.4
Compile Options:
TCQ Enabled By Default : Disabled
AIC7XXX_PROC_STATS     : Enabled
AIC7XXX_RESET_DELAY    : 5
Adapter Configuration:
SCSI Adapter: Adaptec AIC-7880 Ultra SCSI host adapter
Ultra Narrow Controller     PCI MMAPed
I/O Base: 0xfcffe000
Adapter SEEPROM Config: SEEPROM found and used.
Adaptec SCSI BIOS: Enabled
IRQ: 30
SCBs: Active 0, Max Active 1, Allocated 15, HW 16, Page 255
Interrupts: 33726
BIOS Control Word: 0x18a6
Adapter Control Word: 0x1c5f
Extended Translation: Enabled
Disconnect Enable Flags: 0x00ff
Ultra Enable Flags: 0x0020
```

```
Tag Queue Enable Flags: 0x0000
Ordered Queue Tag Flags: 0x0000
Default Tag Queue Depth: 8
Tagged Queue By Device array for aic7xxx
host instance 1:
{255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255}
Actual queue depth per device for aic7xxx host instance 1:
{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1}
Statistics:

(scsi1:0:5:0) Device using Narrow/Sync transfers at 20.0 MByte/sec, offset
15
Transinfo settings: current(12/15/0/0), goal(12/15/0/0), user(12/15/0/0)
Total transfers 0 (0 reads and 0 writes)
  < 2K      2K+     4K+     8K+    16K+    32K+    64K+   128K+
Reads:        0       0       0       0       0       0       0       0
Writes:       0       0       0       0       0       0       0       0

(scsi1:0:6:0) Device using Narrow/Sync transfers at 10.0 MByte/sec, offset
15
Transinfo settings: current(25/15/0/0), goal(12/15/0/0), user(12/15/0/0)
Total transfers 132 (0 reads and 132 writes)
  < 2K      2K+     4K+     8K+    16K+    32K+    64K+   128K+
Reads:        0       0       0       0       0       0       0       0
Writes:       0       0       0       1     131       0       0       0
```

This output reveals the transfer speed to the SCSI devices connected to the controller based on channel ID, as well as detailed statistics concerning the amount and sizes of files read or written by that device. For example, this controller is communicating with the CD-ROM at 20 megabytes per second, while the tape drive is only communicating at 10 megabytes per second.

5.3.9. `/proc/sys/`

The **`/proc/sys/`** directory is different from others in **`/proc/`** because it not only provides information about the system but also allows the system administrator to immediately enable and disable kernel features.

Warning

Use caution when changing settings on a production system using the various files in the **`/proc/sys/`** directory. Changing the wrong setting may render the kernel unstable, requiring a system reboot.

For this reason, be sure the options are valid for that file before attempting to change any value in **`/proc/sys/`**.

A good way to determine if a particular file can be configured, or if it is only designed to provide information, is to list it with the **`-l`** option at the shell prompt. If the file is writable, it may be used to configure the kernel. For example, a partial listing of **`/proc/sys/fs`** looks like the following:

```
-r--r--r--    1 root     root            0 May 10 16:14 dentry-state
-rw-r--r--    1 root     root            0 May 10 16:14 dir-notify-enable
-r--r--r--    1 root     root            0 May 10 16:14 dquot-nr
```

```
-rw-r--r--    1 root     root            0 May 10 16:14 file-max
-r--r--r--    1 root     root            0 May 10 16:14 file-nr
```

In this listing, the files **`dir-notify-enable`** and **`file-max`** can be written to and, therefore, can be used to configure the kernel. The other files only provide feedback on current settings.

Changing a value within a **`/proc/sys/`** file is done by echoing the new value into the file. For example, to enable the System Request Key on a running kernel, type the command:

```
echo 1 > /proc/sys/kernel/sysrq
```

This changes the value for **`sysrq`** from **`0`** (off) to **`1`** (on).

A few **`/proc/sys/`** configuration files contain more than one value. To correctly send new values to them, place a space character between each value passed with the **`echo`** command, such as is done in this example:

```
echo 4 2 45 > /proc/sys/kernel/acct
```

Note

Any configuration changes made using the **`echo`** command disappear when the system is restarted. To make configuration changes take effect after the system is rebooted, refer to Section 5.4, "Using the **`sysctl`** Command".

The **`/proc/sys/`** directory contains several subdirectories controlling different aspects of a running kernel.

5.3.9.1. `/proc/sys/dev/`

This directory provides parameters for particular devices on the system. Most systems have at least two directories, **`cdrom/`** and **`raid/`**. Customized kernels can have other directories, such as **`parport/`**, which provides the ability to share one parallel port between multiple device drivers.

The **`cdrom/`** directory contains a file called **`info`**, which reveals a number of important CD-ROM parameters:

```
CD-ROM information, Id: cdrom.c 3.20 2003/12/17
drive name:             hdc
drive speed:            48
drive # of slots:       1
Can close tray:         1
Can open tray:          1
Can lock tray:          1
Can change speed:       1
Can select disk:        0
Can read multisession:  1
Can read MCN:           1
Reports media changed:  1
Can play audio:         1
Can write CD-R:         0
Can write CD-RW:        0
Can read DVD:           0
Can write DVD-R:        0
Can write DVD-RAM:      0
Can read MRW:           0
Can write MRW:          0
Can write RAM:          0
```

This file can be quickly scanned to discover the qualities of an unknown CD-ROM. If multiple CD-ROMs are available on a system, each device is given its own column of information.

Various files in **`/proc/sys/dev/cdrom`**, such as **`autoclose`** and **`checkmedia`**, can be used to control the system's CD-ROM. Use the **`echo`** command to enable or disable these features.

If RAID support is compiled into the kernel, a **`/proc/sys/dev/raid/`** directory becomes available with at least two files in it: **`speed_limit_min`** and **`speed_limit_max`**. These settings determine the acceleration of RAID devices for I/O intensive tasks, such as resyncing the disks.

5.3.9.2. `/proc/sys/fs/`

This directory contains an array of options and information concerning various aspects of the file system, including quota, file handle, inode, and dentry information.

The **`binfmt_misc/`** directory is used to provide kernel support for miscellaneous binary formats.

The important files in **`/proc/sys/fs/`** include:

- **`dentry-state`** — Provides the status of the directory cache. The file looks similar to the following:

  ```
  57411 52939 45 0 0 0
  ```

  The first number reveals the total number of directory cache entries, while the second number displays the number of unused entries. The third number tells the number of seconds between when a directory has been freed and when it can be reclaimed, and the fourth measures the pages currently requested by the system. The last two numbers are not used and display only zeros.

- **`dquot-nr`** — Lists the maximum number of cached disk quota entries.

- **`file-max`** — Lists the maximum number of file handles that the kernel allocates. Raising the value in this file can resolve errors caused by a lack of available file handles.

- **`file-nr`** — Lists the number of allocated file handles, used file handles, and the maximum number of file handles.

- **`overflowgid`** and **`overflowuid`** — Defines the fixed group ID and user ID, respectively, for use with file systems that only support 16-bit group and user IDs.

- **`super-max`** — Controls the maximum number of superblocks available.

- **`super-nr`** — Displays the current number of superblocks in use.

5.3.9.3. `/proc/sys/kernel/`

This directory contains a variety of different configuration files that directly affect the operation of the kernel. Some of the most important files include:

- **`acct`** — Controls the suspension of process accounting based on the percentage of free space available on the file system containing the log. By default, the file looks like the following:

  ```
  4 2 30
  ```

The first value dictates the percentage of free space required for logging to resume, while the second value sets the threshold percentage of free space when logging is suspended. The third value sets the interval, in seconds, that the kernel polls the file system to see if logging should be suspended or resumed.

- **`cap-bound`** — Controls the *capability bounding* settings, which provides a list of capabilities for any process on the system. If a capability is not listed here, then no process, no matter how privileged, can do it. The idea is to make the system more secure by ensuring that certain things cannot happen, at least beyond a certain point in the boot process.

  For a valid list of values for this virtual file, refer to the following installed documentation:

  `/lib/modules/<`*`kernel-version`*`>/build/include/linux/capability.h`.

- **`ctrl-alt-del`** — Controls whether **`Ctrl`**+**`Alt`**+**`Delete`** gracefully restarts the computer using **`init`** (**`0`**) or forces an immediate reboot without syncing the dirty buffers to disk (**`1`**).

- **`domainname`** — Configures the system domain name, such as **`example.com`**.

- **`exec-shield`** — Configures the Exec Shield feature of the kernel. Exec Shield provides protection against certain types of buffer overflow attacks.

  There are two possible values for this virtual file:

  - **`0`** — Disables Exec Shield.
  - **`1`** — Enables Exec Shield. This is the default value.

  Important

  If a system is running security-sensitive applications that were started while Exec Shield was disabled, these applications must be restarted when Exec Shield is enabled in order for Exec Shield to take effect.

- **`exec-shield-randomize`** — Enables location randomization of various items in memory. This helps deter potential attackers from locating programs and daemons in memory. Each time a program or daemon starts, it is put into a different memory location each time, never in a static or absolute memory address.

  There are two possible values for this virtual file:

  - **`0`** — Disables randomization of Exec Shield. This may be useful for application debugging purposes.
  - **`1`** — Enables randomization of Exec Shield. This is the default value. Note: The **`exec-shield`** file must also be set to **`1`** for **`exec-shield-randomize`** to be effective.

- **`hostname`** — Configures the system hostname, such as **`www.example.com`**.

- **`hotplug`** — Configures the utility to be used when a configuration change is detected by the system. This is primarily used with USB and Cardbus PCI. The default value of **`/sbin/hotplug`** should not be changed unless testing a new program to fulfill this role.

- **`modprobe`** — Sets the location of the program used to load kernel modules. The default value is **`/sbin/modprobe`** which means **`kmod`** calls it to load the module when a kernel thread calls **`kmod`**.

- **`msgmax`** — Sets the maximum size of any message sent from one process to another and is set to **`8192`** bytes by default. Be careful when raising this value, as queued messages between processes are stored in non-swappable kernel memory. Any increase in **`msgmax`** would increase RAM requirements for the system.

- **`msgmnb`** — Sets the maximum number of bytes in a single message queue. The default is **`16384`**.

- **`msgmni`** — Sets the maximum number of message queue identifiers. The default is **`16`**.

- **`osrelease`** — Lists the Linux kernel release number. This file can only be altered by changing the kernel source and recompiling.

- **`ostype`** — Displays the type of operating system. By default, this file is set to **`Linux`**, and this value can only be changed by changing the kernel source and recompiling.

- **`overflowgid`** and **`overflowuid`** — Defines the fixed group ID and user ID, respectively, for use with system calls on architectures that only support 16-bit group and user IDs.

- **`panic`** — Defines the number of seconds the kernel postpones rebooting when the system experiences a kernel panic. By default, the value is set to **`0`**, which disables automatic rebooting after a panic.

- **`printk`** — This file controls a variety of settings related to printing or logging error messages. Each error message reported by the kernel has a *loglevel* associated with it that defines the importance of the message. The loglevel values break down in this order:

  - **`0`** — Kernel emergency. The system is unusable.
  - **`1`** — Kernel alert. Action must be taken immediately.
  - **`2`** — Condition of the kernel is considered critical.
  - **`3`** — General kernel error condition.
  - **`4`** — General kernel warning condition.
  - **`5`** — Kernel notice of a normal but significant condition.
  - **`6`** — Kernel informational message.
  - **`7`** — Kernel debug-level messages.

  Four values are found in the **`printk`** file:

  ```
  6       4       1       7
  ```

  Each of these values defines a different rule for dealing with error messages. The first value, called the *console loglevel*, defines the lowest priority of messages printed to the console. (Note that, the lower the priority, the higher the loglevel number.) The second value sets the default loglevel for messages without an explicit loglevel attached to them. The third value sets the lowest possible loglevel configuration for the console loglevel. The last value sets the default value for the console loglevel.

- **`random/`** directory — Lists a number of values related to generating random numbers for the kernel.

- **`rtsig-max`** — Configures the maximum number of POSIX real-time signals that the system may have queued at any one time. The default value is **`1024`**.

- **`rtsig-nr`** — Lists the current number of POSIX real-time signals queued by the kernel.

- **`sem`** — Configures *semaphore* settings within the kernel. A semaphore is a System V IPC object that is used to control utilization of a particular process.
- **`shmall`**— Sets the total amount of shared memory pages that can be used at one time, system-wide. By default, this value is **`2097152`**.
- **`shmmax`** — Sets the largest shared memory segment size allowed by the kernel. By default, this value is **`33554432`**. However, the kernel supports much larger values than this.
- **`shmmni`** — Sets the maximum number of shared memory segments for the whole system. By default, this value is **`4096`**.
- **`sysrq`** — Activates the System Request Key, if this value is set to anything other than zero (**`0`**), the default.

  The System Request Key allows immediate input to the kernel through simple key combinations. For example, the System Request Key can be used to immediately shut down or restart a system, sync all mounted file systems, or dump important information to the console. To initiate a System Request Key, type **`Alt`**+**SysRq**+ ***`<system request code>`*** . Replace <*system request code*> with one of the following system request codes:

  - **`r`** — Disables raw mode for the keyboard and sets it to XLATE (a limited keyboard mode which does not recognize modifiers such as **`Alt`**, **`Ctrl`**, or **`Shift`** for all keys).
  - **`k`** — Kills all processes active in a virtual console. Also called *Secure Access Key* (*SAK*), it is often used to verify that the login prompt is spawned from **`init`** and not a Trojan copy designed to capture usernames and passwords.
  - **`b`** — Reboots the kernel without first unmounting file systems or syncing disks attached to the system.
  - **`c`** — Crashes the system without first unmounting file systems or syncing disks attached to the system.
  - **`o`** — Shuts off the system.
  - **`s`** — Attempts to sync disks attached to the system.
  - **`u`** — Attempts to unmount and remount all file systems as read-only.
  - **`p`** — Outputs all flags and registers to the console.
  - **`t`** — Outputs a list of processes to the console.
  - **`m`** — Outputs memory statistics to the console.
  - **`0`** through **`9`** — Sets the log level for the console.
  - **`e`** — Kills all processes except **`init`** using SIGTERM.
  - **`i`** — Kills all processes except **`init`** using SIGKILL.
  - **`l`** — Kills all processes using SIGKILL (including **`init`**). *The system is unusable after issuing this System Request Key code.*
  - **`h`** — Displays help text.

  This feature is most beneficial when using a development kernel or when experiencing system freezes.

> **Warning**
>
> The System Request Key feature is considered a security risk because an unattended console provides an attacker with access to the system. For this reason, it is turned off by default.

Refer to **`/usr/share/doc/kernel-doc-<version>/Documentation/sysrq.txt`** for more information about the System Request Key.

- **`sysrq-key`** — Defines the key code for the System Request Key (**`84`** is the default).
- **`sysrq-sticky`** — Defines whether the System Request Key is a chorded key combination. The accepted values are as follows:
  - **`0`** — **`Alt`**+**`SysRq`** and the system request code must be pressed simultaneously. This is the default value.
  - **`1`** — **`Alt`**+**`SysRq`** must be pressed simultaneously, but the system request code can be pressed anytime before the number of seconds specified in **`/proc/sys/kernel/sysrq-timer`** elapses.
- **`sysrq-timer`** — Specifies the number of seconds allowed to pass before the system request code must be pressed. The default value is **`10`**.
- **`tainted`** — Indicates whether a non-GPL module is loaded.
  - **`0`** — No non-GPL modules are loaded.
  - **`1`** — At least one module without a GPL license (including modules with no license) is loaded.
  - **`2`** — At least one module was force-loaded with the command **`insmod -f`**.
- **`threads-max`** — Sets the maximum number of threads to be used by the kernel, with a default value of **`2048`**.
- **`version`** — Displays the date and time the kernel was last compiled. The first field in this file, such as **`#3`**, relates to the number of times a kernel was built from the source base.

5.3.9.4. `/proc/sys/net/`

This directory contains subdirectories concerning various networking topics. Various configurations at the time of kernel compilation make different directories available here, such as **`ethernet/`**, **`ipv4/`**, **`ipx/`**, and **`ipv6/`**. By altering the files within these directories, system administrators are able to adjust the network configuration on a running system.

Given the wide variety of possible networking options available with Linux, only the most common **`/proc/sys/net/`** directories are discussed.

The **`/proc/sys/net/core/`** directory contains a variety of settings that control the interaction between the kernel and networking layers. The most important of these files are:

- **`message_burst`** — Sets the amount of time in tenths of a second required to write a new warning message. This setting is used to mitigate *Denial of Service* (*DoS*) attacks. The default setting is **`50`**.
- **`message_cost`** — Sets a cost on every warning message. The higher the value of this file (default of **`5`**), the more likely the warning message is ignored. This setting is used to mitigate DoS attacks.

The idea of a DoS attack is to bombard the targeted system with requests that generate errors and fill up disk partitions with log files or require all of the system's resources to handle the error logging. The settings in **`message_burst`** and **`message_cost`** are designed to be modified based on the system's acceptable risk versus the need for comprehensive logging.

- **`netdev_max_backlog`** — Sets the maximum number of packets allowed to queue when a particular interface receives packets faster than the kernel can process them. The default value for this file is **`300`**.
- **`optmem_max`** — Configures the maximum ancillary buffer size allowed per socket.
- **`rmem_default`** — Sets the receive socket buffer default size in bytes.
- **`rmem_max`** — Sets the receive socket buffer maximum size in bytes.
- **`wmem_default`** — Sets the send socket buffer default size in bytes.
- **`wmem_max`** — Sets the send socket buffer maximum size in bytes.

The **`/proc/sys/net/ipv4/`** directory contains additional networking settings. Many of these settings, used in conjunction with one another, are useful in preventing attacks on the system or when using the system to act as a router.

Warning

An erroneous change to these files may affect remote connectivity to the system.

The following is a list of some of the more important files within the **`/proc/sys/net/ipv4/`** directory:

- **`icmp_destunreach_rate`**, **`icmp_echoreply_rate`**, **`icmp_paramprob_rate`**, and **`icmp_timeexeed_rate`** — Set the maximum ICMP send packet rate, in 1/100 of a second, to hosts under certain conditions. A setting of **`0`** removes any delay and is not a good idea.
- **`icmp_echo_ignore_all`** and **`icmp_echo_ignore_broadcasts`** — Allows the kernel to ignore ICMP ECHO packets from every host or only those originating from broadcast and multicast addresses, respectively. A value of **`0`** allows the kernel to respond, while a value of **`1`** ignores the packets.
- **`ip_default_ttl`** — Sets the default *Time To Live (TTL)*, which limits the number of hops a packet may make before reaching its destination. Increasing this value can diminish system performance.
- **`ip_forward`** — Permits interfaces on the system to forward packets to one other. By default, this file is set to **`0`**. Setting this file to **`1`** enables network packet forwarding.
- **`ip_local_port_range`** — Specifies the range of ports to be used by TCP or UDP when a local port is needed. The first number is the lowest port to be used and the second number specifies the highest port. Any systems that expect to require more ports than the default 1024 to 4999 should use a range from 32768 to 61000.
- **`tcp_syn_retries`** — Provides a limit on the number of times the system re-transmits a SYN packet when attempting to make a connection.
- **`tcp_retries1`** — Sets the number of permitted re-transmissions attempting to answer an incoming connection. Default of **`3`**.
- **`tcp_retries2`** — Sets the number of permitted re-transmissions of TCP packets. Default of **`15`**.

The file called

```
/usr/share/doc/kernel-doc-<version>/Documentation/networking/ ip-sysctl.txt
```

contains a complete list of files and options available in the **`/proc/sys/net/ipv4/`** directory.

A number of other directories exist within the **`/proc/sys/net/ipv4/`** directory and each covers a different aspect of the network stack. The **`/proc/sys/net/ipv4/conf/`** directory allows each system interface to be configured in different ways, including the use of default settings for unconfigured devices (in the **`/proc/sys/net/ipv4/conf/default/`** subdirectory) and settings that override all special configurations (in the **`/proc/sys/net/ipv4/conf/all/`** subdirectory).

The **`/proc/sys/net/ipv4/neigh/`** directory contains settings for communicating with a host directly connected to the system (called a network neighbor) and also contains different settings for systems more than one hop away.

Routing over IPV4 also has its own directory, **`/proc/sys/net/ipv4/route/`**. Unlike **`conf/`** and **`neigh/`**, the **`/proc/sys/net/ipv4/route/`** directory contains specifications that apply to routing with any interfaces on the system. Many of these settings, such as **`max_size`**, **`max_delay`**, and **`min_delay`**, relate to controlling the size of the routing cache. To clear the routing cache, write any value to the **`flush`** file.

Additional information about these directories and the possible values for their configuration files can be found in:

```
/usr/share/doc/kernel-doc-<version>/Documentation/filesystems/proc.txt
```

5.3.9.5. `/proc/sys/vm/`

This directory facilitates the configuration of the Linux kernel's virtual memory (VM) subsystem. The kernel makes extensive and intelligent use of virtual memory, which is commonly referred to as swap space.

The following files are commonly found in the **`/proc/sys/vm/`** directory:

- **`block_dump`** — Configures block I/O debugging when enabled. All read/write and block dirtying operations done to files are logged accordingly. This can be useful if diagnosing disk spin up and spin downs for laptop battery conservation. All output when **`block_dump`** is enabled can be retrieved via **`dmesg`**. The default value is **`0`**.

  > **Note**
  >
  > If **`block_dump`** is enabled at the same time as kernel debugging, it is prudent to stop the **`klogd`** daemon, as it generates erroneous disk activity caused by **`block_dump`**.

- **`dirty_background_ratio`** — Starts background writeback of dirty data at this percentage of total memory, via a pdflush daemon. The default value is **`10`**.

- **`dirty_expire_centisecs`** — Defines when dirty in-memory data is old enough to be eligible for writeout. Data which has been dirty in-memory for longer than this interval is written out next time a pdflush daemon wakes up. The default value is **`3000`**, expressed in hundredths of a second.

- **`dirty_ratio`** — Starts active writeback of dirty data at this percentage of total memory for the generator of dirty data, via pdflush. The default value is **`40`**.

- **`dirty_writeback_centisecs`** — Defines the interval between pdflush daemon wakeups, which periodically writes dirty in-memory data out to disk. The default value is **`500`**, expressed in hundredths of a second.

- **`laptop_mode`** — Minimizes the number of times that a hard disk needs to spin up by keeping the disk spun down for as long as possible, therefore conserving battery power on laptops. This increases efficiency by combining all future I/O processes together, reducing the frequency of spin ups. The default value is **`0`**, but is automatically enabled in case a battery on a laptop is used.

  This value is controlled automatically by the acpid daemon once a user is notified battery power is enabled. No user modifications or interactions are necessary if the laptop supports the ACPI (Advanced Configuration and Power Interface) specification.

  For more information, refer to the following installed documentation:

  `/usr/share/doc/kernel-doc-<`*`version`*`>/Documentation/laptop-mode.txt`

- **`lower_zone_protection`** — Determines how aggressive the kernel is in defending lower memory allocation zones. This is effective when utilized with machines configured with **`highmem`** memory space enabled. The default value is **`0`**, no protection at all. All other integer values are in megabytes, and **`lowmem`** memory is therefore protected from being allocated by users.

  For more information, refer to the following installed documentation:

  `/usr/share/doc/kernel-doc-<`*`version`*`>/Documentation/filesystems/proc.txt`

- **`max_map_count`** — Configures the maximum number of memory map areas a process may have. In most cases, the default value of **`65536`** is appropriate.

- **`min_free_kbytes`** — Forces the Linux VM (virtual memory manager) to keep a minimum number of kilobytes free. The VM uses this number to compute a **`pages_min`** value for each **`lowmem`** zone in the system. The default value is in respect to the total memory on the machine.

- **`nr_hugepages`** — Indicates the current number of configured **`hugetlb`** pages in the kernel.

  For more information, refer to the following installed documentation:

  `/usr/share/doc/kernel-doc-<`*`version`*`>/Documentation/vm/hugetlbpage.txt`

- **`nr_pdflush_threads`** — Indicates the number of pdflush daemons that are currently running. This file is read-only, and should not be changed by the user. Under heavy I/O loads, the default value of two is increased by the kernel.

- **`overcommit_memory`** — Configures the conditions under which a large memory request is accepted or denied. The following three modes are available:

  - **`0`** — The kernel performs heuristic memory over commit handling by estimating the amount of memory available and failing requests that are blatantly invalid. Unfortunately, since memory is allocated using a heuristic rather than a precise algorithm, this setting can sometimes allow available memory on the system to be overloaded. This is the default setting.

  - **`1`** — The kernel performs no memory over commit handling. Under this setting, the potential for memory overload is increased, but so is performance for memory intensive tasks (such as those executed by some scientific software).

  - **`2`** — The kernel fails requests for memory that add up to all of swap plus the percent of physical RAM specified in **`/proc/sys/vm/overcommit_ratio`**. This setting is best for those who desire less risk of memory overcommitment.

Note

This setting is only recommended for systems with swap areas larger than physical memory.

- **`overcommit_ratio`** — Specifies the percentage of physical RAM considered when **`/proc/sys/vm/overcommit_memory`** is set to **`2`**. The default value is **`50`**.
- **`page-cluster`** — Sets the number of pages read in a single attempt. The default value of **`3`**, which actually relates to 16 pages, is appropriate for most systems.
- **`swappiness`** — Determines how much a machine should swap. The higher the value, the more swapping occurs. The default value, as a percentage, is set to **`60`**.

All kernel-based documentation can be found in the following locally installed location:

`/usr/share/doc/kernel-doc-<`*`version`*`>/Documentation/`, which contains additional information.

5.3.10. `/proc/sysvipc/`

This directory contains information about System V IPC resources. The files in this directory relate to System V IPC calls for messages (**`msg`**), semaphores (**`sem`**), and shared memory (**`shm`**).

5.3.11. `/proc/tty/`

This directory contains information about the available and currently used *tty devices* on the system. Originally called *teletype devices*, any character-based data terminals are called tty devices.

In Linux, there are three different kinds of tty devices. *Serial devices* are used with serial connections, such as over a modem or using a serial cable. *Virtual terminals* create the common console connection, such as the virtual consoles available when pressing **`Alt`**+**`<F-key>`** at the system console. *Pseudo terminals* create a two-way communication that is used by some higher level applications, such as XFree86. The **`drivers`** file is a list of the current tty devices in use, as in the following example:

```
serial                /dev/cua        5   64-127 serial:callout
serial                /dev/ttyS       4   64-127 serial
pty_slave             /dev/pts      136    0-255 pty:slave
pty_master            /dev/ptm      128    0-255 pty:master
pty_slave             /dev/ttyp       3    0-255 pty:slave
pty_master            /dev/pty        2    0-255 pty:master
/dev/vc/0             /dev/vc/0       4        0 system:vtmaster
/dev/ptmx             /dev/ptmx       5        2 system
/dev/console          /dev/console    5        1 system:console
/dev/tty              /dev/tty        5        0 system:/dev/tty
unknown               /dev/vc/%d      4     1-63 console
```

The **`/proc/tty/driver/serial`** file lists the usage statistics and status of each of the serial tty lines.

In order for tty devices to be used as network devices, the Linux kernel enforces *line discipline* on the device. This allows the driver to place a specific type of header with every block of data transmitted over the device, making it possible for the remote end of the connection to a block of data as just one in a stream of data blocks. SLIP and PPP are common line disciplines, and each are commonly used to connect systems to one other over a serial link.

Registered line disciplines are stored in the **`ldiscs`** file, and more detailed information is available within the

ldisc/ directory.

5.3.12. /proc/<PID>/

Out of Memory (OOM) refers to a computing state where all available memory, including swap space, has been allocated. When this situation occurs, it will cause the system to panic and stop functioning as expected. There is a switch that controls OOM behavior in **/proc/sys/vm/panic_on_oom**. When set to **1** the kernel will panic on OOM. A setting of **0** instructs the kernel to call a function named **oom_killer** on an OOM. Usually, **oom_killer** can kill rogue processes and the system will survive.

The easiest way to change this is to echo the new value to **/proc/sys/vm/panic_on_oom**.

```
~]# cat /proc/sys/vm/panic_on_oom
1
~]# echo 0 > /proc/sys/vm/panic_on_oom
~]# cat /proc/sys/vm/panic_on_oom
0
```

It is also possible to prioritize which processes get killed by adjusting the **oom_killer** score. In **/proc/<PID>/** there are two tools labelled **oom_adj** and **oom_score**. Valid scores for **oom_adj** are in the range -16 to +15. To see the current **oom_killer** score, view the **oom_score** for the process. **oom_killer** will kill processes with the highest scores first.

This example adjusts the oom_score of a process with a PID of 12465 to make it less likely that **oom_killer** will kill it.

```
~]# cat /proc/12465/oom_score
79872
~]# echo -5 > /proc/12465/oom_adj
~]# cat /proc/12465/oom_score
78
```

There is also a special value of -17, which disables **oom_killer** for that process. In the example below, **oom_score** returns a value of 0, indicating that this process would not be killed.

```
~]# cat /proc/12465/oom_score
78
~]# echo -17 > /proc/12465/oom_adj
~]# cat /proc/12465/oom_score
0
```

A function called **badness()** is used to determine the actual score for each process. This is done by adding up 'points' for each examined process. The process scoring is done in the following way:

1. The basis of each process's score is its memory size.
2. The memory size of any of the process's children (not including a kernel thread) is also added to the score
3. The process's score is increased for 'niced' processes and decreased for long running processes.
4. Processes with the **CAP_SYS_ADMIN** and **CAP_SYS_RAWIO** capabilities have their scores reduced.
5. The final score is then bitshifted by the value saved in the **oom_adj** file.

Thus, a process with the highest **oom_score** value will most probably be a non-privileged, recently started process that, along with its children, uses a large amount of memory, has been 'niced', and handles no raw I/O.

5.4. Using the `sysctl` Command

The **/sbin/sysctl** command is used to view, set, and automate kernel settings in the **/proc/sys/** directory.

For a quick overview of all settings configurable in the **/proc/sys/** directory, type the **/sbin/sysctl -a** command as root. This creates a large, comprehensive list, a small portion of which looks something like the following:

```
net.ipv4.route.min_delay = 2 kernel.sysrq = 0 kernel.sem = 250     32000
32     128
```

This is the same information seen if each of the files were viewed individually. The only difference is the file location. For example, the **/proc/sys/net/ipv4/route/min_delay** file is listed as **net.ipv4.route.min_delay**, with the directory slashes replaced by dots and the **proc.sys** portion assumed.

The **sysctl** command can be used in place of **echo** to assign values to writable files in the **/proc/sys/** directory. For example, instead of using the command

```
echo 1 > /proc/sys/kernel/sysrq
```

use the equivalent **sysctl** command as follows:

```
~]# sysctl -w kernel.sysrq="1"
kernel.sysrq = 1
```

While quickly setting single values like this in **/proc/sys/** is helpful during testing, this method does not work as well on a production system as special settings within **/proc/sys/** are lost when the machine is rebooted. To preserve custom settings, add them to the **/etc/sysctl.conf** file.

Each time the system boots, the **init** program runs the **/etc/rc.d/rc.sysinit** script. This script contains a command to execute **sysctl** using **/etc/sysctl.conf** to determine the values passed to the kernel. Any values added to **/etc/sysctl.conf** therefore take effect each time the system boots.

5.5. Additional Resources

Below are additional sources of information about **proc** file system.

5.5.1. Installed Documentation

Some of the best documentation about the **proc** file system is installed on the system by default.

- **/usr/share/doc/kernel-doc-<*version*>/Documentation/filesystems/proc.txt** — Contains assorted, but limited, information about all aspects of the **/proc/** directory.

- **/usr/share/doc/kernel-doc-<*version*>/Documentation/sysrq.txt** — An overview of System Request Key options.

- **/usr/share/doc/kernel-doc-<*version*>/Documentation/sysctl/** — A directory containing a variety of **sysctl** tips, including modifying values that concern the kernel (**kernel.txt**), accessing file systems (**fs.txt**), and virtual memory use (**vm.txt**).
- **/usr/share/doc/kernel-doc-<*version*>/Documentation/networking/ip-sysctl.txt** — A detailed overview of IP networking options.

5.5.2. Useful Websites

- http://www.linuxhq.com/ — This website maintains a complete database of source, patches, and documentation for various versions of the Linux kernel.

Chapter 6. Redundant Array of Independent Disks (RAID)

The basic idea behind RAID is to combine multiple small, inexpensive disk drives into an array to accomplish performance or redundancy goals not attainable with one large and expensive drive. This array of drives appears to the computer as a single logical storage unit or drive.

6.1. What is RAID?

RAID allows information to access several disks. RAID uses techniques such as *disk striping* (RAID Level 0), *disk mirroring* (RAID Level 1), and *disk striping with parity* (RAID Level 5) to achieve redundancy, lower latency, increased bandwidth, and maximized ability to recover from hard disk crashes.

RAID consistently distributes data across each drive in the array. RAID then breaks down the data into consistently-sized chunks (commonly 32K or 64k, although other values are acceptable). Each chunk is then written to a hard drive in the RAID array according to the RAID level employed. When the data is read, the process is reversed, giving the illusion that the multiple drives in the array are actually one large drive.

6.1.1. Who Should Use RAID?

System Administrators and others who manage large amounts of data would benefit from using RAID technology. Primary reasons to deploy RAID include:

- Enhances speed
- Increases storage capacity using a single virtual disk
- Minimizes disk failure

6.1.2. Hardware RAID versus Software RAID

There are two possible RAID approaches: hardware RAID and software RAID.

Hardware RAID

The hardware-based array manages the RAID subsystem independently from the host. It presents a single disk per RAID array to the host.

A hardware RAID device connects to the SCSI controller and presents the RAID arrays as a single SCSI drive. An external RAID system moves all RAID handling “intelligence” into a controller located in the external disk subsystem. The whole subsystem is connected to the host via a normal SCSI controller and appears to the host as a single disk.

RAID controller cards function like a SCSI controller to the operating system, and handle all the actual drive communications. The user plugs the drives into the RAID controller (just like a normal SCSI controller) and then adds them to the RAID controllers configuration, and the operating system won't know the difference.

Software RAID

Software RAID implements the various RAID levels in the kernel disk (block device) code. It offers the cheapest possible solution, as expensive disk controller cards or hot-swap chassis [1] are not required. Software RAID also works with cheaper IDE disks as well as SCSI disks. With today's faster CPUs, software RAID outperforms hardware RAID.

The Linux kernel contains an MD driver that allows the RAID solution to be completely hardware independent. The performance of a software-based array depends on the server CPU performance and load.

To learn more about software RAID, here are the key features:

- Threaded rebuild process
- Kernel-based configuration
- Portability of arrays between Linux machines without reconstruction
- Backgrounded array reconstruction using idle system resources
- Hot-swappable drive support
- Automatic CPU detection to take advantage of certain CPU optimizations

6.1.3. RAID Levels and Linear Support

RAID supports various configurations, including levels 0, 1, 4, 5, and linear. These RAID types are defined as follows:

Level 0

RAID level 0, often called “striping”, is a performance-oriented striped data mapping technique. This means the data being written to the array is broken down into strips and written across the member disks of the array, allowing high I/O performance at low inherent cost but provides no redundancy. The storage capacity of a level 0 array is equal to the total capacity of the member disks in a hardware RAID or the total capacity of member partitions in a software RAID.

Level 1

RAID level 1, or “mirroring”, has been used longer than any other form of RAID. Level 1 provides redundancy by writing identical data to each member disk of the array, leaving a “mirrored” copy on each disk. Mirroring remains popular due to its simplicity and high level of data availability. Level 1 operates with two or more disks that may use parallel access for high data-transfer rates when reading but more commonly operate independently to provide high I/O transaction rates. Level 1 provides very good data reliability and improves performance for read-intensive applications but at a relatively high cost. The storage capacity of the level 1 array is equal to the capacity of one of the mirrored hard disks in a hardware RAID or one of the mirrored partitions in a software RAID.

Note

RAID level 1 comes at a high cost because you write the same information to all of the disks in the array, which wastes drive space. For example, if you have RAID level 1 set up so that your root (`/`) partition exists on two 40G drives, you have 80G total but are only able to access 40G of that 80G. The other 40G acts like a mirror of the first 40G.

Level 4

RAID level 4 uses parity [2] concentrated on a single disk drive to protect data. It is better suited to transaction I/O rather than large file transfers. Because the dedicated parity disk represents an inherent bottleneck, level 4 is seldom used without accompanying technologies such as write-back caching. Although RAID level 4 is an option in some RAID partitioning schemes, it is not an option allowed in Red Hat Enterprise Linux RAID installations. The storage capacity of hardware RAID level 4 is equal to the capacity of member disks, minus the capacity of one member disk. The storage capacity of software RAID level 4 is equal to the capacity of the member partitions, minus the size of one of the partitions if they are of equal size.

Note

RAID level 4 takes up the same amount of space as RAID level 5, but level 5 has more advantages. For this reason, level 4 is not supported.

Level 5

RAID level 5 is the most common type of RAID. By distributing parity across some or all of an array's member disk drives, RAID level 5 eliminates the write bottleneck inherent in level 4. The only performance bottleneck is the parity calculation process. With modern CPUs and software RAID, that usually is not a very big problem. As with level 4, the result is asymmetrical performance, with reads substantially outperforming writes. Level 5 is often used with write-back caching to reduce the asymmetry. The storage capacity of hardware RAID level 5 is equal to the capacity of member disks, minus the capacity of one member disk. The storage capacity of software RAID level 5 is equal to the capacity of the member partitions, minus the size of one of the partitions if they are of equal size.

Linear RAID

Linear RAID is a simple grouping of drives to create a larger virtual drive. In linear RAID, the chunks are allocated sequentially from one member drive, going to the next drive only when the first is completely filled. This grouping provides no performance benefit, as it is unlikely that any I/O operations will be split between member drives. Linear RAID also offers no redundancy and, in fact, decreases reliability — if any one member drive fails, the entire array cannot be used. The capacity is the total of all member disks.

6.2. Configuring Software RAID

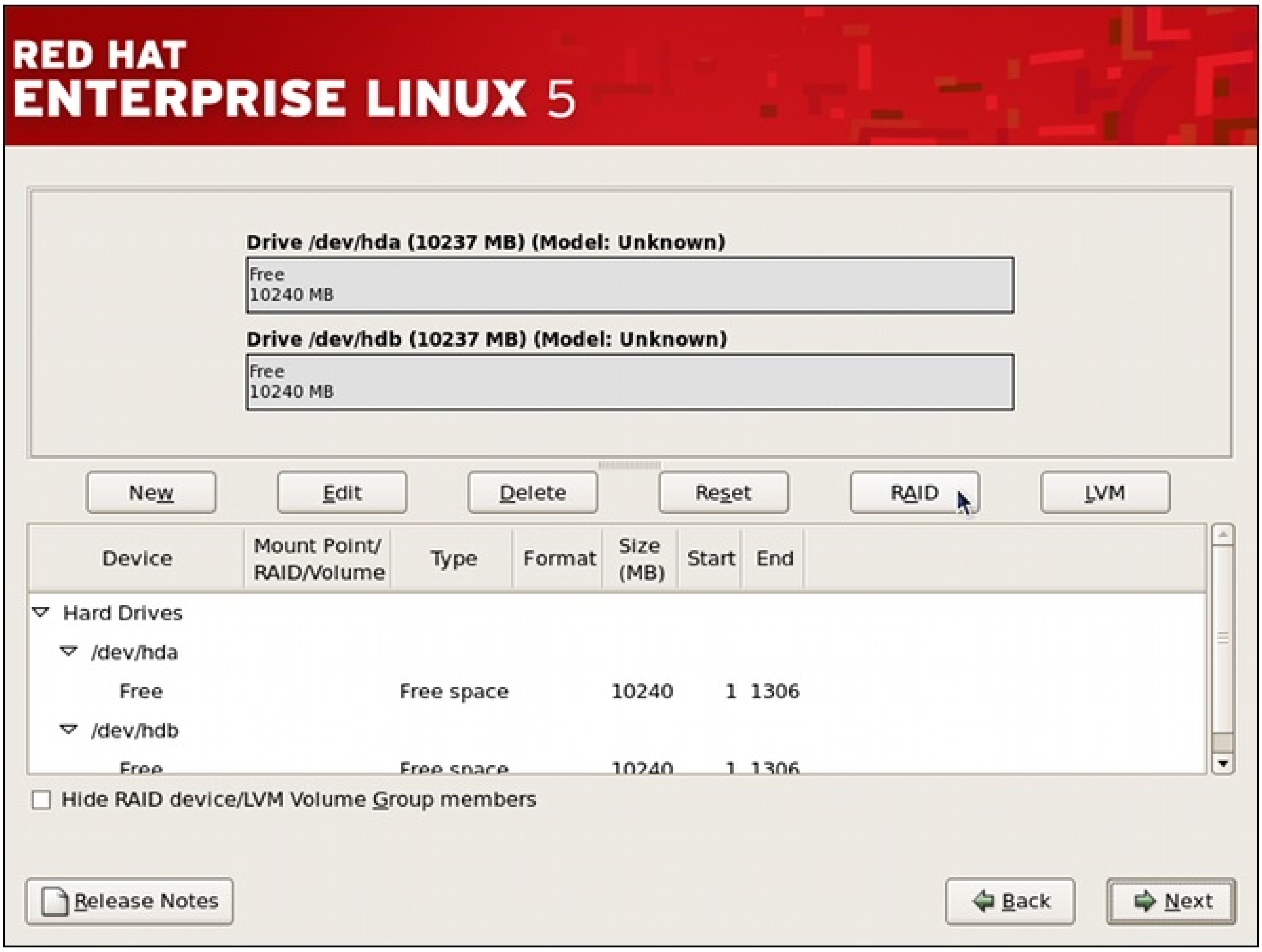
Users can configure software RAID during the graphical installation process, the text-based installation process, or during a kickstart installation. This section discusses software RAID configuration during the installation process using the **Disk Druid** application, and covers the following steps:

1. Creating *software RAID partitions* on physical hard drives.
2. Creating *RAID devices* from the software RAID partitions.
3. (Optional) Configuring *LVM* from the RAID devices.
4. Creating *file systems* from the RAID devices.

To configure software RAID, select **Create custom layout** from the pulldown list on the **`Disk Partitioning Setup`** screen, click the **`Next`** button, and follow the instructions in the rest of this section. The example screenshots in this section use two 10 GB disk drives (**`/dev/hda`** and **`/dev/hdb`**) to illustrate the creation of simple RAID 1 and RAID 0 configurations, and detail how to create a simple RAID configuration by implementing multiple RAID devices.

6.2.1. Creating the RAID Partitions

In a typical situation, the disk drives are new or are formatted. Both drives are shown as raw devices with no partition configuration in Figure 6.1, “Two Blank Drives, Ready For Configuration”.



Drive /dev/hda (10237 MB) (Model: Unknown)
Free
10240 MB

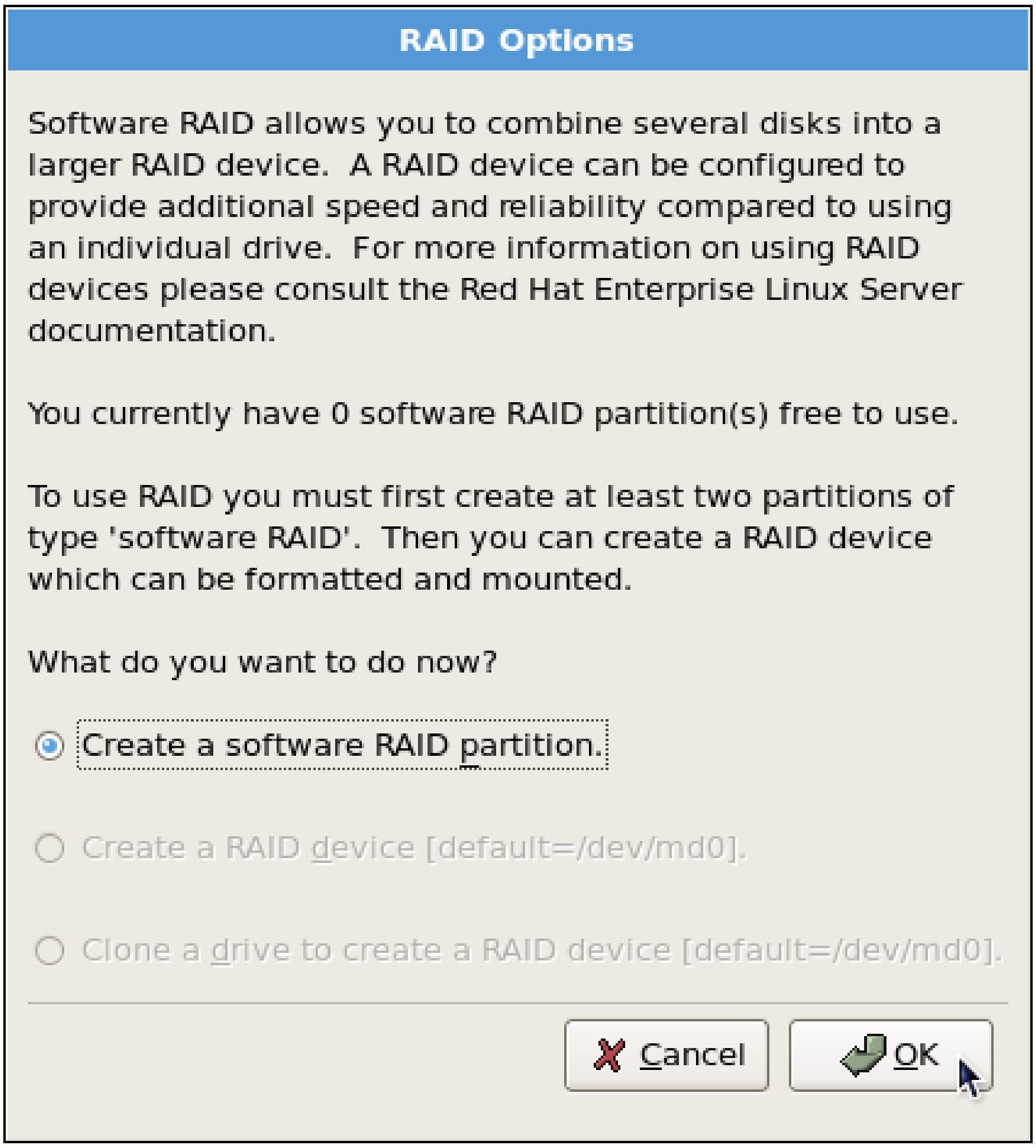
Drive /dev/hdb (10237 MB) (Model: Unknown)
Free
10240 MB

Device	Mount Point/ RAID/Volume	Type	Format	Size (MB)	Start	End
Hard Drives						
/dev/hda						
Free		Free space		10240	1	1306
/dev/hdb						
Free		Free space		10240	1	1306

☐ Hide RAID device/LVM Volume Group members

Figure 6.1. Two Blank Drives, Ready For Configuration

1. In **Disk Druid**, click the **`RAID`** button to enter the software RAID creation screen.
2. Choose **Create a software RAID partition** to create a RAID partition as shown in Figure 6.2, “RAID Partition Options”. Note that no other RAID options (such as entering a mount point) are available until RAID partitions, as well as RAID devices, are created. Click **`OK`** to confirm the choice.


RAID Options

Software RAID allows you to combine several disks into a larger RAID device. A RAID device can be configured to provide additional speed and reliability compared to using an individual drive. For more information on using RAID devices please consult the Red Hat Enterprise Linux Server documentation.

You currently have 0 software RAID partition(s) free to use.

To use RAID you must first create at least two partitions of type 'software RAID'. Then you can create a RAID device which can be formatted and mounted.

What do you want to do now?

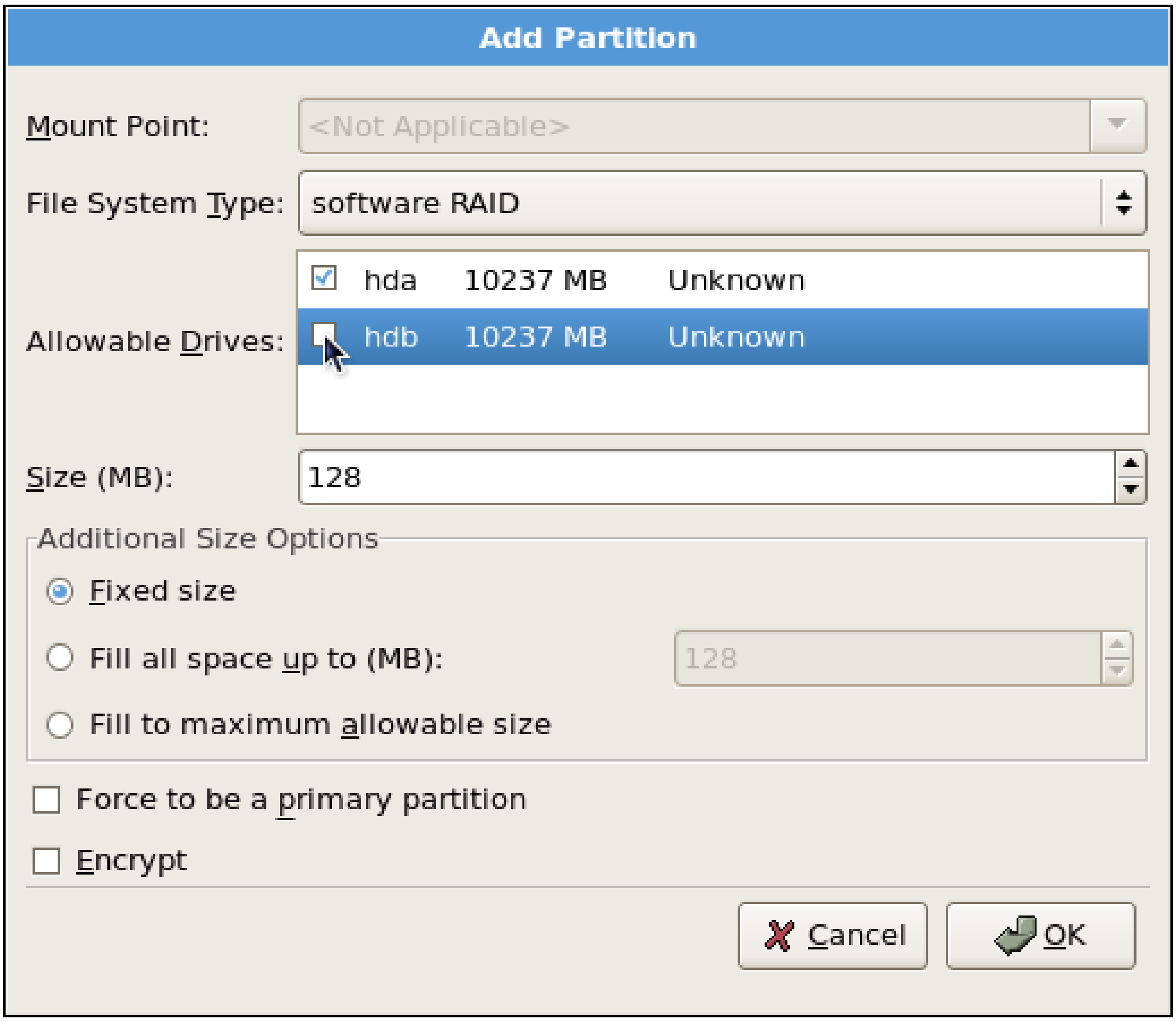
[✓ Create a software RAID partition.]

Create a RAID device [default=/dev/md0].

Clone a drive to create a RAID device [default=/dev/md0].

Figure 6.2. RAID Partition Options

3. A software RAID partition must be constrained to one drive. For **Allowable Drives**, select the drive to use for RAID. If you have multiple drives, by default all drives are selected and you must deselect the drives you do not want.


Add Partition
Mount Point: <Not Applicable>
File System Type: software RAID
Allowable Drives:
Size (MB): 128
Additional Size Options
Fixed size
Fill all space up to (MB): 128
Fill to maximum allowable size
Force to be a primary partition
Encrypt
Cancel
OK

✓	hda	10237 MB	Unknown
☐	hdb	10237 MB	Unknown

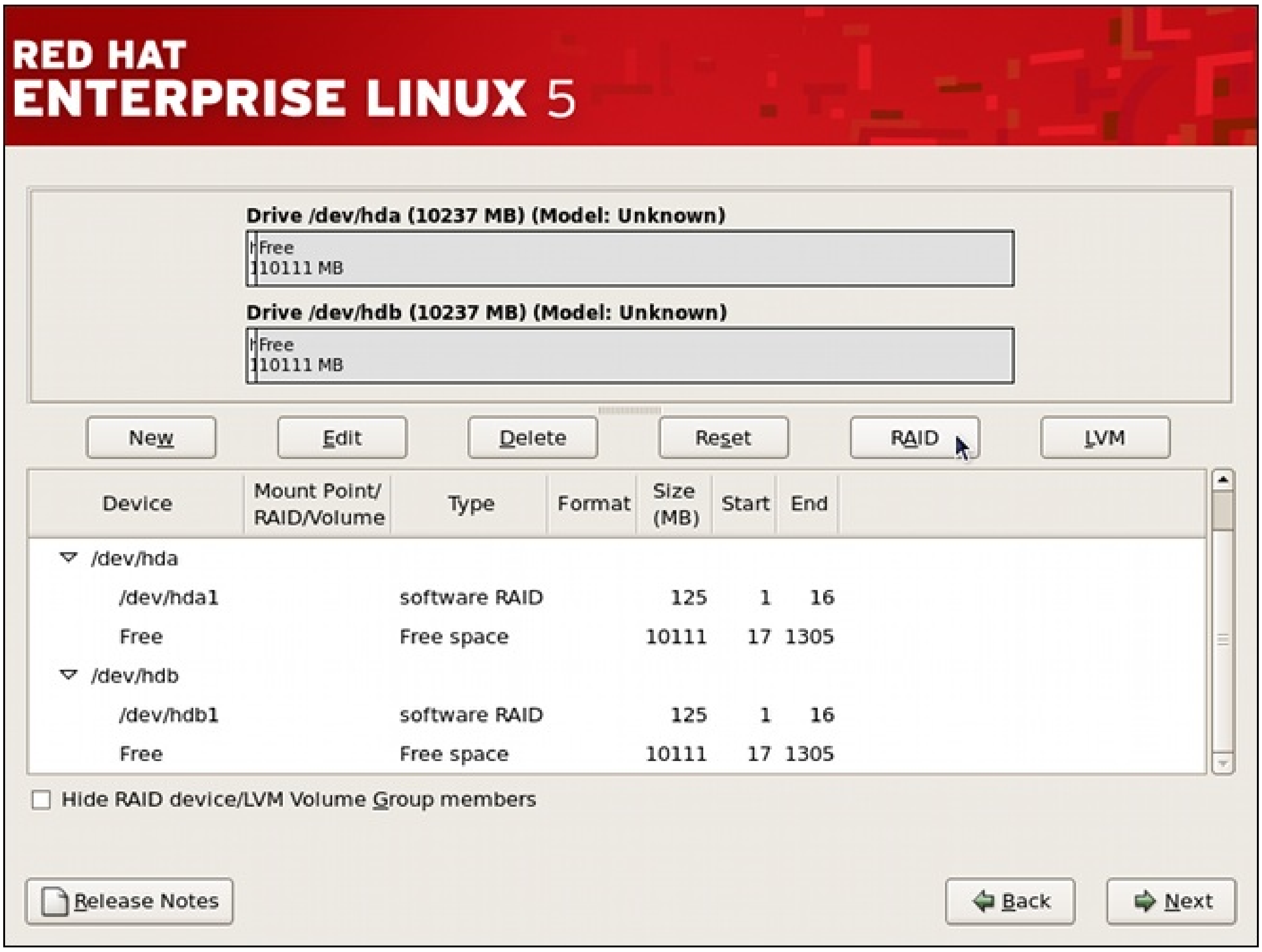
Figure 6.3. Adding a RAID Partition

4. Edit the **`Size (MB)`** field, and enter the size that you want the partition to be (in MB).

5. Select **`Fixed Size`** to specify partition size. Select **`Fill all space up to (MB)`** and enter a value (in MB) to specify partition size range. Select **`Fill to maximum allowable size`** to allow maximum available space of the hard disk. Note that if you make more than one space growable, they share the available free space on the disk.

6. Select **`Force to be a primary partition`** if you want the partition to be a primary partition. A primary partition is one of the first four partitions on the hard drive. If unselected, the partition is created as a logical partition. If other operating systems are already on the system, unselecting this option should be considered. For more information on primary versus logical/extended partitions, refer to the appendix section of the *Red Hat Enterprise Linux Installation Guide*.

Repeat these steps to create as many partitions as needed for your RAID setup. Notice that all the partitions do not have to be RAID partitions. For example, you can configure only the **`/boot`** partition as a software RAID device, leaving the root partition (`/`), **`/home`**, and **`swap`** as regular file systems. Figure 6.4, “RAID 1 Partitions Ready, Pre-Device and Mount Point Creation” shows successfully allocated space for the RAID 1 configuration (for **`/boot`**), which is now ready for RAID device and mount point creation:


RED HAT
ENTERPRISE LINUX 5
Drive /dev/hda (10237 MB) (Model: Unknown)
Free
10111 MB
Drive /dev/hdb (10237 MB) (Model: Unknown)
Free
10111 MB
New
Edit
Delete
Reset
RAID
LVM
Hide RAID device/LVM Volume Group members
Release Notes
Back
Next

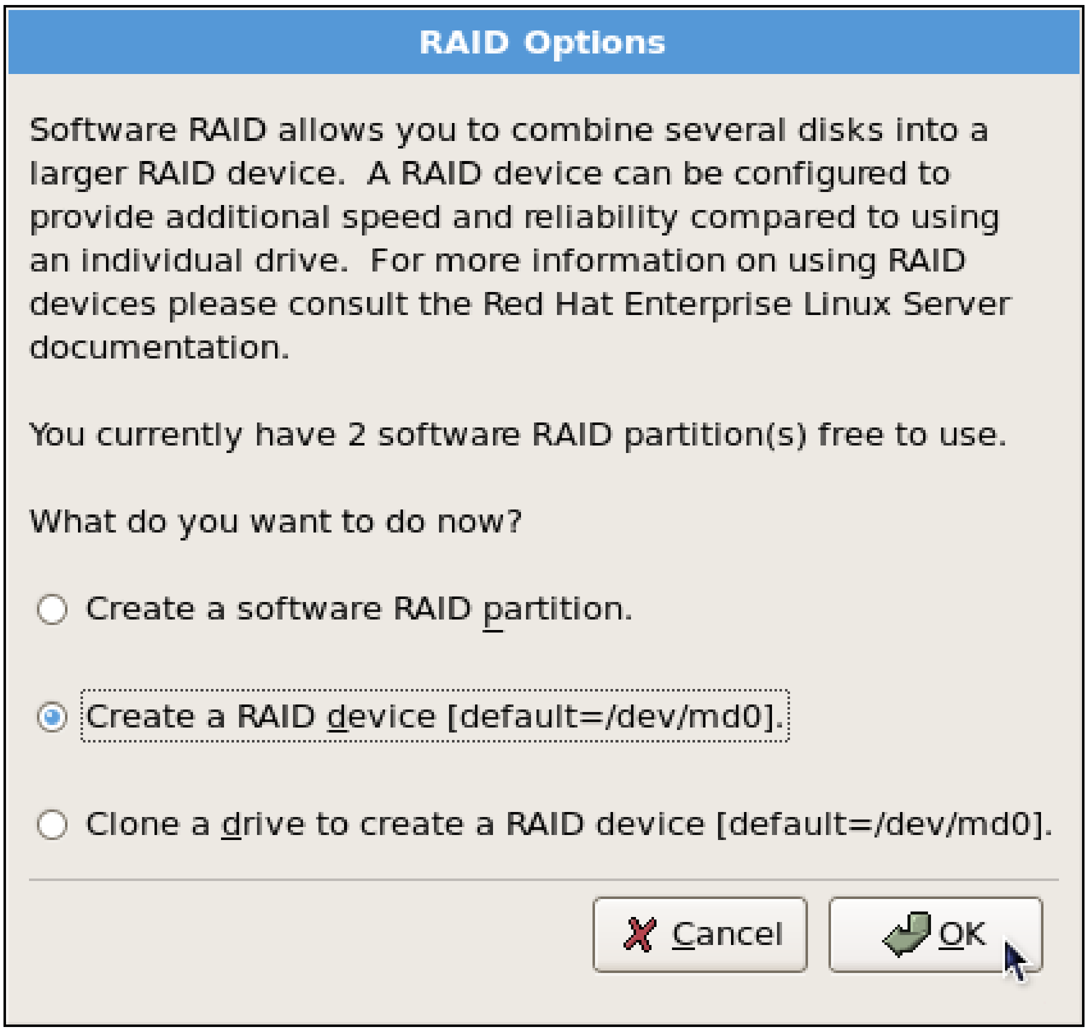
Device	Mount Point/ RAID/Volume	Type	Format	Size (MB)	Start	End
/dev/hda						
/dev/hda1		software RAID		125	1	16
Free		Free space		10111	17	1305
/dev/hdb						
/dev/hdb1		software RAID		125	1	16
Free		Free space		10111	17	1305

Figure 6.4. RAID 1 Partitions Ready, Pre-Device and Mount Point Creation

6.2.2. Creating the RAID Devices and Mount Points

Once you create all of your partitions as software RAID partitions, you must create the RAID device and mount point.

1. On the main partitioning screen, click the **`RAID`** button. The **`RAID Options`** dialog appears as shown in Figure 6.5, “RAID Options”.


RAID Options

Software RAID allows you to combine several disks into a larger RAID device. A RAID device can be configured to provide additional speed and reliability compared to using an individual drive. For more information on using RAID devices please consult the Red Hat Enterprise Linux Server documentation.

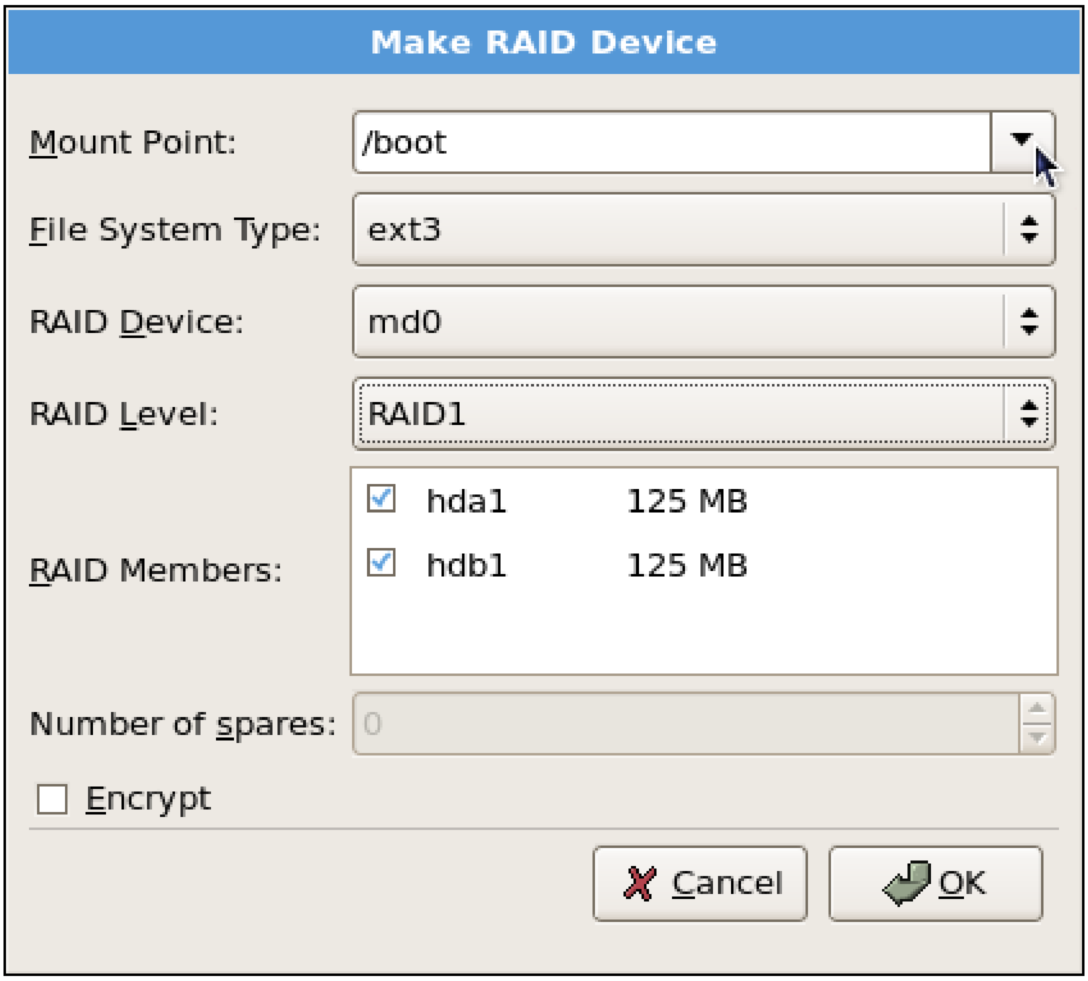
You currently have 2 software RAID partition(s) free to use.

What do you want to do now?

- ○ Create a software RAID partition.
- ● Create a RAID device [default=/dev/md0].
- ○ Clone a drive to create a RAID device [default=/dev/md0].

Figure 6.5. RAID Options

2. Select the **`Create a RAID device`** option, and click **`OK`**. As shown in Figure 6.6, “Making a RAID Device and Assigning a Mount Point”, the **`Make RAID Device`** dialog appears, allowing you to make a RAID device and assign a mount point.


Make RAID Device

Mount Point: /boot

File System Type: ext3

RAID Device: md0

RAID Level: RAID1

RAID Members:

✓	hda1	125 MB
✓	hdb1	125 MB

Number of spares: 0

☐ Encrypt

Figure 6.6. Making a RAID Device and Assigning a Mount Point

3. Select a mount point from the **`Mount Point`** pulldown list.

4. Choose the file system type for the partition from the **`File System Type`** pulldown list. At this point you can either configure a dynamic LVM file system or a traditional static ext2/ext3 file system. For more information on LVM and its configuration during the installation process, refer to *Chapter 11, LVM (Logical Volume Manager)*. If LVM is not required, continue on with the following instructions.

5. From the **`RAID Device`** pulldown list, select a device name such as **`md0`**.

6. From the **`RAID Level`**, choose the required RAID level.

Note

If you are making a RAID partition of **`/boot`**, you must choose RAID level 1, and it must use one of the first two drives (IDE first, SCSI second). If you are not creating a separate RAID partition of **`/boot`**, and you are making a RAID partition for the root file system (that is, **`/`**), it must be RAID level 1 and must use one of the first two drives (IDE first, SCSI second).

7. The RAID partitions created appear in the **`RAID Members`** list. Select which of these partitions should be used to create the RAID device.

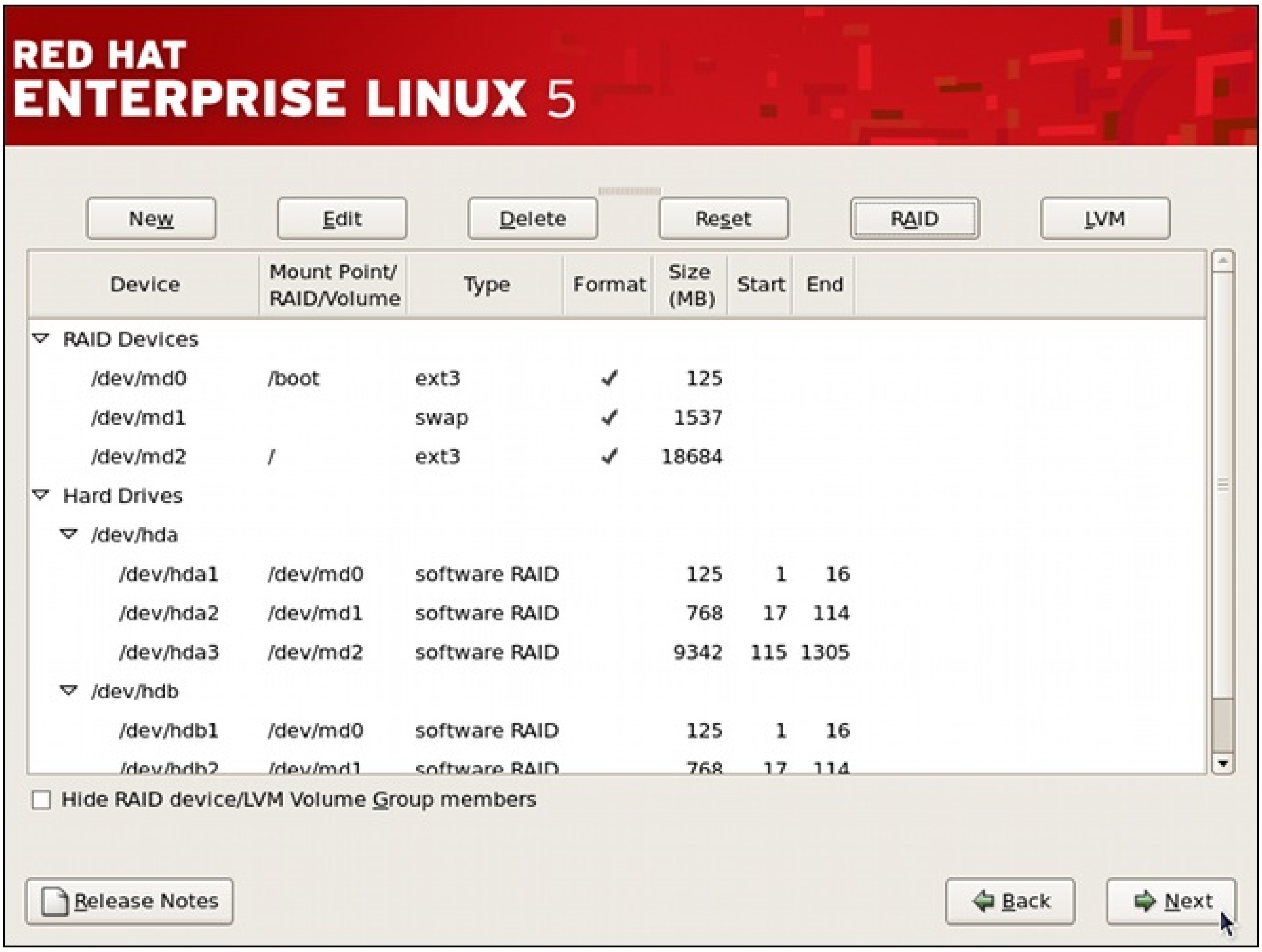
8. If configuring RAID 1 or RAID 5, specify the number of spare partitions in the **`Number of spares`** field. If a software RAID partition fails, the spare is automatically used as a replacement. For each spare you want to specify, you must create an additional software RAID partition (in addition to the

partitions for the RAID device). Select the partitions for the RAID device and the partition(s) for the spare(s).

9. Click **OK** to confirm the setup. The RAID device appears in the **Drive Summary** list.

10. Repeat this chapter's entire process for configuring additional partitions, devices, and mount points, such as the root partition (**/**), home partition (**/home**), or **swap**.

After completing the entire configuration, the figure as shown in Figure 6.7, “Sample RAID Configuration” resembles the default configuration, except for the use of RAID.

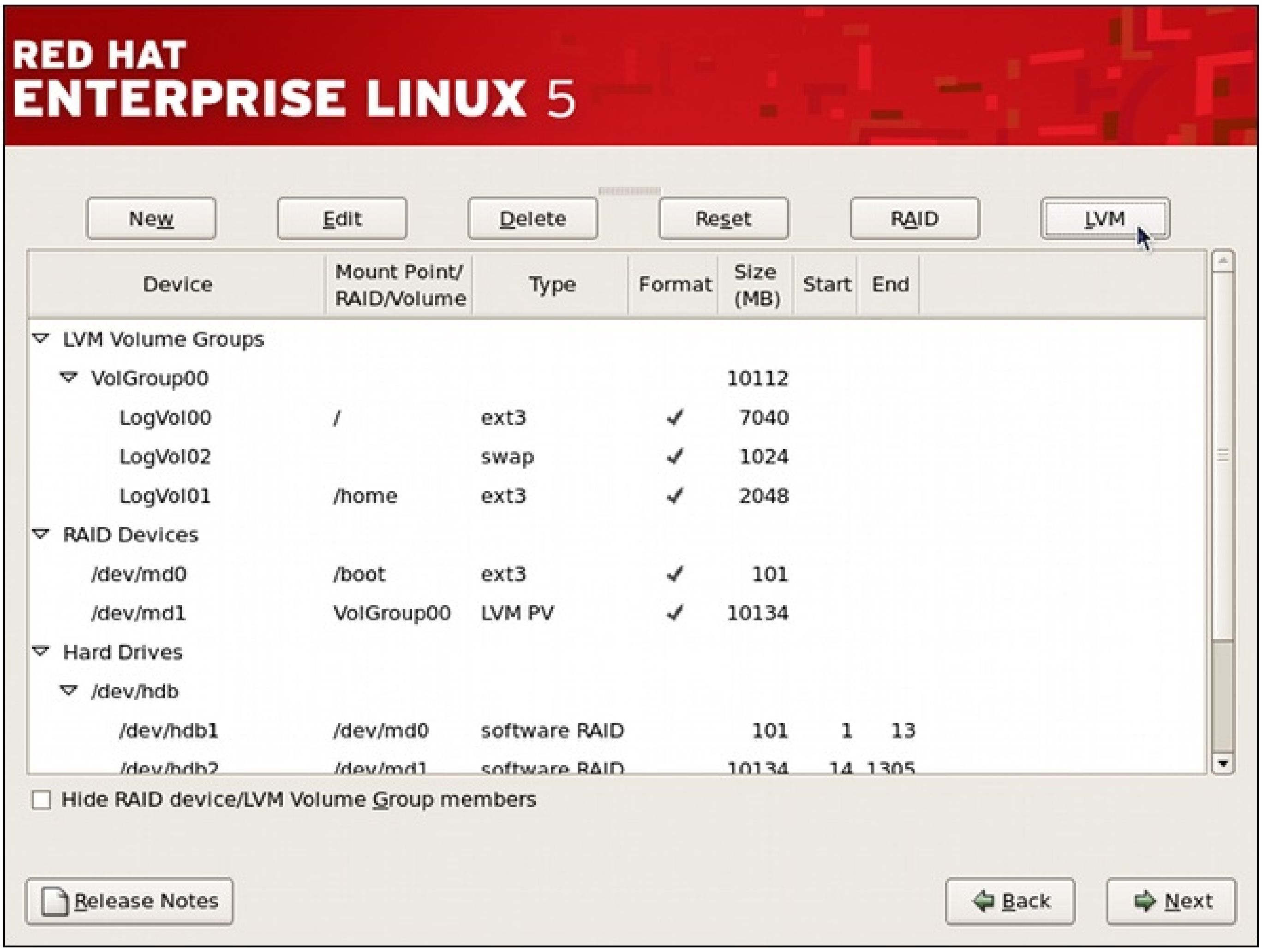


Device	Mount Point/ RAID/Volume	Type	Format	Size (MB)	Start	End
RAID Devices						
/dev/md0	/boot	ext3	✓	125		
/dev/md1		swap	✓	1537		
/dev/md2	/	ext3	✓	18684		
Hard Drives						
/dev/hda						
/dev/hda1	/dev/md0	software RAID		125	1	16
/dev/hda2	/dev/md1	software RAID		768	17	114
/dev/hda3	/dev/md2	software RAID		9342	115	1305
/dev/hdb						
/dev/hdb1	/dev/md0	software RAID		125	1	16
/dev/hdb2	/dev/md1	software RAID		768	17	114

☐ Hide RAID device/LVM Volume Group members

Figure 6.7. Sample RAID Configuration

The figure as shown in Figure 6.8, “Sample RAID With LVM Configuration” is an example of a RAID and LVM configuration.



Device	Mount Point/ RAID/Volume	Type	Format	Size (MB)	Start	End
LVM Volume Groups						
VolGroup00				10112		
LogVol00	/	ext3	✓	7040		
LogVol02		swap	✓	1024		
LogVol01	/home	ext3	✓	2048		
RAID Devices						
/dev/md0	/boot	ext3	✓	101		
/dev/md1	VolGroup00	LVM PV	✓	10134		
Hard Drives						
/dev/hdb						
/dev/hdb1	/dev/md0	software RAID		101	1	13
/dev/hdb2	/dev/md1	software RAID		10134	14	1305

☐ Hide RAID device/LVM Volume Group members

Figure 6.8. Sample RAID With LVM Configuration

You can proceed with your installation process by clicking **`Next`**. Refer to the *Red Hat Enterprise Linux Installation Guide* for further instructions.

6.3. Managing Software RAID

This section discusses software RAID configuration and management after the installation, and covers the following topics:

- Reviewing existing software RAID configuration.
- Creating a new RAID device.
- Replacing a faulty device in an array.
- Adding a new device to an existing array.
- Deactivating and removing an existing RAID device.
- Saving the configuration.

All examples in this section use the software RAID configuration from the previous section.

6.3.1. Reviewing RAID Configuration

When a software RAID is in use, basic information about all presently active RAID devices are stored in the **/proc/mdstat** special file. To list these devices, display the content of this file by typing the following at a shell prompt:

```
cat /proc/mdstat
```

To determine whether a certain device is a RAID device or a component device, run the command in the following form as **root**:

```
mdadm --query device…
```

In order to examine a RAID device in more detail, use the following command:

```
mdadm --detail raid_device…
```

Similarly, to examine a component device, type:

```
mdadm --examine component_device…
```

While the **mdadm --detail** command displays information about a RAID device, **mdadm --examine** only relays information about a RAID device as it relates to a given component device. This distinction is particularly important when working with a RAID device that itself is a component of another RAID device.

The **mdadm --query** command, as well as both **mdadm --detail** and **mdadm --examine** commands allow you to specify multiple devices at once.

Example 6.1. Reviewing RAID configuration

Assume the system uses configuration from Figure 6.7, “Sample RAID Configuration”. You can verify that **/dev/md0** is a RAID device by typing the following at a shell prompt:

```
~]# mdadm --query /dev/md0
/dev/md0: 125.38MiB raid1 2 devices, 0 spares. Use mdadm --detail for more
detail.
/dev/md0: No md super block found, not an md component.
```

As you can see, the above command produces only a brief overview of the RAID device and its configuration. To display more detailed information, use the following command instead:

```
~]# mdadm --detail /dev/md0
/dev/md0:
        Version : 0.90
  Creation Time : Tue Jun 28 16:05:49 2011
     Raid Level : raid1
     Array Size : 128384 (125.40 MiB 131.47 MB)
  Used Dev Size : 128384 (125.40 MiB 131.47 MB)
   Raid Devices : 2
  Total Devices : 2
Preferred Minor : 0
    Persistence : Superblock is persistent

    Update Time : Thu Jun 30 17:06:34 2011
          State : clean
  Active Devices : 2
Working Devices : 2
 Failed Devices : 0
  Spare Devices : 0

           UUID : 49c5ac74:c2b79501:5c28cb9c:16a6dd9f
         Events : 0.6

    Number   Major   Minor   RaidDevice State
       0       3        1        0      active sync   /dev/hda1
       1       3       65        1      active sync   /dev/hdb1
```

Finally, to list all presently active RAID devices, type:

```
~]$ cat /proc/mdstat
Personalities : [raid0] [raid1]
md0 : active raid1 hdb1[1] hda1[0]
      128384 blocks [2/2] [UU]

md1 : active raid0 hdb2[1] hda2[0]
      1573888 blocks 256k chunks

md2 : active raid0 hdb3[1] hda3[0]
      19132928 blocks 256k chunks

unused devices: <none>
```

6.3.2. Creating a New RAID Device

To create a new RAID device, use the command in the following form as **root**:

```
mdadm --create raid_device --level=level --raid-devices=number
component_device…
```

This is the simplest way to create a RAID array. There are many more options that allow you to specify the number of spare devices, the block size of a stripe array, if the array has a write-intent bitmap, and much more. All these options can have a significant impact on the performance, but are beyond the scope of this document. For more detailed information, refer to the *CREATE MODE* section of the **mdadm**(8) manual page.

Example 6.2. Creating a new RAID device

Assume that the system has two unused SCSI disk drives available, and that each of these devices has exactly one partition of the same size:

```
~]# ls /dev/sd*
/dev/sda  /dev/sda1  /dev/sdb  /dev/sdb1
```

To create **/dev/md3** as a new RAID level 1 array from **/dev/sda1** and **/dev/sdb1**, run the following command:

```
~]# mdadm --create /dev/md3 --level=1 --raid-devices=2 /dev/sda1 /dev/sdb1
mdadm: array /dev/md3 started.
```

6.3.3. Replacing a Faulty Device

To replace a particular device in a software RAID, first make sure it is marked as faulty by running the following command as **`root`**:

```
mdadm raid_device --fail component_device
```

Then remove the faulty device from the array by using the command in the following form:

```
mdadm raid_device --remove component_device
```

Once the device is operational again, you can re-add it to the array:

```
mdadm raid_device --add component_device
```

Example 6.3. Replacing a faulty device

Assume the system has an active RAID device, **`/dev/md3`**, with the following layout (that is, the RAID device created in Example 6.2, “Creating a new RAID device”):

```
~]# mdadm --detail /dev/md3 | tail -n 3
    Number   Major   Minor   RaidDevice State
       0       8        1        0      active sync   /dev/sda1
       1       8       17        1      active sync   /dev/sdb1
```

Imagine the first disk drive fails and needs to be replaced. To do so, first mark the **`/dev/sdb1`** device as faulty:

```
~]# mdadm /dev/md3 --fail /dev/sdb1
mdadm: set /dev/sdb1 faulty in /dev/md3
```

Then remove it from the RAID device:

```
~]# mdadm /dev/md3 --remove /dev/sdb1
mdadm: hot removed /dev/sdb1
```

As soon as the hardware is replaced, you can add the device back to the array by using the following command:

```
~]# mdadm /dev/md3 --add /dev/sdb1
mdadm: added /dev/sdb1
```

6.3.4. Extending a RAID Device

To add a new device to an existing array, use the command in the following form as **`root`**:

```
mdadm raid_device --add component_device
```

This will add the device as a spare device. To grow the array to use this device actively, type the following at a shell prompt:

```
mdadm --grow raid_device --raid-devices=number
```

Example 6.4. Extending a RAID device

Assume the system has an active RAID device, **/dev/md3**, with the following layout (that is, the RAID device created in Example 6.2, "Creating a new RAID device"):

```
~]# mdadm --detail /dev/md3 | tail -n 3
    Number   Major   Minor   RaidDevice State
       0       8        1        0      active sync   /dev/sda1
       1       8       17        1      active sync   /dev/sdb1
```

Also assume that a new SCSI disk drive, **/dev/sdc**, has been added and has exactly one partition. To add it to the **/dev/md3** array, type the following at a shell prompt:

```
~]# mdadm /dev/md3 --add /dev/sdc1
mdadm: added /dev/sdc1
```

This will add **/dev/sdc1** as a spare device. To change the size of the array to actually use it, type:

```
~]# mdadm --grow /dev/md3 --raid-devices=3
```

6.3.5. Removing a RAID Device

To remove an existing RAID device, first deactivate it by running the following command as **root**:

```
mdadm --stop raid_device
```

Once deactivated, remove the RAID device itself:

```
mdadm --remove raid_device
```

Finally, zero superblocks on all devices that were associated with the particular array:

```
mdadm --zero-superblock component_device…
```

Example 6.5. Removing a RAID device

Assume the system has an active RAID device, **/dev/md3**, with the following layout (that is, the RAID device created in Example 6.4, "Extending a RAID device"):

```
~]# mdadm --detail /dev/md3 | tail -n 4
    Number   Major   Minor   RaidDevice State
       0       8        1        0      active sync   /dev/sda1
       1       8       17        1      active sync   /dev/sdb1
       2       8       33        2      active sync   /dev/sdc1
```

In order to remove this device, first stop it by typing the following at a shell prompt:

```
~]# mdadm --stop /dev/md3
mdadm: stopped /dev/md3
```

Once stopped, you can remove the **/dev/md3** device by running the following command:

```
~]# mdadm --remove /dev/md3
```

Finally, to remove the superblocks from all associated devices, type:

```
~]# mdadm --zero-superblock /dev/sda1 /dev/sdb1 /dev/sdc1
```

6.3.6. Preserving the Configuration

By default, changes made by the **mdadm** command only apply to the current session, and will not survive a system restart. At boot time, the **mdmonitor** service reads the content of the **/etc/mdadm.conf** configuration file to see which RAID devices to start. If the software RAID was configured during the graphical installation process, this file contains directives listed in Table 6.1, “Common mdadm.conf directives” by default.

Table 6.1. Common mdadm.conf directives

Option	Description
ARRAY	Allows you to identify a particular array.
DEVICE	Allows you to specify a list of devices to scan for a RAID component (for example, “/dev/hda1”). You can also use the keyword **partitions** to use all partitions listed in **/proc/partitions**, or **containers** to specify an array container.
MAILADDR	Allows you to specify an email address to use in case of an alert.

To list what **ARRAY** lines are presently in use regardless of the configuration, run the following command as **root**:

```
mdadm --detail --scan
```

Use the output of this command to determine which lines to add to the **/etc/mdadm.conf** file. You can also display the **ARRAY** line for a particular device:

```
mdadm --detail --brief raid_device
```

By redirecting the output of this command, you can add such a line to the configuration file with a single command:

```
mdadm --detail --brief raid_device >> /etc/mdadm.conf
```

Example 6.6. Preserving the configuration

By default, the **/etc/mdadm.conf** contains the software RAID configuration created during the system installation:

```
# mdadm.conf written out by anaconda
DEVICE partitions
MAILADDR root
ARRAY /dev/md0 level=raid1 num-devices=2
UUID=49c5ac74:c2b79501:5c28cb9c:16a6dd9f
ARRAY /dev/md1 level=raid0 num-devices=2
UUID=76914c11:5bfa2c00:dc6097d1:a1f4506d
ARRAY /dev/md2 level=raid0 num-devices=2
UUID=2b5d38d0:aea898bf:92be20e2:f9d893c5
```

Assuming you have created the **/dev/md3** device as shown in Example 6.2, “Creating a new RAID device”, you can make it persistent by running the following command:

```
~]# mdadm --detail --brief /dev/md3 >> /etc/mdadm.conf
```

6.4. Additional Resources

For more information on RAID, refer to the following resources.

6.4.1. Installed Documentation

- **mdadm** man page — A manual page for the**mdadm** utility.
- **mdadm.conf** man page — A manual page that provides a comprehensive list of available **/etc/mdadm.conf** configuration options.

[1] A hot-swap chassis allows you to remove a hard drive without having to power-down your system.

[2] Parity information is calculated based on the contents of the rest of the member disks in the array. This information can then be used to reconstruct data when one disk in the array fails. The reconstructed data can then be used to satisfy I/O requests to the failed disk before it is replaced and to repopulate the failed disk after it has been replaced.

Chapter 7. Swap Space

7.1. What is Swap Space?

Swap space in Linux is used when the amount of physical memory (RAM) is full. If the system needs more memory resources and the RAM is full, inactive pages in memory are moved to the swap space. While swap space can help machines with a small amount of RAM, it should not be considered a replacement for more RAM. Swap space is located on hard drives, which have a slower access time than physical memory.

Swap space can be a dedicated swap partition (recommended), a swap file, or a combination of swap partitions and swap files.

In years past, the recommended amount of swap space increased linearly with the amount of RAM in the system. But because the amount of memory in modern systems has increased into the hundreds of gigabytes, it is now recognized that the amount of swap space that a system needs is a function of the memory workload running on that system. However, given that swap space is usually designated at install time, and that it can be difficult to determine beforehand the memory workload of a system, we recommend determining system swap using the following table.

Table 7.1. Recommended System Swap Space

Amount of RAM in the System	Recommended Amount of Swap Space
4GB of RAM or less	a minimum of 2GB of swap space
4GB to 16GB of RAM	a minimum of 4GB of swap space
16GB to 64GB of RAM	a minimum of 8GB of swap space
64GB to 256GB of RAM	a minimum of 16GB of swap space
256GB to 512GB of RAM	a minimum of 32GB of swap space

Important

File systems and LVM2 volumes assigned as swap space *cannot* be in use when being modified. For example, no system processes can be assigned the swap space, as well as no amount of swap should be allocated and used by the kernel. Use the **`free`** and **`cat /proc/swaps`** commands to verify how much and where swap is in use.

The best way to achieve swap space modifications is to boot your system in rescue mode, and then follow the instructions (for each scenario) in the remainder of this chapter. Refer to the Red Hat Enterprise Linux Installation Guide for instructions on booting into rescue mode. When prompted to mount the file system, select **`Skip`**.

7.2. Adding Swap Space

Sometimes it is necessary to add more swap space after installation. For example, you may upgrade the amount of RAM in your system from 128 MB to 256 MB, but there is only 256 MB of swap space. It might be advantageous to increase the amount of swap space to 512 MB if you perform memory-intense operations or run applications that require a large amount of memory.

You have three options: create a new swap partition, create a new swap file, or extend swap on an existing LVM2 logical volume. It is recommended that you extend an existing logical volume.

7.2.1. Extending Swap on an LVM2 Logical Volume

To extend an LVM2 swap logical volume (assuming **`/dev/VolGroup00/LogVol01`** is the volume you want to extend):

1. Disable swapping for the associated logical volume:

   ```
   swapoff -v /dev/VolGroup00/LogVol01
   ```

2. Resize the LVM2 logical volume by 256 MB:

   ```
   lvm lvresize /dev/VolGroup00/LogVol01 -L +256M
   ```

3. Format the new swap space:

   ```
   mkswap /dev/VolGroup00/LogVol01
   ```

4. Enable the extended logical volume:

   ```
   swapon -va
   ```

5. Test that the logical volume has been extended properly:

   ```
   cat /proc/swaps
   free
   ```

7.2.2. Creating an LVM2 Logical Volume for Swap

To add a swap volume group (assuming **`/dev/VolGroup00/LogVol02`** is the swap volume you want to add):

1. Create the LVM2 logical volume of size 256 MB:

   ```
   lvm lvcreate VolGroup00 -n LogVol02 -L 256M
   ```

2. Format the new swap space:

   ```
   mkswap /dev/VolGroup00/LogVol02
   ```

3. Add the following entry to the **`/etc/fstab`** file:

   ```
   /dev/VolGroup00/LogVol02   swap     swap    defaults     0 0
   ```

4. Enable the extended logical volume:

   ```
   swapon -va
   ```

5. Test that the logical volume has been extended properly:

   ```
   cat /proc/swaps
   free
   ```

7.2.3. Creating a Swap File

To add a swap file:

1. Determine the size of the new swap file in megabytes and multiply by 1024 to determine the number of blocks. For example, the block size of a 64 MB swap file is 65536.

2. At a shell prompt as root, type the following command with **`count`** being equal to the desired block size:

   ```
   dd if=/dev/zero of=/swapfile bs=1024 count=65536
   ```

3. Change the persmissions of the newly created file:

   ```
   chmod 0600 /swapfile
   ```

4. Setup the swap file with the command:

   ```
   mkswap /swapfile
   ```

5. To enable the swap file immediately but not automatically at boot time:

   ```
   swapon /swapfile
   ```

6. To enable it at boot time, edit **`/etc/fstab`** to include the following entry:

   ```
   /swapfile          swap            swap    defaults        0 0
   ```

   The next time the system boots, it enables the new swap file.

7. After adding the new swap file and enabling it, verify it is enabled by viewing the output of the command **`cat /proc/swaps`** or **`free`**.

7.3. Removing Swap Space

Sometimes it can be prudent to reduce swap space after installation. For example, say you downgraded the amount of RAM in your system from 1 GB to 512 MB, but there is 2 GB of swap space still assigned. It might be advantageous to reduce the amount of swap space to 1 GB, since the larger 2 GB could be wasting disk space.

You have three options: remove an entire LVM2 logical volume used for swap, remove a swap file, or reduce swap space on an existing LVM2 logical volume.

7.3.1. Reducing Swap on an LVM2 Logical Volume

To reduce an LVM2 swap logical volume (assuming **`/dev/VolGroup00/LogVol01`** is the volume you want to reduce):

1. Disable swapping for the associated logical volume:

   ```
   swapoff -v /dev/VolGroup00/LogVol01
   ```

2. Reduce the LVM2 logical volume by 512 MB:

```
lvm lvreduce /dev/VolGroup00/LogVol01 -L -512M
```

3. Format the new swap space:

```
mkswap /dev/VolGroup00/LogVol01
```

4. Enable the extended logical volume:

```
swapon -va
```

5. Test that the logical volume has been reduced properly:

```
cat /proc/swaps
free
```

7.3.2. Removing an LVM2 Logical Volume for Swap

The swap logical volume cannot be in use (no system locks or processes on the volume). The easiest way to achieve this is to boot your system in rescue mode. Refer to the *Red Hat Enterprise Linux Installation Guide* for instructions on booting into rescue mode. When prompted to mount the file system, select **`Skip`**.

To remove a swap volume group (assuming **`/dev/VolGroup00/LogVol02`** is the swap volume you want to remove):

1. Disable swapping for the associated logical volume:

```
swapoff -v /dev/VolGroup00/LogVol02
```

2. Remove the LVM2 logical volume of size 512 MB:

```
lvm lvremove /dev/VolGroup00/LogVol02
```

3. Remove the following entry from the **`/etc/fstab`** file:

```
/dev/VolGroup00/LogVol02   swap     swap    defaults     0 0
```

4. Test that the logical volume has been removed:

```
cat /proc/swaps
free
```

7.3.3. Removing a Swap File

To remove a swap file:

1. At a shell prompt as root, execute the following command to disable the swap file (where **`/swapfile`** is the swap file):

```
swapoff -v /swapfile
```

2. Remove its entry from the **`/etc/fstab`** file.

3. Remove the actual file:

   ```
   rm /swapfile
   ```

7.4. Moving Swap Space

To move swap space from one location to another, follow the steps for removing swap space, and then follow the steps for adding swap space.

Chapter 8. Managing Disk Storage

8.1. Standard Partitions using `parted`

The utility **`parted`** allows users to:

- View the existing partition table
- Change the size of existing partitions
- Add partitions from free space or additional hard drives

If you want to view the system's disk space usage or monitor the disk space usage, refer to Section 42.3, "File Systems".

By default, the **`parted`** package is included when installing Red Hat Enterprise Linux. To start **`parted`**, log in as root and type the command **`parted`** ***`/dev/sda`*** at a shell prompt (where ***`/dev/sda`*** is the device name for the drive you want to configure).

If you want to remove or resize a partition, the device on which that partition resides must not be in use. Creating a new partition on a device which is in use—while possible—is not recommended.

For a device to not be in use, none of the partitions on the device can be mounted, and any swap space on the device must not be enabled.

As well, the partition table should not be modified while it is in use because the kernel may not properly recognize the changes. If the partition table does not match the actual state of the mounted partitions, information could be written to the wrong partition, resulting in lost and overwritten data.

The easiest way to achieve this is to boot your system in rescue mode. When prompted to mount the file system, select **`Skip`**.

Alternately, if the drive does not contain any partitions in use (system processes that use or lock the file system from being unmounted), you can unmount them with the **`umount`** command and turn off all the swap space on the hard drive with the **`swapoff`** command.

Table 8.1, "**`parted`** commands" contains a list of commonly used **`parted`** commands. The sections that follow explain some of these commands and arguments in more detail.

Table 8.1. `parted` commands

Command	Description
`check` ***`minor-num`***	Perform a simple check of the file system
`cp` ***`from to`***	Copy file system from one partition to another; *from* and *to* are the minor numbers of the partitions
`help`	Display list of available commands
`mklabel` ***`label`***	Create a disk label for the partition table
`mkfs` ***`minor-num file-system-type`***	Create a file system of type *file-system-type*
`mkpart` ***`part-type fs-type start-mb end-mb`***	Make a partition without creating a new file system
`mkpartfs` ***`part-type fs-type start-mb end-mb`***	Make a partition and create the specified file system
`move` ***`minor-num start-mb end-mb`***	Move the partition
`name` ***`minor-num name`***	Name the partition for Mac and PC98 disklabels only

Command	Description
`print`	Display the partition table
`quit`	Quit **`parted`**
`rescue` *start-mb end-mb*	Rescue a lost partition from *start-mb* to *end-mb*
`resize` ***`minor-num start-mb end-mb`***	Resize the partition from *start-mb* to *end-mb*
`rm` ***`minor-num`***	Remove the partition
`select` ***`device`***	Select a different device to configure
`set` ***`minor-num flag state`***	Set the flag on a partition; *state* is either on or off
`toggle [`*`NUMBER`*` [`*`FLAG`*`]`	Toggle the state of *FLAG* on partition *NUMBER*
`unit` ***`UNIT`***	Set the default unit to *UNIT*

8.1.1. Viewing the Partition Table

After starting **`parted`**, use the command **`print`** to view the partition table. A table similar to the following appears:

```
Model: ATA ST3160812AS (scsi)
Disk /dev/sda: 160GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos

Number  Start   End    Size    Type      File system  Flags
 1      32.3kB  107MB  107MB   primary   ext3         boot
 2      107MB   105GB  105GB   primary   ext3
 3      105GB   107GB  2147MB  primary   linux-swap
 4      107GB   160GB  52.9GB  extended        root
 5      107GB   133GB  26.2GB  logical   ext3
 6      133GB   133GB  107MB   logical   ext3
 7      133GB   160GB  26.6GB  logical                lvm
```

The first line contains the disk type, manufacturer, model number and interface, and the second line displays the disk label type. The remaining output below the fourth line shows the partition table.

In the partition table, the *Minor* number is the partition **`number`**. For example, the partition with minor number 1 corresponds to **`/dev/sda1`**. The **`Start`** and **`End`** values are in megabytes. Valid **`Type`** are metadata, free, primary, extended, or logical. The **`Filesystem`** is the file system type, which can be any of the following:

- ext2
- ext3
- fat16
- fat32
- hfs
- jfs
- linux-swap
- ntfs
- reiserfs

- hp-ufs
- sun-ufs
- xfs

If a **`Filesystem`** of a device shows no value, this means that its file system type is unknown.

The **`Flags`** column lists the flags set for the partition. Available flags are boot, root, swap, hidden, raid, lvm, or lba.

Note

To select a different device without having to restart **`parted`**, use the **`select`** command followed by the device name (for example, **`/dev/sda`**). Doing so allows you to view or configure the partition table of a device.

8.1.2. Creating a Partition

Warning

Do not attempt to create a partition on a device that is in use.

Before creating a partition, boot into rescue mode (or unmount any partitions on the device and turn off any swap space on the device).

Start **`parted`**, where /dev/*sda* is the device on which to create the partition:

```
parted /dev/sda
```

View the current partition table to determine if there is enough free space:

```
print
```

If there is not enough free space, you can resize an existing partition. Refer to Section 8.1.4, “Resizing a Partition” for details.

8.1.2.1. Making the Partition

From the partition table, determine the start and end points of the new partition and what partition type it should be. You can only have four primary partitions (with no extended partition) on a device. If you need more than four partitions, you can have three primary partitions, one extended partition, and multiple logical partitions within the extended. For an overview of disk partitions, refer to the appendix *An Introduction to Disk Partitions* in the *Red Hat Enterprise Linux Installation Guide*.

For example, to create a primary partition with an ext3 file system from 1024 megabytes until 2048 megabytes on a hard drive type the following command:

```
mkpart primary ext3 1024 2048
```

Note

If you use the **`mkpartfs`** command instead, the file system is created after the partition is created. However, **`parted`** does not support creating an ext3 file system. Thus, if you wish to create an ext3 file system, use **`mkpart`** and create the file system with the **`mkfs`** command as described later.

The changes start taking place as soon as you press **`Enter`**, so review the command before executing to it.

After creating the partition, use the **`print`** command to confirm that it is in the partition table with the correct partition type, file system type, and size. Also remember the minor number of the new partition so that you can label it. You should also view the output of

```
cat /proc/partitions
```

to make sure the kernel recognizes the new partition.

8.1.2.2. Formatting the Partition

The partition still does not have a file system. Create the file system:

```
mkfs -t ext3 /dev/sda6
```

Warning

Formatting the partition permanently destroys any data that currently exists on the partition.

8.1.2.3. Labeling the Partition

Next, give the partition a label. For example, if the new partition is **`/dev/sda6`** and you want to label it **`/work`**:

```
e2label /dev/sda6 /work
```

By default, the installation program uses the mount point of the partition as the label to make sure the label is unique. You can use any label you want.

8.1.2.4. Creating the Mount Point

As root, create the mount point:

```
mkdir /work
```

8.1.2.5. Add to `/etc/fstab`

As root, edit the **`/etc/fstab`** file to include the new partition. The new line should look similar to the following:

```
LABEL=/work           /work                 ext3    defaults        1 2
```

The first column should contain **LABEL=** followed by the label you gave the partition. The second column should contain the mount point for the new partition, and the next column should be the file system type (for example, ext3 or swap). If you need more information about the format, read the man page with the commanc **man fstab**.

If the fourth column is the word **defaults**, the partition is mounted at boot time. To mount the partition without rebooting, as root, type the command:

```
mount /work
```

8.1.3. Removing a Partition

Warning

Do not attempt to remove a partition on a device that is in use.

Before removing a partition, boot into rescue mode (or unmount any partitions on the device and turn off any swap space on the device).

Start **parted**, where /dev/*sda* is the device on which to remove the partition:

```
parted /dev/sda
```

View the current partition table to determine the minor number of the partition to remove:

```
print
```

Remove the partition with the command **rm**. For example, to remove the partition with minor number 3:

```
rm 3
```

The changes start taking place as soon as you press **Enter**, so review the command before committing to it.

After removing the partition, use the **print** command to confirm that it is removed from the partition table. You should also view the output of

```
cat /proc/partitions
```

to make sure the kernel knows the partition is removed.

The last step is to remove it from the **/etc/fstab** file. Find the line that declares the removed partition, and remove it from the file.

8.1.4. Resizing a Partition

Warning

Do not attempt to resize a partition on a device that is in use.

Before resizing a partition, boot into rescue mode (or unmount any partitions on the device and turn off any swap space on the device).

Start **`parted`**, where /dev/*sda* is the device on which to resize the partition:

```
parted /dev/sda
```

View the current partition table to determine the minor number of the partition to resize as well as the start and end points for the partition:

```
print
```

To resize the partition, use the **`resize`** command followed by the minor number for the partition, the starting place in megabytes, and the end place in megabytes. For example:

```
resize 3 1024 2048
```

Warning

A partition cannot be made larger than the space available on the device

After resizing the partition, use the **`print`** command to confirm that the partition has been resized correctly, is the correct partition type, and is the correct file system type.

After rebooting the system into normal mode, use the command **`df`** to make sure the partition was mounted and is recognized with the new size.

8.2. LVM Partition Management

The following commands can be found by issuing **`lvm help`** at a command prompt.

Table 8.2. `LVM` commands

Command	Description
`dumpconfig`	Dump the active configuration
`formats`	List the available metadata formats
`help`	Display the help commands
`lvchange`	Change the attributes of logical volume(s)
`lvcreate`	Create a logical volume
`lvdisplay`	Display information about a logical volume
`lvextend`	Add space to a logical volume
`lvmchange`	*Due to use of the device mapper, this command has been deprecated*
`lvmdiskscan`	List devices that may be used as physical volumes
`lvmsadc`	Collect activity data
`lvmsar`	Create activity report
`lvreduce`	Reduce the size of a logical volume
`lvremove`	Remove logical volume(s) from the system

Command	Description
`lvrename`	Rename a logical volume
`lvresize`	Resize a logical volume
`lvs`	Display information about logical volumes
`lvscan`	List all logical volumes in all volume groups
`pvchange`	Change attributes of physical volume(s)
`pvcreate`	Initialize physical volume(s) for use by LVM
`pvdata`	Display the on-disk metadata for physical volume(s)
`pvdisplay`	Display various attributes of physical volume(s)
`pvmove`	Move extents from one physical volume to another
`pvremove`	Remove LVM label(s) from physical volume(s)
`pvresize`	Resize a physical volume in use by a volume group
`pvs`	Display information about physical volumes
`pvscan`	List all physical volumes
`segtypes`	List available segment types
`vgcfgbackup`	Backup volume group configuration
`vgcfgrestore`	Restore volume group configuration
`vgchange`	Change volume group attributes
`vgck`	Check the consistency of a volume group
`vgconvert`	Change volume group metadata format
`vgcreate`	Create a volume group
`vgdisplay`	Display volume group information
`vgexport`	Unregister a volume group from the system
`vgextend`	Add physical volumes to a volume group
`vgimport`	Register exported volume group with system
`vgmerge`	Merge volume groups
`vgmknodes`	Create the special files for volume group devices in /dev/
`vgreduce`	Remove a physical volume from a volume group
`vgremove`	Remove a volume group
`vgrename`	Rename a volume group
`vgs`	Display information about volume groups
`vgscan`	Search for all volume groups
`vgsplit`	Move physical volumes into a new volume group
`version`	Display software and driver version information

Chapter 9. Implementing Disk Quotas

Disk space can be restricted by implementing disk quotas which alert a system administrator before a user consumes too much disk space or a partition becomes full.

Disk quotas can be configured for individual users as well as user groups. This makes it possible to manage the space allocated for user-specific files (such as email) separately from the space allocated to the projects a user works on (assuming the projects are given their own groups).

In addition, quotas can be set not just to control the number of disk blocks consumed but to control the number of inodes (data structures that contain information about files in UNIX file systems). Because inodes are used to contain file-related information, this allows control over the number of files that can be created.

The **quota** RPM must be installed to implement disk quotas.

Note

For more information on installing RPM packages, refer to Part II, “Package Management”.

9.1. Configuring Disk Quotas

To implement disk quotas, use the following steps:

1. Enable quotas per file system by modifying the **/etc/fstab** file.
2. Remount the file system(s).
3. Create the quota database files and generate the disk usage table.
4. Assign quota policies.

Each of these steps is discussed in detail in the following sections.

9.1.1. Enabling Quotas

As root, using a text editor, edit the **/etc/fstab** file. Add the **usrquota** and/or **grpquota** options to the file systems that require quotas:

```
/dev/VolGroup00/LogVol00 /         ext3    defaults        1 1
LABEL=/boot              /boot     ext3    defaults        1 2
none                     /dev/pts  devpts  gid=5,mode=620  0 0
none                     /dev/shm  tmpfs   defaults        0 0
none                     /proc     proc    defaults        0 0
none                     /sys      sysfs   defaults        0 0
/dev/VolGroup00/LogVol02 /home     ext3    defaults,usrquota,grpquota  1 2
/dev/VolGroup00/LogVol01 swap      swap    defaults        0 0 . . .
```

In this example, the **/home** file system has both user and group quotas enabled.

Note

The following examples assume that a separate **/home** partition was created during the installation of Red Hat Enterprise Linux. The root (**/**) partition can be used for setting quota policies in the **/etc/fstab** file.

9.1.2. Remounting the File Systems

After adding the **usrquota** and/or **grpquota** options, remount each file system whose **fstab** entry has been modified. If the file system is not in use by any process, use one of the following methods:

- Issue the **umount** command followed by the **mount** command to remount the file system.(See the **man** page for both **umount** and **mount** for the specific syntax for mounting and unmounting various filesystem types.)
- Issue the **mount -o remount *<file-system>*** command (where ***<file-system>*** is the name of the file system) to remount the file system. For example, to remount the **/home** file system, the command to issue is **mount -o remount /home**.

If the file system is currently in use, the easiest method for remounting the file system is to reboot the system.

9.1.3. Creating the Quota Database Files

After each quota-enabled file system is remounted, the system is capable of working with disk quotas. However, the file system itself is not yet ready to support quotas. The next step is to run the **quotacheck** command.

The **quotacheck** command examines quota-enabled file systems and builds a table of the current disk usage per file system. The table is then used to update the operating system's copy of disk usage. In addition, the file system's disk quota files are updated.

To create the quota files (**aquota.user** and **aquota.group**) on the file system, use the **-c** option of the **quotacheck** command. For example, if user and group quotas are enabled for the **/home** file system, create the files in the **/home** directory:

```
quotacheck -cug /home
```

The **-c** option specifies that the quota files should be created for each file system with quotas enabled, the **-u** option specifies to check for user quotas, and the **-g** option specifies to check for group quotas.

If neither the **-u** or **-g** options are specified, only the user quota file is created. If only **-g** is specified, only the group quota file is created.

After the files are created, run the following command to generate the table of current disk usage per file system with quotas enabled:

```
quotacheck -avug
```

The options used are as follows:

- **a** — Check all quota-enabled, locally-mounted file systems
- **v** — Display verbose status information as the quota check proceeds

- **u** — Check user disk quota information
- **g** — Check group disk quota information

After **quotacheck** has finished running, the quota files corresponding to the enabled quotas (user and/or group) are populated with data for each quota-enabled locally-mounted file system such as **/home**.

9.1.4. Assigning Quotas per User

The last step is assigning the disk quotas with the **edquota** command.

To configure the quota for a user, as root in a shell prompt, execute the command:

```
edquota username
```

Perform this step for each user who needs a quota. For example, if a quota is enabled in **/etc/fstab** for the **/home** partition (**/dev/VolGroup00/LogVol02** in the example below) and the command **edquota testuser** is executed, the following is shown in the editor configured as the default for the system:

```
Disk quotas for user testuser (uid 501):
Filesystem                blocks      soft      hard     inodes    soft    hard
/dev/VolGroup00/LogVol02  440436         0         0      37418       0       0
```

Note

The text editor defined by the **EDITOR** environment variable is used by **edquota**. To change the editor, set the **EDITOR** environment variable in your **~/.bash_profile** file to the full path of the editor of your choice.

The first column is the name of the file system that has a quota enabled for it. The second column shows how many blocks the user is currently using. The next two columns are used to set soft and hard block limits for the user on the file system. The **inodes** column shows how many inodes the user is currently using. The last two columns are used to set the soft and hard inode limits for the user on the file system.

The hard block limit is the absolute maximum amount of disk space that a user or group can use. Once this limit is reached, no further disk space can be used.

The soft block limit defines the maximum amount of disk space that can be used. However, unlike the hard limit, the soft limit can be exceeded for a certain amount of time. That time is known as the *grace period*. The grace period can be expressed in seconds, minutes, hours, days, weeks, or months.

If any of the values are set to 0, that limit is not set. In the text editor, change the desired limits. For example:

```
Disk quotas for user testuser (uid 501):
Filesystem                blocks      soft      hard    inodes    soft    hard
/dev/VolGroup00/LogVol02  440436    500000    550000     37418       0       0
```

To verify that the quota for the user has been set, use the command:

```
quota testuser
```

9.1.5. Assigning Quotas per Group

Quotas can also be assigned on a per-group basis. For example, to set a group quota for the **devel** group (the group must exist prior to setting the group quota), use the command:

```
edquota -g devel
```

This command displays the existing quota for the group in the text editor:

```
Disk quotas for group devel (gid 505):
Filesystem                blocks    soft      hard    inodes    soft     hard
/dev/VolGroup00/LogVol02  440400       0         0     37418       0        0
```

Modify the limits, then save the file.

To verify that the group quota has been set, use the command:

```
quota -g devel
```

9.1.6. Setting the Grace Period for Soft Limits

If soft limits are set for a given quota (whether inode or block and for either users or groups) the grace period, or amount of time a soft limit can be exceeded, should be set with the command:

```
edquota -t
```

While other **edquota** commands operate on a particular user's or group's quota, the **-t** option operates on every filesystem with quotas enabled.

9.2. Managing Disk Quotas

If quotas are implemented, they need some maintenance — mostly in the form of watching to see if the quotas are exceeded and making sure the quotas are accurate.

Of course, if users repeatedly exceed their quotas or consistently reach their soft limits, a system administrator has a few choices to make depending on what type of users they are and how much disk space impacts their work. The administrator can either help the user determine how to use less disk space or increase the user's disk quota.

9.2.1. Enabling and Disabling

It is possible to disable quotas without setting them to 0. To turn all user and group quotas off, use the following command:

```
quotaoff -vaug
```

If neither the **-u** or **-g** options are specified, only the user quotas are disabled. If only **-g** is specified, only group quotas are disabled. The **-v** switch causes verbose status information to display as the command executes.

To enable quotas again, use the **quotaon** command with the same options.

For example, to enable user and group quotas for all file systems, use the following command:

```
quotaon -vaug
```

To enable quotas for a specific file system, such as **/home**, use the following command:

```
quotaon -vug /home
```

If neither the **-u** or **-g** options are specified, only the user quotas are enabled. If only **-g** is specified, only group quotas are enabled.

9.2.2. Reporting on Disk Quotas

Creating a disk usage report entails running the **repquota** utility. For example, the command **repquota /home** produces this output:

```
*** Report for user quotas on device /dev/mapper/VolGroup00-LogVol02
Block grace time: 7days; Inode grace time: 7days
                        Block limits                File limits
User            used    soft    hard  grace    used  soft  hard  grace
----------------------------------------------------------------------
root      --      36       0       0              4     0     0
kristin   --     540       0       0            125     0     0
testuser  --  440400  500000  550000          37418     0     0
```

To view the disk usage report for all (option **-a**) quota-enabled file systems, use the command:

```
repquota -a
```

While the report is easy to read, a few points should be explained. The **--** displayed after each user is a quick way to determine whether the block or inode limits have been exceeded. If either soft limit is exceeded, a **+** appears in place of the corresponding **-**; the first **-** represents the block limit, and the second represents the inode limit.

The **grace** columns are normally blank. If a soft limit has been exceeded, the column contains a time specification equal to the amount of time remaining on the grace period. If the grace period has expired, **none** appears in its place.

9.2.3. Keeping Quotas Accurate

Whenever a file system is not unmounted cleanly (due to a system crash, for example), it is necessary to run quotacheck. However, quotacheck can be run on a regular basis, even if the system has not crashed. Safe methods for periodically running **quotacheck** include:

Ensuring quotacheck runs on next reboot

> **Note**
>
> This method works best for (busy) multiuser systems which are periodically rebooted.

As root, place a shell script into the **/etc/cron.daily/** or **/etc/cron.weekly/** directory—or schedule one using the **crontab -e** command—that contains the **touch /forcequotacheck** command. This creates an empty **forcequotacheck** file in the root directory, which the system init script looks for at boot time. If it is found, the init script runs **quotacheck**. Afterward, the init script removes the **/forcequotacheck** file; thus, scheduling this file to be created periodically with **cron** ensures that **quotacheck** is run during the next reboot.

Refer to Chapter 39, *Automated Tasks* for more information about configuring **cron**.

Running quotacheck in single user mode

An alternative way to safely run **quotacheck** is to (re-)boot the system into single-user mode to prevent the possibility of data corruption in quota files and run:

```
~]# quotaoff -vaug /<file_system>
~]# quotacheck -vaug /<file_system>
~]# quotaon -vaug /<file_system>
```

Running quotacheck on a running system

If necessary, it is possible to run **quotacheck** on a machine during a time when no users are logged in, and thus have no open files on the file system being checked. Run the command **quotacheck -vaug *<file_system>*** ; this command will fail if **quotacheck** cannot remount the given *<file_system>* as read-only. Note that, following the check, the file system will be remounted read-write.

Important

Running **quotacheck** on a live file system mounted read-write is not recommended due to the possibility of quota file corruption.

Refer to Chapter 39, *Automated Tasks* for more information about configuring **cron**.

9.3. Additional Resources

For more information on disk quotas, refer to the following resources.

9.3.1. Installed Documentation

- The **quotacheck**, **edquota**, **repquota**, **quota**, **quotaon**, and **quotaoff** man pages

9.3.2. Related Books

- *Red Hat Enterprise Linux Introduction to System Administration*; Red Hat, Inc. — Available at http://www.redhat.com/docs/ and on the Documentation CD, this manual contains background information on storage management (including disk quotas) for new Red Hat Enterprise Linux system administrators.

Chapter 10. Access Control Lists

Files and directories have permission sets for the owner of the file, the group associated with the file, and all other users for the system. However, these permission sets have limitations. For example, different permissions cannot be configured for different users. Thus, *Access Control Lists* (ACLs) were implemented.

The Red Hat Enterprise Linux 5 kernel provides ACL support for the ext3 file system and NFS-exported file systems. ACLs are also recognized on ext3 file systems accessed via Samba.

Along with support in the kernel, the **`acl`** package is required to implement ACLs. It contains the utilities used to add, modify, remove, and retrieve ACL information.

The **`cp`** and **`mv`** commands copy or move any ACLs associated with files and directories.

10.1. Mounting File Systems

Before using ACLs for a file or directory, the partition for the file or directory must be mounted with ACL support. If it is a local ext3 file system, it can mounted with the following command:

```
mount -t ext3 -o acl <device-name> <partition>
```

For example:

```
mount -t ext3 -o acl /dev/VolGroup00/LogVol02 /work
```

Alternatively, if the partition is listed in the **`/etc/fstab`** file, the entry for the partition can include the **`acl`** option:

```
LABEL=/work      /work       ext3    acl        1 2
```

If an ext3 file system is accessed via Samba and ACLs have been enabled for it, the ACLs are recognized because Samba has been compiled with the **`--with-acl-support`** option. No special flags are required when accessing or mounting a Samba share.

10.1.1. NFS

By default, if the file system being exported by an NFS server supports ACLs and the NFS client can read ACLs, ACLs are utilized by the client system.

To disable ACLs on NFS shares when configuring the server, include the **`no_acl`** option in the **`/etc/exports`** file. To disable ACLs on an NFS share when mounting it on a client, mount it with the **`no_acl`** option via the command line or the **`/etc/fstab`** file.

10.2. Setting Access ACLs

There are two types of ACLs: *access ACLs* and *default ACLs*. An access ACL is the access control list for a specific file or directory. A default ACL can only be associated with a directory; if a file within the directory does not have an access ACL, it uses the rules of the default ACL for the directory. Default ACLs are optional.

ACLs can be configured:

1. Per user

2. Per group
3. Via the effective rights mask
4. For users not in the user group for the file

The **setfacl** utility sets ACLs for files and directories. Use the **-m** option to add or modify the ACL of a file or directory:

```
setfacl -m <rules> <files>
```

Rules (*<rules>*) must be specified in the following formats. Multiple rules can be specified in the same command if they are separated by commas.

u:*<uid>*:*<perms>*

Sets the access ACL for a user. The user name or UID may be specified. The user may be any valid user on the system.

g:*<gid>*:*<perms>*

Sets the access ACL for a group. The group name or GID may be specified. The group may be any valid group on the system.

m:*<perms>*

Sets the effective rights mask. The mask is the union of all permissions of the owning group and all of the user and group entries.

o:*<perms>*

Sets the access ACL for users other than the ones in the group for the file.

White space is ignored. Permissions (*<perms>*) must be a combination of the characters **r**, **w**, and **x** for read, write, and execute.

If a file or directory already has an ACL, and the **setfacl** command is used, the additional rules are added to the existing ACL or the existing rule is modified.

For example, to give read and write permissions to user andrius:

```
setfacl -m u:andrius:rw /project/somefile
```

To remove all the permissions for a user, group, or others, use the **-x** option and do not specify any permissions:

```
setfacl -x <rules> <files>
```

For example, to remove all permissions from the user with UID 500:

```
setfacl -x u:500 /project/somefile
```

10.3. Setting Default ACLs

To set a default ACL, add **d:** before the rule and specify a directory instead of a file name.

For example, to set the default ACL for the **/share/** directory to read and execute for users not in the user group (an access ACL for an individual file can override it):

```
setfacl -m d:o:rx /share
```

10.4. Retrieving ACLs

To determine the existing ACLs for a file or directory, use the **getfacl** command. In the example below, the **getfacl** is used to determine the existing ACLs for a file.

```
getfacl home/john/picture.png
```

The above command returns the following output:

```
# file: home/john/picture.png
# owner: john
# group: john
user::rw-
group::r--
other::r--
```

If a directory with a default ACL is specified, the default ACL is also displayed as illustrated below.

```
[john@main /]$ getfacl home/sales/
# file: home/sales/
# owner: john
# group: john
user::rw-
user:barryg:r--
group::r--
mask::r--
other::r--
default:user::rwx
default:user:john:rwx
default:group::r-x
default:mask::rwx
default:other::r-x
```

10.5. Archiving File Systems With ACLs

Warning

The **tar** and **dump** commands do *not* backup ACLs.

The **star** utility is similar to the **tar** utility in that it can be used to generate archives of files; however, some of its options are different. Refer to Table 10.1, “Command Line Options for **star**” for a listing of more commonly used options. For all available options, refer to the **star** man page. The **star** package is required to use this utility.

Table 10.1. Command Line Options for `star`

Option	Description
`-c`	Creates an archive file.
`-n`	Do not extract the files; use in conjunction with **`-x`** to show what extracting the files does.
`-r`	Replaces files in the archive. The files are written to the end of the archive file, replacing any files with the same path and file name.
`-t`	Displays the contents of the archive file.
`-u`	Updates the archive file. The files are written to the end of the archive if they do not exist in the archive or if the files are newer than the files of the same name in the archive. This option only work if the archive is a file or an unblocked tape that may backspace.
`-x`	Extracts the files from the archive. If used with **`-U`** and a file in the archive is older than the corresponding file on the file system, the file is not extracted.
`-help`	Displays the most important options.
`-xhelp`	Displays the least important options.
`-/`	Do not strip leading slashes from file names when extracting the files from an archive. By default, they are striped when files are extracted.
`-acl`	When creating or extracting, archive or restore any ACLs associated with the files and directories.

10.6. Compatibility with Older Systems

If an ACL has been set on any file on a given file system, that file system has the **`ext_attr`** attribute. This attribute can be seen using the following command:

```
tune2fs -l <filesystem-device>
```

A file system that has acquired the **`ext_attr`** attribute can be mounted with older kernels, but those kernels do not enforce any ACLs which have been set.

Versions of the **`e2fsck`** utility included in version 1.22 and higher of the **`e2fsprogs`** package (including the versions in Red Hat Enterprise Linux 2.1 and 4) can check a file system with the **`ext_attr`** attribute. Older versions refuse to check it.

10.7. Additional Resources

Refer to the follow resources for more information.

10.7.1. Installed Documentation

- **`acl`** man page — Description of ACLs
- **`getfacl`** man page — Discusses how to get file access control lists
- **`setfacl`** man page — Explains how to set file access control lists

- **`star`** man page — Explains more about the **`star`** utility and its many options

10.7.2. Useful Websites

- http://acl.bestbits.at/ — Website for ACLs

Chapter 11. LVM (Logical Volume Manager)

11.1. What is LVM?

LVM is a tool for logical volume management which includes allocating disks, striping, mirroring and resizing logical volumes.

With LVM, a hard drive or set of hard drives is allocated to one or more *physical volumes*. LVM physical volumes can be placed on other block devices which might span two or more disks.

The physical volumes are combined into *logical volumes*, with the exception of the **`/boot`** partition. The **`/boot`** partition cannot be on a logical volume group because the boot loader cannot read it. If the root (`/`) partition is on a logical volume, create a separate **`/boot`** partition which is not a part of a volume group.

Since a physical volume cannot span over multiple drives, to span over more than one drive, create one or more physical volumes per drive.

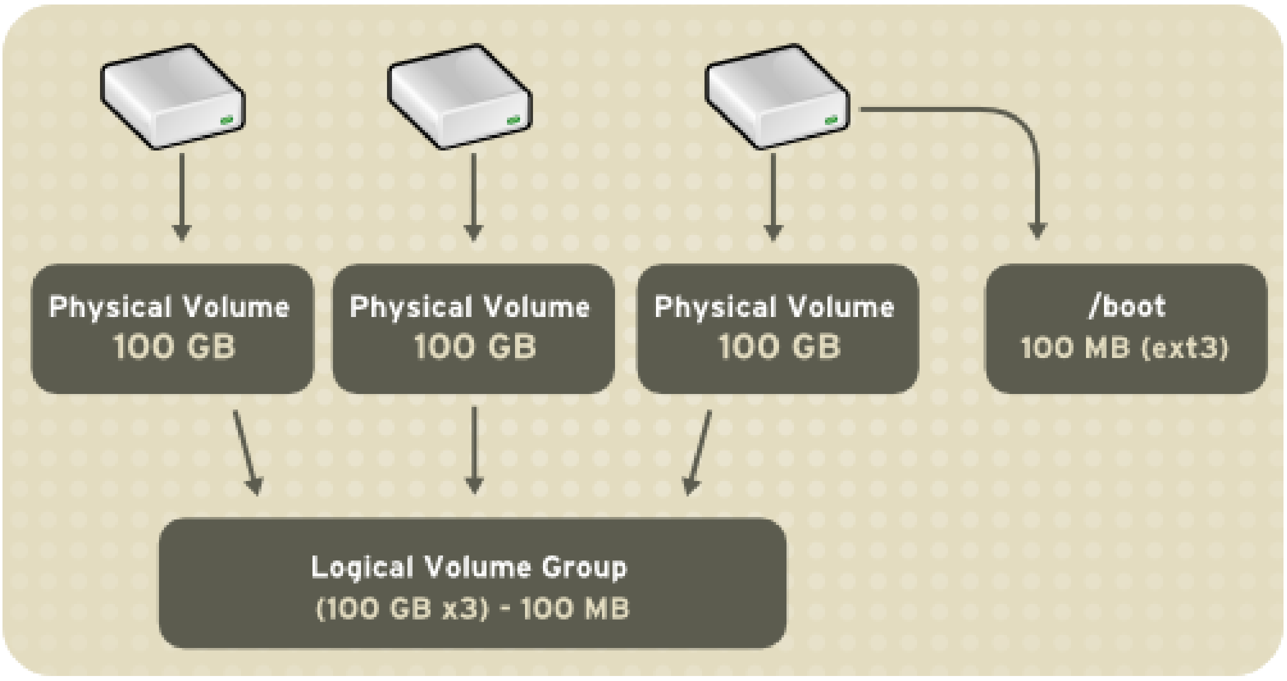


Figure 11.1. Logical Volumes

The volume groups can be divided into *logical volumes*, which are assigned mount points, such as **`/home`** and `/` and file system types, such as ext2 or ext3. When "partitions" reach their full capacity, free space from the volume group can be added to the logical volume to increase the size of the partition. When a new hard drive is added to the system, it can be added to the volume group, and partitions that are logical volumes can be increased in size.

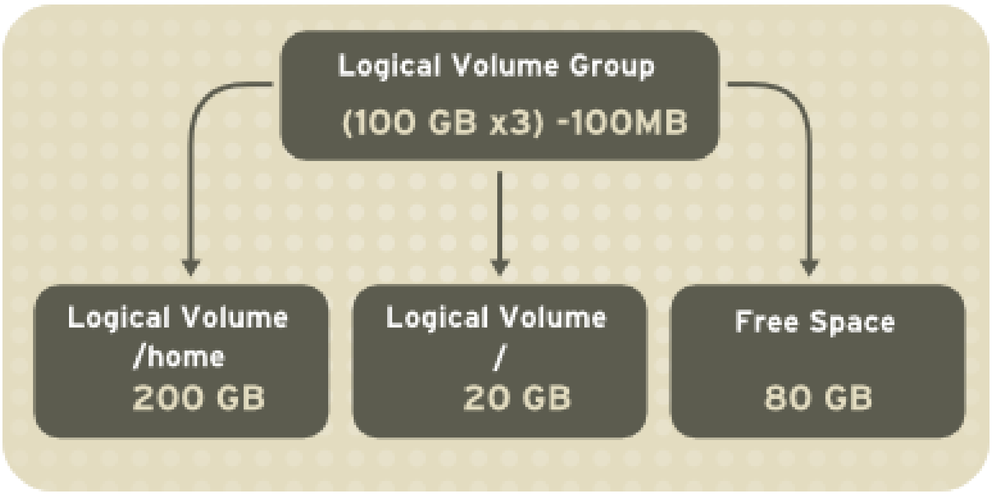


Figure 11.2. Logical Volumes

On the other hand, if a system is partitioned with the ext3 file system, the hard drive is divided into partitions of defined sizes. If a partition becomes full, it is not easy to expand the size of the partition. Even if the partition is moved to another hard drive, the original hard drive space has to be reallocated as a different partition or not used.

To learn how to configure LVM during the installation process, refer to Section 11.2, “LVM Configuration”.

11.1.1. What is LVM2?

LVM version 2, or LVM2, is the default for Red Hat Enterprise Linux 5, which uses the device mapper driver contained in the 2.6 kernel. LVM2 can be upgraded from versions of Red Hat Enterprise Linux running the 2.4 kernel.

11.2. LVM Configuration

LVM can be configured during the graphical installation process, the text-based installation process, or during a kickstart installation. You can use the **`system-config-lvm`** utility to create your own LVM configuration post-installation. The next two sections focus on using **Disk Druid** during installation to complete this task. The third section introduces the LVM utility (**`system-config-lvm`**) which allows you to manage your LVM volumes in X windows or graphically.

Read Section 11.1, “What is LVM?” first to learn about LVM. An overview of the steps required to configure LVM include:

- Creating *physical volumes* from the hard drives.
- Creating *volume groups* from the physical volumes.
- Creating *logical volumes* from the volume groups and assign the logical volumes mount points.

Two 9.1 GB SCSI drives (**`/dev/sda`** and **`/dev/sdb`**) are used in the following examples. They detail how to create a simple configuration using a single LVM volume group with associated logical volumes during installation.

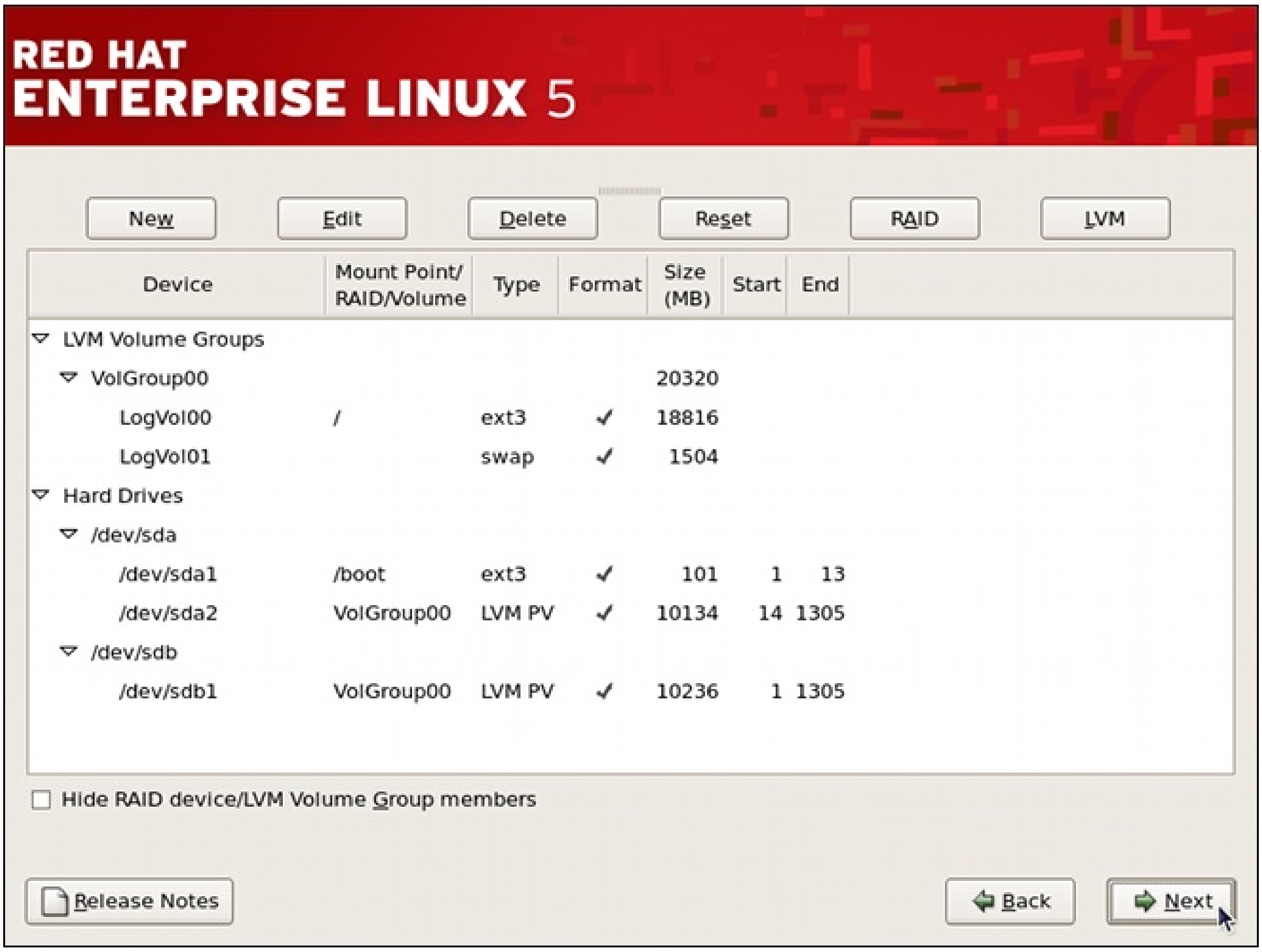
11.3. Automatic Partitioning

On the **`Disk Partitioning Setup`** screen, select **Remove linux partitions on selected drives and create default layout** from the pulldown list.

For Red Hat Enterprise Linux, LVM is the default method for disk partitioning. If you do not wish to have LVM implemented, or if you require RAID partitioning, manual disk partitioning through **Disk Druid** is required.

The following properties make up the automatically created configuration:

- The **`/boot`** partition resides on its own non-LVM partition. In the following example, it is the first partition on the first drive (**`/dev/sda1`**). Bootable partitions *cannot* reside on LVM logical volumes.
- A single LVM volume group (**`VolGroup00`**) is created, which spans all selected drives and all remaining space available. In the following example, the remainder of the first drive (**`/dev/sda2`**), and the entire second drive (**`/dev/sdb1`**) are allocated to the volume group.
- Two LVM logical volumes (**`LogVol00`** and **`LogVol01`**) are created from the newly created spanned volume group. In the following example, the recommended swap space is automatically calculated and assigned to **`LogVol01`**, and the remainder is allocated to the root file system, **`LogVol00`**.



Device	Mount Point/ RAID/Volume	Type	Format	Size (MB)	Start	End
LVM Volume Groups						
VolGroup00				20320		
LogVol00	/	ext3	✓	18816		
LogVol01		swap	✓	1504		
Hard Drives						
/dev/sda						
/dev/sda1	/boot	ext3	✓	101	1	13
/dev/sda2	VolGroup00	LVM PV	✓	10134	14	1305
/dev/sdb						
/dev/sdb1	VolGroup00	LVM PV	✓	10236	1	1305

Figure 11.3. Automatic LVM Configuration With Two SCSI Drives

Note

If enabling quotas are of interest to you, it may be best to modify the automatic configuration to include other mount points, such as **`/home`** or **`/var`**, so that each file system has its own independent quota configuration limits.

In most cases, the default automatic LVM partitioning is sufficient, but advanced implementations could warrant modification or manual configuration of the partition tables.

Note

If you anticipate future memory upgrades, leaving some free space in the volume group would allow for easy future expansion of the swap space logical volume on the system; in which case, the automatic LVM configuration should be modified to leave available space for future growth.

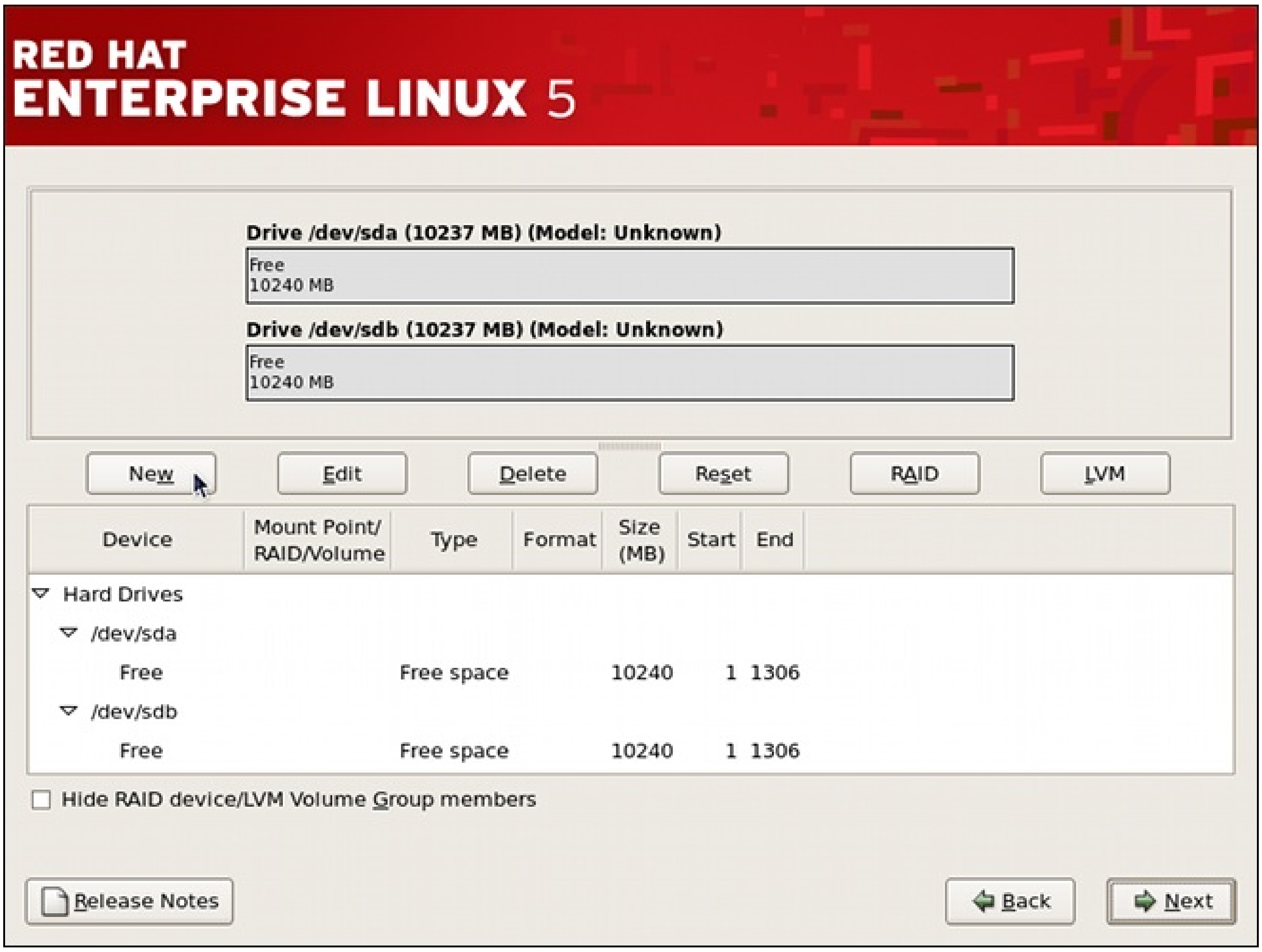
11.4. Manual LVM Partitioning

The following section explains how to manually configure LVM for Red Hat Enterprise Linux. Because there are numerous ways to manually configure a system with LVM, the following example is similar to the default configuration done in Section 11.3, “Automatic Partitioning”.

On the **`Disk Partitioning Setup`** screen, select **Create custom layout** from the pulldown list and click the **`Next`** button in the bottom right corner of the screen.

11.4.1. Creating the `/boot` Partition

In a typical situation, the disk drives are new, or formatted clean. The following figure, Figure 11.4, “Two Blank Drives, Ready for Configuration”, shows both drives as raw devices with no partitioning configured.



Drive /dev/sda (10237 MB) (Model: Unknown)
Free
10240 MB

Drive /dev/sdb (10237 MB) (Model: Unknown)
Free
10240 MB

Device	Mount Point/ RAID/Volume	Type	Format	Size (MB)	Start	End
▽ Hard Drives						
▽ /dev/sda						
Free		Free space		10240	1	1306
▽ /dev/sdb						
Free		Free space		10240	1	1306

☐ Hide RAID device/LVM Volume Group members

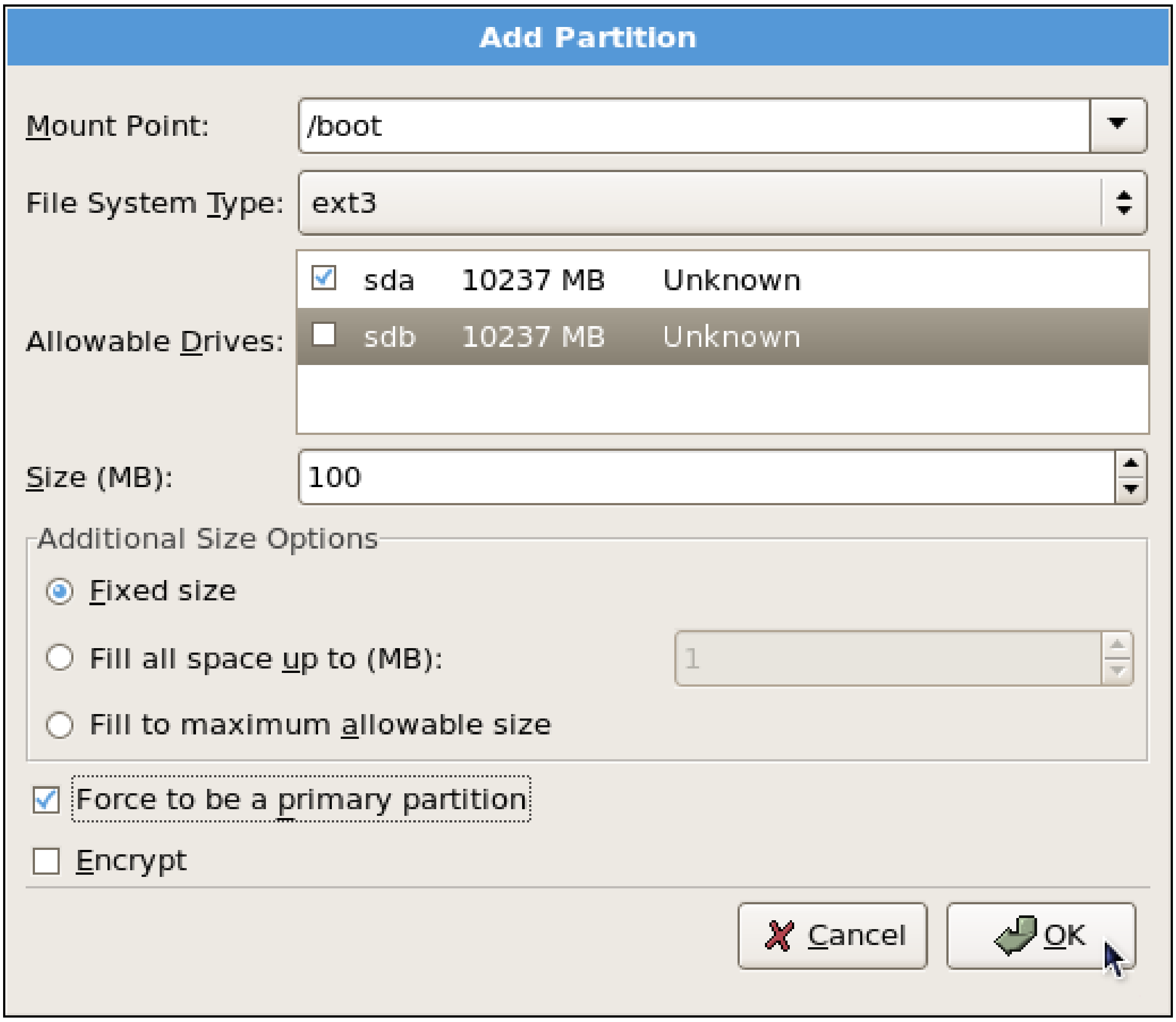
Figure 11.4. Two Blank Drives, Ready for Configuration

Warning

The **`/boot`** partition cannot reside on an LVM volume because the GRUB boot loader cannot read it.

1. Select **`New`**.
2. Select **`/boot`** from the **`Mount Point`** pulldown menu.
3. Select **`ext3`** from the **`File System Type`** pulldown menu.
4. Select only the **`sda`** checkbox from the **`Allowable Drives`** area.
5. Leave **`100`** (the default) in the **`Size (MB)`** menu.
6. Leave the **`Fixed size`** (the default) radio button selected in the **`Additional Size Options`** area.
7. Select **`Force to be a primary partition`** to make the partition be a primary partition. A primary partition is one of the first four partitions on the hard drive. If unselected, the partition is created as a logical partition. If other operating systems are already on the system, unselecting this option should be considered. For more information on primary versus logical/extended partitions, refer to the appendix section of the *Red Hat Enterprise Linux Installation Guide*.

Refer to Figure 11.5, “Creation of the Boot Partition” to verify your inputted values:


Add Partition

Mount Point: /boot

File System Type: ext3

Allowable Drives:

✓	sda	10237 MB	Unknown
☐	sdb	10237 MB	Unknown

Size (MB): 100

Additional Size Options

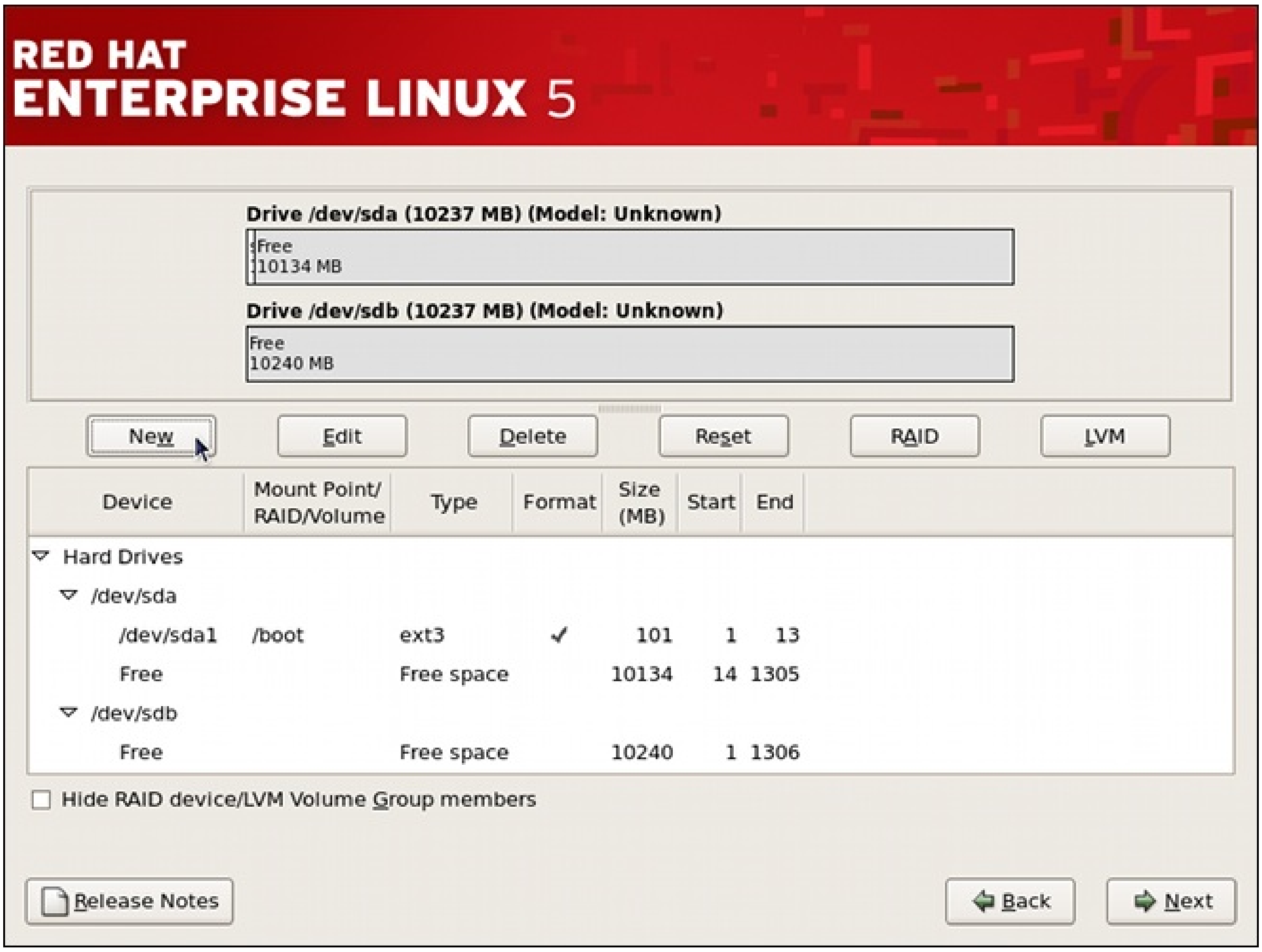
- [✓ Fixed size]
- Fill all space up to (MB): 1
- Fill to maximum allowable size

✓ Force to be a primary partition

☐ Encrypt

Figure 11.5. Creation of the Boot Partition

Click **OK** to return to the main screen. The following figure displays the boot partition correctly set:



Drive /dev/sda (10237 MB) (Model: Unknown)
Free
10134 MB

Drive /dev/sdb (10237 MB) (Model: Unknown)
Free
10240 MB

Device	Mount Point/ RAID/Volume	Type	Format	Size (MB)	Start	End
Hard Drives						
/dev/sda						
/dev/sda1	/boot	ext3	✓	101	1	13
Free		Free space		10134	14	1305
/dev/sdb						
Free		Free space		10240	1	1306

☐ Hide RAID device/LVM Volume Group members

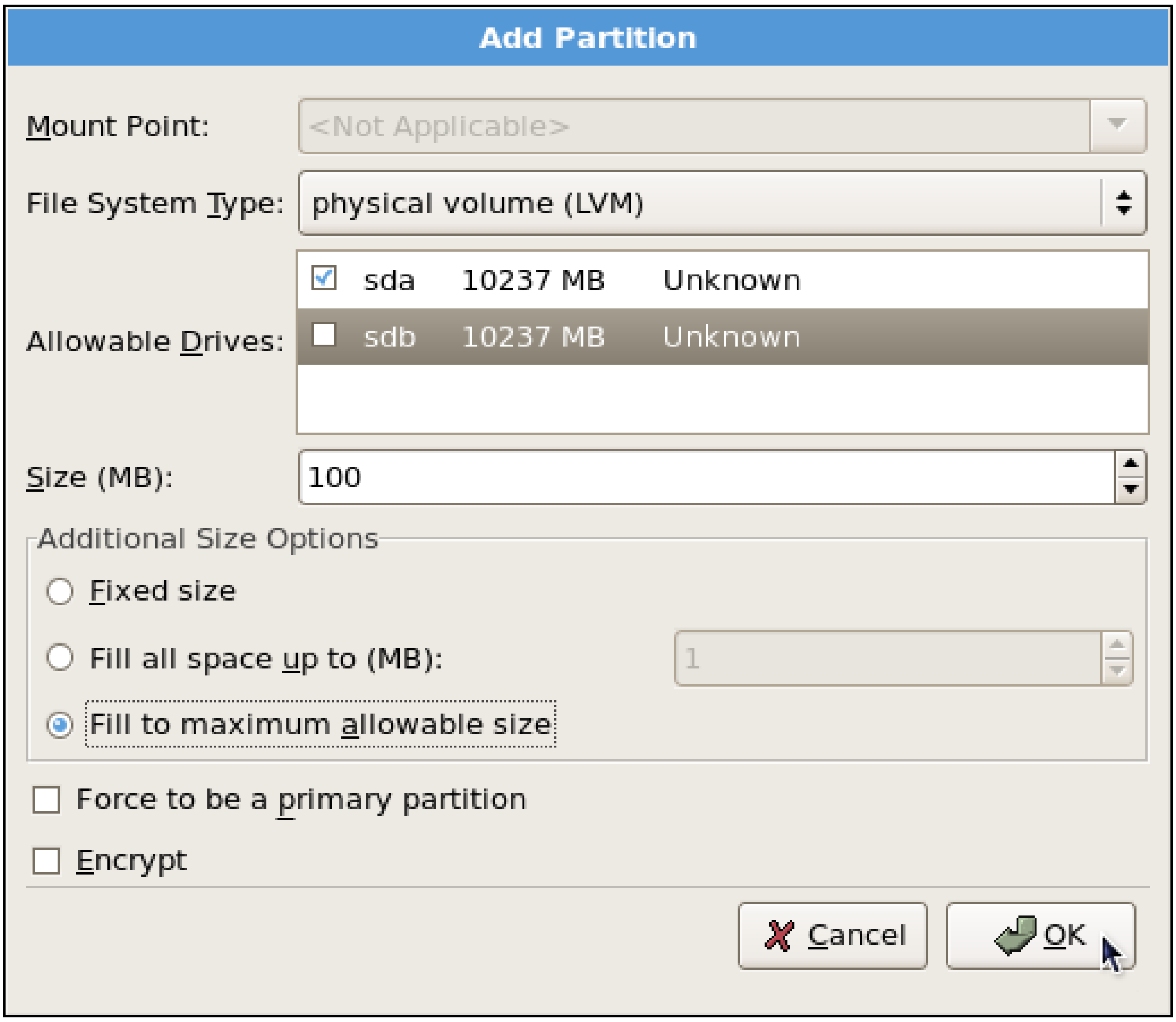
Figure 11.6. The `/boot` Partition Displayed

11.4.2. Creating the LVM Physical Volumes

Once the boot partition is created, the remainder of all disk space can be allocated to LVM partitions. The first step in creating a successful LVM implementation is the creation of the physical volume(s).

1. Select **`New`**.

2. Select **`physical volume (LVM)`** from the **`File System Type`** pulldown menu as shown in Figure 11.7, “Creating a Physical Volume”.


Add Partition

Mount Point: <Not Applicable>

File System Type: physical volume (LVM)

Allowable Drives:

✓	sda	10237 MB	Unknown
☐	sdb	10237 MB	Unknown

Size (MB): 100

Additional Size Options

○ Fixed size

○ Fill all space up to (MB): 1

● Fill to maximum allowable size

☐ Force to be a primary partition

☐ Encrypt

Figure 11.7. Creating a Physical Volume

3. You cannot enter a mount point yet (you can once you have created all your physical volumes and then all volume groups).

4. A physical volume must be constrained to one drive. For **Allowable Drives**, select the drive on which the physical volume are created. If you have multiple drives, all drives are selected, and you must deselect all but one drive.

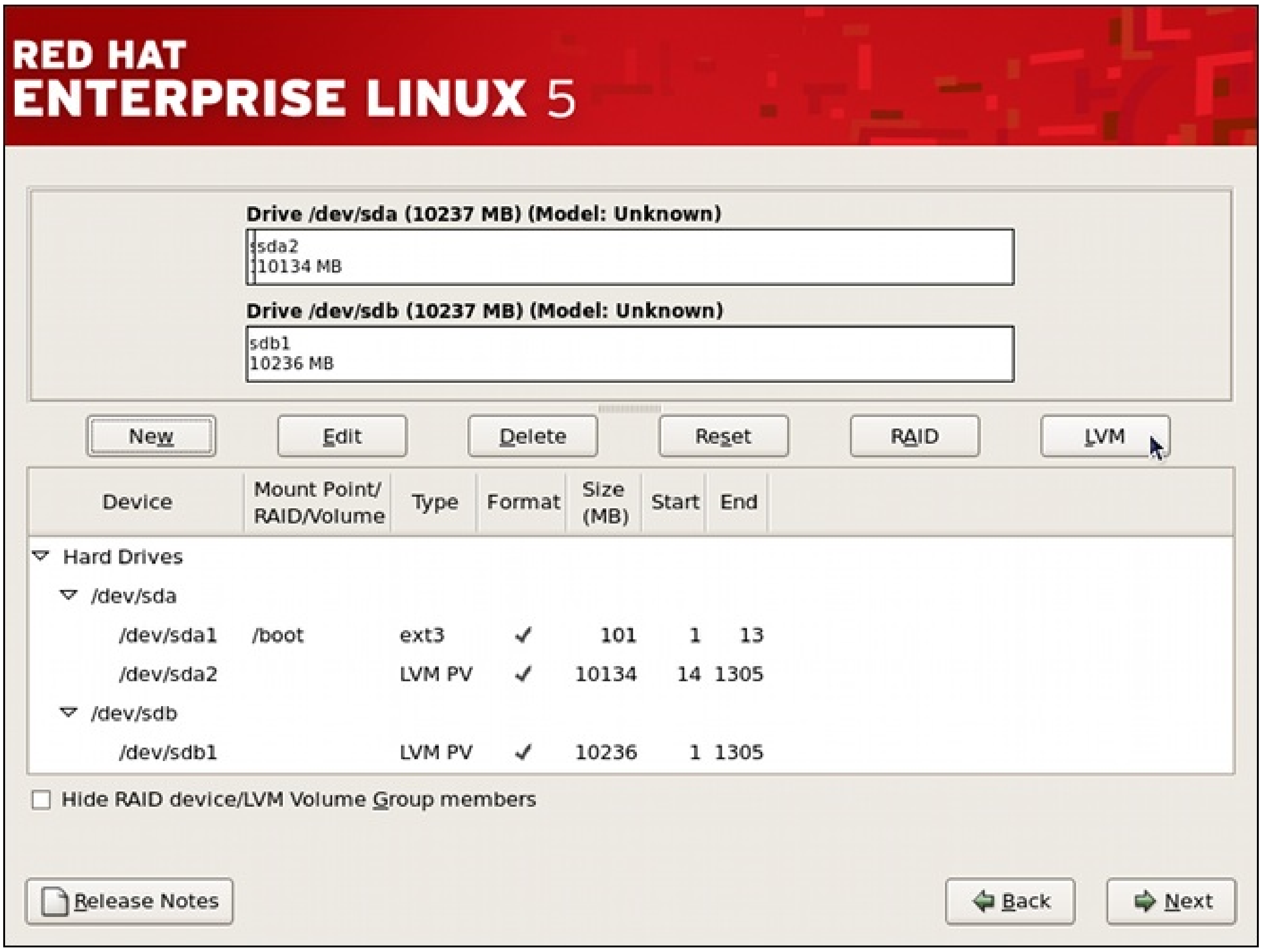
5. Enter the size that you want the physical volume to be.

6. Select **`Fixed size`** to make the physical volume the specified size, select **`Fill all space up to (MB)`** and enter a size in MBs to give range for the physical volume size, or select **`Fill to maximum allowable size`** to make it grow to fill all available space on the hard disk. If you make more than one growable, they share the available free space on the disk.

7. Select **`Force to be a primary partition`** if you want the partition to be a primary partition.

8. Click **`OK`** to return to the main screen.

Repeat these steps to create as many physical volumes as needed for your LVM setup. For example, if you want the volume group to span over more than one drive, create a physical volume on each of the drives. The following figure shows both drives completed after the repeated process:



Drive /dev/sda (10237 MB) (Model: Unknown)

sda2
10134 MB

Drive /dev/sdb (10237 MB) (Model: Unknown)

sdb1
10236 MB

Device	Mount Point/ RAID/Volume	Type	Format	Size (MB)	Start	End
Hard Drives						
/dev/sda						
/dev/sda1	/boot	ext3	✓	101	1	13
/dev/sda2		LVM PV	✓	10134	14	1305
/dev/sdb						
/dev/sdb1		LVM PV	✓	10236	1	1305

☐ Hide RAID device/LVM Volume Group members

Figure 11.8. Two Physical Volumes Created

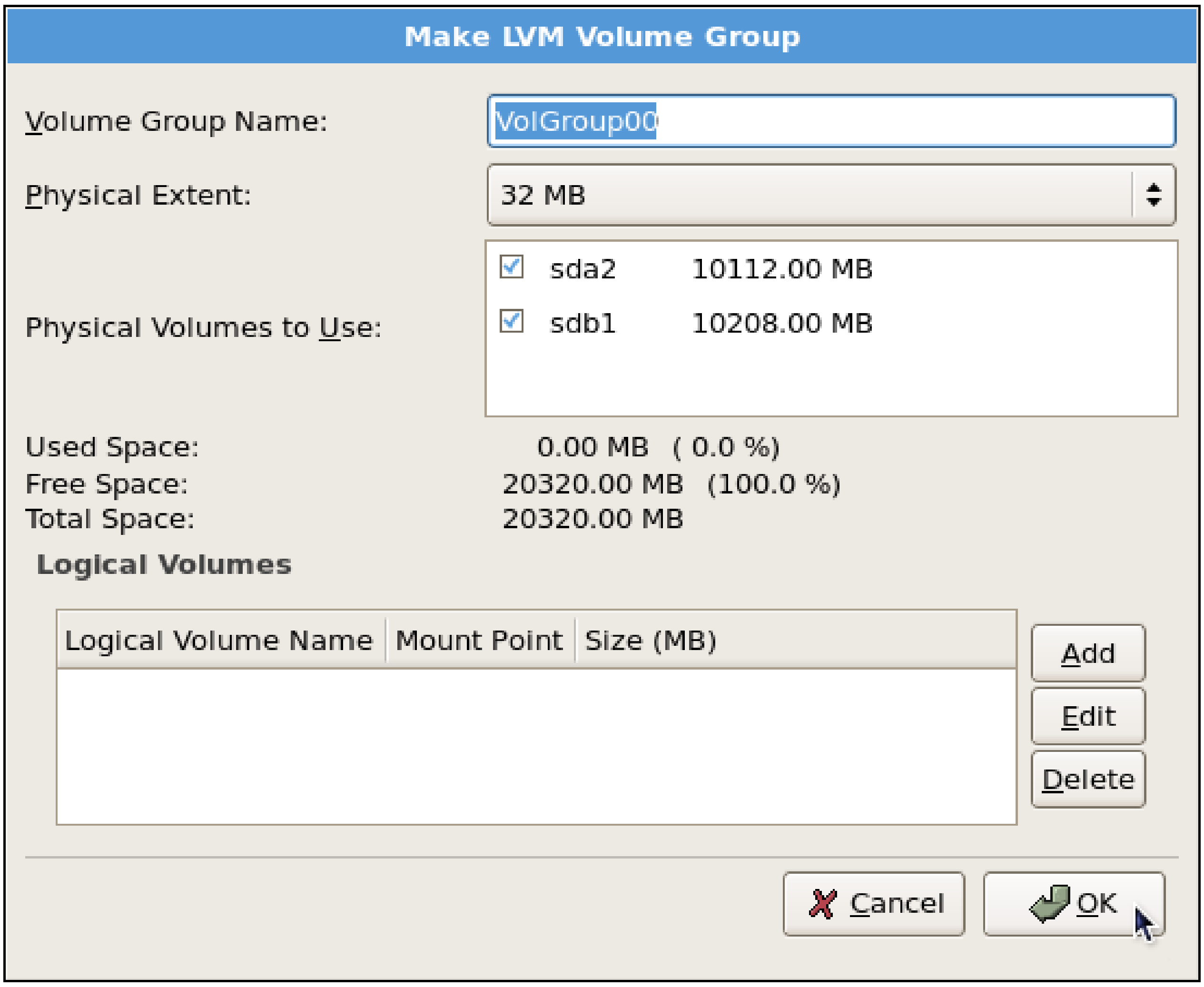
11.4.3. Creating the LVM Volume Groups

Once all the physical volumes are created, the volume groups can be created:

1. Click the **`LVM`** button to collect the physical volumes into volume groups. A volume group is basically a collection of physical volumes. You can have multiple logical volumes, but a physical volume can only be in one volume group.

Note

There is overhead disk space reserved in the volume group. The volume group size is slightly less than the total of physical volume sizes.



Make LVM Volume Group

Volume Group Name:	VolGroup00
Physical Extent:	32 MB
Physical Volumes to Use:	✓ sda2 10112.00 MB ✓ sdb1 10208.00 MB
Used Space:	0.00 MB (0.0 %)
Free Space:	20320.00 MB (100.0 %)
Total Space:	20320.00 MB

Logical Volumes

Logical Volume Name	Mount Point	Size (MB)

Figure 11.9. Creating an LVM Volume Group

2. Change the **`Volume Group Name`** if desired.

3. All logical volumes inside the volume group must be allocated in *physical extent (PE)* units. A physical extent is an allocation unit for data.

4. Select which physical volumes to use for the volume group.

11.4.4. Creating the LVM Logical Volumes

Create logical volumes with mount points such as `/`, **`/home`**, and swap space. Remember that **`/boot`** cannot be a logical volume. To add a logical volume, click the **`Add`** button in the **`Logical Volumes`** section. A dialog window as shown in Figure 11.10, “Creating a Logical Volume” appears.

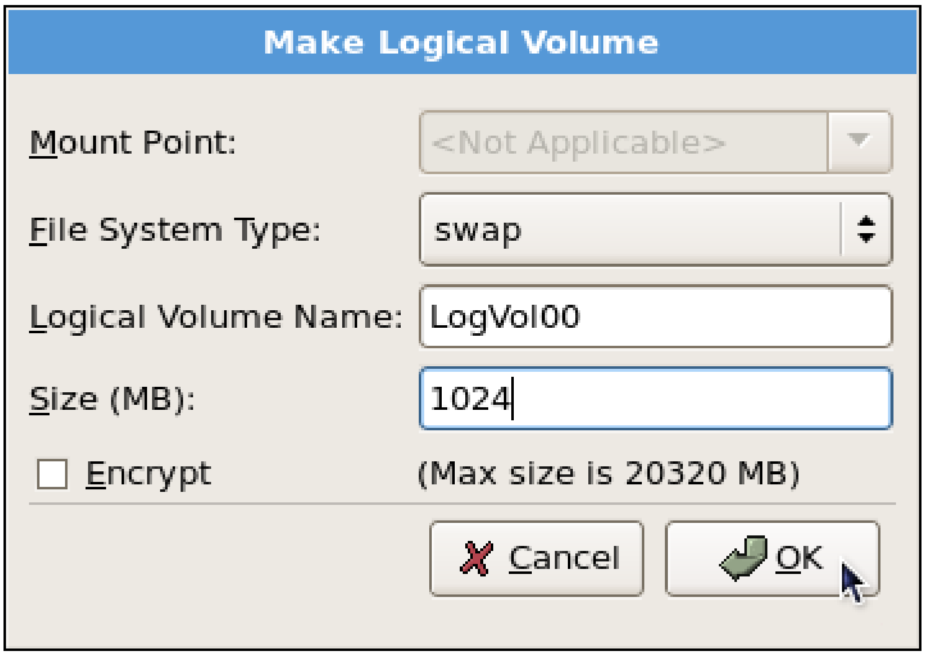
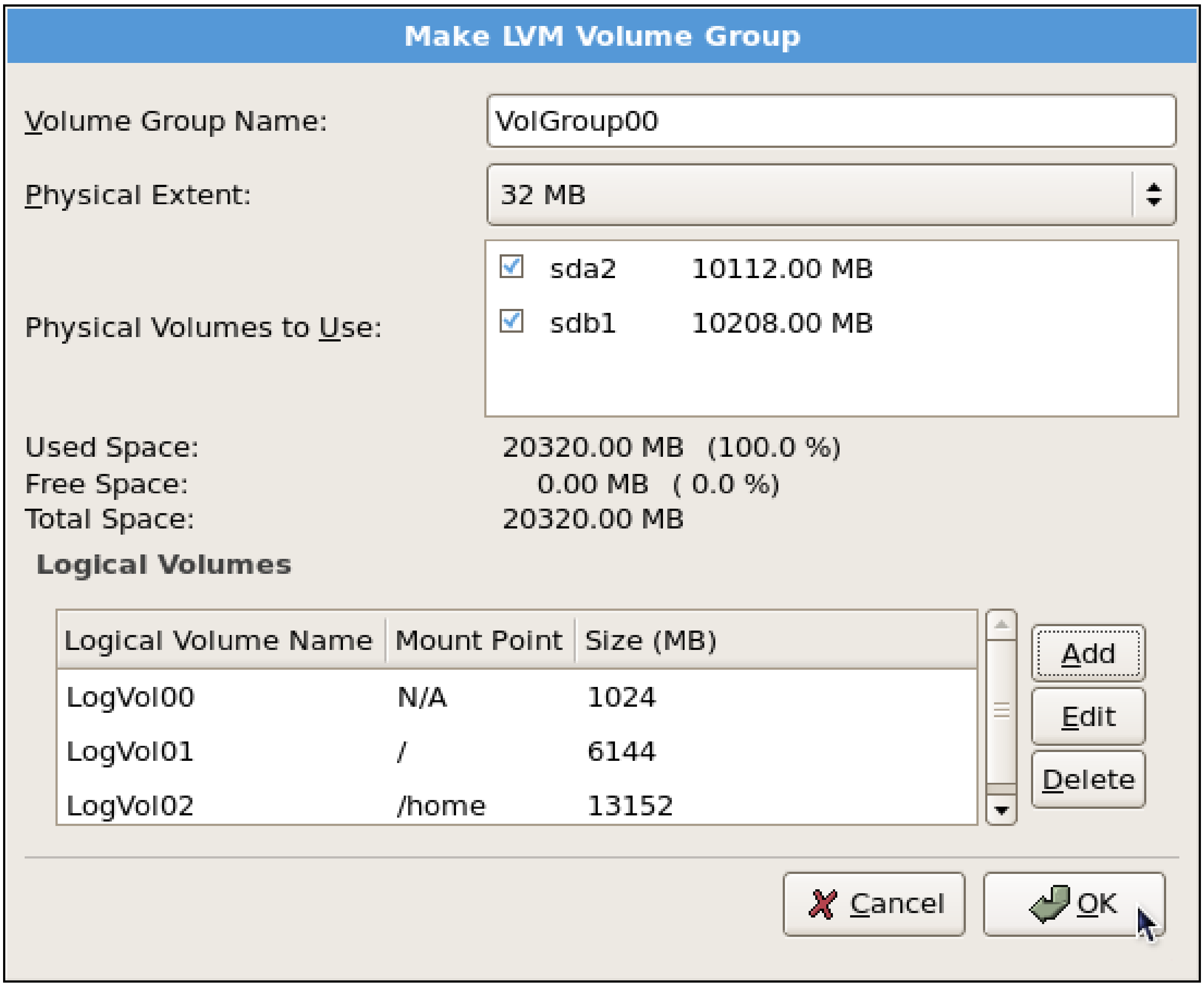


Figure 11.10. Creating a Logical Volume

Repeat these steps for each volume group you want to create.

Note

You may want to leave some free space in the volume group so you can expand the logical volumes later. The default automatic configuration does not do this, but this manual configuration example does — approximately 1 GB is left as free space for future expansion.



Volume Group Name: VolGroup00

Physical Extent: 32 MB

Physical Volumes to Use:

✓	sda2	10112.00 MB
✓	sdb1	10208.00 MB

Used Space: 20320.00 MB (100.0 %)
Free Space: 0.00 MB (0.0 %)
Total Space: 20320.00 MB

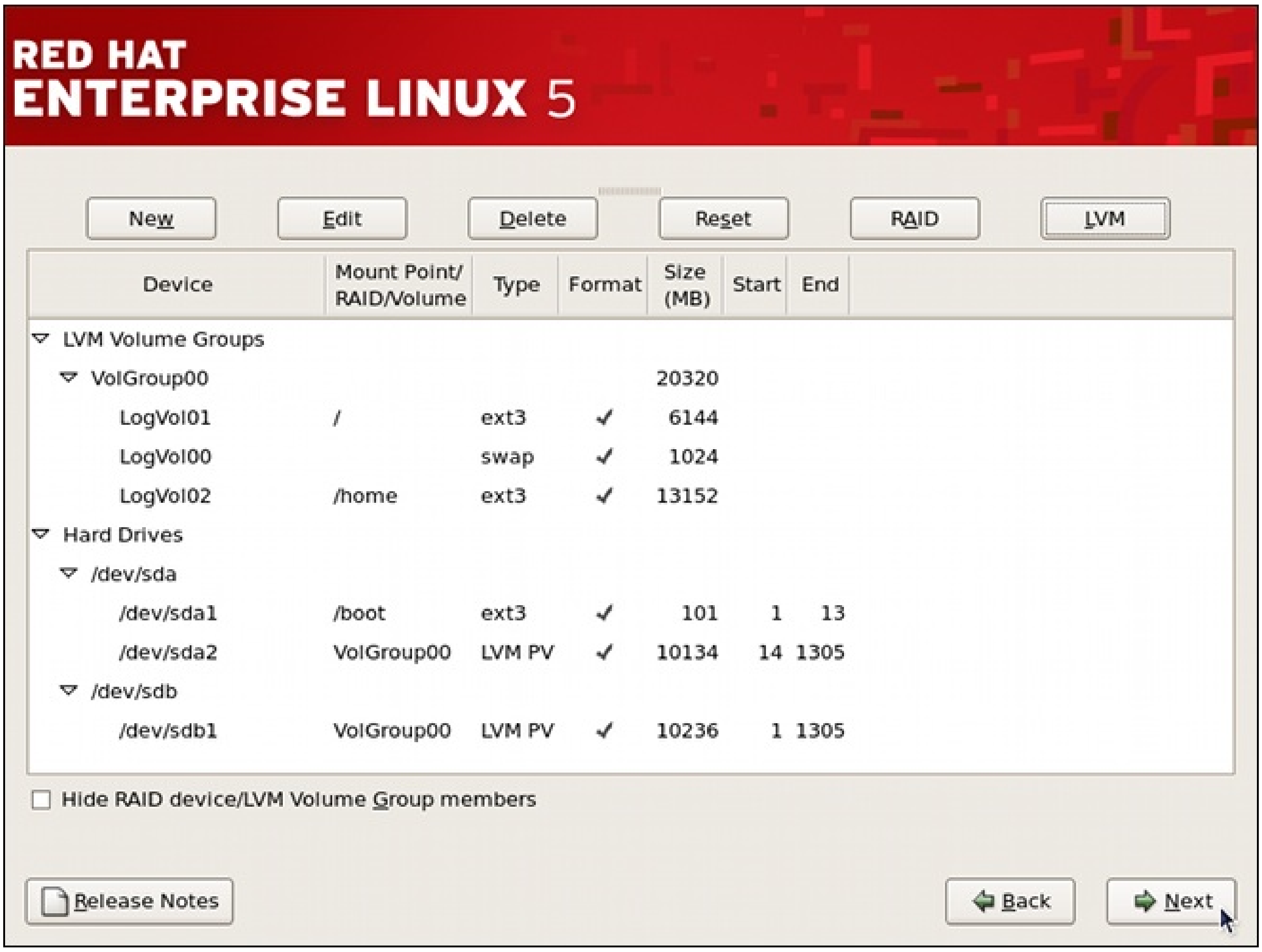
Logical Volumes

Logical Volume Name	Mount Point	Size (MB)
LogVol00	N/A	1024
LogVol01	/	6144
LogVol02	/home	13152

Figure 11.11. Pending Logical Volumes

Click **OK** to apply the volume group and all associated logical volumes.

The following figure shows the final manual configuration:



Device	Mount Point/ RAID/Volume	Type	Format	Size (MB)	Start	End
LVM Volume Groups						
VolGroup00				20320		
LogVol01	/	ext3	✓	6144		
LogVol00		swap	✓	1024		
LogVol02	/home	ext3	✓	13152		
Hard Drives						
/dev/sda						
/dev/sda1	/boot	ext3	✓	101	1	13
/dev/sda2	VolGroup00	LVM PV	✓	10134	14	1305
/dev/sdb						
/dev/sdb1	VolGroup00	LVM PV	✓	10236	1	1305

Figure 11.12. Final Manual Configuration

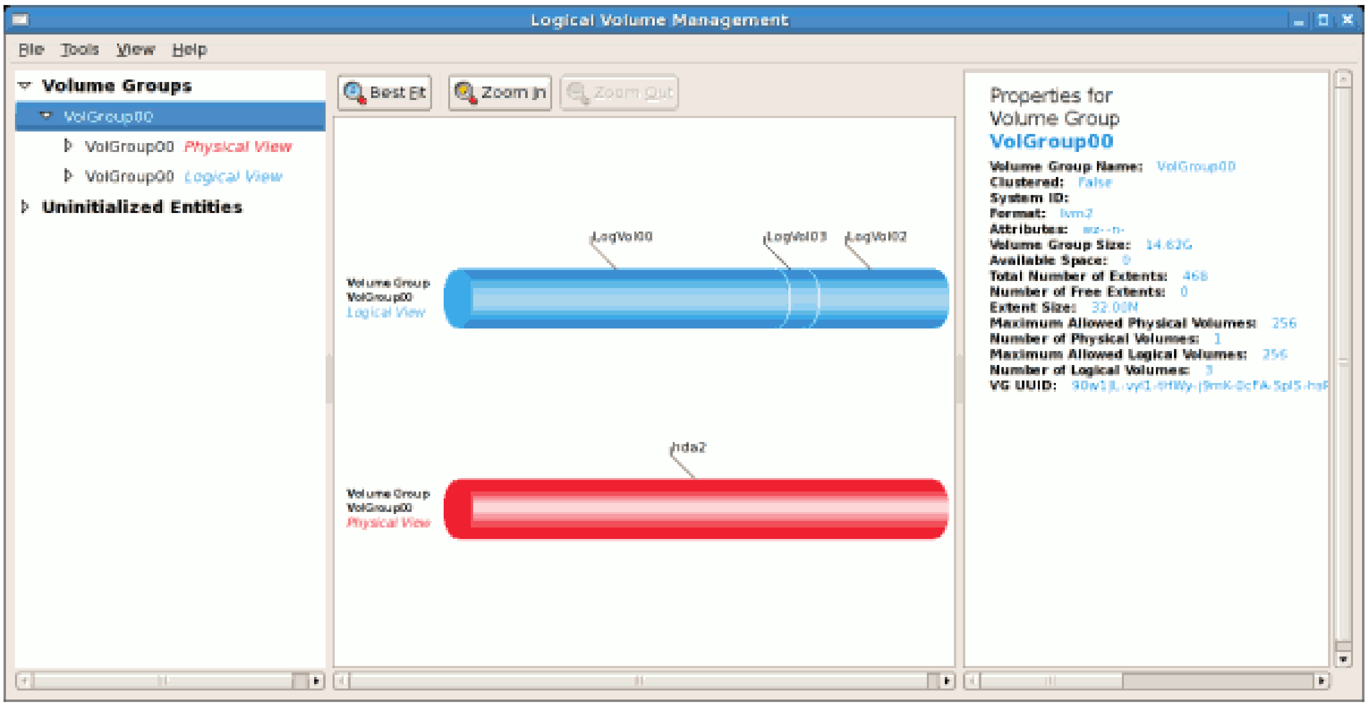
11.5. Using the LVM utility `system-config-lvm`

The LVM utility allows you to manage logical volumes within X windows or graphically. You can access the application by selecting from your menu panel **System** > **Administration** > **Logical Volume Management**. Alternatively you can start the Logical Volume Management utility by typing **`system-config-lvm`** from a terminal.

In the example used in this section, the following are the details for the volume group that was created during the installation:

```
/boot - (Ext3) file system. Displayed under 'Uninitialized Entities'. (DO
NOT initialize this partition).
LogVol00 - (LVM) contains the (/) directory (312 extents).
LogVol02 - (LVM) contains the (/home) directory (128 extents).
LogVol03 - (LVM) swap (28 extents).
```

The logical volumes above were created in disk entity **`/dev/hda2`** while **`/boot`** was created in **`/dev/hda1`**. The system also consists of 'Uninitialized Entities' which are illustrated in Figure 11.17, “Uninitialized Entities”. The figure below illustrates the main window in the LVM utility. The logical and the physical views of the above configuration are illustrated below. The three logical volumes exist on the same physical volume (hda2).

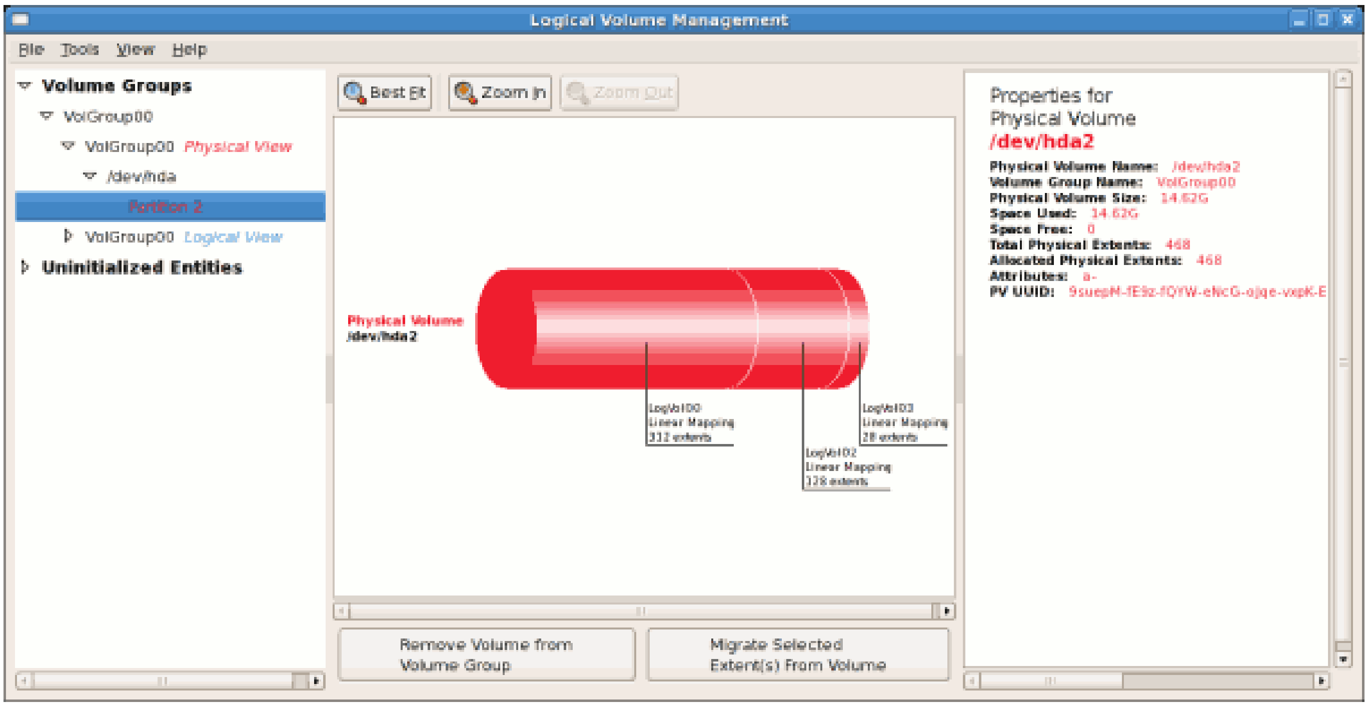


Properties for Volume Group
VolGroup00

Volume Group Name: VolGroup00
Clustered: False
System ID:
Format: lvm2
Attributes: wz--n-
Volume Group Size: 14.62G
Available Space: 0
Total Number of Extents: 468
Number of Free Extents: 0
Extent Size: 32.00M
Maximum Allowed Physical Volumes: 256
Number of Physical Volumes: 1
Maximum Allowed Logical Volumes: 256
Number of Logical Volumes: 3
VG UUID: 90w1jL-vyt1-tHWy-j9mK-0cFA-5pl5-haF

Figure 11.13. Main LVM Window

The figure below illustrates the physical view for the volume. In this window, you can select and remove a volume from the volume group or migrate extents from the volume to another volume group. Steps to migrate extents are discussed in Figure 11.22, "Migrate Extents".

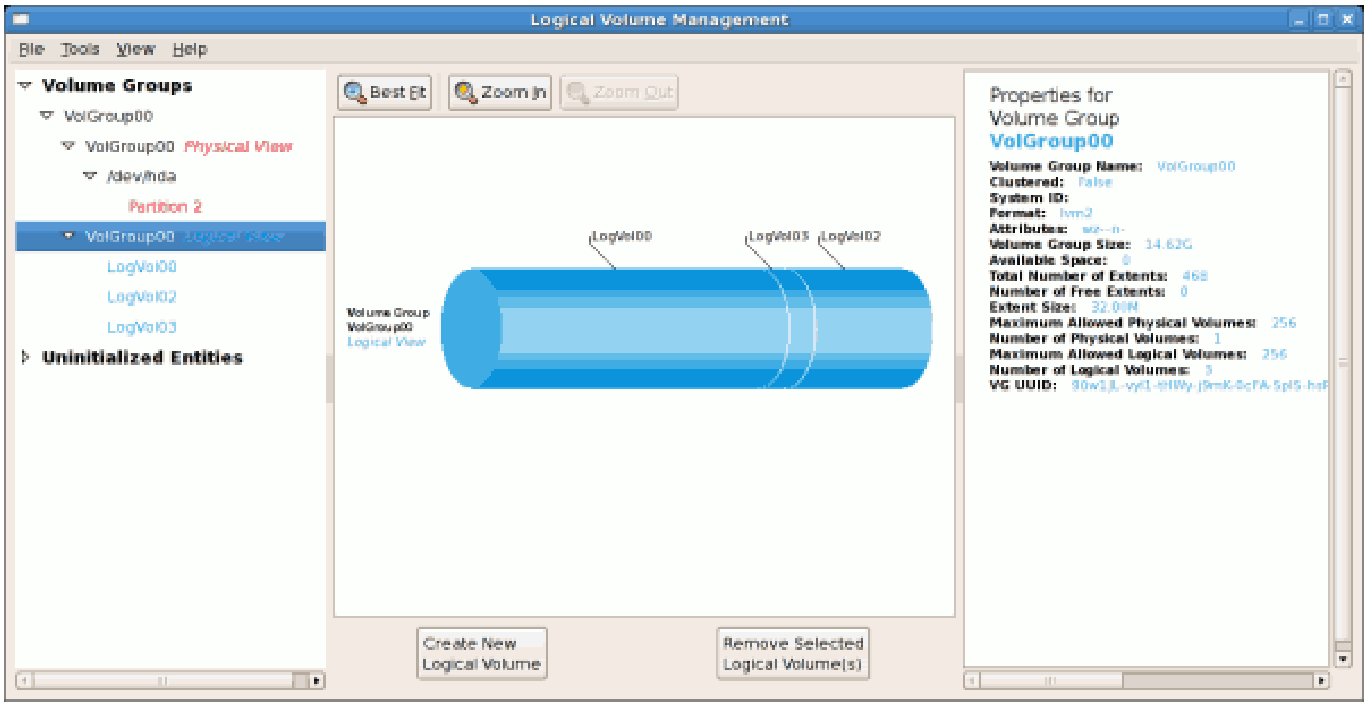

Logical Volume Management
File Tools View Help
Volume Groups
VolGroup00
VolGroup00 Physical View
/dev/hda
Partition 2
VolGroup00 Logical View
Uninitialized Entities
Best Fit
Zoom In
Zoom Out
Physical Volume /dev/hda2
LogVol00 Linear Mapping 312 extents
LogVol03 Linear Mapping 28 extents
LogVol02 Linear Mapping 128 extents
Remove Volume from Volume Group
Migrate Selected Extent(s) From Volume

Properties for Physical Volume
/dev/hda2

Physical Volume Name: /dev/hda2
Volume Group Name: VolGroup00
Physical Volume Size: 14.62G
Space Used: 14.62G
Space Free: 0
Total Physical Extents: 468
Allocated Physical Extents: 468
Attributes: a-
PV UUID: 9suepM-fE9z-fQYW-eNcG-ojqe-vxpK-E

Figure 11.14. Physical View Window

The figure below illustrates the logical view for the selected volume group. The logical volume size is also indicated with the individual logical volume sizes illustrated.



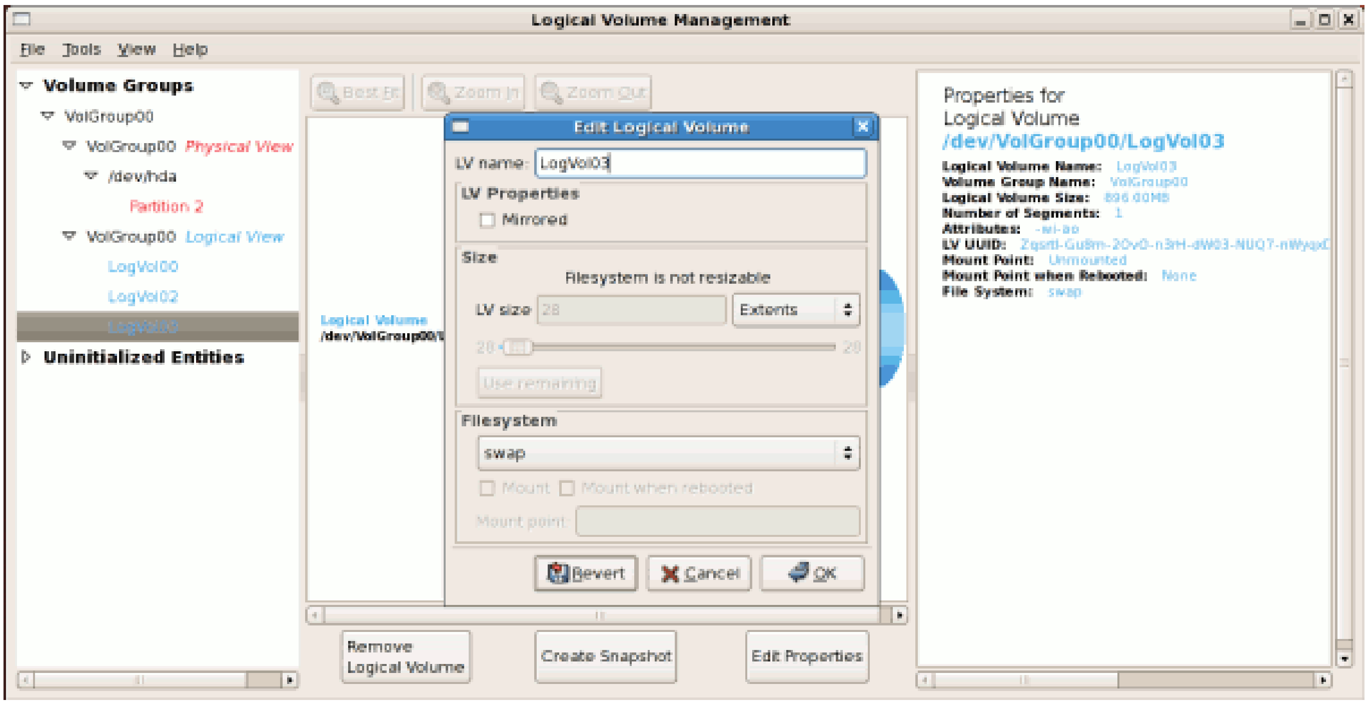
Properties for
Volume Group
VolGroup00

Volume Group Name: VolGroup00
Clustered: False
System ID:
Format: lvm2
Attributes: wz--n-
Volume Group Size: 14.62G
Available Space: 0
Total Number of Extents: 468
Number of Free Extents: 0
Extent Size: 32.00M
Maximum Allowed Physical Volumes: 256
Number of Physical Volumes: 1
Maximum Allowed Logical Volumes: 256
Number of Logical Volumes: 3
VG UUID: 90w1JL-vyKl-4HWy-j9mK-0cFA-5pl5-haF

Figure 11.15. Logical View Window

On the left side column, you can select the individual logical volumes in the volume group to view more details about each. In this example the objective is to rename the logical volume name for 'LogVol03' to 'Swap'. To perform this operation select the respective logical volume and click on the `Edit Properties` button. This will display the Edit Logical Volume window from which you can modify the Logical volume name, size (in extents) and also use the remaining space available in a logical volume group. The figure below illustrates this.

Please note that this logical volume cannot be changed in size as there is currently no free space in the volume group. If there was remaining space, this option would be enabled (see Figure 11.31, “Edit logical volume”). Click on the `OK` button to save your changes (this will remount the volume). To cancel your changes click on the `Cancel` button. To revert to the last snapshot settings click on the `Revert` button. A snapshot can be created by clicking on the `Create Snapshot` button on the LVM utility window. If the selected logical volume is in use by the system (for example) the `/` (root) directory, this task will not be successful as the volume cannot be unmounted.



LV name: LogVol03

LV Properties

☐ Mirrored

Size

Filesystem is not resizable

LV size: 28 Extents

20 28

Use remaining

Filesystem

swap

☐ Mount ☐ Mount when rebooted

Mount point:

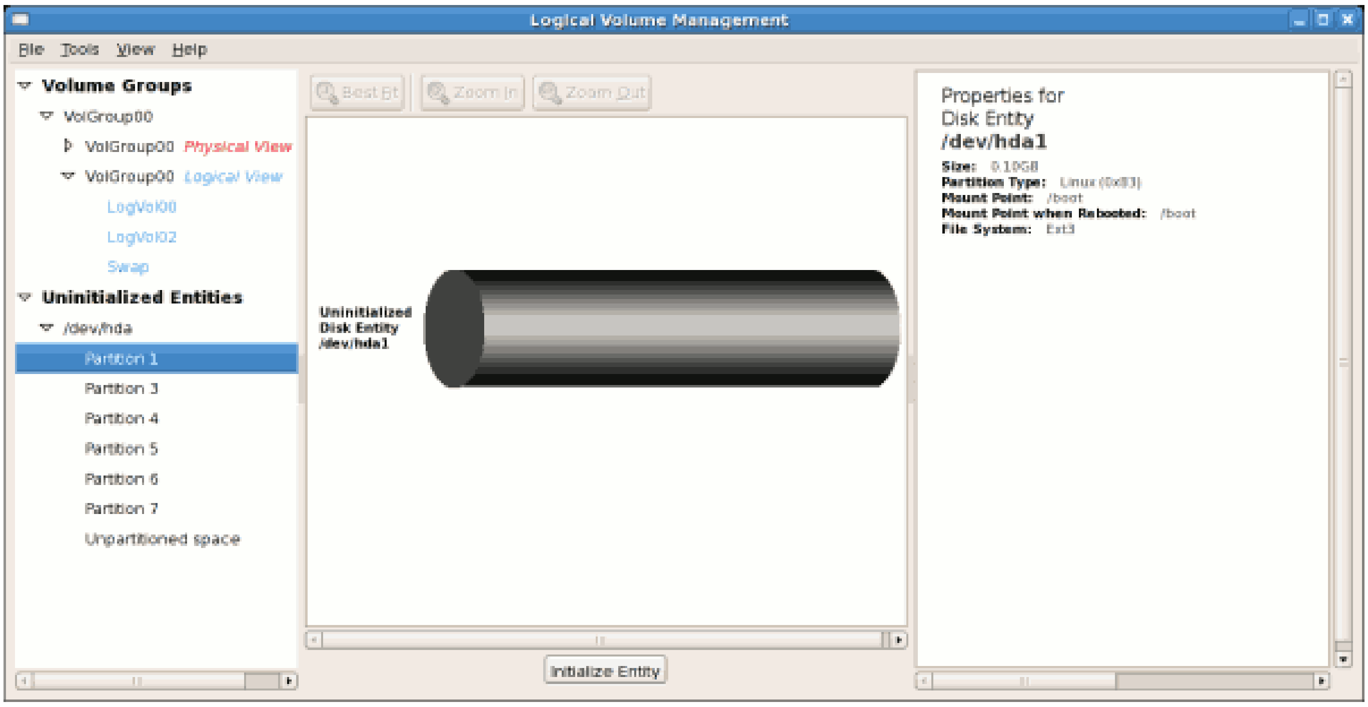
Properties for
Logical Volume
/dev/VolGroup00/LogVol03

Logical Volume Name: LogVol03
Volume Group Name: VolGroup00
Logical Volume Size: 896.00MB
Number of Segments: 1
Attributes: -wi-ao
LV UUID: Zgsrtl-Gu8m-2Ov0-n3rH-dW03-NUQ7-nWyqxC
Mount Point: Unmounted
Mount Point when Rebooted: None
File System: swap

Figure 11.16. Edit Logical Volume

11.5.1. Utilizing uninitialized entities

'Uninitialized Entities' consist of unpartitioned space and non LVM file systems. In this example partitions 3, 4, 5, 6 and 7 were created during installation and some unpartitioned space was left on the hard disk. Please view each partition and ensure that you read the 'Properties for Disk Entity' on the right column of the window to ensure that you do not delete critical data. In this example partition 1 cannot be initialized as it is `/boot`. Uninitialized entities are illustrated below.



Properties for
Disk Entity
/dev/hda1

Size: 0.10GB
Partition Type: Linux (0x83)
Mount Point: /boot
Mount Point when Rebooted: /boot
File System: Ext3

Figure 11.17. Uninitialized Entities

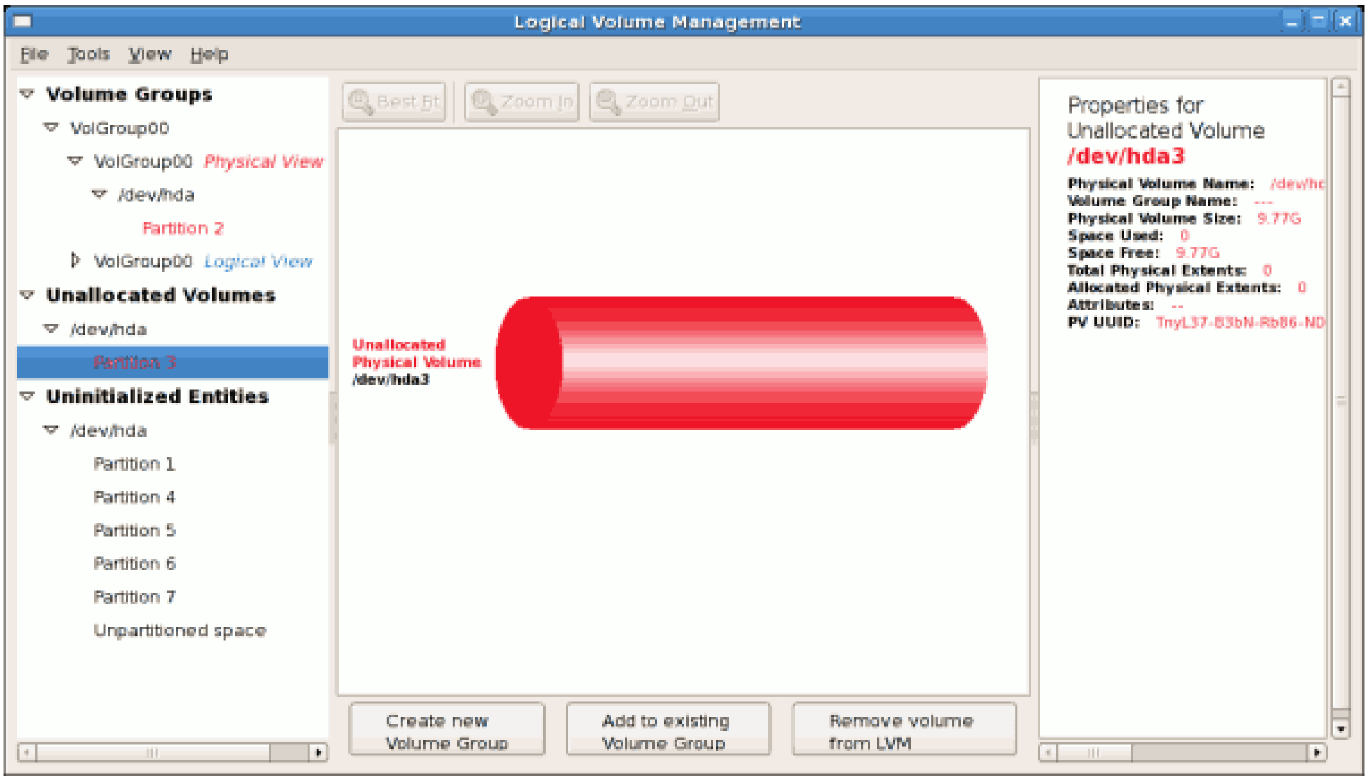
In this example, partition 3 will be initialized and added to an existing volume group. To initialize a partition or unpartioned space, select the partition and click on the **`Initialize Entity`** button. Once initialized, a volume will be listed in the 'Unallocated Volumes' list.

11.5.2. Adding Unallocated Volumes to a volume group

Once initialized, a volume will be listed in the 'Unallocated Volumes' list. The figure below illustrates an unallocated partition (Partition 3). The respective buttons at the bottom of the window allow you to:

- create a new volume group,
- add the unallocated volume to an existing volume group,
- remove the volume from LVM.

To add the volume to an existing volume group, click on the **`Add to Existing Volume Group`** button.



Properties for Unallocated Volume /dev/hda3

Physical Volume Name: /dev/hc
Volume Group Name: ---
Physical Volume Size: 9.77G
Space Used: 0
Space Free: 9.77G
Total Physical Extents: 0
Allocated Physical Extents: 0
Attributes: --
PV UUID: TnyL37-B3bN-Rb86-ND

Figure 11.18. Unallocated Volumes

Clicking on the **`Add to Existing Volume Group`** button will display a pop up window listing the existing volume groups to which you can add the physical volume you are about to initialize. A volume group may span across one or more hard disks. In this example only one volume group exists as illustrated below.

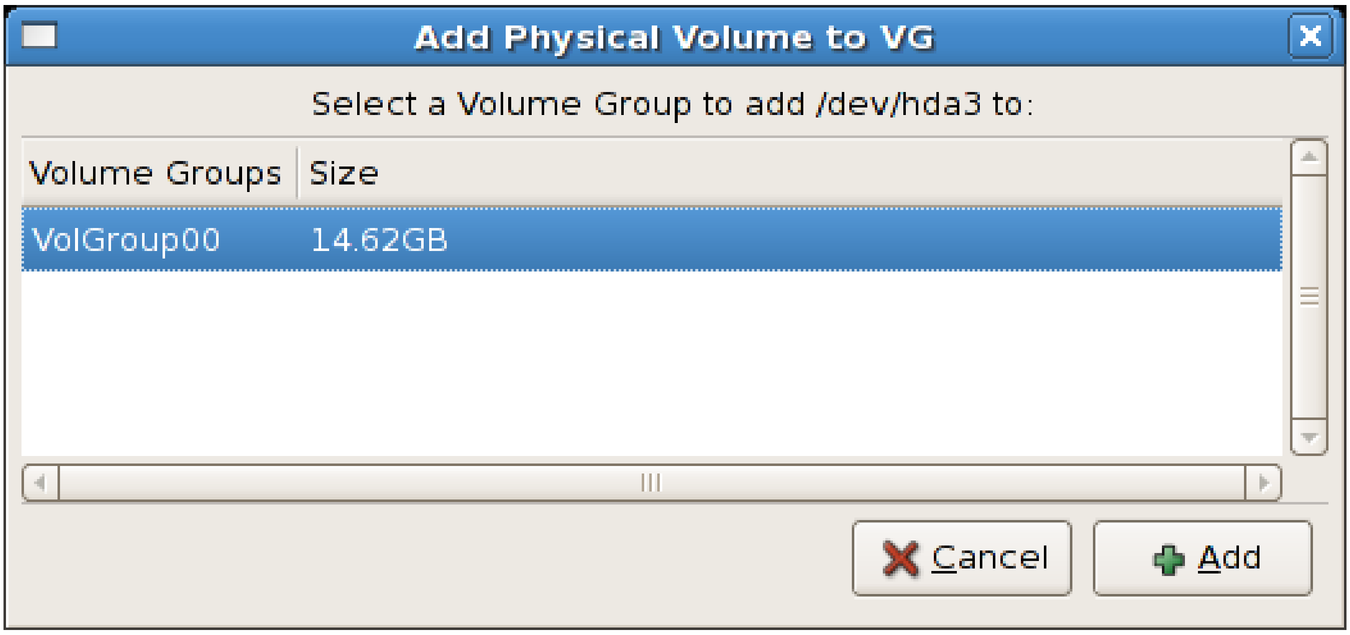
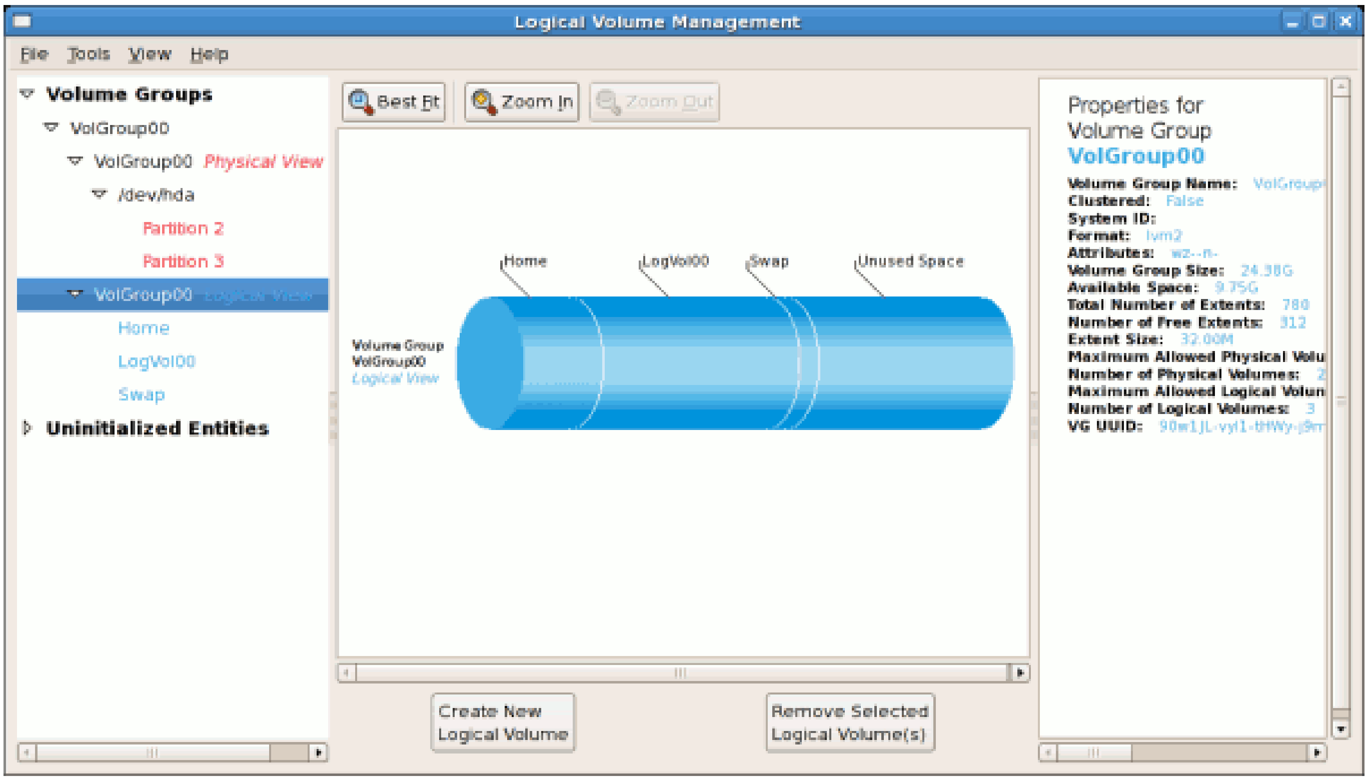

Add Physical Volume to VG
Select a Volume Group to add /dev/hda3 to:
Volume Groups
Size
VolGroup00
14.62GB
Cancel
Add

Figure 11.19. Add physical volume to volume group

Once added to an existing volume group the new logical volume is automatically added to the unused space of the selected volume group. You can use the unused space to:

- create a new logical volume (click on the **`Create New Logical Volume(s)`** button,
- select one of the existing logical volumes and increase the extents (see Section 11.5.6, "Extending a volume group"),
- select an existing logical volume and remove it from the volume group by clicking on the **`Remove Selected Logical Volume(s)`** button. Please note that you cannot select unused space to perform this operation.

The figure below illustrates the logical view of 'VolGroup00' after adding the new volume group.

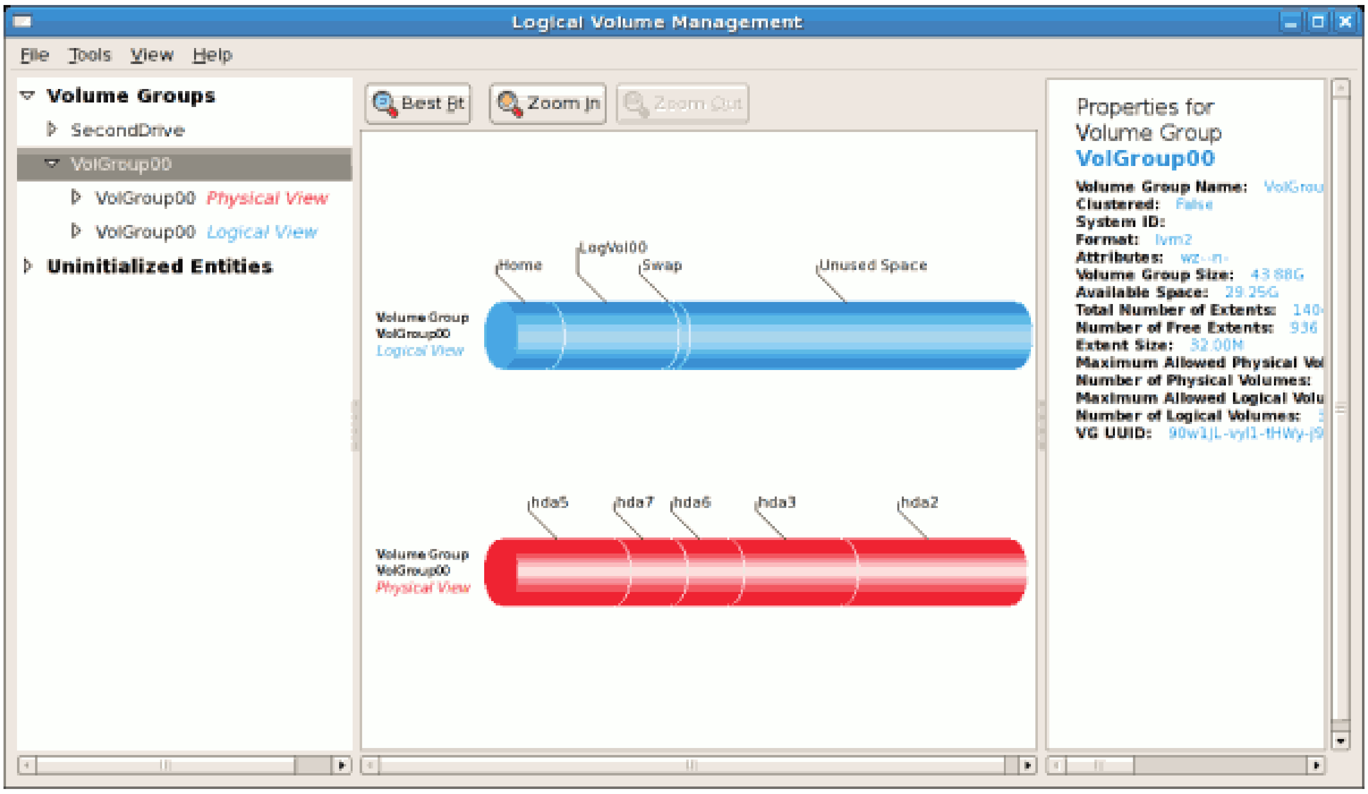


Properties for Volume Group **VolGroup00**

Volume Group Name: VolGroup
Clustered: False
System ID:
Format: lvm2
Attributes: wz--n-
Volume Group Size: 24.38G
Available Space: 9.75G
Total Number of Extents: 780
Number of Free Extents: 312
Extent Size: 32.00M
Maximum Allowed Physical Volu
Number of Physical Volumes: 2
Maximum Allowed Logical Volun
Number of Logical Volumes: 3
VG UUID: 90w1JL-vyl1-tHWy-j9m

Figure 11.20. Logical view of volume group

In the figure below, the uninitialized entities (partitions 3, 5, 6 and 7) were added to 'VolGroup00'.



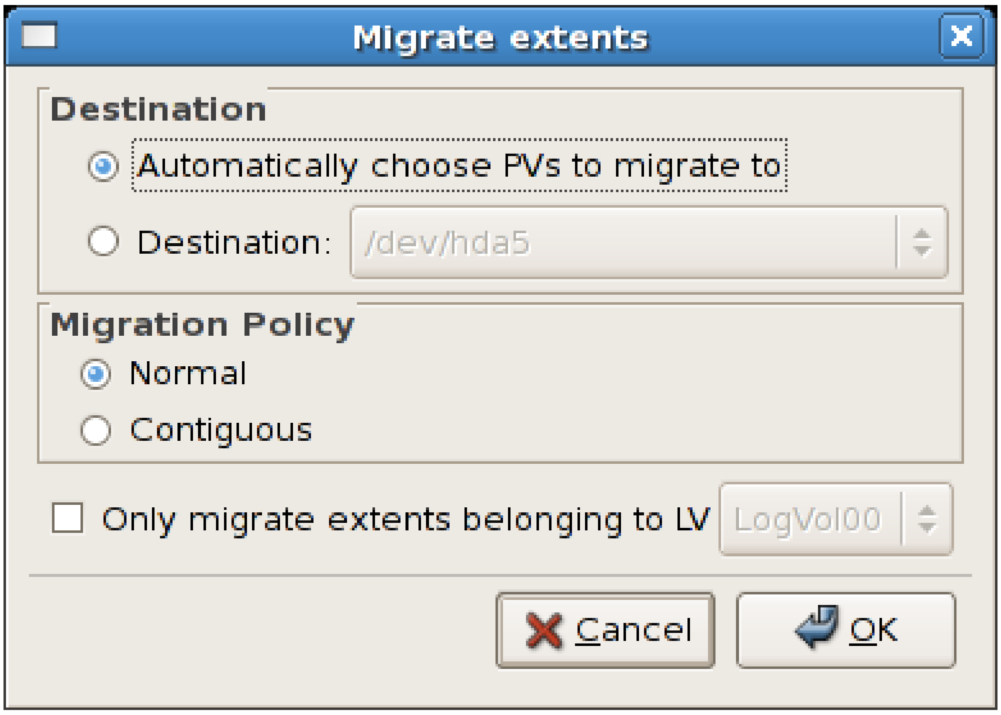
Properties for
Volume Group
VolGroup00

```
Volume Group Name:   VolGrou
Clustered:   False
System ID:
Format:   lvm2
Attributes:   wz--n-
Volume Group Size:   43.88G
Available Space:   29.25G
Total Number of Extents:   140
Number of Free Extents:   936
Extent Size:   32.00M
Maximum Allowed Physical Vol
Number of Physical Volumes:
Maximum Allowed Logical Volu
Number of Logical Volumes:   
VG UUID:   90w1JL-vyl1-tHWy-j9
```

Figure 11.21. Logical view of volume group

11.5.3. Migrating extents

To migrate extents from a physical volume, select the volume and click on the **`Migrate Selected Extent(s) From Volume`** button. Please note that you need to have a sufficient number of free extents to migrate extents within a volume group. An error message will be displayed if you do not have a sufficient number of free extents. To resolve this problem, please extend your volume group (see Section 11.5.6, "Extending a volume group"). If a sufficient number of free extents is detected in the volume group, a pop up window will be displayed from which you can select the destination for the extents or automatically let LVM choose the physical volumes (PVs) to migrate them to. This is illustrated below.



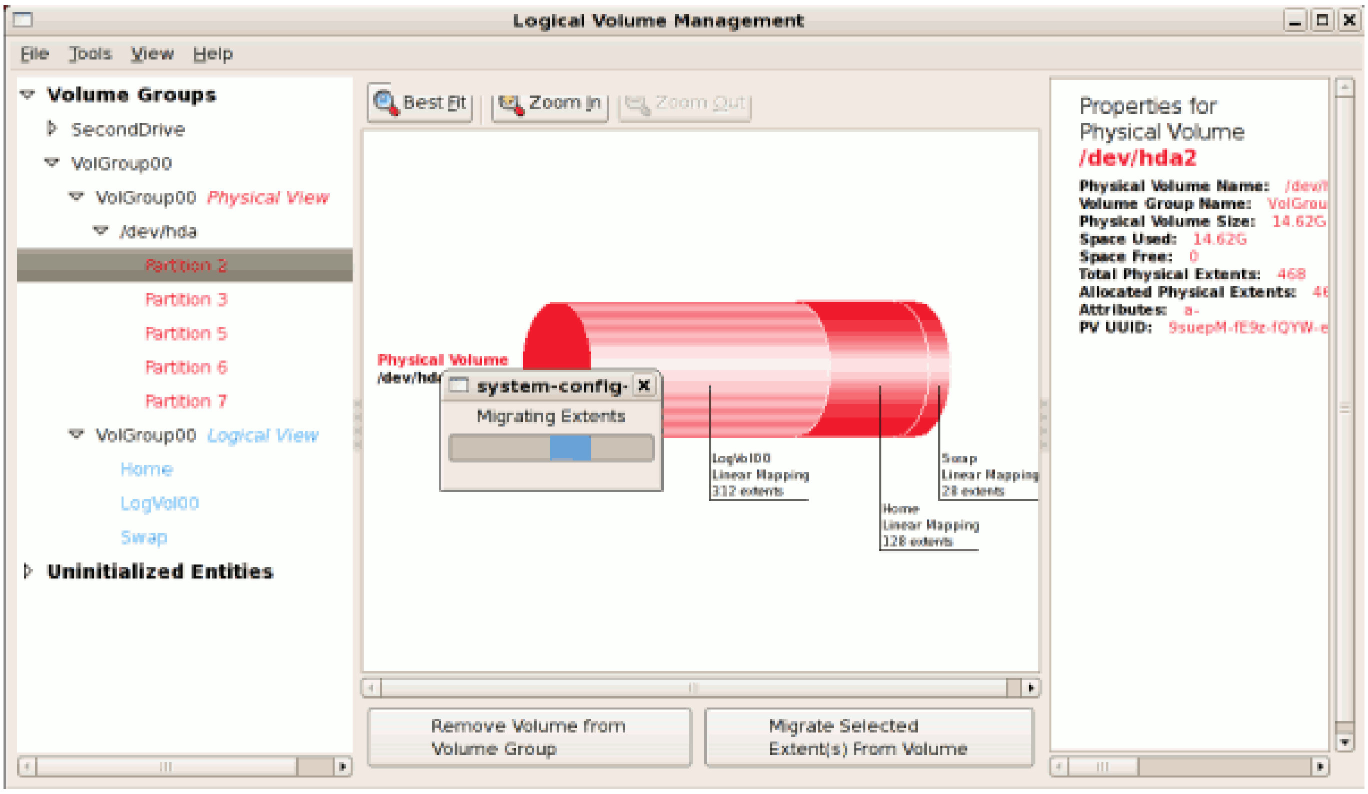
Destination: [✓ Automatically choose PVs to migrate to] Destination: /dev/hda5

Migration Policy: [✓ Normal] Contiguous

☐ Only migrate extents belonging to LV LogVol00

Figure 11.22. Migrate Extents

The figure below illustrates a migration of extents in progress. In this example, the extents were migrated to 'Partition 3'.

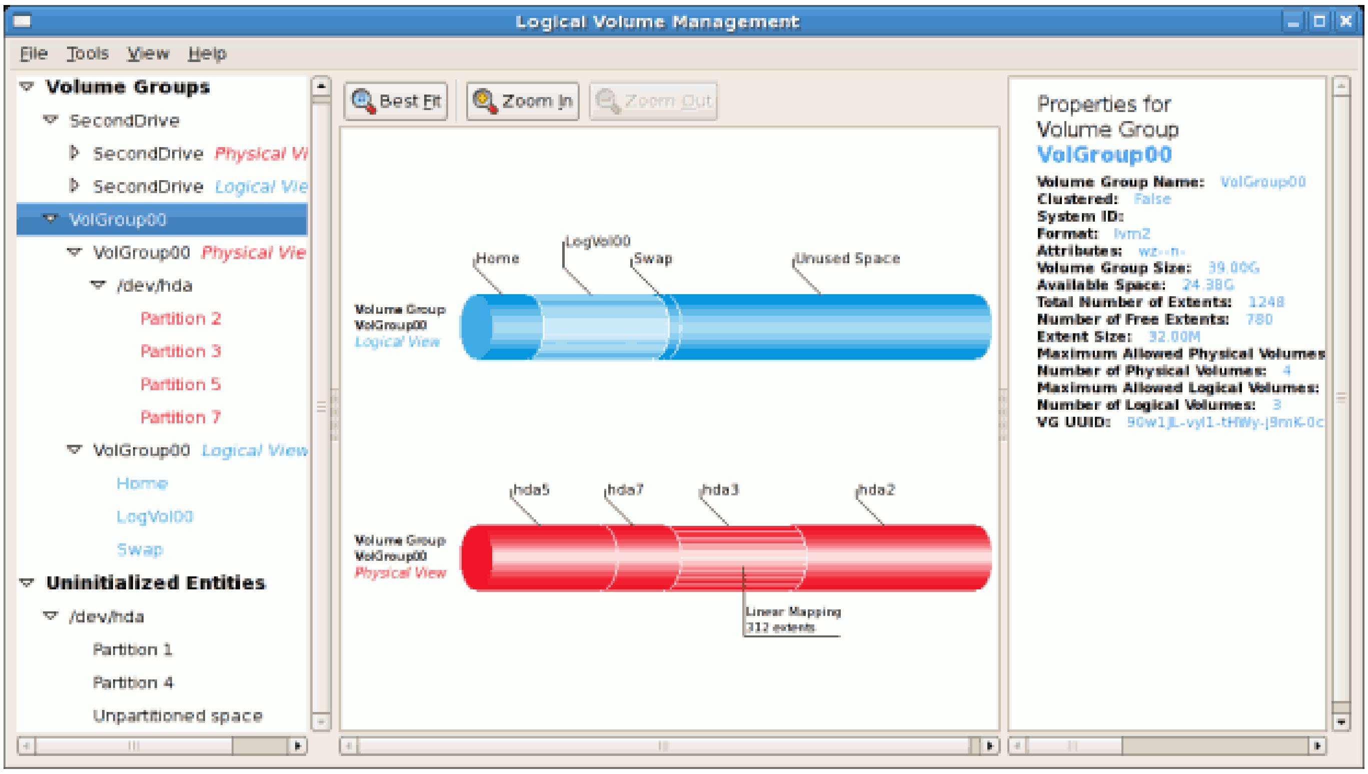


Properties for Physical Volume **/dev/hda2**

Physical Volume Name: /dev/
Volume Group Name: VolGrou
Physical Volume Size: 14.62G
Space Used: 14.62G
Space Free: 0
Total Physical Extents: 468
Allocated Physical Extents: 46
Attributes: a-
PV UUID: 9suepM-fE9z-fQYW-e

Figure 11.23. Migrating extents in progress

Once the extents have been migrated, unused space is left on the physical volume. The figure below illustrates the physical and logical view for the volume group. Please note that the extents of LogVol00 which were initially in hda2 are now in hda3. Migrating extents allows you to move logical volumes in case of hard disk upgrades or to manage your disk space better.



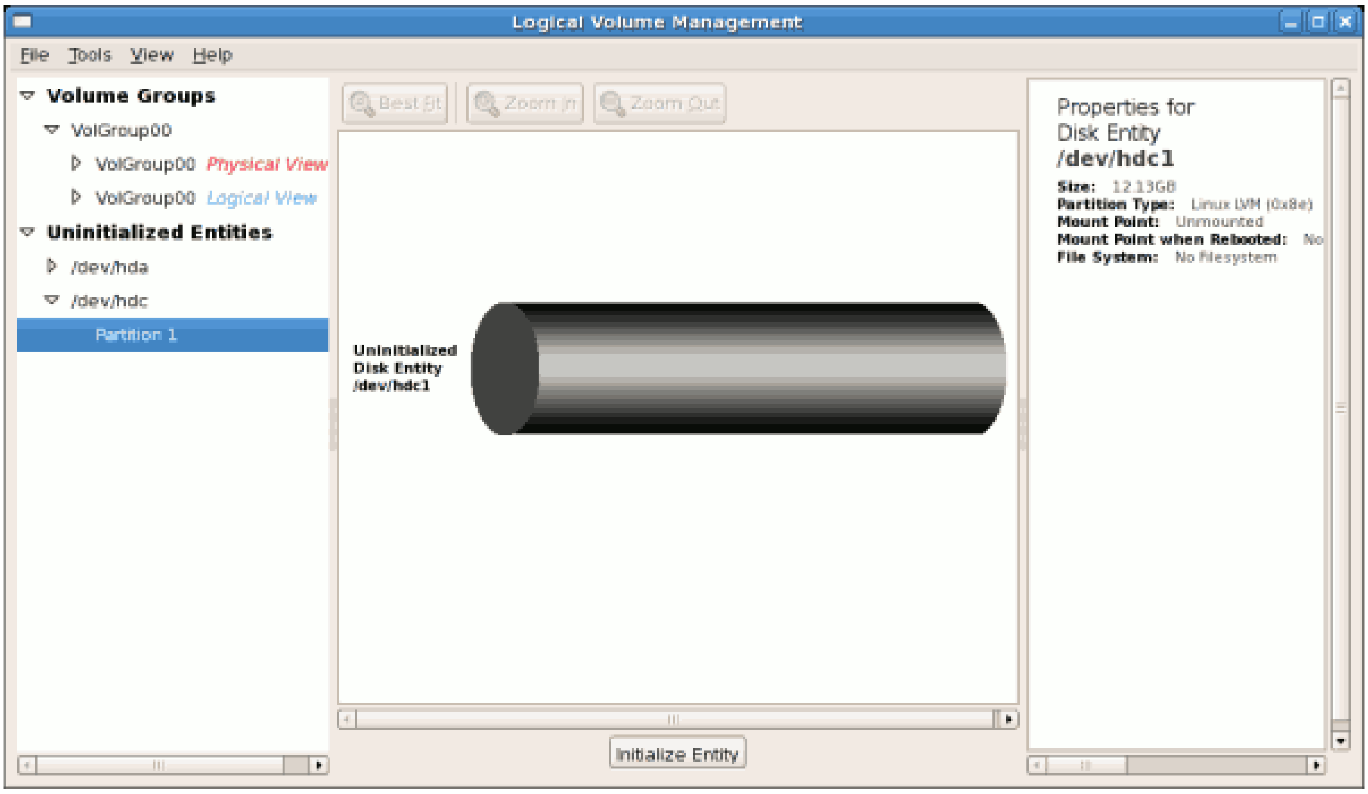
Properties for Volume Group **VolGroup00**

Volume Group Name: VolGroup00
Clustered: False
System ID:
Format: lvm2
Attributes: wz--n-
Volume Group Size: 39.00G
Available Space: 24.38G
Total Number of Extents: 1248
Number of Free Extents: 780
Extent Size: 32.00M
Maximum Allowed Physical Volumes
Number of Physical Volumes: 4
Maximum Allowed Logical Volumes:
Number of Logical Volumes: 3
VG UUID: 90w1JL-vyl1-tHWy-J9mK-0c

Figure 11.24. Logical and physical view of volume group

11.5.4. Adding a new hard disk using LVM

In this example, a new IDE hard disk was added. The figure below illustrates the details for the new hard disk. From the figure below, the disk is uninitialized and not mounted. To initialize a partition, click on the **`Initialize Entity`** button. For more details, see Section 11.5.1, "Utilizing uninitialized entities". Once initialized, LVM will add the new volume to the list of unallocated volumes as illustrated in Figure 11.26, "Create new volume group".



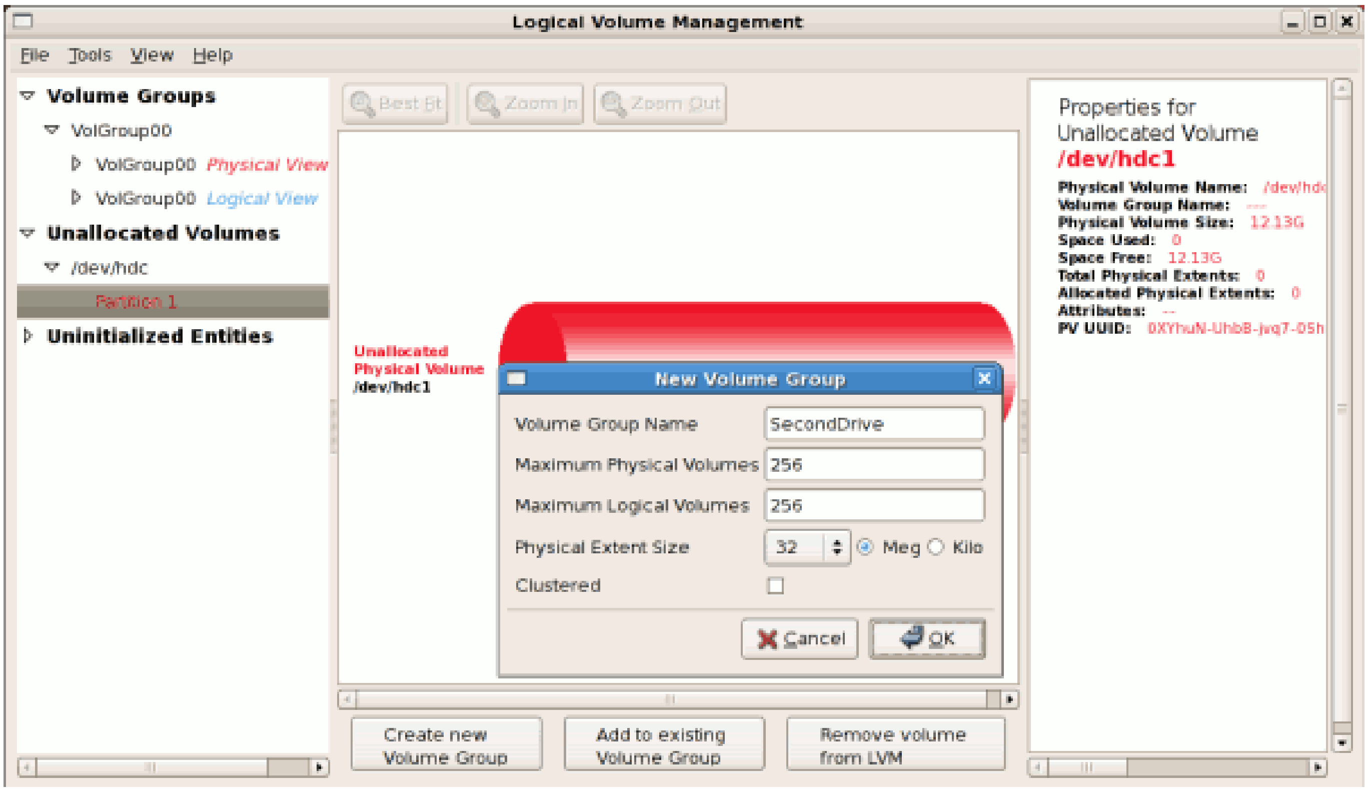
Properties for Disk Entity **/dev/hdc1**

Size: 12.13GB
Partition Type: Linux LVM (0x8e)
Mount Point: Unmounted
Mount Point when Rebooted: No
File System: No Filesystem

Figure 11.25. Uninitialized hard disk

11.5.5. Adding a new volume group

Once initialized, LVM will add the new volume to the list of unallocated volumes where you can add it to an existing volume group or create a new volume group. You can also remove the volume from LVM. The volume if removed from LVM will be listed in the list of 'Uninitialized Entities' as illustrated in Figure 11.25, “Uninitialized hard disk”. In this example, a new volume group was created as illustrated below.

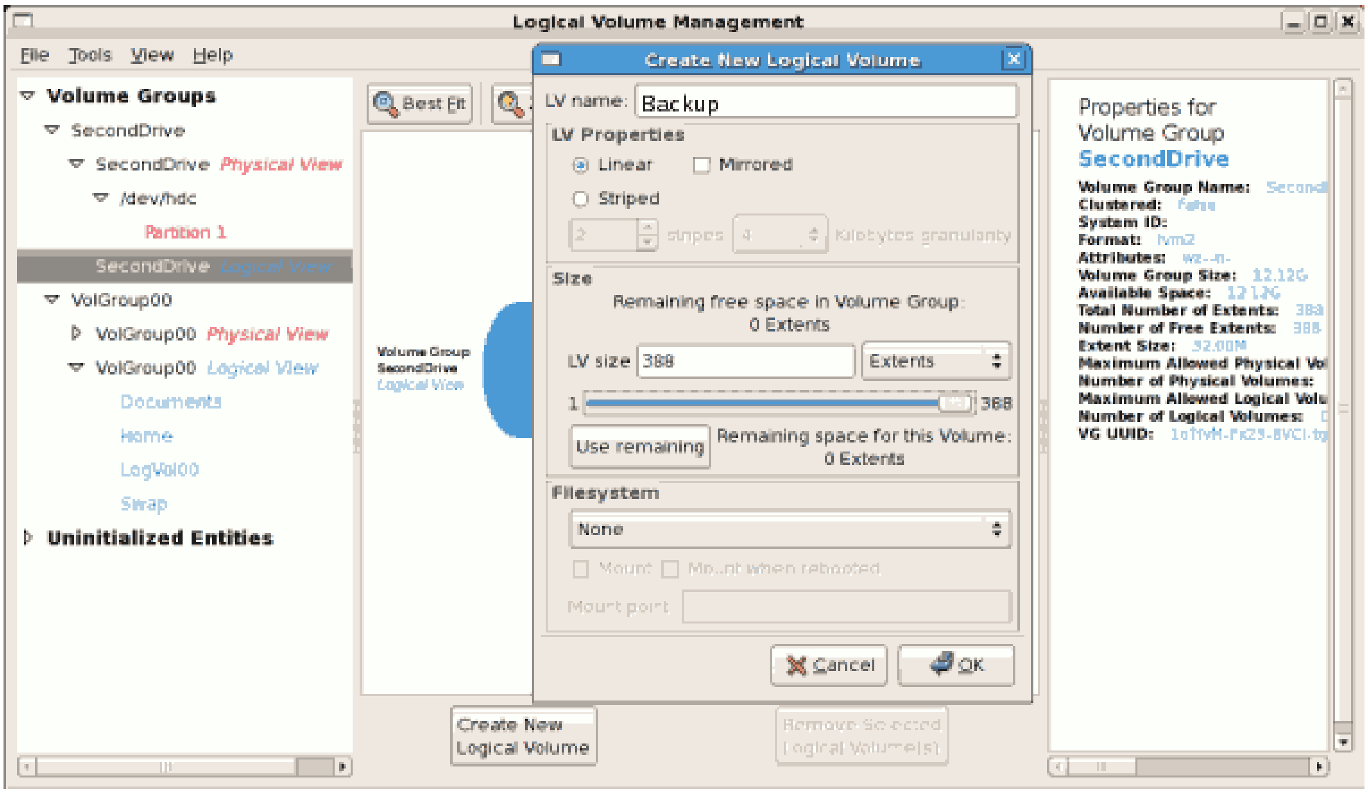


Volume Group Name: SecondDrive
Maximum Physical Volumes: 256
Maximum Logical Volumes: 256
Physical Extent Size: 32 [✓ Meg] Kilo
Clustered: ☐

Properties for Unallocated Volume
/dev/hdc1
Physical Volume Name: /dev/hdc
Volume Group Name: ---
Physical Volume Size: 12.13G
Space Used: 0
Space Free: 12.13G
Total Physical Extents: 0
Allocated Physical Extents: 0
Attributes: --
PV UUID: 0XYhuN-UhbB-jvq7-05h

Figure 11.26. Create new volume group

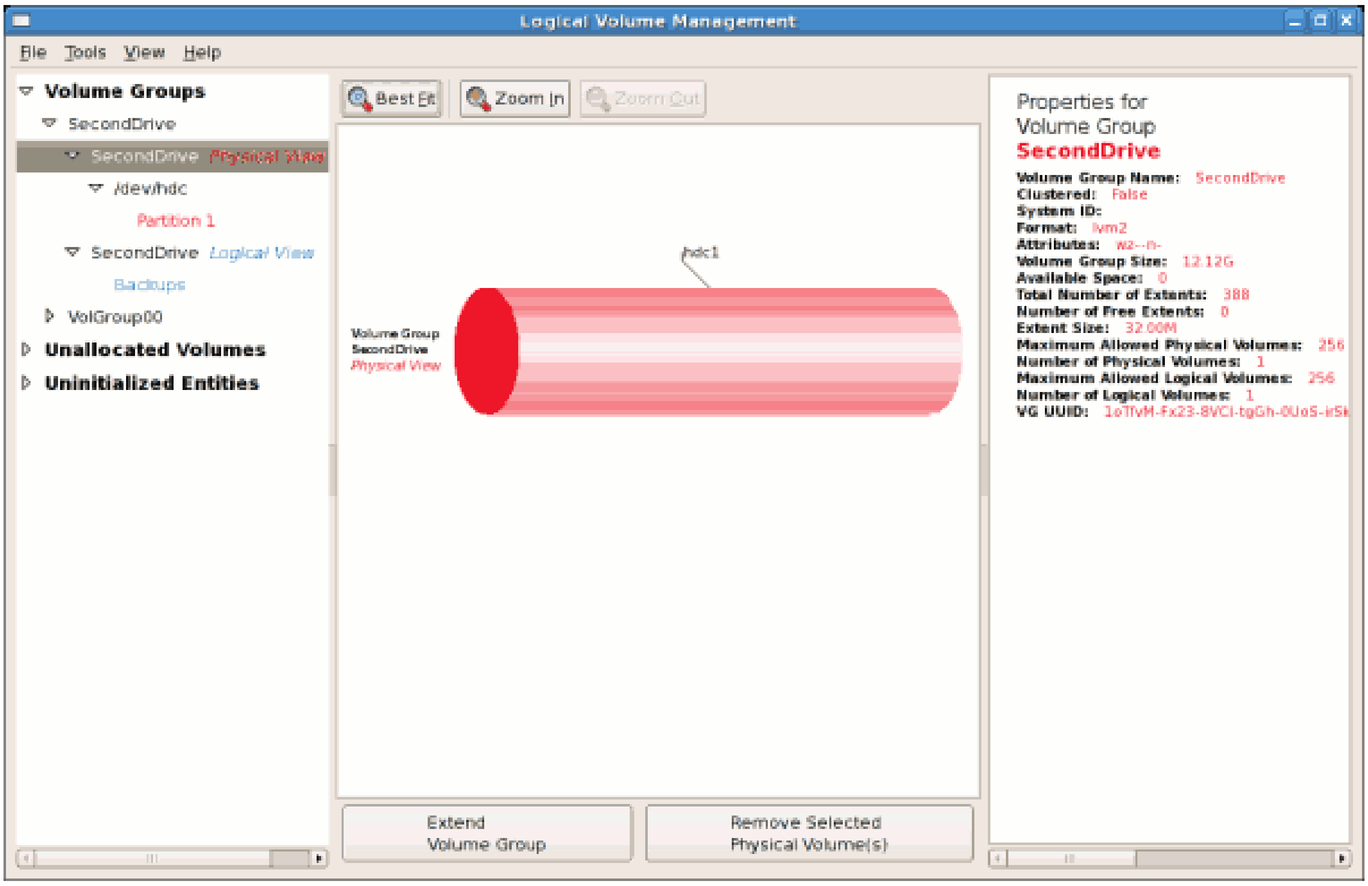
Once created a new volume group will be displayed in the list of existing volume groups as illustrated below. The logical view will display the new volume group with unused space as no logical volumes have been created. To create a logical volume, select the volume group and click on the **`Create New Logical Volume`** button as illustrated below. Please select the extents you wish to use on the volume group. In this example, all the extents in the volume group were used to create the new logical volume.



Volume Group Name: SecondDrive
Clustered: False
System ID:
Format: lvm2
Attributes: wz--n-
Volume Group Size: 12.12G
Available Space: 12.12G
Total Number of Extents: 388
Number of Free Extents: 388
Extent Size: 32.00M
Maximum Allowed Physical Vol
Number of Physical Volumes:
Maximum Allowed Logical Volu
Number of Logical Volumes:
VG UUID: 1oTfvM-Fx23-8VCI-tg

Figure 11.27. Create new logical volume

The figure below illustrates the physical view of the new volume group. The new logical volume named 'Backups' in this volume group is also listed.

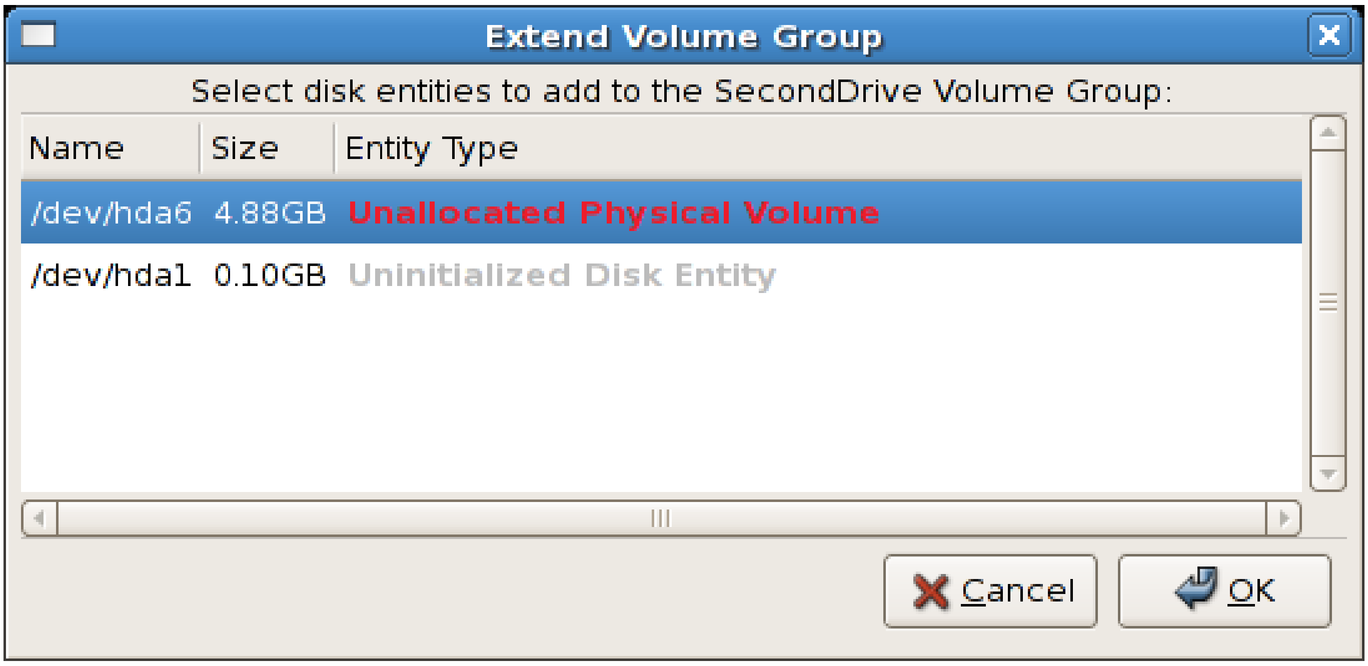


Volume Group Name: SecondDrive
Clustered: False
System ID:
Format: lvm2
Attributes: wz--n-
Volume Group Size: 12.12G
Available Space: 0
Total Number of Extents: 388
Number of Free Extents: 0
Extent Size: 32.00M
Maximum Allowed Physical Volumes: 256
Number of Physical Volumes: 1
Maximum Allowed Logical Volumes: 256
Number of Logical Volumes: 1
VG UUID: 1oTfvM-Fx23-8VCI-tgGh-0UoS-irSk

Figure 11.28. Physical view of new volume group

11.5.6. Extending a volume group

In this example, the objective was to extend the new volume group to include an uninitialized entity (partition). This was to increase the size or number of extents for the volume group. To extend the volume group, click on the **`Extend Volume Group`** button. This will display the 'Extend Volume Group' window as illustrated below. On the 'Extend Volume Group' window, you can select disk entities (partitions) to add to the volume group. Please ensure that you check the contents of any 'Uninitialized Disk Entities' (partitions) to avoid deleting any critical data (see Figure 11.25, "Uninitialized hard disk"). In the example, the disk entity (partition) **`/dev/hda6`** was selected as illustrated below.

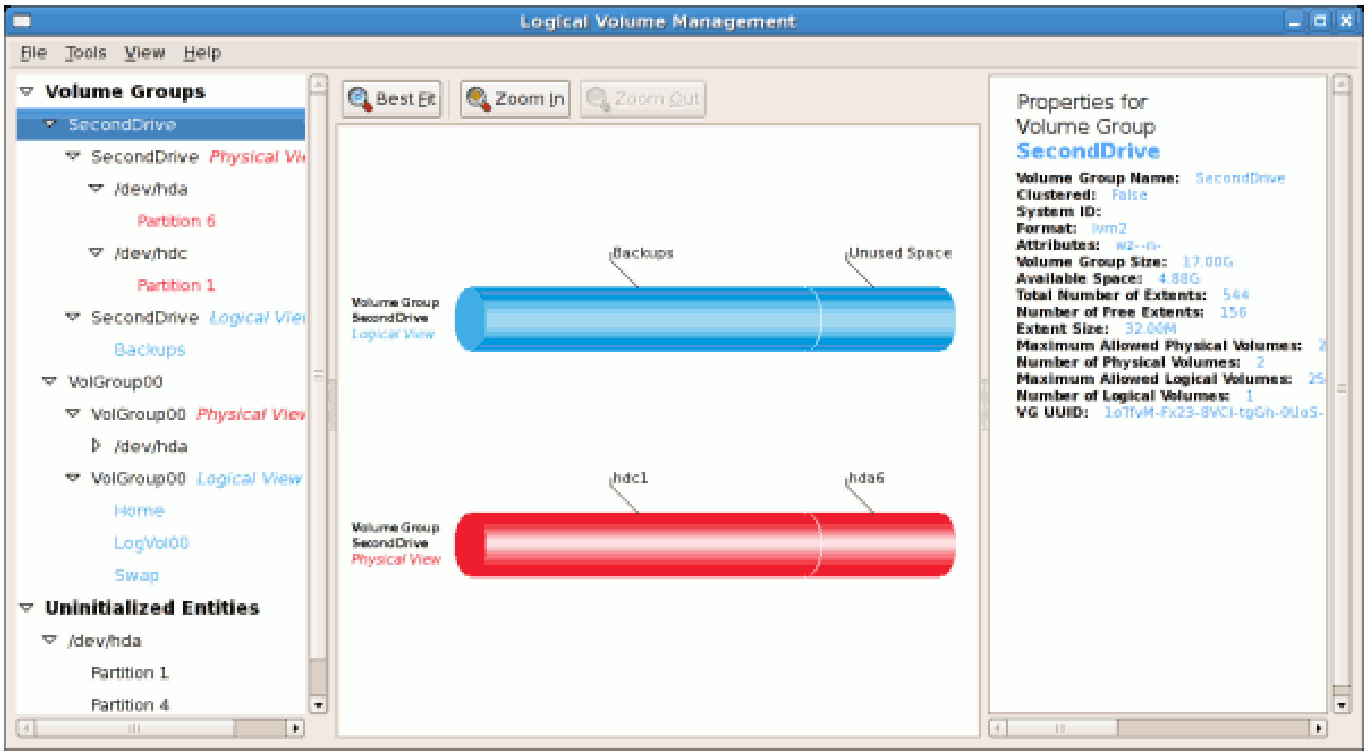


Select disk entities to add to the SecondDrive Volume Group:

Name	Size	Entity Type
/dev/hda6	4.88GB	Unallocated Physical Volume
/dev/hda1	0.10GB	Uninitialized Disk Entity

Figure 11.29. Select disk entities

Once added, the new volume will be added as 'Unused Space' in the volume group. The figure below illustrates the logical and physical view of the volume group after it was extended.



Properties for Volume Group **SecondDrive**

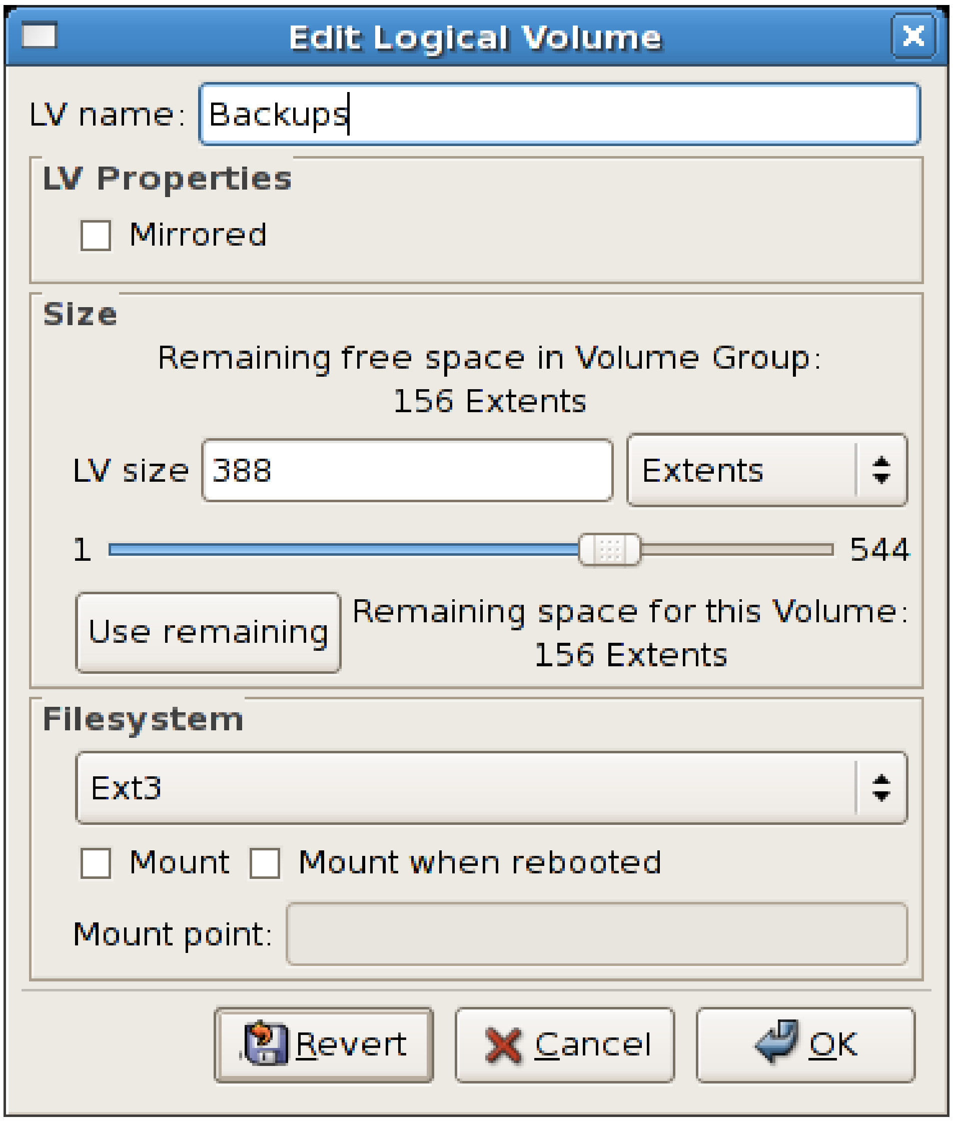
Volume Group Name: SecondDrive
Clustered: False
System ID:
Format: lvm2
Attributes: wz--n-
Volume Group Size: 17.00G
Available Space: 4.88G
Total Number of Extents: 544
Number of Free Extents: 156
Extent Size: 32.00M
Maximum Allowed Physical Volumes: 2
Number of Physical Volumes: 2
Maximum Allowed Logical Volumes: 25
Number of Logical Volumes: 1
VG UUID: 1oTfvM-Fx23-8VCI-tgGh-0UoS-

Figure 11.30. Logical and physical view of an extended volume group

11.5.7. Editing a Logical Volume

The LVM utility allows you to select a logical volume in the volume group and modify its name, size and specify filesystem options. In this example, the logical volume named 'Backups" was extended onto the remaining space for the volume group.

Clicking on the **`Edit Properties`** button will display the 'Edit Logical Volume' popup window from which you can edit the properties of the logical volume. On this window, you can also mount the volume after making the changes and mount it when the system is rebooted. Please note that you should indicate the mount point. If the mount point you specify does not exist, a popup window will be displayed prompting you to create it. The 'Edit Logical Volume' window is illustrated below.


Edit Logical Volume

LV name: Backups

LV Properties

☐ Mirrored

Size

Remaining free space in Volume Group:
156 Extents

LV size 388 Extents

1 544

Use remaining

Remaining space for this Volume:
156 Extents

Filesystem

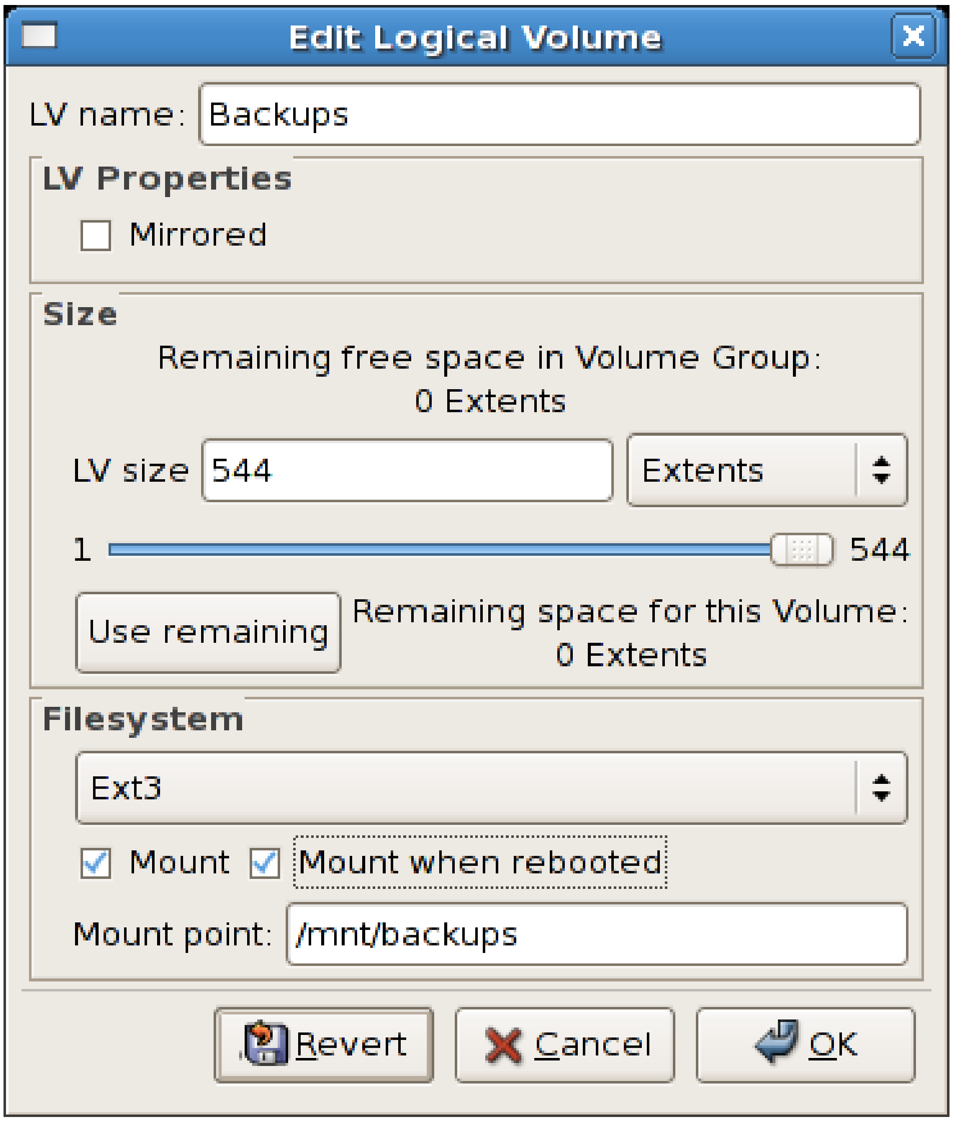
Ext3

☐ Mount ☐ Mount when rebooted

Mount point:

Figure 11.31. Edit logical volume

If you wish to mount the volume, select the 'Mount' checkbox indicating the preferred mount point. To mount the volume when the system is rebooted, select the 'Mount when rebooted' checkbox. In this example, the new volume will be mounted in **`/mnt/backups`**. This is illustrated in the figure below.



Edit Logical Volume

LV name: Backups

LV Properties

☐ Mirrored

Size

Remaining free space in Volume Group:
0 Extents

LV size 544 Extents

1 544

Use remaining

Remaining space for this Volume:
0 Extents

Filesystem

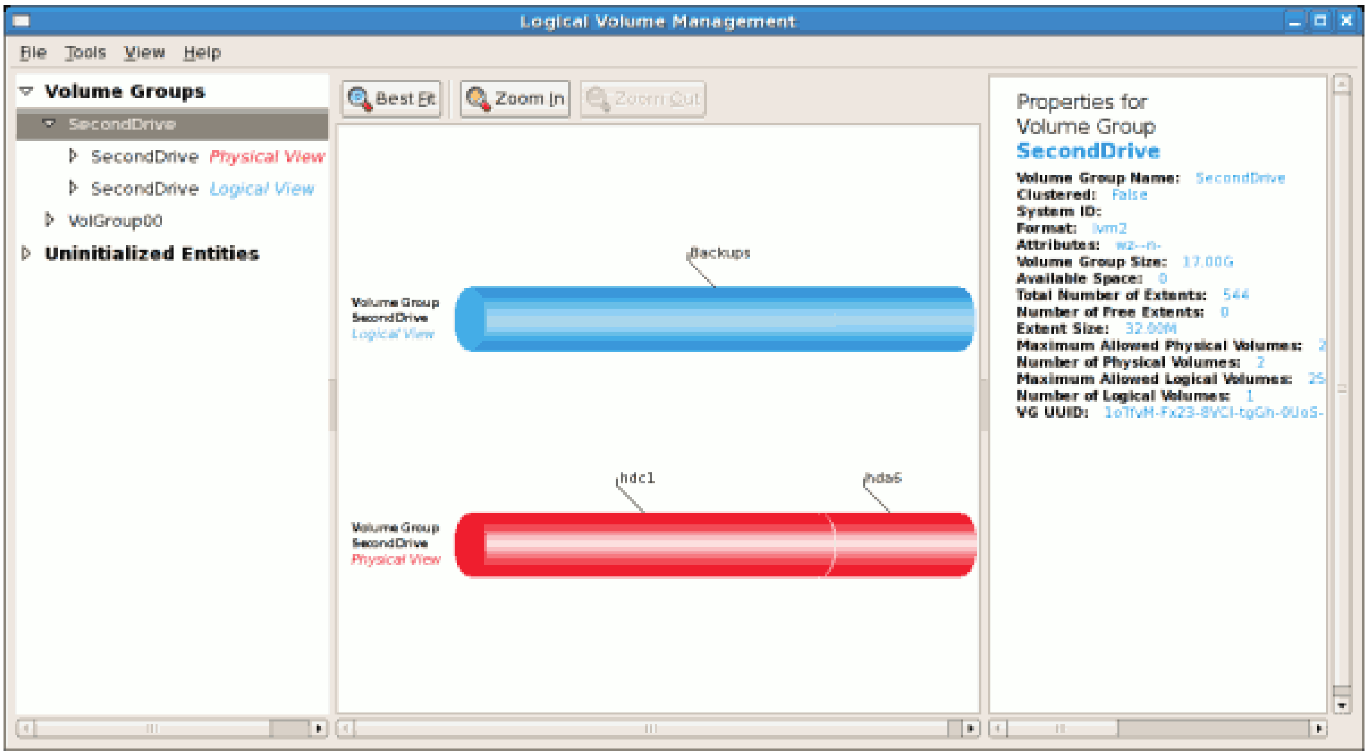
Ext3

☑ Mount ☑ Mount when rebooted

Mount point: /mnt/backups

Figure 11.32. Edit logical volume - specifying mount options

The figure below illustrates the logical and physical view of the volume group after the logical volume was extended to the unused space. Please note in this example that the logical volume named 'Backups' spans across two hard disks. A volume can be striped across two or more physical devices using LVM.



Properties for Volume Group **SecondDrive**

Volume Group Name: SecondDrive
Clustered: False
System ID:
Format: lvm2
Attributes: wz--n-
Volume Group Size: 17.00G
Available Space: 0
Total Number of Extents: 544
Number of Free Extents: 0
Extent Size: 32.00M
Maximum Allowed Physical Volumes: 2
Number of Physical Volumes: 2
Maximum Allowed Logical Volumes: 25
Number of Logical Volumes: 1
VG UUID: 1oTfvM-Fx23-8VCl-tgGh-0UoS-

Figure 11.33. Edit logical volume

11.6. Additional Resources

Use these sources to learn more about LVM.

11.6.1. Installed Documentation

- **`rpm -qd lvm2`** — This command shows all the documentation available from the **`lvm`** package, including man pages.
- **`lvm help`** — This command shows all LVM commands available.

11.6.2. Useful Websites

- http://sources.redhat.com/lvm2 — LVM2 webpage, which contains an overview, link to the mailing lists, and more.
- http://tldp.org/HOWTO/LVM-HOWTO/ — LVM HOWTO from the Linux Documentation Project.

Part II. Package Management

All software on a Red Hat Enterprise Linux system is divided into RPM packages which can be installed, upgraded, or removed. This part describes how to manage the RPM packages on a Red Hat Enterprise Linux system using graphical and command line tools.

Chapter 12. Package Management with RPM

The *RPM Package Manager* (RPM) is an open packaging system, which runs on Red Hat Enterprise Linux as well as other Linux and UNIX systems. Red Hat, Inc. encourages other vendors to use RPM for their own products. RPM is distributed under the terms of the GPL.

The utility works only with packages built for processing by the `rpm` package. For the end user, RPM makes system updates easy. Installing, uninstalling, and upgrading RPM packages can be accomplished with short commands. RPM maintains a database of installed packages and their files, so you can invoke powerful queries and verifications on your system. If you prefer a graphical interface, you can use the **Package Management Tool** to perform many RPM commands. Refer to Chapter 13, ***Package Management Tool*** for details.

Important

When installing a package, please ensure it is compatible with your operating system and architecture. This can usually be determined by checking the package name.

During upgrades, RPM handles configuration files carefully, so that you never lose your customizations — something that you cannot accomplish with regular `.tar.gz` files.

For the developer, RPM allows you to take software source code and package it into source and binary packages for end users. This process is quite simple and is driven from a single file and optional patches that you create. This clear delineation between *pristine* sources and your patches along with build instructions eases the maintenance of the package as new versions of the software are released.

Note

Because RPM makes changes to your system, you must be logged in as root to install, remove, or upgrade an RPM package.

12.1. RPM Design Goals

To understand how to use RPM, it can be helpful to understand the design goals of RPM:

Upgradability

With RPM, you can upgrade individual components of your system without completely reinstalling. When you get a new release of an operating system based on RPM (such as Red Hat Enterprise Linux), you do not need to reinstall on your machine (as you do with operating systems based on other packaging systems). RPM allows intelligent, fully-automated, in-place upgrades of your system. Configuration files in packages are preserved across upgrades, so you do not lose your customizations. There are no special upgrade files needed to upgrade a package because the same RPM file is used to install and upgrade the package on your system.

Powerful Querying

RPM is designed to provide powerful querying options. You can do searches through your entire database for packages or just for certain files. You can also easily find out what package a file belongs to and from where the package came. The files an RPM package contains are in a compressed archive, with a custom binary header containing useful information about the package and its contents, allowing you to query individual packages quickly and easily.

System Verification

Another powerful RPM feature is the ability to verify packages. If you are worried that you deleted an important file for some package, you can verify the package. You are then notified of any anomalies, if any — at which point, you can reinstall the package if necessary. Any configuration files that you modified are preserved during reinstallation.

Pristine Sources

A crucial design goal was to allow the use of *pristine* software sources, as distributed by the original authors of the software. With RPM, you have the pristine sources along with any patches that were used, plus complete build instructions. This is an important advantage for several reasons. For instance, if a new version of a program is released, you do not necessarily have to start from scratch to get it to compile. You can look at the patch to see what you *might* need to do. All the compiled-in defaults, and all of the changes that were made to get the software to build properly, are easily visible using this technique.

The goal of keeping sources pristine may seem important only for developers, but it results in higher quality software for end users, too.

12.2. Using RPM

RPM has five basic modes of operation (not counting package building): installing, uninstalling, upgrading, querying, and verifying. This section contains an overview of each mode. For complete details and options, try **`rpm --help`** or **`man rpm`**. You can also refer to Section 12.5, “Additional Resources” for more information on RPM.

12.2.1. Finding RPM Packages

Before using any RPM packages, you must know where to find them. An Internet search returns many RPM repositories, but if you are looking for RPM packages built by Red Hat, they can be found at the following locations:

- The Red Hat Enterprise Linux CD-ROMs
- The Red Hat Errata Page available at http://www.redhat.com/apps/support/errata/
- Red Hat Network — Refer to Chapter 15, *Registering a System and Managing Subscriptions* for more details on Red Hat Network.

12.2.2. Installing

RPM packages typically have file names like **`foo-1.0-1.i386.rpm`**. The file name includes the package name (**`foo`**), version (**`1.0`**), release (**`1`**), and architecture (**`i386`**). To install a package, log in as root and type the following command at a shell prompt:

```
rpm -ivh foo-1.0-1.i386.rpm
```

Alternatively, the following command can also be used:

```
rpm -Uvh foo-1.0-1.i386.rpm
```

If the installation is successful, the following output is displayed:

```
Preparing...                ###########################################
[100%]
   1:foo                    ###########################################
[100%]
```

As you can see, RPM prints out the name of the package and then prints a succession of hash marks as a progress meter while the package is installed.

The signature of a package is checked automatically when installing or upgrading a package. The signature confirms that the package was signed by an authorized party. For example, if the verification of the signature fails, an error message such as the following is displayed:

```
error: V3 DSA signature: BAD, key ID 0352860f
```

If it is a new, header-only, signature, an error message such as the following is displayed:

```
error: Header V3 DSA signature: BAD, key ID 0352860f
```

If you do not have the appropriate key installed to verify the signature, the message contains the word **`NOKEY`** such as:

```
warning: V3 DSA signature: NOKEY, key ID 0352860f
```

Refer to Section 12.3, “Checking a Package's Signature” for more information on checking a package's signature.

Warning

If you are installing a kernel package, you should use **`rpm -ivh`** instead. Refer to Chapter 44, *Manually Upgrading the Kernel* for details.

12.2.2.1. Package Already Installed

If a package of the same name and version is already installed, the following output is displayed:

```
Preparing...                ###########################################
[100%]
package foo-1.0-1 is already installed
```

However, if you want to install the package anyway, you can use the **`--replacepkgs`** option, which tells RPM to ignore the error:

```
rpm -ivh --replacepkgs foo-1.0-1.i386.rpm
```

This option is helpful if files installed from the RPM were deleted or if you want the original configuration files from the RPM to be installed.

12.2.2.2. Conflicting Files

If you attempt to install a package that contains a file which has already been installed by another package, the following is displayed:

```
Preparing...                ############################################
[100%]
file /usr/bin/foo from install of foo-1.0-1 conflicts with file from package
bar-2.0.20
```

To make RPM ignore this error, use the **`--replacefiles`** option:

```
rpm -ivh --replacefiles foo-1.0-1.i386.rpm
```

12.2.2.3. Unresolved Dependency

RPM packages may sometimes depend on other packages, which means that they require other packages to be installed to run properly. If you try to install a package which has an unresolved dependency, output similar to the following is displayed:

```
error: Failed dependencies:
        bar.so.2 is needed by foo-1.0-1
Suggested resolutions:
 bar-2.0.20-3.i386.rpm
```

If you are installing a package from the Red Hat Enterprise Linux CD-ROM set, it usually suggest the package(s) needed to resolve the dependency. Find the suggested package(s) on the Red Hat Enterprise Linux CD-ROMs or from Red Hat Network , and add it to the command:

```
rpm -ivh foo-1.0-1.i386.rpm bar-2.0.20-3.i386.rpm
```

If installation of both packages is successful, output similar to the following is displayed:

```
Preparing...                ############################################
[100%]
   1:foo                    ############################################ [
50%]
   2:bar                    ############################################
[100%]
```

If it does not suggest a package to resolve the dependency, you can try the **`-q --whatprovides`** option combination to determine which package contains the required file.

```
rpm -q --whatprovides bar.so.2
```

To force the installation anyway (which is not recommended since the package may not run correctly), use the **`--nodeps`** option.

12.2.3. Uninstalling

Uninstalling a package is just as simple as installing one. Type the following command at a shell prompt:

```
rpm -e foo
```

Note

Notice that we used the package *name* **`foo`**, not the name of the original package *file* **`foo-1.0-1.i386.rpm`**. To uninstall a package, replace **`foo`** with the actual package name of the original package.

You can encounter a dependency error when uninstalling a package if another installed package depends on the one you are trying to remove. For example:

```
error: Failed dependencies:
 foo is needed by (installed) bar-2.0.20-3.i386.rpm
```

To make RPM ignore this error and uninstall the package anyway (which may break the package dependent on it) use the **`--nodeps`** option.

12.2.4. Upgrading

Upgrading a package is similar to installing one. Type the following command at a shell prompt:

```
rpm -Uvh foo-2.0-1.i386.rpm
```

As part of upgrading a package, RPM automatically uninstalls any old versions of the **`foo`** package. Note that **`-U`** will also install a package even when there are no previous versions of the package installed.

Note

It is not advisable to use the **`-U`** option for installing kernel packages, because RPM replaces the previous kernel package. This does not affect a running system, but if the new kernel is unable to boot during your next restart, there would be no other kernel to boot instead.

Using the **`-i`** option adds the kernel to your GRUB boot menu (**`/etc/grub.conf`**). Similarly, removing an old, unneeded kernel removes the kernel from GRUB.

Because RPM performs intelligent upgrading of packages with configuration files, you may see a message like the following:

```
saving /etc/foo.conf as /etc/foo.conf.rpmsave
```

This message means that changes you made to the configuration file may not be *forward compatible* with the new configuration file in the package, so RPM saved your original file and installed a new one. You should investigate the differences between the two configuration files and resolve them as soon as possible, to ensure that your system continues to function properly.

If you attempt to upgrade to a package with an *older* version number (that is, if a more updated version of the package is already installed), the output is similar to the following:

```
package foo-2.0-1 (which is newer than foo-1.0-1) is already installed
```

To force RPM to upgrade anyway, use the **`--oldpackage`** option:

```
rpm -Uvh --oldpackage foo-1.0-1.i386.rpm
```

12.2.5. Freshening

Freshening is similar to upgrading, except that only existent packages are upgraded. Type the following command at a shell prompt:

```
rpm -Fvh foo-1.2-1.i386.rpm
```

RPM's freshen option checks the versions of the packages specified on the command line against the versions of packages that have already been installed on your system. When a newer version of an already-installed package is processed by RPM's freshen option, it is upgraded to the newer version. However, RPM's freshen option does not install a package if no previously-installed package of the same name exists. This differs from RPM's upgrade option, as an upgrade *does* install packages whether or not an older version of the package was already installed.

Freshening works for single packages or package groups. If you have just downloaded a large number of different packages, and you only want to upgrade those packages that are already installed on your system, freshening does the job. Thus, you do not have to delete any unwanted packages from the group that you downloaded before using RPM.

In this case, issue the following command:

```
rpm -Fvh *.rpm
```

RPM automatically upgrades only those packages that are already installed.

12.2.6. Querying

The RPM database stores information about all RPM packages installed in your system. It is stored in the directory **`/var/lib/rpm/`**, and is used to query what packages are installed, what versions each package is, and any changes to any files in the package since installation, among others.

To query this database, use the **`-q`** option. The **`rpm -q`** ***`package name`*** command displays the package name, version, and release number of the installed package ***`package name`*** . For example, using **`rpm -q foo`** to query installed package **`foo`** might generate the following output:

```
foo-2.0-1
```

You can also use the following *Package Selection Options* with **`-q`** to further refine or qualify your query:

- **`-a`** — queries all currently installed packages.
- **`-f`** ***`<filename>`*** — queries the RPM database for which package owns **`f`*****`<filename>`*** . When specifying a file, specify the absolute path of the file (for example, **`rpm -qf /bin/ls`**).
- **`-p`** ***`<packagefile>`*** — queries the uninstalled package ***`<packagefile>`*** .

There are a number of ways to specify what information to display about queried packages. The following options are used to select the type of information for which you are searching. These are called *Package Query Options*.

- **`-i`** displays package information including name, description, release, size, build date, install date, vendor, and other miscellaneous information.

- **`-l`** displays the list of files that the package contains.
- **`-s`** displays the state of all the files in the package.
- **`-d`** displays a list of files marked as documentation (man pages, info pages, READMEs, etc.).
- **`-c`** displays a list of files marked as configuration files. These are the files you edit after installation to adapt and customize the package to your system (for example, **`sendmail.cf`**, **`passwd`**, **`inittab`**, etc.).

For options that display lists of files, add **`-v`** to the command to display the lists in a familiar **`ls -l`** format.

12.2.7. Verifying

Verifying a package compares information about files installed from a package with the same information from the original package. Among other things, verifying compares the size, MD5 sum, permissions, type, owner, and group of each file.

The command **`rpm -V`** verifies a package. You can use any of the *Verify Options* listed for querying to specify the packages you wish to verify. A simple use of verifying is **`rpm -V foo`**, which verifies that all the files in the **`foo`** package are as they were when they were originally installed. For example:

- To verify a package containing a particular file:

  ```
  rpm -Vf /usr/bin/foo
  ```

  In this example, **`/usr/bin/foo`** is the absolute path to the file used to query a package.

- To verify ALL installed packages throughout the system:

  ```
  rpm -Va
  ```

- To verify an installed package against an RPM package file:

  ```
  rpm -Vp foo-1.0-1.i386.rpm
  ```

  This command can be useful if you suspect that your RPM databases are corrupt.

If everything verified properly, there is no output. If there are any discrepancies, they are displayed. The format of the output is a string of eight characters (a **`c`** denotes a configuration file) and then the file name. Each of the eight characters denotes the result of a comparison of one attribute of the file to the value of that attribute recorded in the RPM database. A single period (`.`) means the test passed. The following characters denote specific discrepancies:

- **`5`** — MD5 checksum
- **`S`** — file size
- **`L`** — symbolic link
- **`T`** — file modification time
- **`D`** — device
- **`U`** — user
- **`G`** — group

- **M** — mode (includes permissions and file type)
- **?** — unreadable file

If you see any output, use your best judgment to determine if you should remove the package, reinstall it, or fix the problem in another way.

12.3. Checking a Package's Signature

If you wish to verify that a package has not been corrupted or tampered with, examine only the md5sum by typing the following command at a shell prompt (where *<rpm-file>* is the file name of the RPM package):

```
rpm -K --nosignature <rpm-file>
```

The message ***<rpm-file>*****: md5 OK** is displayed. This brief message means that the file was not corrupted by the download. To see a more verbose message, replace **-K** with **-Kvv** in the command.

On the other hand, how trustworthy is the developer who created the package? If the package is *signed* with the developer's GnuPG *key*, you know that the developer really is who they say they are.

An RPM package can be signed using *Gnu Privacy Guard* (or GnuPG), to help you make certain your downloaded package is trustworthy.

GnuPG is a tool for secure communication; it is a complete and free replacement for the encryption technology of PGP, an electronic privacy program. With GnuPG, you can authenticate the validity of documents and encrypt/decrypt data to and from other recipients. GnuPG is capable of decrypting and verifying PGP 5.*x* files as well.

During installation, GnuPG is installed by default. That way you can immediately start using GnuPG to verify any packages that you receive from Red Hat. Before doing so, you must first import Red Hat's public key.

12.3.1. Importing Keys

To verify Red Hat packages, you must import the Red Hat GPG key. To do so, execute the following command at a shell prompt:

```
rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
```

To display a list of all keys installed for RPM verification, execute the command:

```
rpm -qa gpg-pubkey*
```

For the Red Hat key, the output includes:

```
gpg-pubkey-37017186-45761324
```

To display details about a specific key, use **rpm -qi** followed by the output from the previous command:

```
rpm -qi gpg-pubkey-37017186-45761324
```

12.3.2. Verifying Signature of Packages

To check the GnuPG signature of an RPM file after importing the builder's GnuPG key, use the following command (replace *<rpm-file>* with the filename of the RPM package):

```
rpm -K <rpm-file>
```

If all goes well, the following message is displayed: **`md5 gpg OK`**. This means that the signature of the package has been verified, and that it is not corrupt.

12.4. Practical and Common Examples of RPM Usage

RPM is a useful tool for both managing your system and diagnosing and fixing problems. The best way to make sense of all of its options is to look at some examples.

- Perhaps you have deleted some files by accident, but you are not sure what you deleted. To verify your entire system and see what might be missing, you could try the following command:

  ```
  rpm -Va
  ```

  If some files are missing or appear to have been corrupted, you should probably either re-install the package or uninstall and then re-install the package.

- At some point, you might see a file that you do not recognize. To find out which package owns it, enter:

  ```
  rpm -qf /usr/bin/ggv
  ```

  The output would look like the following:

  ```
  ggv-2.6.0-2
  ```

- We can combine the above two examples in the following scenario. Say you are having problems with **`/usr/bin/paste`**. You would like to verify the package that owns that program, but you do not know which package owns **`paste`**. Enter the following command,

  ```
  rpm -Vf /usr/bin/paste
  ```

  and the appropriate package is verified.

- Do you want to find out more information about a particular program? You can try the following command to locate the documentation which came with the package that owns that program:

  ```
  rpm -qdf /usr/bin/free
  ```

  The output would be similar to the following:

  ```
  /usr/share/doc/procps-3.2.3/BUGS
  /usr/share/doc/procps-3.2.3/FAQ
  /usr/share/doc/procps-3.2.3/NEWS
  /usr/share/doc/procps-3.2.3/TODO
  /usr/share/man/man1/free.1.gz
  /usr/share/man/man1/pgrep.1.gz
  /usr/share/man/man1/pkill.1.gz
  /usr/share/man/man1/pmap.1.gz
  /usr/share/man/man1/ps.1.gz
  ```

```
/usr/share/man/man1/skill.1.gz
/usr/share/man/man1/slabtop.1.gz
/usr/share/man/man1/snice.1.gz
/usr/share/man/man1/tload.1.gz
/usr/share/man/man1/top.1.gz
/usr/share/man/man1/uptime.1.gz
/usr/share/man/man1/w.1.gz
/usr/share/man/man1/watch.1.gz
/usr/share/man/man5/sysctl.conf.5.gz
/usr/share/man/man8/sysctl.8.gz
/usr/share/man/man8/vmstat.8.gz
```

- You may find a new RPM, but you do not know what it does. To find information about it, use the following command:

  ```
  rpm -qip crontabs-1.10-7.noarch.rpm
  ```

  The output would be similar to the following:

  ```
  Name        : crontabs                     Relocations: (not
  relocatable)
  Version     : 1.10                              Vendor: Red Hat, Inc.
  Release     : 7                            Build Date: Mon 20 Sep 2004
  05:58:10 PM EDT
  Install Date: (not installed)              Build Host:
  tweety.build.redhat.com
  Group       : System Environment/Base      Source RPM: crontabs-1.10-
  7.src.rpm
  Size        : 1004                            License: Public Domain
  Signature   : DSA/SHA1, Wed 05 Jan 2005 06:05:25 PM EST, Key ID
  219180cddb42a60e
  Packager    : Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla>
  Summary     : Root crontab files used to schedule the execution of
  programs.
  Description : The crontabs package contains root crontab files. Crontab is
  the
  program used to install, uninstall, or list the tables used to drive the
  cron daemon. The cron daemon checks the crontab files to see when
  particular commands are scheduled to be executed. If commands are
  scheduled, then it executes them.
  ```

- Perhaps you now want to see what files the **`crontabs`** RPM installs. You would enter the following:

  ```
  rpm -qlp crontabs-1.10-5.noarch.rpm
  ```

  The output is similar to the following:

  ```
  /etc/cron.daily
  /etc/cron.hourly
  /etc/cron.monthly
  /etc/cron.weekly
  /etc/crontab
  /usr/bin/run-parts
  ```

These are just a few examples. As you use RPM, you may find more uses for it.

12.5. Additional Resources

RPM is an extremely complex utility with many options and methods for querying, installing, upgrading, and removing packages. Refer to the following resources to learn more about RPM.

12.5.1. Installed Documentation

- **`rpm --help`** — This command displays a quick reference of RPM parameters.
- **`man rpm`** — The RPM man page gives more detail about RPM parameters than the **`rpm --help`** command.

12.5.2. Useful Websites

- http://www.rpm.org/ — The RPM website.
- https://lists.rpm.org/mailman/listinfo/rpm-list — Visit this link to subscribe to the RPM mailing list, which is archived there.

12.5.3. Related Books

- The *Red Hat RPM Guide* by Eric Foster-Johnson is an excellent resource on all details of the RPM package format and the RPM package management utility. It is available online at http://docs.fedoraproject.org/drafts/rpm-guide-en/.

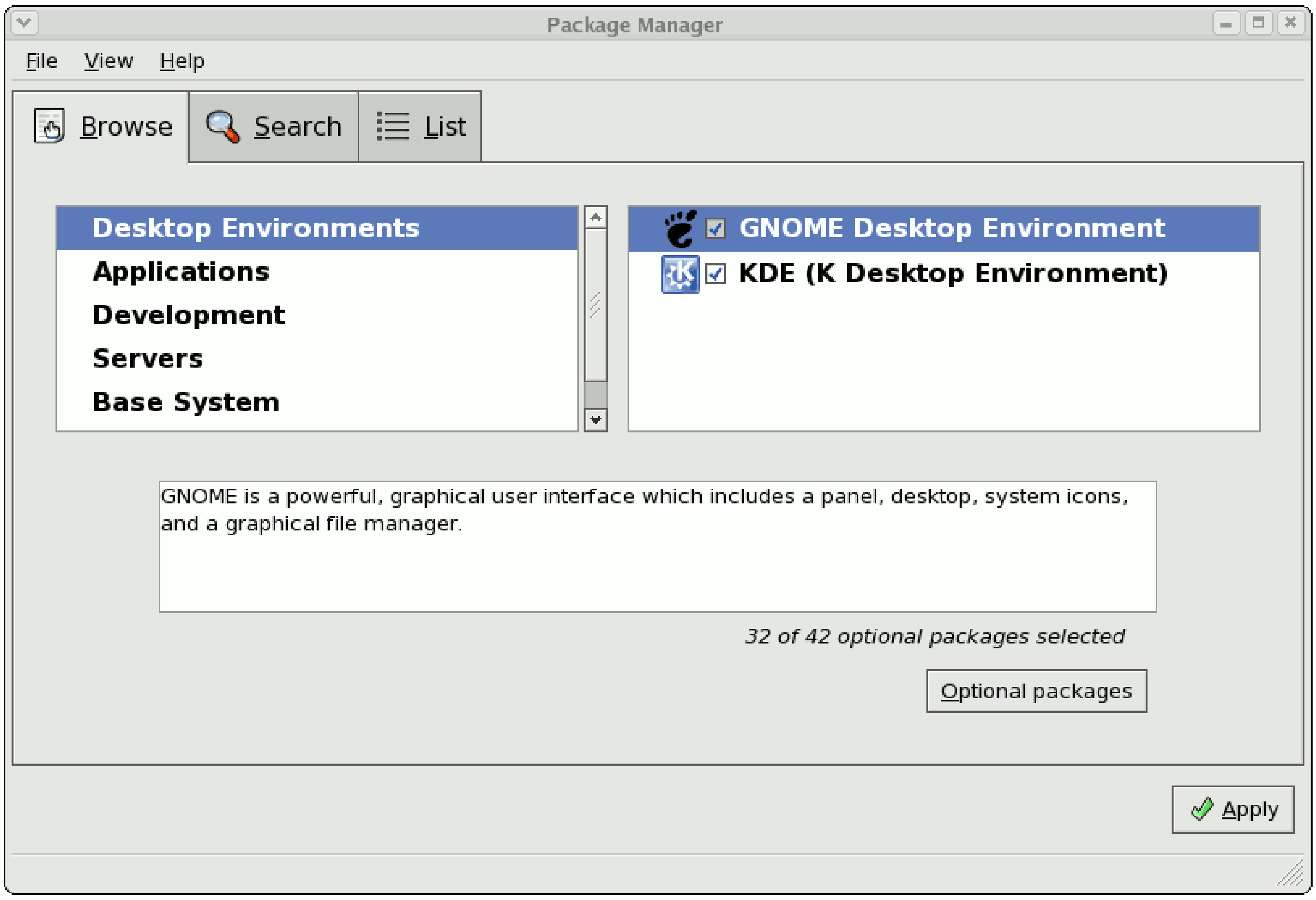
Chapter 13. Package Management Tool

If you prefer to use a graphical interface to view and manage packages in your system, you can use the **Package Management Tool**, better known as *pirut*. This tool allows you to perform basic package management of your system through an easy-to-use interface to remove installed packages or download (and install) packages compatible to your system. It also allows you to view what packages are installed in your system and which ones are available for download from Red Hat Network. In addition, the **Package Management Tool** also automatically resolves any critical dependencies when you install or remove packages in the same way that the `rpm` command does.

Note

While the **Package Management Tool** can automatically resolve dependencies during package installation and removal, it *cannot* perform a forced install / remove the same way that `rpm -e --nodeps` or `rpm -U --nodeps` can.

The X Window System is required to run the **Package Management Tool**. To start the application, go to **Applications** (the main menu on the panel) > **Add/Remove Software**. Alternatively, you can type the commands `system-config-packages` or `pirut` at shell prompt.



GNOME is a powerful, graphical user interface which includes a panel, desktop, system icons, and a graphical file manager.

32 of 42 optional packages selected

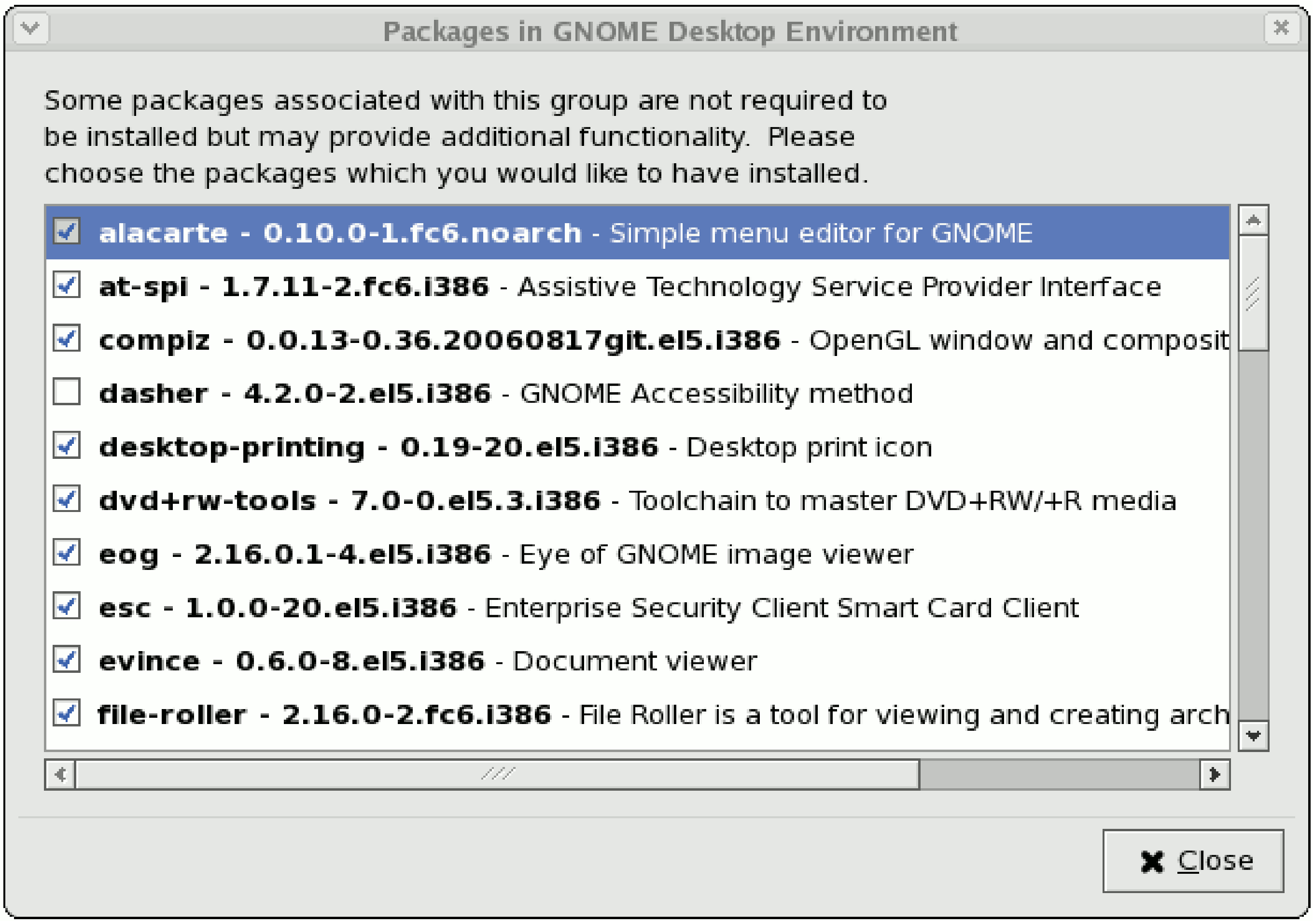
Figure 13.1. Package Management Tool

13.1. Listing and Analyzing Packages

You can use the **Package Management Tool** to search and list all packages installed in your system, as well as any packages available for you to download. The **`Browse`**, **`Search`**, and **`List`** tabs present different options in viewing, analyzing, installing or removing packages.

The **`Browse`** tab allows you to view packages by group. In Figure 13.1, "Package Management Tool", the left window shows the different package group types you can choose from (for example, Desktop Environments, Applications, Development and more). When a package group type is selected, the right window displays the different package groups of that type.

To view what packages are included in a package group, click **`Optional packages`**. Installed packages are checked.



Some packages associated with this group are not required to be installed but may provide additional functionality. Please choose the packages which you would like to have installed.

- ☑ **alacarte - 0.10.0-1.fc6.noarch** - Simple menu editor for GNOME
- ☑ **at-spi - 1.7.11-2.fc6.i386** - Assistive Technology Service Provider Interface
- ☑ **compiz - 0.0.13-0.36.20060817git.el5.i386** - OpenGL window and composit
- ☐ **dasher - 4.2.0-2.el5.i386** - GNOME Accessibility method
- ☑ **desktop-printing - 0.19-20.el5.i386** - Desktop print icon
- ☑ **dvd+rw-tools - 7.0-0.el5.3.i386** - Toolchain to master DVD+RW/+R media
- ☑ **eog - 2.16.0.1-4.el5.i386** - Eye of GNOME image viewer
- ☑ **esc - 1.0.0-20.el5.i386** - Enterprise Security Client Smart Card Client
- ☑ **evince - 0.6.0-8.el5.i386** - Document viewer
- ☑ **file-roller - 2.16.0-2.fc6.i386** - File Roller is a tool for viewing and creating arch

Figure 13.2. Optional Packages

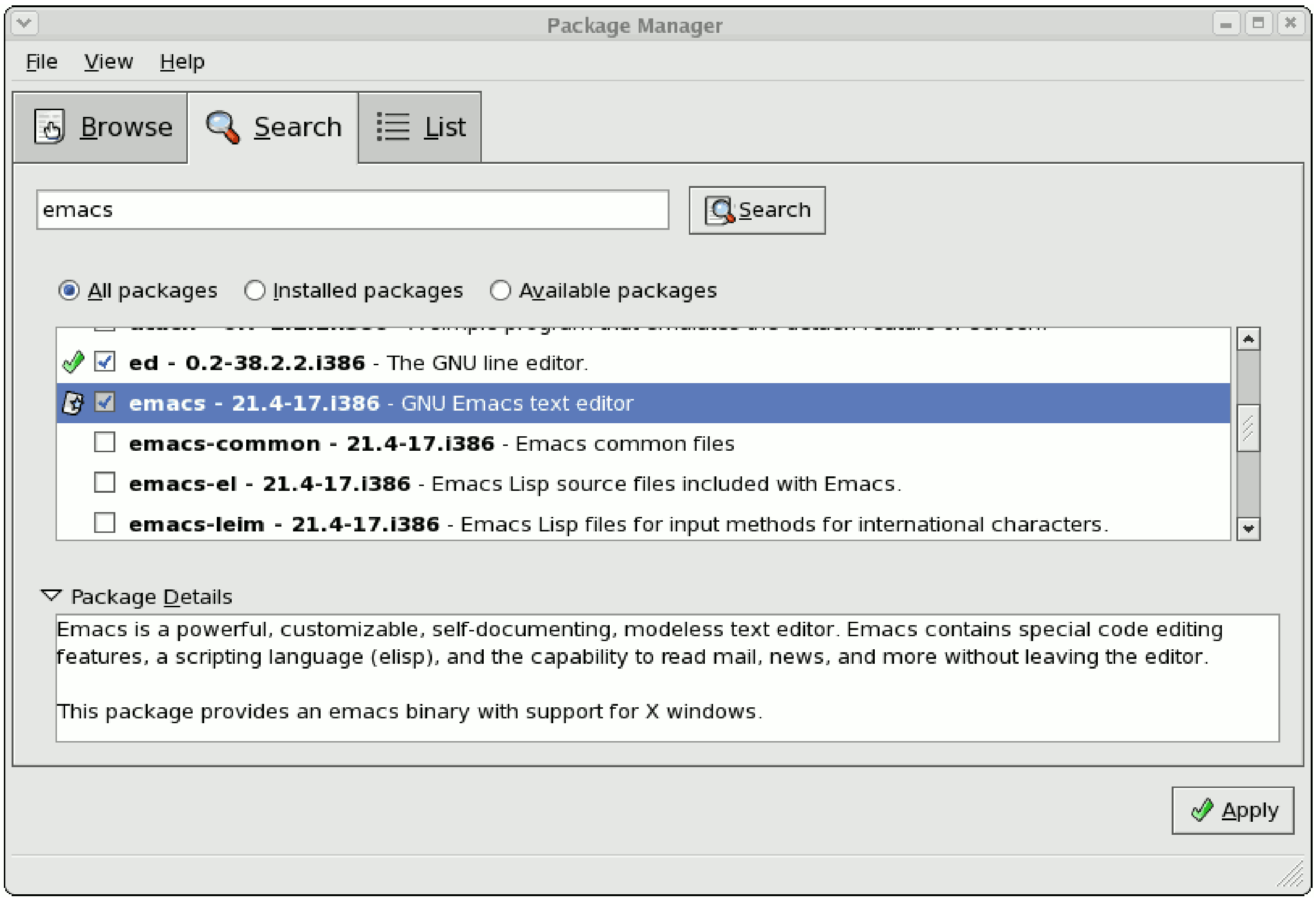
The **`List`** tab displays a list of packages installed or available for download. Packages already installed in your system are marked with a green check (✓).

By default, the **`All packages`** option above the main window is selected; this specifies that all packages be displayed. Use the **`Installed packages`** option to display only packages that are already installed in your system, and the **`Available packages`** option to view what packages you can download and install.

The **`Search`** tab allows you to use keywords to search for particular packages. This tab also allows you to view a short description of a package. To do so, simply select a package and click the **`Package Details`** button below the main window.

13.2. Installing and Removing Packages

To install a package available for download, click the checkbox beside the package name. When you do so, an installation icon ( ) appears beside its checkbox. This indicates that the package is queued for download and installation. You can select multiple packages to download and install; once you have made your selection, click the **`Apply`** button.



emacs

[✓ All packages] Installed packages Available packages

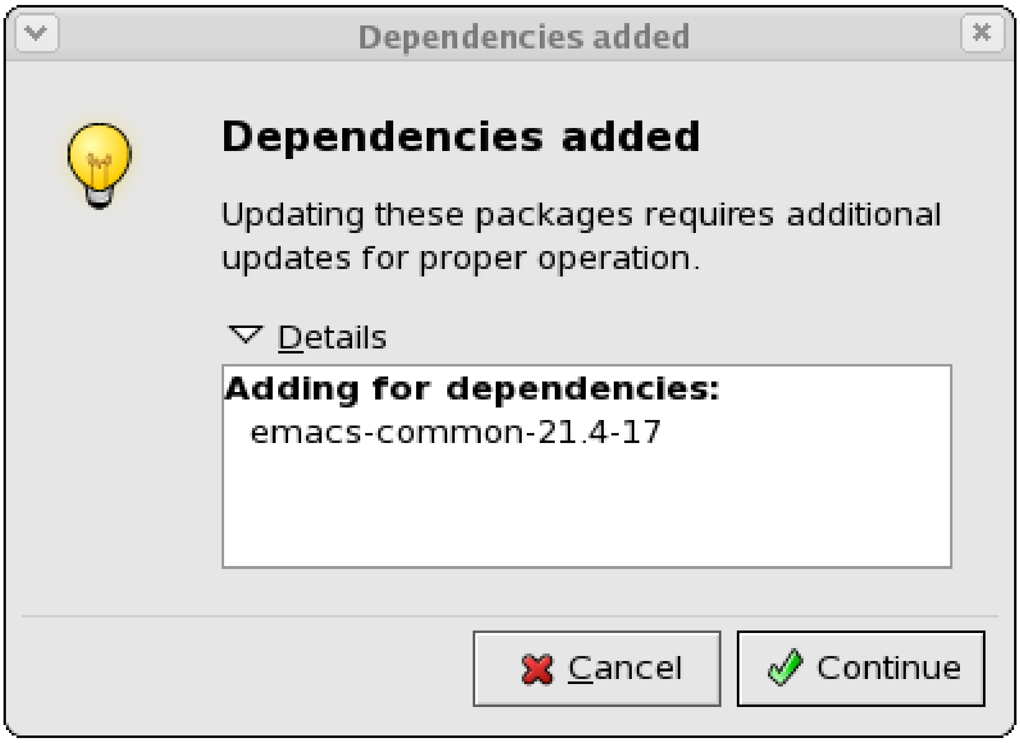
- [✓] **ed - 0.2-38.2.2.i386** - The GNU line editor.
- [✓] **emacs - 21.4-17.i386** - GNU Emacs text editor
- [ ] **emacs-common - 21.4-17.i386** - Emacs common files
- [ ] **emacs-el - 21.4-17.i386** - Emacs Lisp source files included with Emacs.
- [ ] **emacs-leim - 21.4-17.i386** - Emacs Lisp files for input methods for international characters.

Emacs is a powerful, customizable, self-documenting, modeless text editor. Emacs contains special code editing features, a scripting language (elisp), and the capability to read mail, news, and more without leaving the editor.

This package provides an emacs binary with support for X windows.

Figure 13.3. Package installation

If there are any package dependencies for your selected downloads, the **Package Management Tool** will notify you accordingly. Click **`Details`** to view what additional packages are needed. To proceed with downloading and installing the package (along with all other dependent packages) click **`Continue`**.



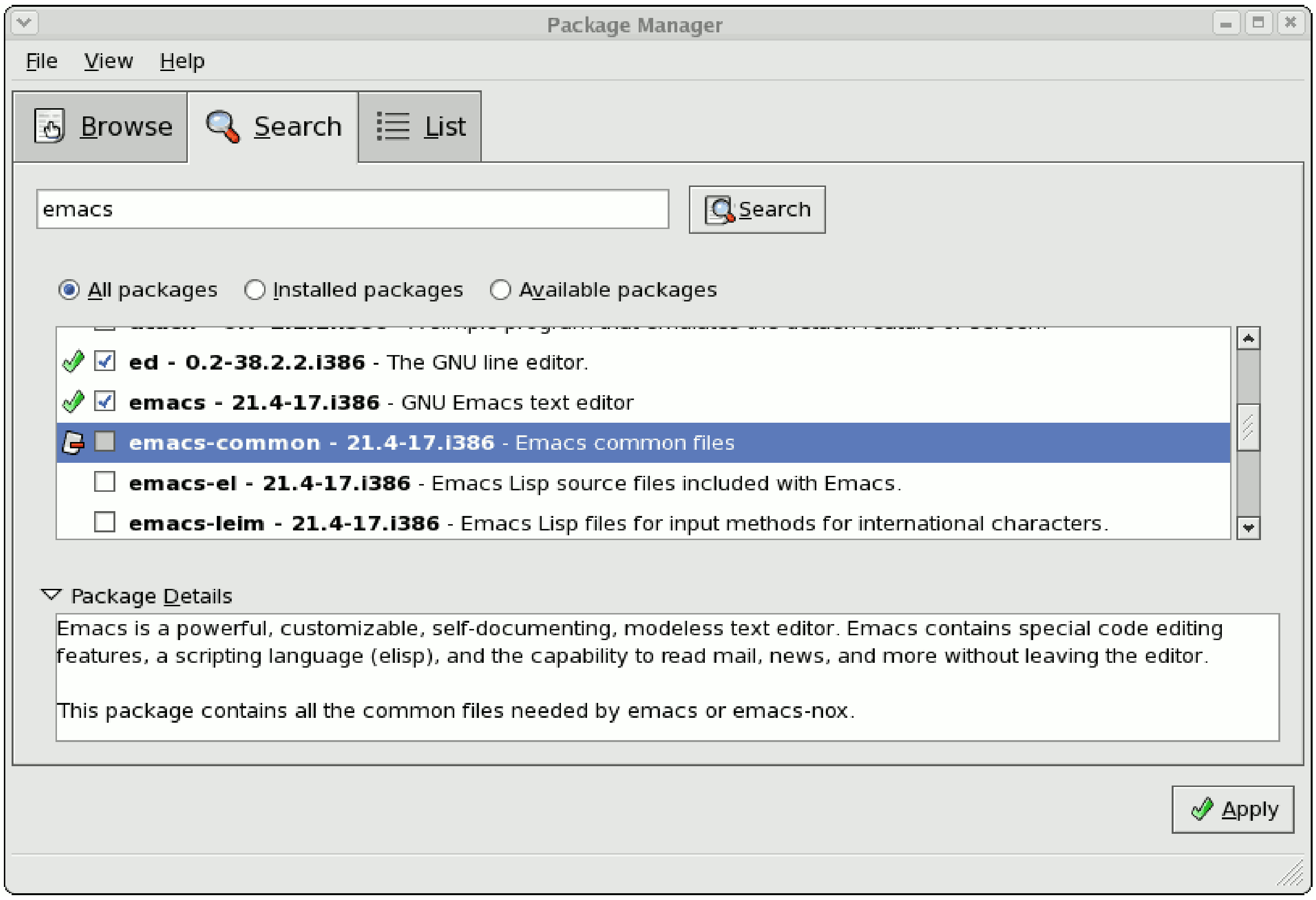
Dependencies added

Updating these packages requires additional updates for proper operation.

Adding for dependencies:
emacs-common-21.4-17

Figure 13.4. Package dependencies: installation

Removing a package can be done in a similar manner. To remove a package installed in your system, click the checkbox beside the package name. The green check appearing beside the package name will be replaced by a package removal icon ( ). This indicates that the package is queued for removal; you can also select multiple packages to be removed at the same time. Once you have selected the packages you want to remove, click the **Apply** button.



emacs

(● All packages) ○ Installed packages ○ Available packages

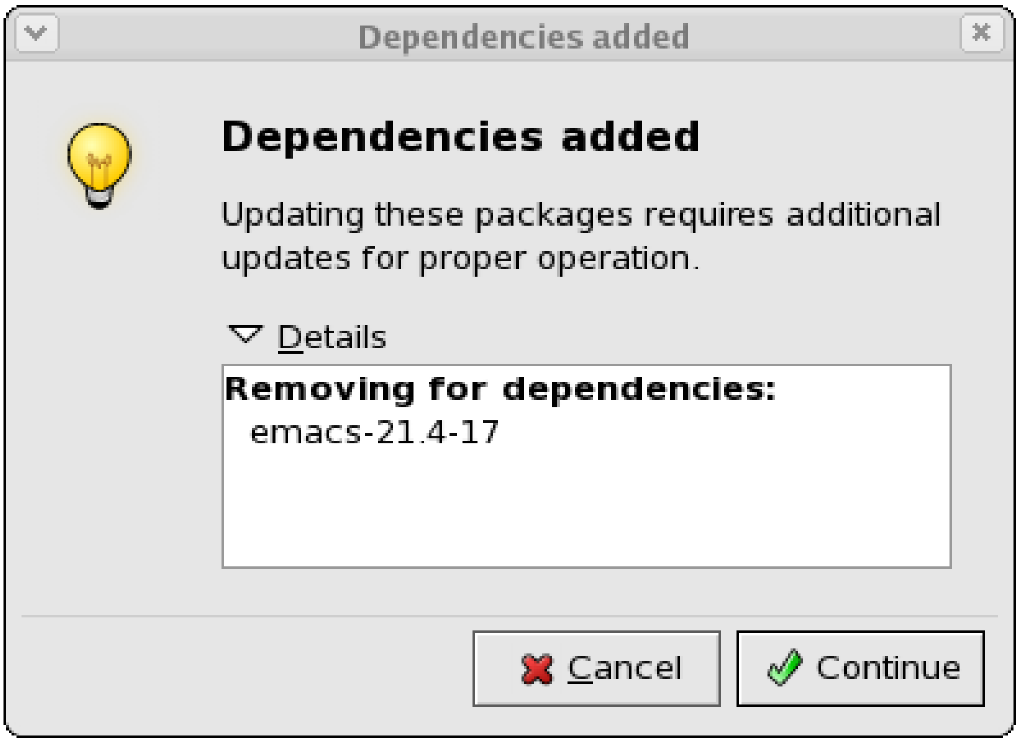
☑ **ed - 0.2-38.2.2.i386** - The GNU line editor.
☑ **emacs - 21.4-17.i386** - GNU Emacs text editor
☐ **emacs-common - 21.4-17.i386** - Emacs common files
☐ **emacs-el - 21.4-17.i386** - Emacs Lisp source files included with Emacs.
☐ **emacs-leim - 21.4-17.i386** - Emacs Lisp files for input methods for international characters.

Emacs is a powerful, customizable, self-documenting, modeless text editor. Emacs contains special code editing features, a scripting language (elisp), and the capability to read mail, news, and more without leaving the editor.

This package contains all the common files needed by emacs or emacs-nox.

Figure 13.5. Package removal

Note that if any other installed packages are dependent on the package you are removing, they will be removed as well. The **Package Management Tool** will notify you if there are any such dependencies. Click **`Details`** to view what packages are dependent on the one you are removing. To proceed with removing your selected package/s (along with all other dependent packages) click **`Continue`**.



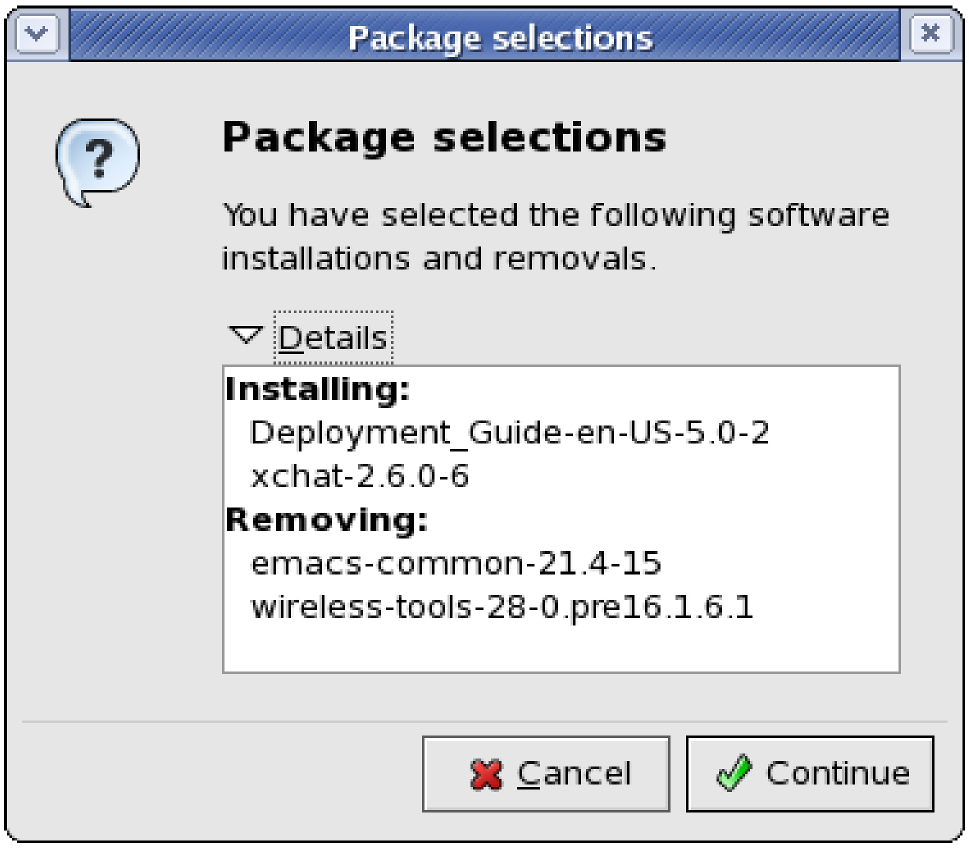
Dependencies added

Updating these packages requires additional updates for proper operation.

Removing for dependencies:
emacs-21.4-17

Figure 13.6. Package dependencies: removal

You can install and remove multiple packages by selecting packages to be installed / removed and then clicking **`Apply`**. The **`Package selections`** window displays the number of packages to be installed and removed.



Package selections

You have selected the following software installations and removals.

Installing:
Deployment_Guide-en-US-5.0-2
xchat-2.6.0-6
Removing:
emacs-common-21.4-15
wireless-tools-28-0.pre16.1.6.1

Figure 13.7. Installing and removing packages simultaneously

Chapter 14. YUM (Yellowdog Updater Modified)

Yellowdog Update, Modified (YUM) is a package manager that was developed by Duke University to improve the installation of RPMs. **yum** searches numerous repositories for packages and their dependencies so they may be installed together in an effort to alleviate dependency issues. Red Hat Enterprise Linux 5.10 uses **yum** to fetch packages and install RPMs.

up2date is now deprecated in favor of **yum** (Yellowdog Updater Modified). The entire stack of tools which installs and updates software in Red Hat Enterprise Linux 5.10 is now based on **yum**. This includes everything, from the initial installation via **Anaconda** to host software management tools like **pirut**.

yum also allows system administrators to configure a local (i.e. available over a local network) repository to supplement packages provided by Red Hat. This is useful for user groups that use applications and packages that are not officially supported by Red Hat.

Aside from being able to supplement available packages for local users, using a local **yum** repository also saves bandwidth for the entire network. Further, clients that use local **yum** repositories do not need to be registered individually to install or update the latest packages from Red Hat Network.

14.1. Setting Up a Yum Repository

To set up a repository for Red Hat Enterprise Linux packages, follow these steps:

1. Install the **createrepo** package:

   ```
   ~]# yum install createrepo
   ```

2. Copy all the packages you want to provide in the repository into one directory (**/mnt/local_repo** for example).

3. Run **createrepo** on that directory (for example, **createrepo /mnt/local_repo**). This will create the necessary metadata for your **Yum** repository.

14.2. yum Commands

yum commands are typically run as **yum *<command> <package name/s>*** . By default, **yum** will automatically attempt to check all configured repositories to resolve all package dependencies during an installation/upgrade.

The following is a list of the most commonly-used **yum** commands. For a complete list of available **yum** commands, refer to **man yum**.

yum install *<package name/s>*

Used to install the latest version of a package or group of packages. If no package matches the specified package name(s), they are assumed to be a shell glob, and any matches are then installed.

yum update *<package name/s>*

Used to update the specified packages to the latest available version. If no package name/s are specified, then **yum** will attempt to update all installed packages.

If the **`--obsoletes`** option is used (i.e. **`yum --obsoletes`** ***`<package name/s>`*** , **`yum`** will process obsolete packages. As such, packages that are obsoleted across updates will be removed and replaced accordingly.

`yum check-update`

This command allows you to determine whether any updates are available for your installed packages. **`yum`** returns a list of all package updates from all repositories if any are available.

`yum remove` ***`<package name/s>`***

Used to remove specified packages, along with any other packages dependent on the packages being removed.

`yum provides` ***`<file name>`***

Used to determine which packages provide a specific file or feature.

`yum search` ***`<keyword>`***

This command is used to find any packages containing the specified keyword in the description, summary, packager and package name fields of RPMs in all repositories.

`yum localinstall` ***`<absolute path to package name/s>`***

Used when using **`yum`** to install a package located locally in the machine.

14.3. `yum` Options

`yum` options are typically stated before specific **`yum`** commands; i.e. **`yum`** ***`<options> <command> <package name/s>`*** . Most of these options can be set as default using the configuration file.

The following is a list of the most commonly-used **`yum`** options. For a complete list of available **`yum`** options, refer to **`man yum`**.

`-y`

Answer "yes" to every question in the transaction.

`-t`

Sets **`yum`** to be "tolerant" of errors with regard to packages specified in the transaction. For example, if you run **`yum update package1 package2`** and **`package2`** is already installed, **`yum`** will continue to install **`package1`**.

`--exclude=`***`<package name>`***

Excludes a specific package by name or glob in a specific transaction.

14.4. Configuring `yum`

By default, **`yum`** is configured through **`/etc/yum.conf`**. The following is an example of a typical **`/etc/yum.conf`** file:

```
[main]
cachedir=/var/cache/yum
```

```
keepcache=0
debuglevel=2
logfile=/var/log/yum.log
distroverpkg=redhat-release
tolerant=1
exactarch=1
obsoletes=1
gpgcheck=1
plugins=1
metadata_expire=1800
[myrepo]
name=RHEL 5 $releasever - $basearch
baseurl=http://local/path/to/yum/repository/
enabled=1
```

A typical **`/etc/yum.conf`** file is made up of two types of sections: a **`[main]`** section, and a repository section. There can only be one **`[main]`** section, but you can specify multiple repositories in a single **`/etc/yum.conf`**.

14.4.1. `[main]` Options

The **`[main]`** section is mandatory, and there must only be one. For a complete list of options you can use in the **`[main]`** section, refer to **`man yum.conf`**.

The following is a list of the most commonly-used options in the **`[main]`** section.

`cachedir`

This option specifies the directory where **`yum`** should store its cache and database files. By default, the cache directory of **`yum`** is **`/var/cache/yum`**.

`keepcache=<`*`1 or 0`*`>`

Setting **`keepcache=1`** instructs **`yum`** to keep the cache of headers and packages after a successful installation. **`keepcache=1`** is the default.

`reposdir=<`*`absolute path to directory of .repo files`*`>`

This option allows you to specify a directory where **`.repo`** files are located. **`.repo`** files contain repository information (similar to the **`[`*`repository`*`]`** section of **`/etc/yum.conf`**).

`yum` collects all repository information from **`.repo`** files and the **`[`*`repository`*`]`** section of the **`/etc/yum.conf`** file to create a master list of repositories to use for each transaction. Refer to Section 14.4.2, “ **`[`*`repository`*`]`** Options” for more information about options you can use for both the **`[`*`repository`*`]`** section and **`.repo`** files.

If **`reposdir`** is not set, **`yum`** uses the default directory **`/etc/yum.repos.d`**.

`gpgcheck=<`*`1 or 0`*`>`

This disables/enables GPG signature checking on packages on all repositories, including local package installation. The default is **`gpgcheck=0`**, which disables GPG checking.

If this option is set in the **`[main]`** section of the **`/etc/yum.conf`** file, it sets the GPG checking rule for all repositories. However, you can also set this on individual repositories instead; i.e., you can enable GPG checking on one repository while disabling it on another.

`assumeyes=<1 or 0>`

This determines whether or not **`yum`** should prompt for confirmation of critical actions. The default if **`assumeyes=0`**, which means **`yum`** will prompt you for confirmation.

If **`assumeyes=1`** is set, **`yum`** behaves in the same way that the command line option **`-y`** does.

`tolerant=<1 or 0>`

When enabled (**`tolerant=1`**), **`yum`** will be tolerant of errors on the command line with regard to packages. This is similar to the **`yum`** command line option **`-t`**.

The default value for this is **`tolerant=0`** (not tolerant).

`exclude=<package name/s>`

This option allows you to exclude packages by keyword during installation/updates. If you are specifying multiple packages, this is a space-delimited list. Shell globs using wildcards (for example, * and ?) are allowed.

`retries=<number of retries>`

This sets the number of times **`yum`** should attempt to retrieve a file before returning an error. Setting this to 0 makes **`yum`** retry forever. The default value is 6.

14.4.2. `[repository]` Options

The **`[repository]`** section of the **`/etc/yum.conf`** file contains information about a repository **`yum`** can use to find packages during package installation, updating and dependency resolution. A repository entry takes the following form:

```
[repository ID]
name=repository name
baseurl=url, file or ftp://path to repository
```

You can also specify repository information in a separate **`.repo`** files (for example, **`rhel5.repo`**). The format of repository information placed in **`.repo`** files is identical with the **`[repository]`** of **`/etc/yum.conf`**.

`.repo` files are typically placed in **`/etc/yum.repos.d`**, unless you specify a different repository path in the **`[main]`** section of **`/etc/yum.conf`** with **`reposdir=`**. **`.repo`** files and the **`/etc/yum.conf`** file can contain multiple repository entries.

Each repository entry consists of the following mandatory parts:

[*repository ID*]

The repository ID is a unique, one-word string that serves as a repository identifier.

`name=repository name`

This is a human-readable string describing the repository.

`baseurl=http, file or ftp://path`

This is a URL to the directory where the **repodata**directory of a repository is located. If the repository is local to the machine, use **baseurl=file://*path to local repository*** . If the repository is located online using HTTP, use **baseurl=http://*link*** . If the repository is online and uses FTP, use **baseurl=ftp://*link*** .

If a specific online repository requires basic HTTP authentication, you can specify your username and password in the **baseurl** line by prepending it as *username*:*password*@*link*. For example, if a repository on http://www.example.com/repo/ requires a username of "user" and a password os "password", then the **baseurl** link can be specified as **baseurl=http://user:password@www.example.com/repo/**.

The following is a list of options most commonly used in repository entries. For a complete list of repository entries, refer to **man yum.conf**.

gpgcheck=<*1 or 0*>

This disables/enables GPG signature checking a specific repository. The default is **gpgcheck=0**, which disables GPG checking.

gpgkey=*URL*

This option allows you to point to a URL of the ASCII-armoured GPG key file for a repository. This option is normally used if **yum** needs a public key to verify a package and the required key was not imported into the RPM database.

If this option is set, **yum** will automatically import the key from the specified URL. You will be prompted before the key is installed unless you set **assumeyes=1** (in the **[main]** section of **/etc/yum.conf**) or **-y** (in a **yum** transaction).

exclude=<*package name/s*>

This option is similar to the **exclude** option in the **[main]** section of **/etc/yum.conf**. However, it only applies to the repository in which it is specified.

includepkgs=<*package name/s*>

This option is the opposite of **exclude**. When this option is set on a repository, **yum** will only be able to see the specified packages in that repository. By default, all packages in a repository are visible to **yum**.

14.5. Upgrading the System Off-line with ISO and Yum

For systems that are disconnected from the Internet or Red Hat Network, using the **yum update** command with the Red Hat Enterprise Linux installation ISO image is an easy and quick way to upgrade systems to the latest minor version. The following steps illustrate the upgrading process:

1. Create a target directory to mount your ISO image. This directory is not automatically created when mounting, so create it before proceeding to the next step, as **root**, type:

   ```
   mkdir mount_dir
   ```

   Replace *mount_dir* with a path to the mount directory. Typicaly, users create it as a subdirectory in the **/media/** directory.

2. Mount the Red Hat Enterprise Linux 5 installation ISO image to the previously created target directory. As **root**, type:

```
mount -o loop iso_name mount_dir
```

Replace *iso_name* with a path to your ISO image and *mount_dir* with a path to the target directory. Here, the **`-o loop`** option is required to mount the file as a block device.

3. Check the numeric value found on the first line of the **`.discinfo`** file from the mount directory:

   ```
   head -n1 mount_dir/.discinfo
   ```

   The output of this command is an identification number of the ISO image, you need to know it to perform the following step.

4. Create a new file in the **`/etc/yum.repos.d/`** directory, named for instance *new.repo*, and add a content in the following form. Note that configuration files in this directory must have the *.repo* extension to function properly.

   ```
   [repository]
   mediaid=media_id
   name=repository_name
   baseurl=repository_url
   gpgkey=gpg_key
   enabled=1
   gpgcheck=1
   ```

   Replace *media_id* with the numeric value found in ***`mount_dir/`*`.discinfo`**. Set the repository name instead of *repository_name*, replace *repository_url* with a path to a repository directory in the mount point and *gpg_key* with a path to the GPG key.

   For example, the repository settings for Red Hat Enterprise Linux 5 Server ISO can look as follows:

   ```
   [rhel5-Server]
   mediaid=1354216429.587870
   name=RHEL5-Server
   baseurl=file:///media/rhel5/Server
   gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
   enabled=1
   gpgcheck=1
   ```

5. Update all yum repositories including **`/etc/yum.repos.d/`*`new.repo`*** created in previous steps. As **`root`**, type:

   ```
   yum update
   ```

   This upgrades your system to the version provided by the mounted ISO image.

6. After successful upgrade, you can unmount the ISO image, with the **`root`** privileges:

   ```
   umount mount_dir
   ```

   where *mount_dir* is a path to your mount directory. Also, you can remove the mount directory created in the first step. As **`root`**, type:

   ```
   rmdir mount_dir
   ```

7. If you will not use the previously created configuration file for another installation or update, you can remove it. As **root**, type:

   ```
   rm /etc/yum.repos.d/new.repo
   ```

Example 14.1. Upgrading from Red Hat Enterprise Linux 5.8 to 5.9

Imagine you need to upgrade your system without access to the Internet connection. To do so, you want to use an ISO image with the newer version of the system, called for instance **RHEL5.9-Server-20121129.0-x86_64-DVD1.iso**. You have crated a target directory **/media/rhel5/**. As **root**, change into the directory with your ISO image and type:

```
~]# mount -o loop RHEL5.9-Server-20121129.0-x86_64-DVD1.iso /media/rhel5/
```

To find the identification number of the mounted image, run:

```
~]# head -n1 /media/rhel5/.discinfo
1354216429.587870
```

You need this number to configure your mount point as a yum repository. Create the**/etc/yum.repos.d/rhel5.repo** file and insert the following text into it:

```
[rhel5-Server]
mediaid=1354216429.587870
name=RHEL5-Server
baseurl=file:///media/rhel5/Server
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
enabled=1
gpgcheck=1
```

Update the yum repository, which effectively upgrades your system to a version provided by **RHEL5.9-Server-20121129.0-x86_64-DVD1.iso**. As **root**, execute:

```
~]# yum update
```

When your system is successfully upgraded, unmount the image, remove the target directory and the configuration file:

```
~]# umount /media/rhel5/
```

```
~]# rmdir /media/rhel5/
```

```
~]# rm /etc/yum.repos.d/rhel5.repo
```

14.6. Useful `yum` Variables

The following is a list of variables you can use for both **yum** commands and **yum** configuration files (i.e. **/etc/yum.conf** and **.repo** files).

`$releasever`

This is replaced with the package's version, as listed in **`distroverpkg`**. This defaults to the version of the **`redhat-release`** package.

`$arch`

This is replaced with your system's architecture, as listed by **`os.uname()`** in Python.

`$basearch`

This is replaced with your base architecture. For example, if **`$arch`**=i686 then **`$basearch`**=i386.

`$YUM`*`0-9`*

This is replaced with the value of the shell environment variable of the same name. If the shell environment variable does not exist, then the configuration file variable will not be replaced.

Chapter 15. Registering a System and Managing Subscriptions

Effective asset management requires a mechanism to handle the software inventory — both the type of products and the number of systems that the software is installed on. The subscription service provides that mechanism and gives transparency into both global allocations of subscriptions for an entire organization and the specific subscriptions assigned to a single system.

Red Hat Subscription Manager works with **`yum`** to unite content delivery with subscription management. The Subscription Manager handles only the subscription-system associations. **`yum`** or other package management tools handle the actual content delivery. Chapter 14, *YUM (Yellowdog Updater Modified)* describes how to use **`yum`**.

15.1. Using Red Hat Subscription Manager Tools

Both registration and subscriptions are managed on the local system through GUI and CLI tools called *Red Hat Subscription Manager*.

Note

The Red Hat Subscription Manager tools are always run as **`root`** because of the nature of the changes to the system. However, Red Hat Subscription Manager connects to the subscription service as a user account for the subscription service.

15.1.1. Launching the Red Hat Subscription Manager GUI

Red Hat Subscription Manager is listed as one of the administrative tools in the **System > Administration** menu in the top management bar.

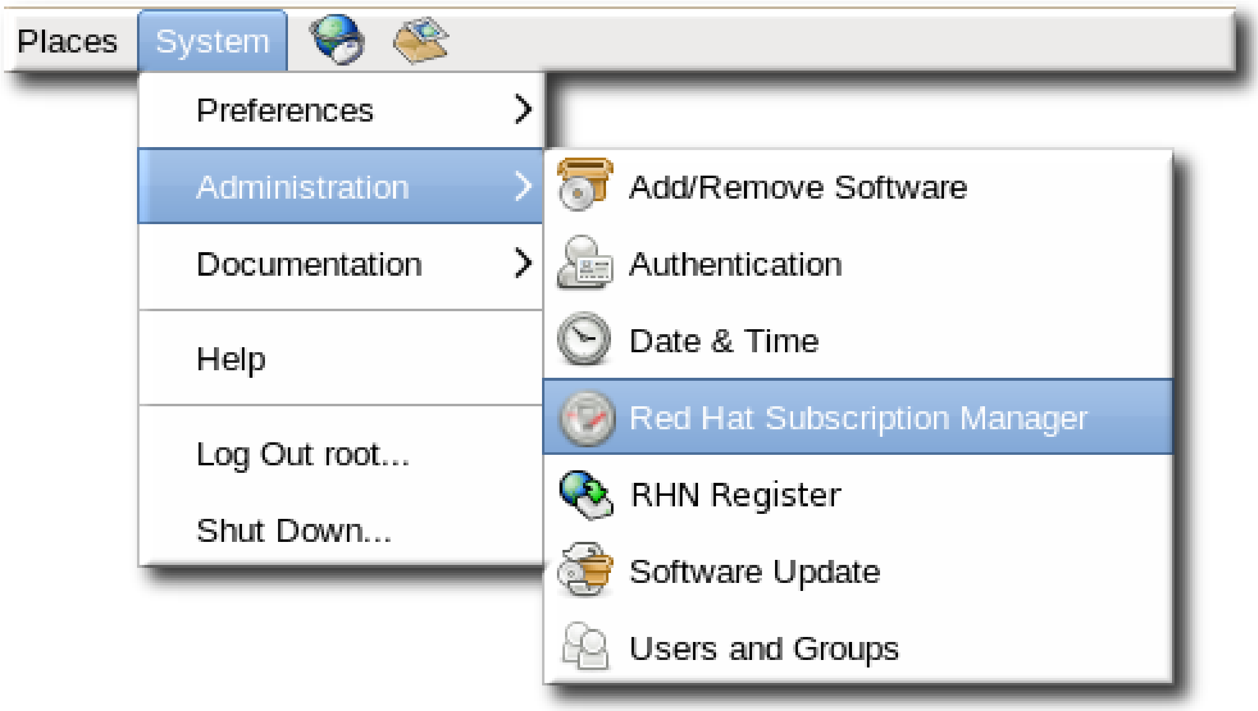


Figure 15.1. Red Hat Subscription Manager Menu Option

Alternatively, the Red Hat Subscription Manager GUI can be opened from the command line with a single command:

```
[root@server1 ~]# subscription-manager-gui
```

15.1.2. Running the subscription-manager Command-Line Tool

Any of the operations that can be performed through the Red Hat Subscription Manager UI can also be performed by running the **subscription-manager** tool. This tool has the following format:

```
[root@server1 ~]# subscription-manager command [options]
```

Each command has its own set of *options* that are used with it. The **subscription-manager** help and manpage have more information.

Table 15.1. Common subscription-manager Commands

Command	Description
register	Registers or identifies a new system to the subscription service.
unregister	Unregisters a machine, which strips its subscriptions and removes the machine from the subscription service.
subscribe	Attaches a specific subscription to the machine.
redeem	Auto-attaches a machine to a pre-specified subscription that was purchased from a vendor, based on its hardware and BIOS information.
unsubscribe	Removes a specific subscription or all subscriptions from the machine.
list	Lists all of the subscriptions that are compatible with a machine, either subscriptions that are actually attached to the machine or unused subscriptions that are available to the machine.

15.2. Registering and Unregistering a System

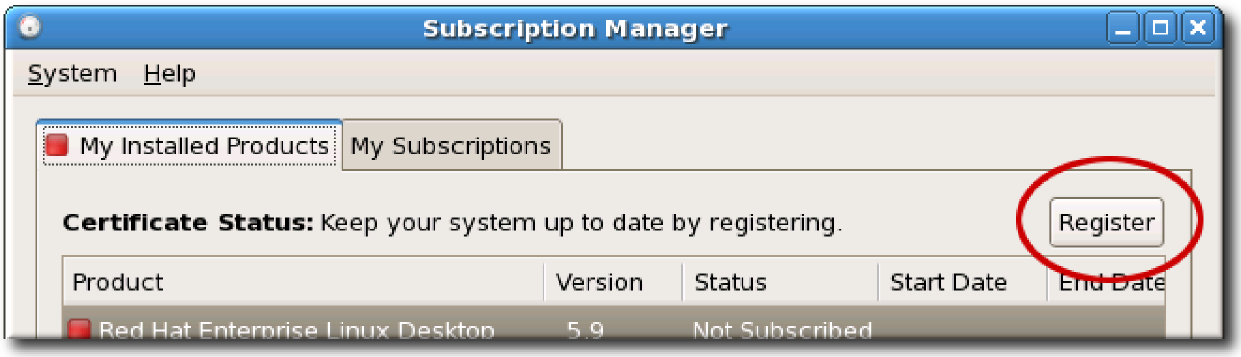
Systems can be registered with a subscription service during the firstboot process or as part of the kickstart setup (both described in the *Installation Guide*). Systems can also be registered after they have been configured or removed from the subscription service inventory (unregistered) if they will no longer be managed within that subscription service.

15.2.1. Registering from the GUI

1. Launch Subscription Manager. For example:

   ```
   [root@server ~]# subscription-manager-gui
   ```

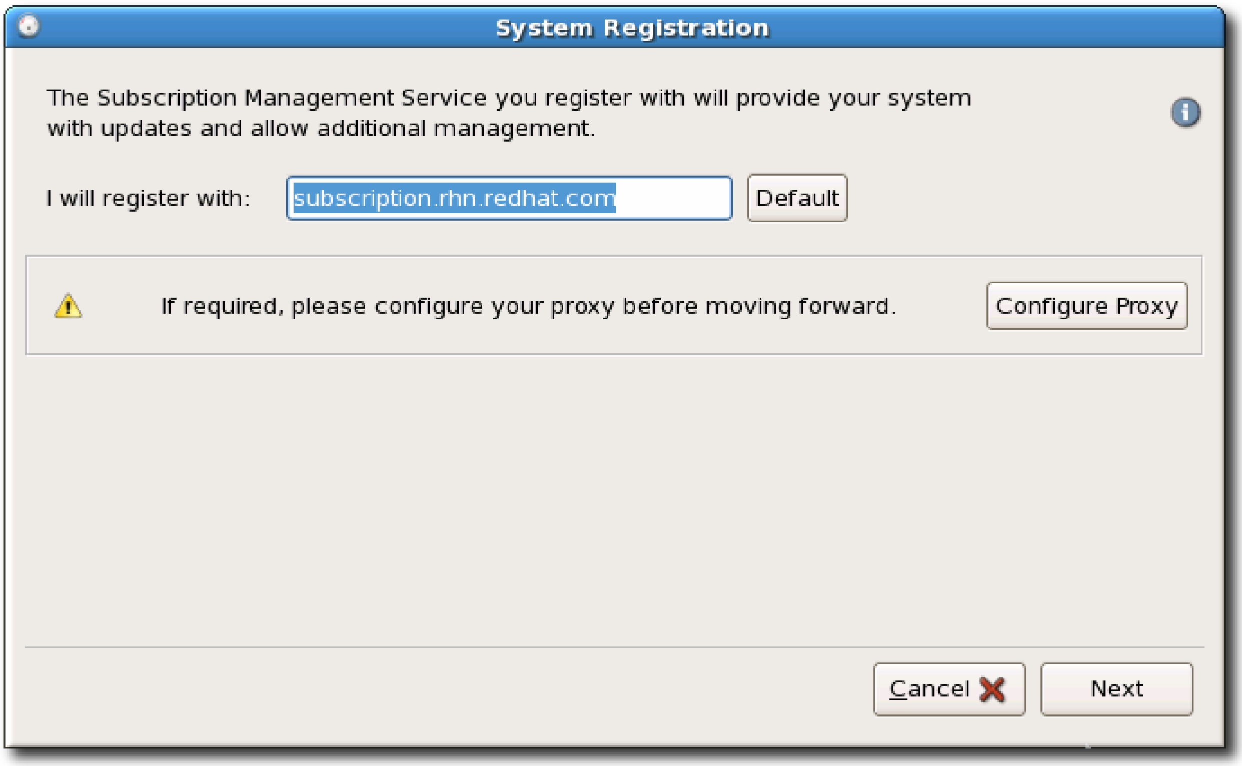
2. If the system is not already registered, then there will be a **Register** button at the top of the window in the top right corner of the **My Installed Products** tab.



Certificate Status: Keep your system up to date by registering.

Product	Version	Status	Start Date	End Date
Red Hat Enterprise Linux Desktop	5.9	Not Subscribed		

3. To identify which subscription server to use for registration, enter the hostname of the service. The default service is Customer Portal Subscription Management, with the hostname **`subscription.rhn.redhat.com`**. To use a different subscription service, such as Subscription Asset Manager, enter the hostname of the local server.



The Subscription Management Service you register with will provide your system with updates and allow additional management.

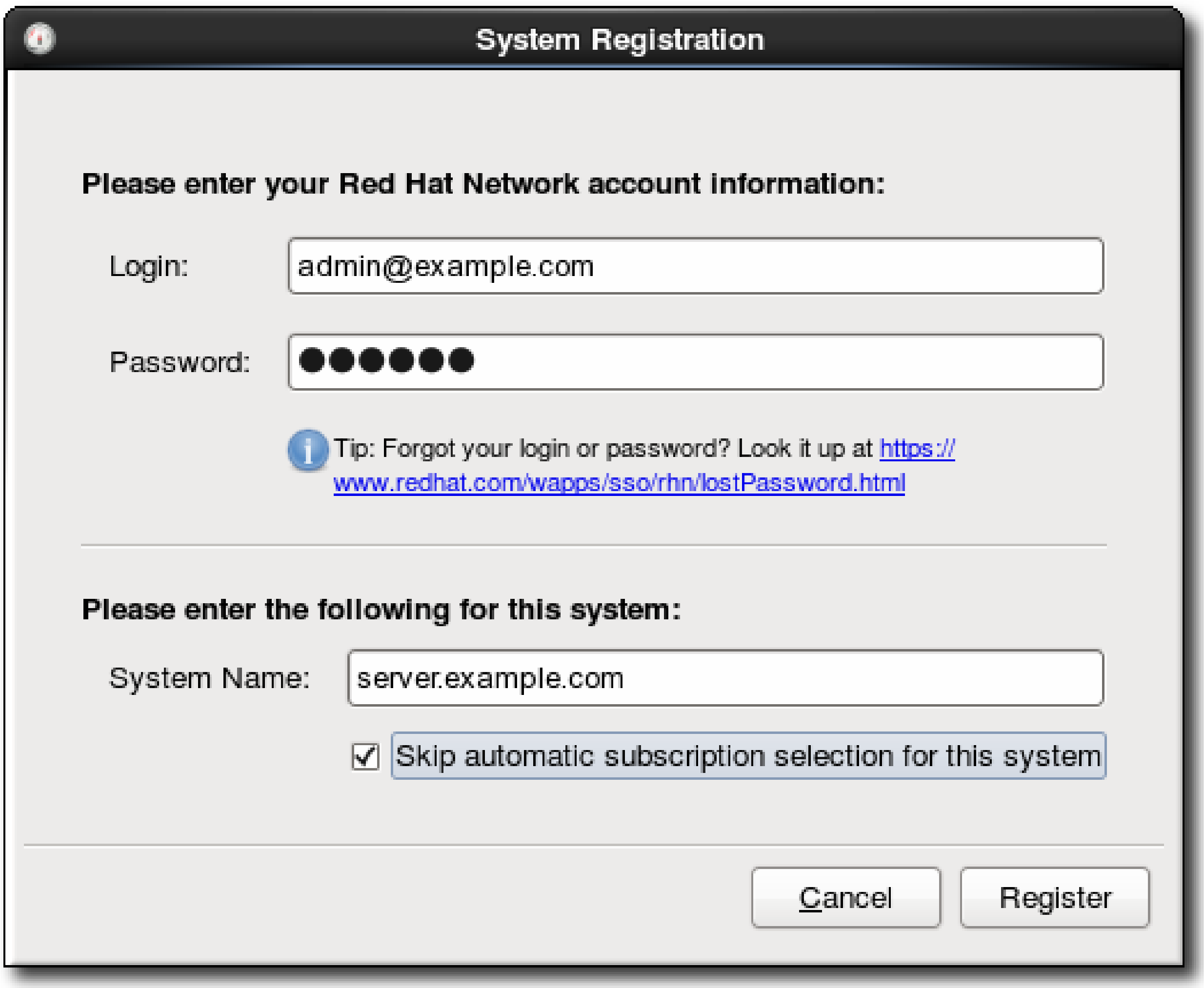
I will register with: subscription.rhn.redhat.com

If required, please configure your proxy before moving forward.

There are seveal different subscription services which use and recognize certificate-based subscriptions, and a system can be registered with any of them in firstboot:

- Customer Portal Subscription Management, hosted services from Red Hat (the default)
- Subscription Asset Manager, an on-premise subscription server which proxies content delivery back to the Customer Portal's services
- CloudForms System Engine, an on-premise service which handles both subscription services and content delivery

4. Enter the user credentials **for the given subscription service** to log in.


System Registration

Please enter your Red Hat Network account information:

Login: admin@example.com

Password: ●●●●●●

Tip: Forgot your login or password? Look it up at https://www.redhat.com/wapps/sso/rhn/lostPassword.html

Please enter the following for this system:

System Name: server.example.com

[✓] Skip automatic subscription selection for this system

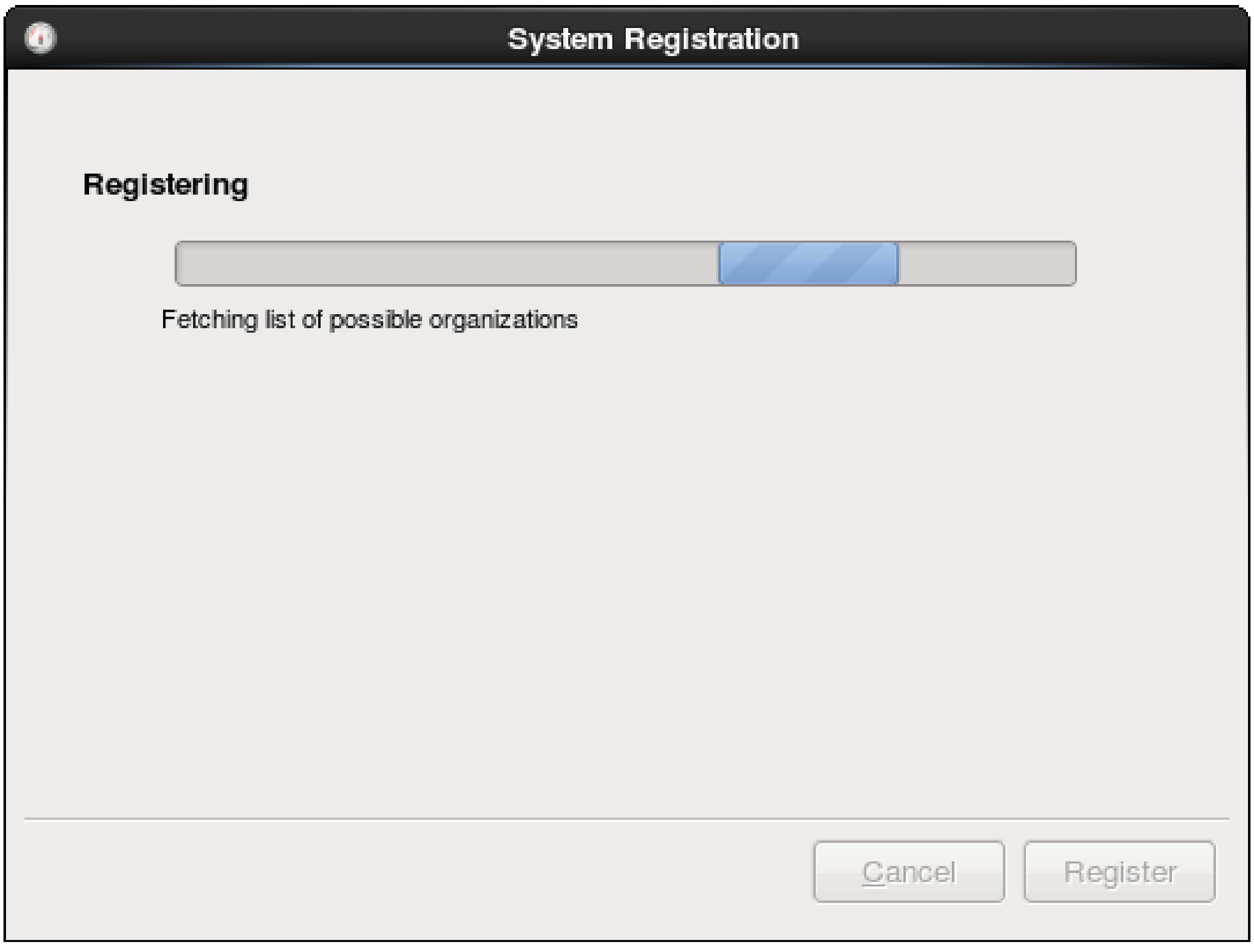
The user credentials to use depend on the subscription service. When registering with the Customer Portal, use the Red Hat Network credentials for the administrator or company account.

However, for Subscription Asset Manager or CloudForms System engine, the user account to use is created within the on-premise service and probably is not the same as the Customer Portal user account.

5. Optionally, select the **`Manually assign subscriptions after registration`** checkbox.

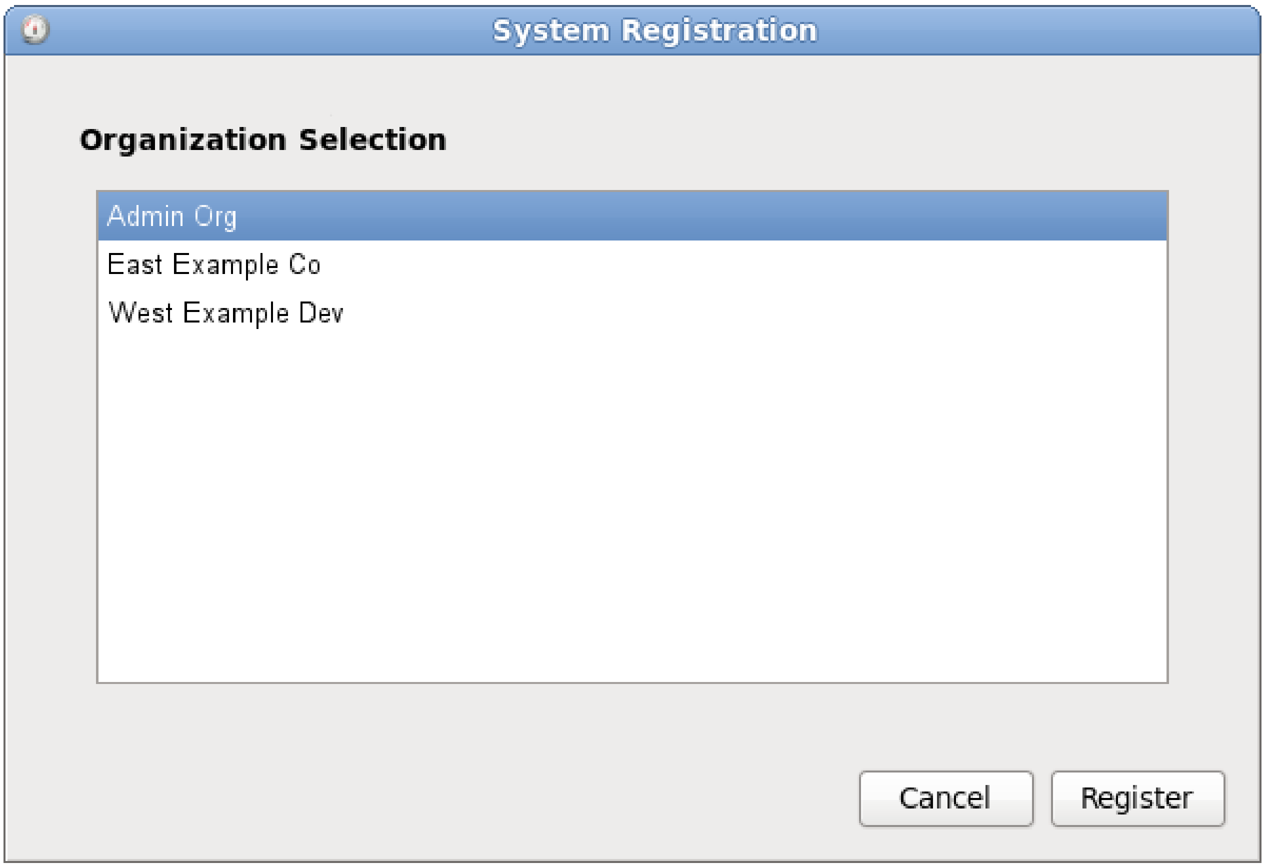
   By default, the registration process automatically attaches the best-matched subscription to the system. This can be turned off so that the subscriptions can be selected manually, as in Section 15.3, “Attaching and Removing Subscriptions”.

6. When registration begins, Subscription Manager scans for organizations and environments (sub-domains within the organization) to which to register the system.



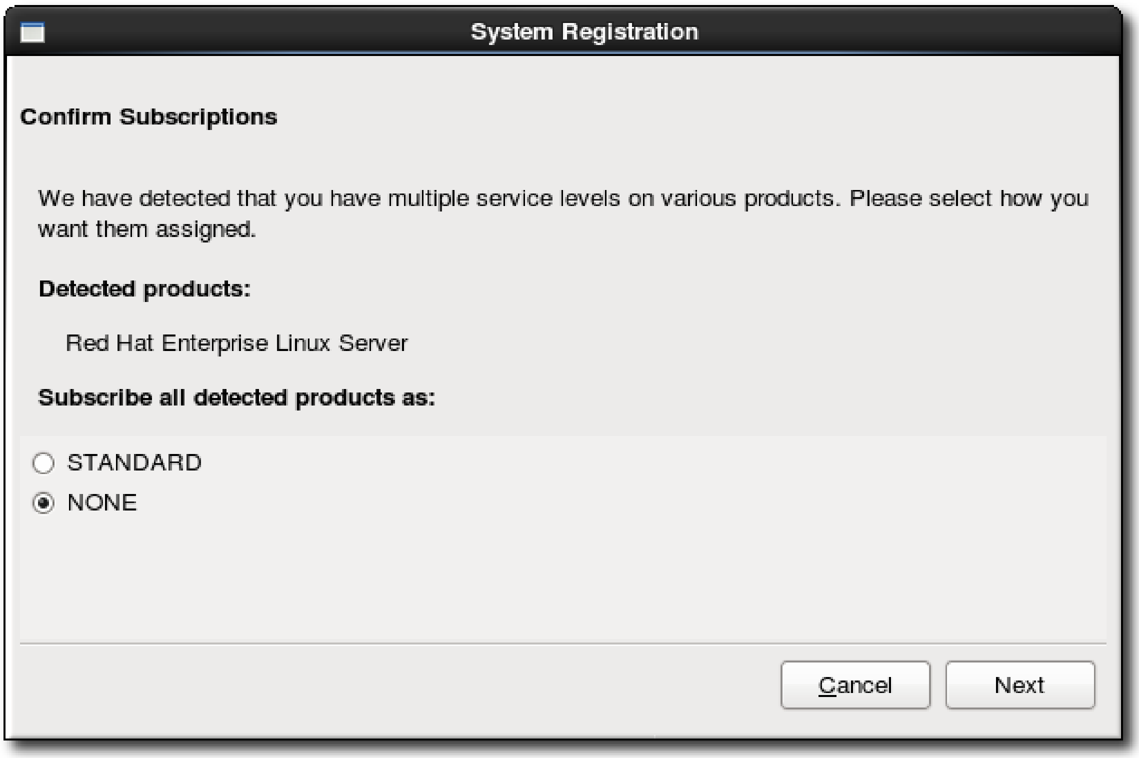
IT environments that use Customer Portal Subscription Management have only a single organization, so no further configuration is necessary. IT infrastructures that use a local subscription service like Subscription Asset Manager might have multiple organizations configured, and those organizations may have multiple environments configured within them.

If multiple organizations are detected, Subscription Manager prompts to select the one to join.



7. With the default setting, subscriptions are automatically selected and attached to the system. Review and confirm the subscriptions to attach to the system.

   a. If prompted, select the service level to use for the discovered subscriptions.



We have detected that you have multiple service levels on various products. Please select how you want them assigned.

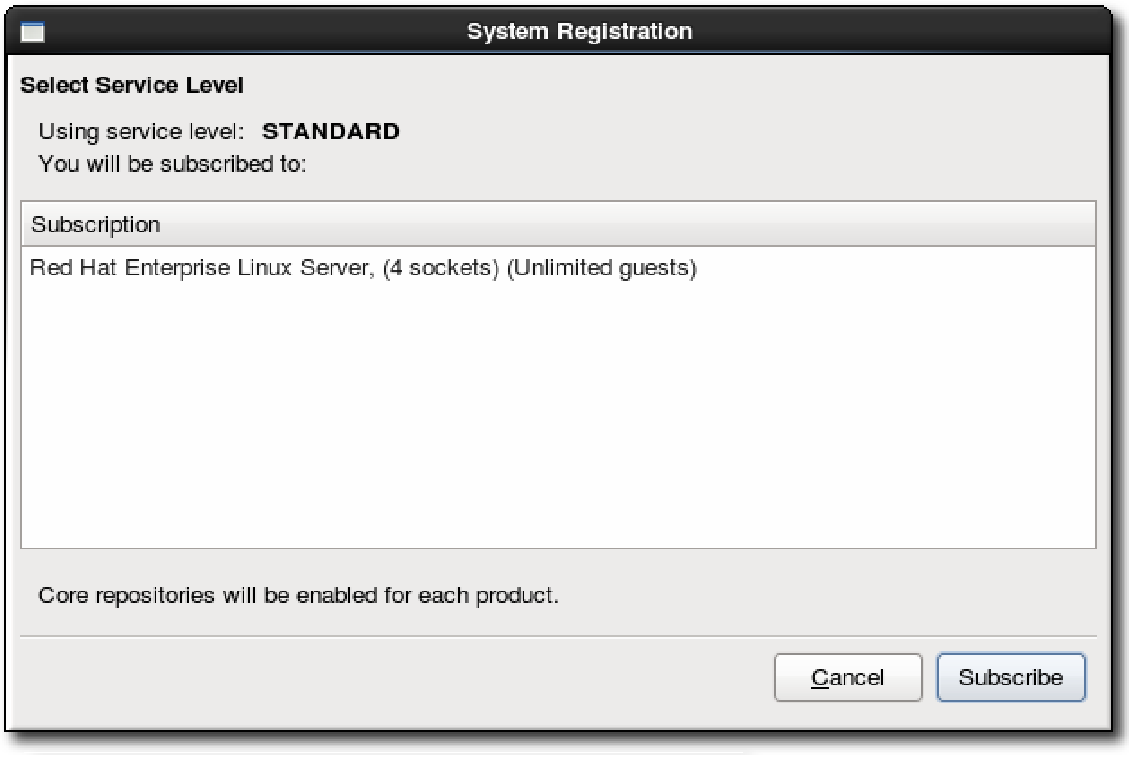
Detected products:

Red Hat Enterprise Linux Server

Subscribe all detected products as:

STANDARD [✓ NONE]

   b. Subscription Manager lists the selected subscription. This subscription selection must be confirmed by clicking the **`Subscribe`** button for the wizard to complete.


System Registration

Select Service Level

Using service level: **STANDARD**
You will be subscribed to:

Subscription
Red Hat Enterprise Linux Server, (4 sockets) (Unlimited guests)

Core repositories will be enabled for each product.

15.2.2. Registering from the Command Line

The simplest way to register a machine is to pass the **`register`** command with the user account information required to authenticate to Customer Portal Subscription Management. When the system is successfully authenticated, it echoes back the newly-assigned system inventory ID and the user account name which registered it.

The **`register`** options are listed in Table 15.2, "register Options".

Example 15.1. Registering a System to the Customer Portal

```
[root@server1 ~]# subscription-manager register --username admin-example -
-password secret

The system has been registered with id: 7d133d55-876f-4f47-83eb-
0ee931cb0a97
```

Example 15.2. Automatically Subscribing While Registering

The **`register`** command has an option, **`--autosubscribe`**, which allows the system to be registered to the subscription service and immediately attaches the subscription which best matches the system's architecture, in a single step.

```
[root@server1 ~]# subscription-manager register --username admin-example -
-password secret --autosubscribe
```

This is the same behavior as when registering with the default settings in the Subscription Manager UI.

Example 15.3. Registering a System with Subscription Asset Manager

With Subscription Asset Managr or CloudForms System Engine, an account can have multiple, independent subdivisions called *organizations*t is required that you specify which organization (essentially an independent group or unit within the main account) to join the system to. This is done by using the **--org** option in addition to the username and password. The given user must also have the access permissions to add systems to that organization.

To register with a subscription service other than Customer Portal Subscription Management, several additional options must be used to identify the environment and organizational divisions that the system is being registered to:

- The username and password **for the user account withint the subscription service itself**
- **--serverurl** to give the hostname of the subscription service
- **--baseurl** to give the hostname of the content delivery service (for CloudForms System Engine only)
- **--org** to give the name of the organization under which to register the system
- **--environment** to give the name of an environment (group) within the organization to which to add the system; this is optional, since a default environment is set for any organization

  A system can only be added to an environment during registration.

```
[root@server1 ~]# subscription-manager register --username=admin-example -
-password=secret --org="IT Department" --environment="dev" --
serverurl=sam-server.example.com

The system has been registered with id: 7d133d55-876f-4f47-83eb-
0ee931cb0a97
```

Note

If the system is in a multi-org environment and no organization is given, the **register** command returns a Remote Server error.

Table 15.2. register Options

Options	Description	Required
--username=*name*	Gives the content server user account name.	Required
--password=*password*	Gives the password for the user account.	Required
--serverurl=*hostname*	Gives the hostname of the subscription service to use. The default is for Customer Portal Subcription Management, subscription.rhn.redhat.com. If this option is not used, the system is registered with Customer Portal Subscription Management.	Required for Subscription Asset Manager or CloudForms System Engine

Options	Description	Required
--baseurl=*URL*	Gives the hostname of the content delivery server to use to receive updates. Both Customer Portal Subscription Management and Subscription Asset Manager use Red Hat's hosted content delivery services, with the URL https://cdn.redhat.com. Since CloudForms System Engine hosts its own content, the URL must be used for systems registered with System Engine.	Required for CloudForms System Engine
--org=*name*	Gives the organization to which to join the system.	Required, except for hosted environments
--environment=*name*	Registers the system to an environment within an organization.	Optional
--name=*machine_name*	Sets the name of the system to register. This defaults to be the same as the hostname.	Optional
--autosubscribe	Automatically ataches the best-matched compatible subscription. This is good for automated setup operations, since the system can be configured in a single step.	Optional
--activationkey=*key*	Attaches existing subscriptions as part of the registration process. The subscriptions are pre-assigned by a vendor or by a systems administrator using Subscription Asset Manager.	Optional
--servicelevel=None|Standard|Premium	Sets the service level to use for subscriptions on that machine. This is only used with the **`--autosubscribe`** option.	Optional
--release=NUMBER	Sets the operating system minor release to use for subscriptions for the system. Products and updates are limited to that specific minor release version. This is used only used with the **`--autosubscribe`** option.	Optional
--force	Registers the system even if it is already registered. Normally, any register operations will fail if the machine is already registered.	Optional

15.2.3. Unregistering

The only thing required to unregister a machine is to run the **`unregister`** command. This removes the system's entry from the subscription service, removes any subscriptions, and, locally, deletes its identity and subscription certificates.

From the command line, this requires only the **`unregister`** command.

Example 15.4. Unregistering a System

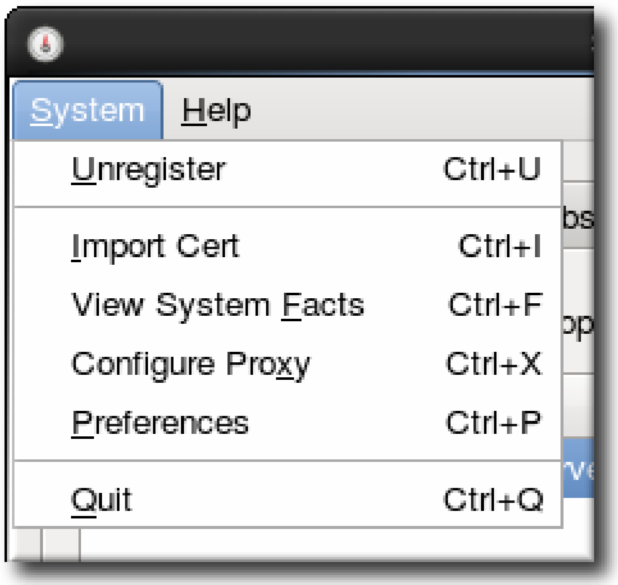
```
[root@server1 ~]# subscription-manager unregister
```

To unregister from the Subscription Manager GUI:

1. Open the Subscription Manager UI.

   ```
   [root@server ~]# subscription-manager-gui
   ```

2. Open the **`System`** menu, and select the **Unregister** item.



3. Confirm that the system should be unregistered.

15.3. Attaching and Removing Subscriptions

Assigning a *subscription* to a system gives the system the ability to install and update any Red Hat product in that subscription. A subscription is a list of all of the products, in all variations, that were purchased at one time, and it defines both the products and the number of times that subscription can be used. When one of those licenses is associated with a system, that subscription is *attached* to the system.

15.3.1. Attaching and Removing Subscriptions through the GUI

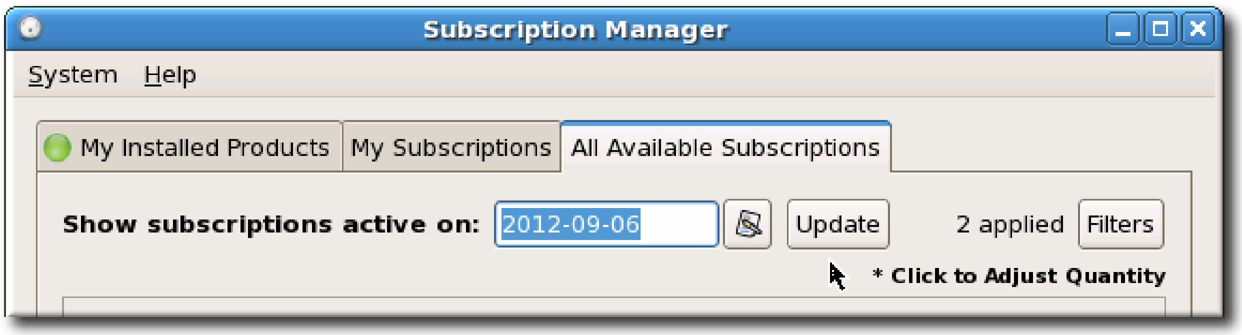
15.3.1.1. Attaching a Subscription

1. Launch Subscription Manager. For example:

   ```
   [root@server ~]# subscription-manager-gui
   ```

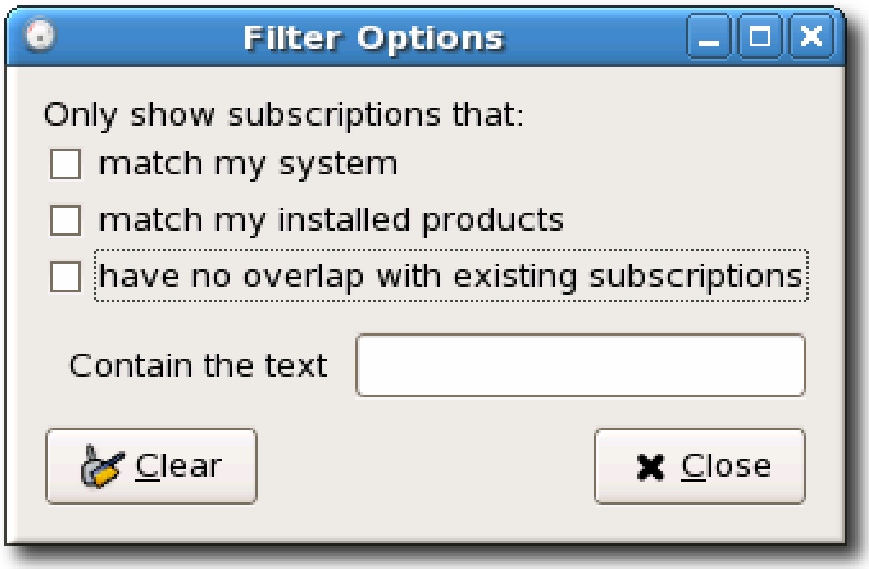
2. Open the **`All Available Subscriptions`** tab.

3. Optionally, set the date range and click the **`Filters`** button to set the filters to use to search for available subscriptions.


Subscription Manager
System Help
My Installed Products
My Subscriptions
All Available Subscriptions
Show subscriptions active on: 2012-09-06
Update
2 applied
Filters
* Click to Adjust Quantity

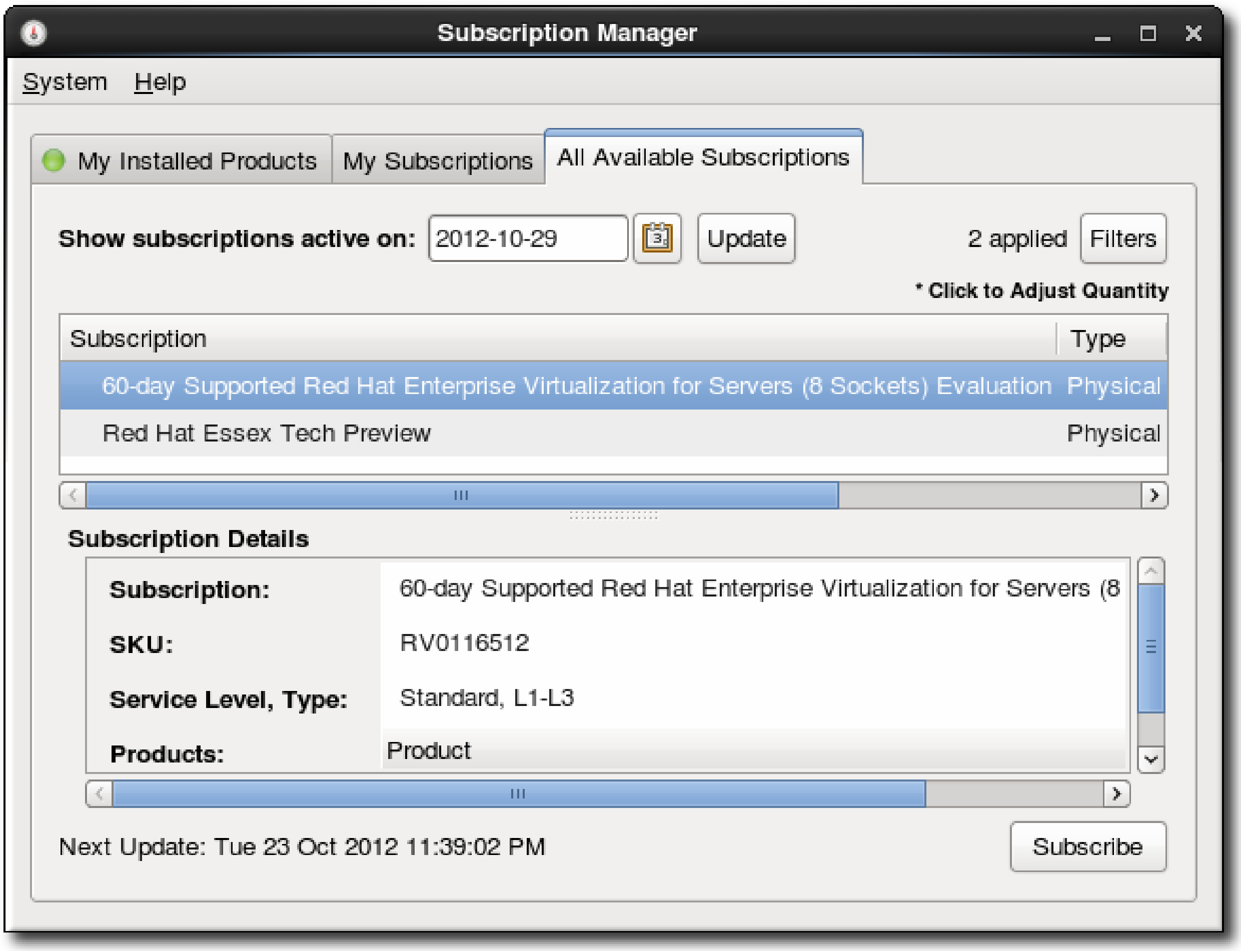
Subscriptions can be filtered by their active date and by their name. The checkboxes provide more fine-grained filtering:

- *match my system* shows only subscriptions which match the system architecture.
- *match my installed products* shows subscriptions which work with currently installed products on the system.
- *have no overlap with existing subscriptions* excludes subscriptions with duplicate products. If a subscription is already attached to the system for a specific product or if multiple subscriptions supply the same product, then the subscription service filters those subscriptions and shows only the best fit.
- *contain the text* searches for strings, such as the product name, within the subscription or pool.


Filter Options
Only show subscriptions that:
match my system
match my installed products
have no overlap with existing subscriptions
Contain the text
Clear
Close

After setting the date and filters, click the **`Update`** button to apply them.

4. Select one of the available subscriptions.



Show subscriptions active on: 2012-10-29

2 applied

* Click to Adjust Quantity

Subscription	Type
60-day Supported Red Hat Enterprise Virtualization for Servers (8 Sockets) Evaluation	Physical
Red Hat Essex Tech Preview	Physical

Subscription Details

Subscription:	60-day Supported Red Hat Enterprise Virtualization for Servers (8
SKU:	RV0116512
Service Level, Type:	Standard, L1-L3
Products:	Product

Next Update: Tue 23 Oct 2012 11:39:02 PM

5. Click the **`Subscribe`** button.

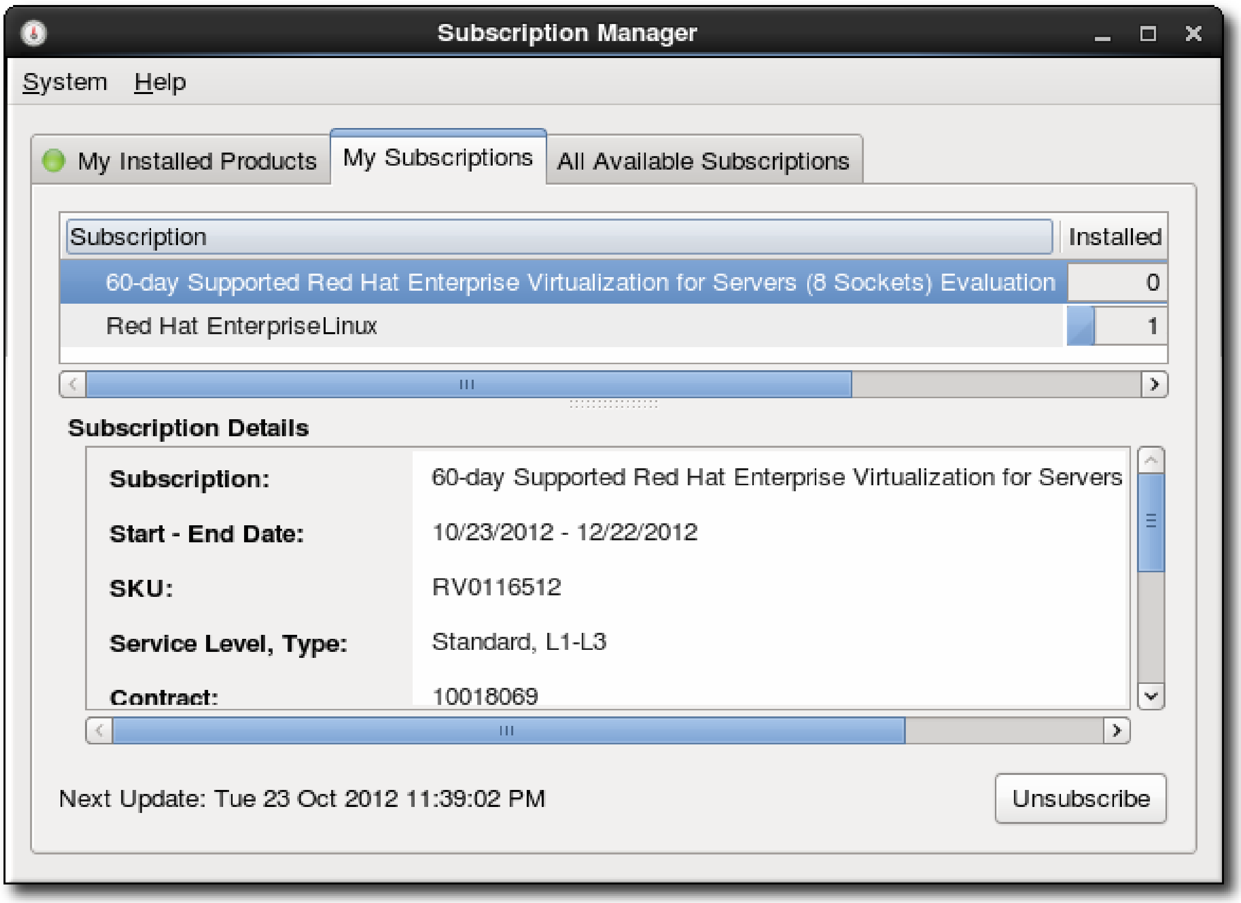
15.3.1.2. Removing Subscriptions

1. Launch Subscription Manager. For example:

```
[root@server ~]# subscription-manager-gui
```

2. Open the **`My Subscriptions`** tab.

   All of the active subscriptions to which the system is currently attached are listed. (The products available through the subscription may or may not be installed.)



Subscription	Installed
60-day Supported Red Hat Enterprise Virtualization for Servers (8 Sockets) Evaluation	0
Red Hat EnterpriseLinux	1

Subscription Details

Subscription:	60-day Supported Red Hat Enterprise Virtualization for Servers
Start - End Date:	10/23/2012 - 12/22/2012
SKU:	RV0116512
Service Level, Type:	Standard, L1-L3
Contract:	10018069

Next Update: Tue 23 Oct 2012 11:39:02 PM

3. Select the subscription to remove.
4. Click the **`Unsubscribe`** button in the bottom right of the window.

15.3.2. Attaching and Removing Subscriptions through the Command Line

15.3.2.1. Attaching Subscriptions

Attaching subscriptions to a system requires specifying the individual product or subscription to attach, using the **`--pool`** option.

```
[root@server1 ~]# subscription-manager subscribe --pool=XYZ01234567
```

The options for the **`subscribe`** command are listed in Table 15.3, “subscribe Options”.

The ID of the subscription pool for the purchased product must be specified. The pool ID is listed with the product subscription information, which is available from running the **`list`** command:

```
[root@server1 ~]# subscription-manager list --available

+-------------------------------------------+
    Available Subscriptions
+-------------------------------------------+
ProductName:            RHEL for Physical Servers
ProductId:              MKT-rhel-server
PoolId:                 ff8080812bc382e3012bc3845ca000cb
Quantity:               10
Expires:                2011-09-20
```

Alternatively,the best-fitting subscriptions, as identified by the subscription service, can be attached to the system by using the **`--auto`** option (which is analogous to the **`--autosubscribe`** option with the **`register`** command).

```
[root@server1 ~]# subscription-manager subscribe --auto
```

Table 15.3. subscribe Options

Options	Description	Required
--pool=*pool-id*	Gives the ID for the subscription to attach to the system.	Required, unless **`--auto`** is used
--auto	Automatically attaches the system to the best-match subscription or subscriptions.	Optional
--quantity=*number*	Attaches multiple counts of a subscription to the system. This is used to cover subscriptions that define a count limit, like using two 2-socket server subscriptions to cover a 4-socket machine.	Optional
--servicelevel=None|Standard|Premium	Sets the service level to use for subscriptions on that machine. This is only used with the **`--auto`** option.	Optional

15.3.2.2. Removing Subscriptions from the Command Line

A system can be attached to multiple subscriptions and products. Similarly, a single subscription or all subscriptions can be removed from the system.

Running the **`unsubscribe`** command with the **`--all`** option removes every product subscription and subscription pool that is currently attached to the system.

```
[root@server1 ~]# subscription-manager unsubscribe --all
```

It is also possible to remove a single product subscription. Each product has an identifying X.509 certificate installed with it. The product subscription to remove is identified in the **`unsubscribe`** command by referencing the ID number of that X.509 certificate.

1. Get the serial number for the product certificate, if you are removing a single product subscription. The serial number can be obtained from the *subscription#*.**`pem`** file (for example, **`392729555585697907.pem`**) or by using the **`list`** command. For example:

```
[root@server1 ~]# subscription-manager list --consumed

+-------------------------------------------+
    Consumed Product Subscriptions
+-------------------------------------------+


ProductName:          High availability (cluster suite)
ContractNumber:       0
```

```
SerialNumber:          112875143586000162
Active:                True
Begins:                2010-09-18
Expires:               2011-11-18
```

2. Run the subscription-manager tool with the **`--serial`** option to specify the certificate.

```
[root@server1 ~]# subscription-manager unsubscribe --
serial=11287514358600162
```

15.4. Redeeming Vendor Subscriptions

Systems can be set up with pre-existing subscriptions already available to that system. For some systems which were purchased through third-party vendors, a subscription to Red Hat products is included with the purchase of the machine.

Red Hat Subscription Manager pulls information about the system hardware and the BIOS into the system facts to recognize the hardware vendor. If the vendor and BIOS information matches a certain configuration, then the subscription can be *redeemed*, which will allow subscriptions to be automatically attached to the system.

15.4.1. Redeeming Subscriptions through the GUI

Note

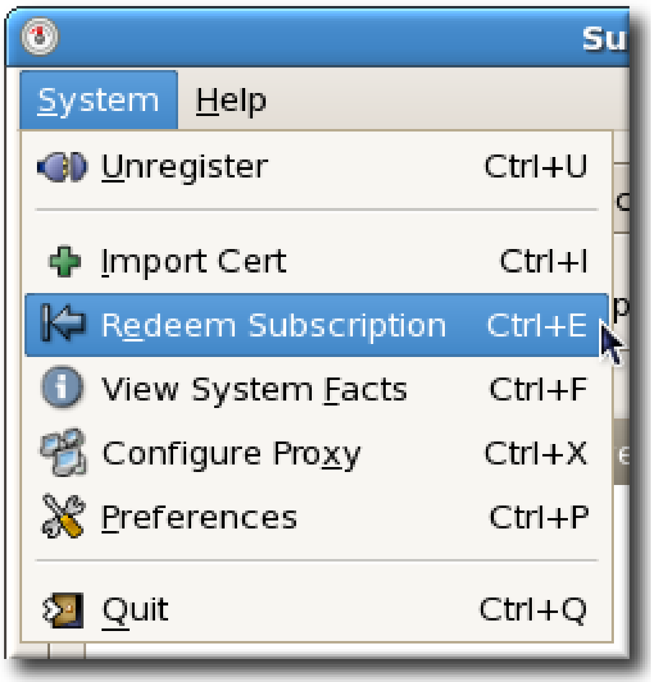
If the machine does not have any subscriptions to be redeemed, then the **`Redeem`** menu item is not there.

1. Launch Subscription Manager. For example:

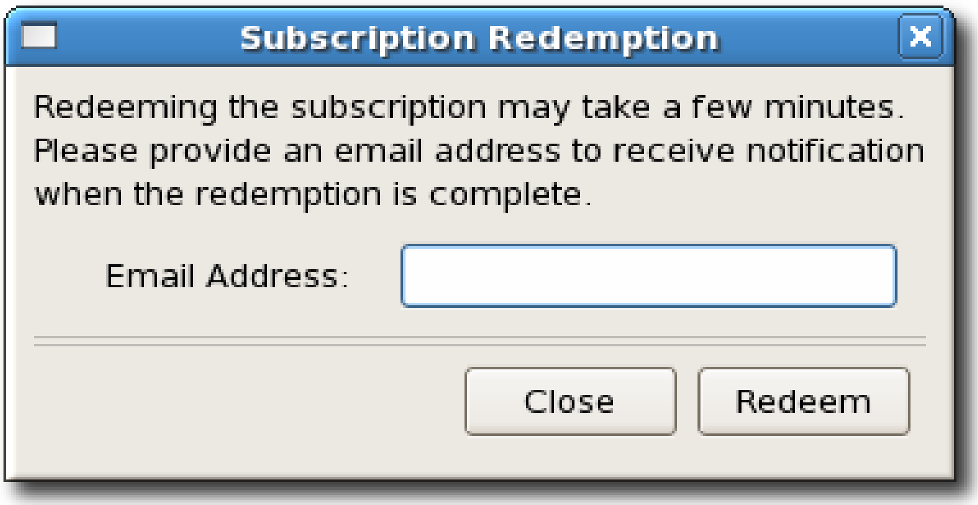
```
[root@server ~]# subscription-manager-gui
```

2. If necessary, register the system, as described in Section 15.2.1, “Registering from the GUI”.

3. Open the **System** menu in the top left of the window, and click the **`Redeem`** item.



4. In the dialog window, enter the email address to send the notification to when the redemption is complete. Because the redemption process can take several minutes to contact the vendor and receive information about the pre-configured subscriptions, the notification message is sent through email rather than through the Subscription Manager dialog window.


Subscription Redemption

Redeeming the subscription may take a few minutes. Please provide an email address to receive notification when the redemption is complete.

Email Address:

5. Click the **Redeem** button.

It can take up to ten minutes for the confirmation email to arrive.

15.4.2. Redeeming Subscriptions through the Command Line

Note

The machine must be registered *first* so that the subscription service can properly identify the system and its subscriptions.

The machine subscriptions are redeemed by running the **redeem** command, with an email address to send the redemption email to when the process is complete.

```
# subscription-manager redeem --email=jsmith@example.com
```

15.5. Attaching Subscriptions from a Subscription Asset Manager Activation Key

A local Subscription Asset Manager can pre-configure subscriptions to use for a system, and that pre-configured set of subscriptions is identified by an activation key. That key can then be used to attach those subscriptions on a local system.

The Subscription Asset Manager activation key can be used as part of the registration process for the new system:

```
# subscription-manager register --username=jsmith --password=secret --
org="IT Dept" --activationkey=abcd1234
```

If there are multiple organizations, it is still necessary to specify the organization for the system. That information is not defined in the activation key.

15.6. Setting Preferences for Systems

Auto-attaching and healing (updating) subscriptions selects what subscriptions to attach to a system based on a variety of criteria, including current installed products, hardware, and architecture. It is possible to set twc additional preferences for Subscription Manager to use:

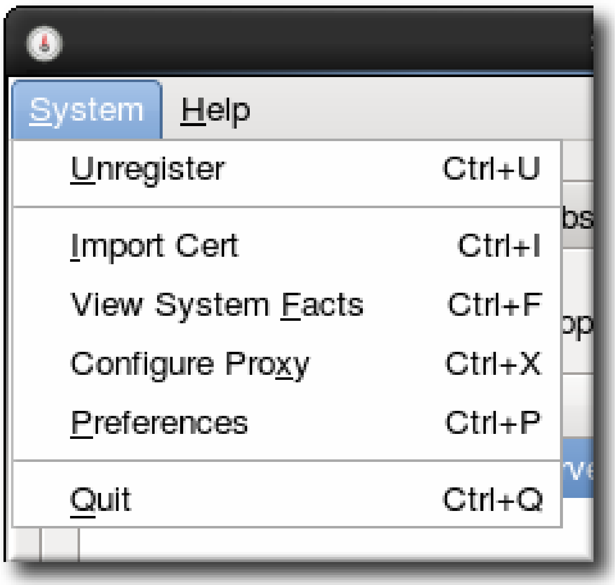
- Service levels for subscriptions
- The operating system minor version (X.Y) to use

This is especially useful for healing, which runs daily to ensure that all installed products and current subscriptions remain active.

15.6.1. Setting Preferences in the UI

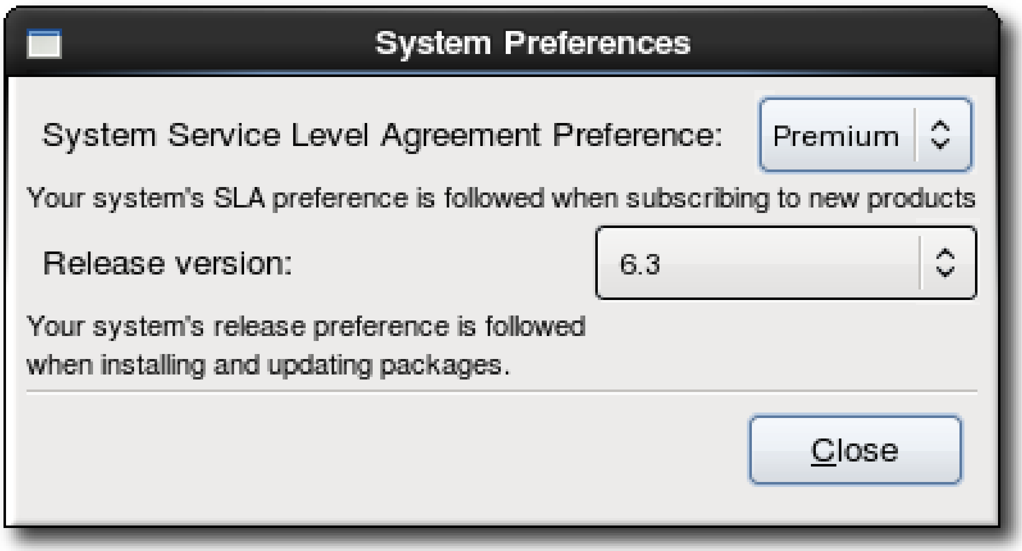
Both a service level preference and an operating system release version preference are set in the **`System Preferences`** dialog box in Subscription Manager.

1. Open the Subscription Manager.
2. Open the **`System`** menu.
3. Select the **`System Preferences`** menu item.



4. Select the desired service level agreement preference from the drop-down menu. Only service levels available to the Red Hat account, based on all of its active subscriptions, are listed.

5. Select the operating system release preference in the **`Release version`** drop-down menu. The only versions listed are Red Hat Enterprise Linux versions for which the account has an active subscription.



System Service Level Agreement Preference: Premium

Your system's SLA preference is followed when subscribing to new products

Release version: 6.3

Your system's release preference is followed when installing and updating packages.

6. The preferences are saved and applied to future subscription operations when they are set. To close the dialog, click **`Close`**.

15.6.2. Setting Service Levels Through the Command Line

A general service level preference can be set using the **`service-level --set`** command.

Example 15.5. Setting a Service Level Preference

First, list the available service levels for the system, using the **`--list`** option with the **`service-level`** command.

```
[root@server ~]# subscription-manager service-level --list
+-------------------------------------------+
          Available Service Levels
+-------------------------------------------+
Standard
None
Premium
Self-Support
```

Then, set the desired level for the system.

```
[root@server ~]# subscription-manager service-level --set=self-support
Service level set to: self-support
```

The current setting for the local system is shown with the **`--show`** option:

```
[root#server ~]# subscription-manager service-level --show
Current service level: self-support
```

A service level preference can be defined when a subscription operation is being run (such as registering a system or attaching subscriptions after registration). This can be used to override a system preference. Both the **`register`** and **`subscribe`** commands have the **`--servicelevel`** option to set a preference for that action.

Example 15.6. Autoattaching Subscriptions with a Premium Service Level

```
[root#server ~]# subscription-manager subscribe --auto --servicelevel
Premium
Service level set to: Premium
Installed Product Current Status:
ProductName:            Red Hat Enterprise Linux 5 Server
Status:                 Subscribed
```

Note

The **`--servicelevel`** option requires the **`--autosubscribe`** option (for register) or **`--auto`** option (for subscribe). It cannot be used when attaching a specified pool or when importing a subscription.

15.6.3. Setting a Preferred Operating System Release Version in the Command Line

Many IT environments have to be certified to meet a certain level of security or other criteria. In that case, major upgrades must be carefully planned and controlled — so administrators cannot simply run **`yum update`** and move from version to version.

Setting a release version preference limits the system access to content repositories associated with that operating system version instead of automatically using the newest or latest version repositories.

For example, if the preferred operating system version is 5.9, then 5.9 content repositories will be preferred for all installed products and attached subscriptions for the system, even as other repositories become available.

Example 15.7. Setting an Operating System Release During Registration

A preference for a release version can be set when the system is registered by using **`--release`** option with the **`register`**. This applies the release preference to any subscriptions selected and auto-attached to the system at registration time.

Setting a preference requires the **`--autosubscribe`** option, because it is one of the criteria used to select subscriptions to auto-attach.

```
[root#server ~]# subscription-manager register --autosubscribe --
release=5.9 --username=admin@example.com...
```

Note

Unlike setting a service level preference, a release preference can only be used during registration or set as a preference. It cannot be specified with the **`subscribe`** command.

Example 15.8. Setting an Operating System Release Preference

The **`release`** command can display the available operating system releases, based on the available, purchased (not only attached) subscriptions for the organization.

```
[root#server ~]# subscription-manager release --list
+-------------------------------------------+
          Available Releases
+-------------------------------------------+
5.8
5.9
5Server
```

The **`--set`** then sets the preference to one of the available release versions:

```
[root#server ~]# subscription-manager release --set=5.9
Release version set to: 5.9
```

15.6.4. Removing a Preference

To remove a preference through the command line, use the **`--unset`** with the appropriate command. For example, to unset a release version preference:

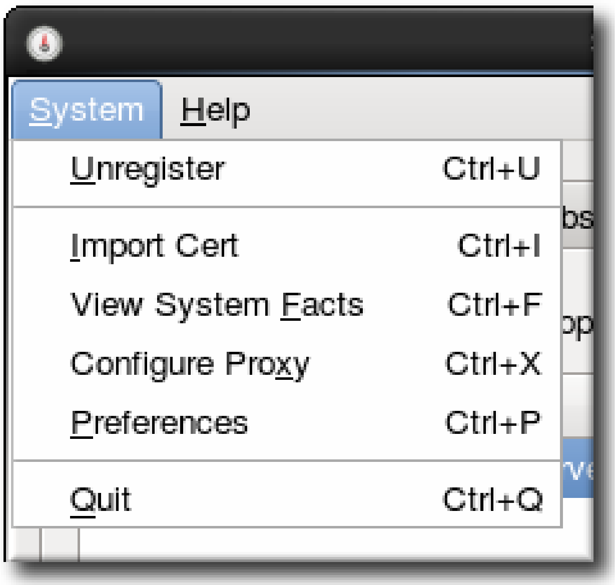
```
[root#server ~]# subscription-manager release --unset
Release version set to:
```

To remove or unset a preference in the UI:

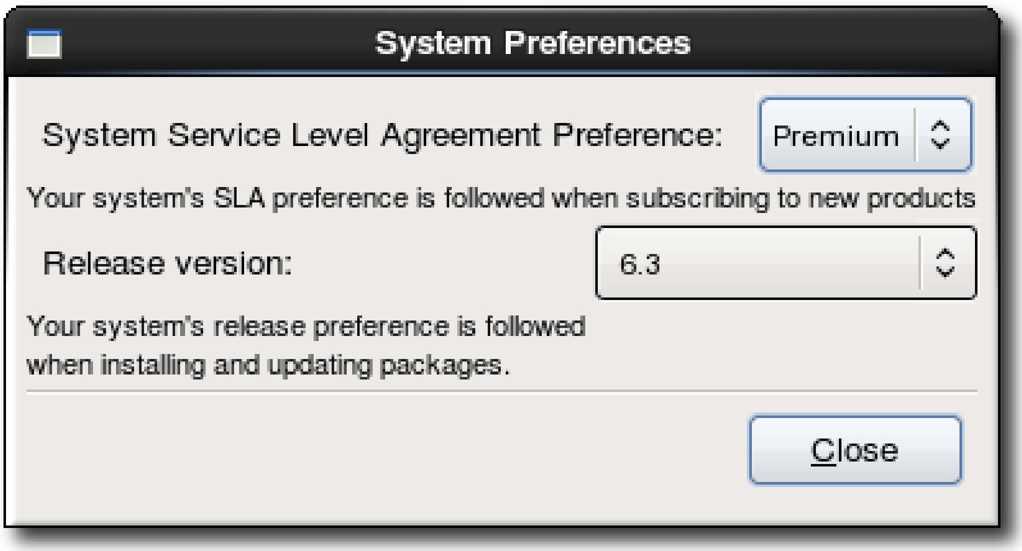
1. Open the Subscription Manager.

2. Open the **`System`** menu.

3. Select the **System Preferences** menu item.



4. Set the service level or release version value to the blank line in the corresponding drop-down menu.


System Preferences

System Service Level Agreement Preference: Premium

Your system's SLA preference is followed when subscribing to new products

Release version: 6.3

Your system's release preference is followed when installing and updating packages.

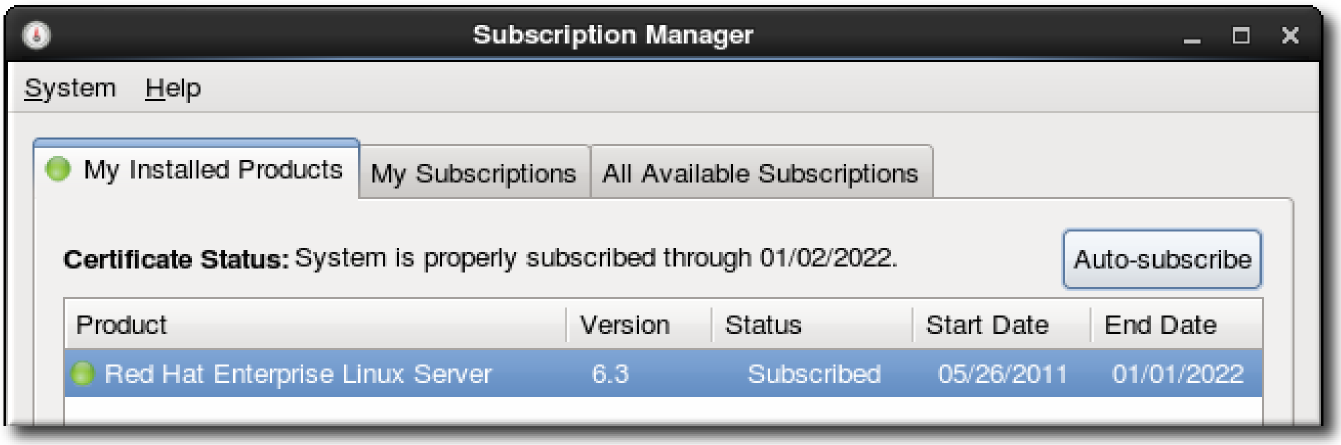
5. Click **Close**.

15.7. Managing Subscription Expiration and Notifications

Subscriptions are active for a certain period of time, called the *validity period*. When a subscription is purchased, the start and end dates for the contract are set.

On a system, there can be multiple subscriptions attached. Each product requires its own subscription. Additionally, some products may require multiple quantities for it to be fully subscribed. For example, a 16 socket machine may require four 4-socket operating system subscriptions to cover the socket count.

The **My Installed Software** tab shows the subscription status for the entire system. It also shows a date; that is the first date that a product subscription goes from valid to invalid (meaning it expires).

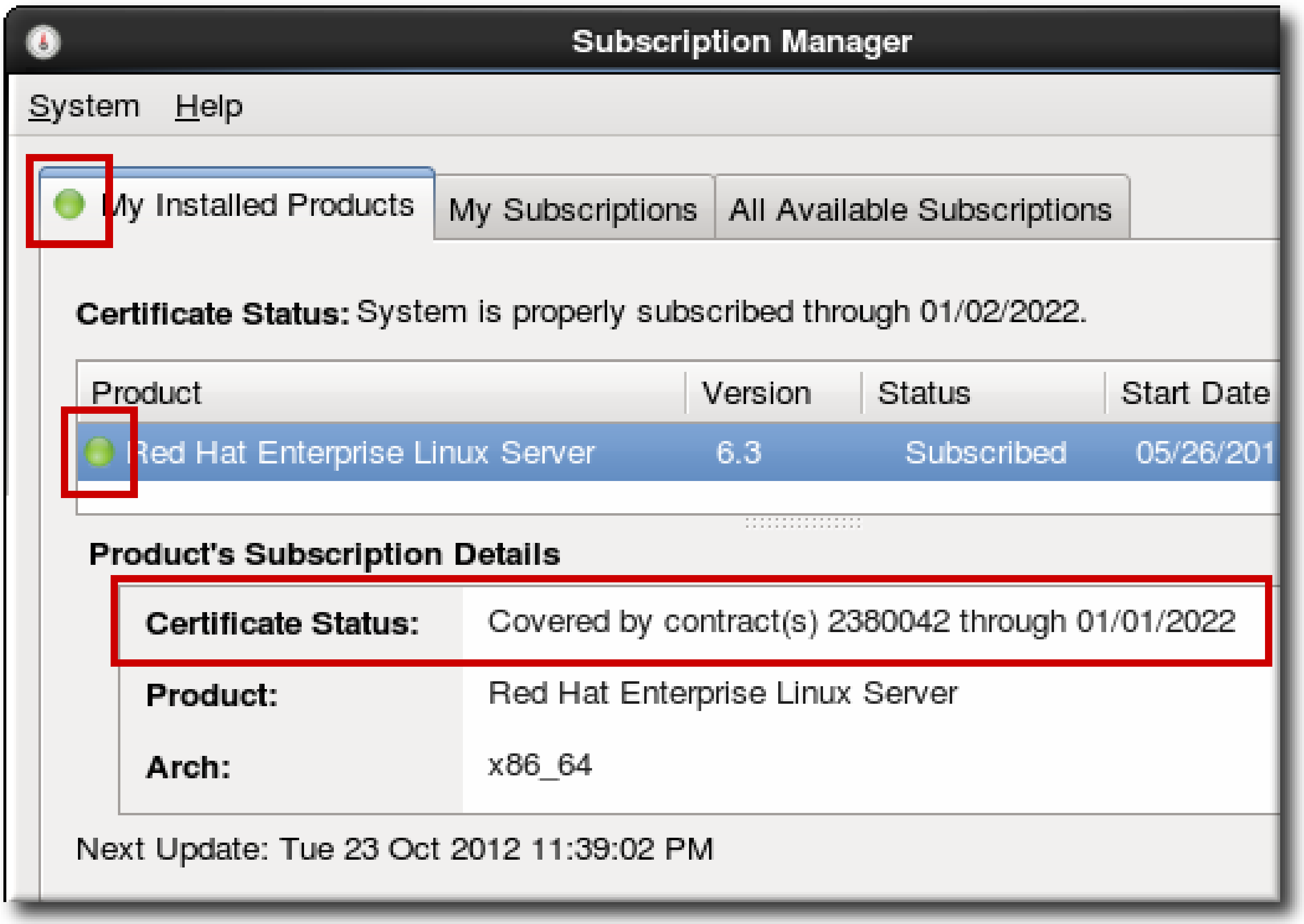


Certificate Status: System is properly subscribed through 01/02/2022.

Product	Version	Status	Start Date	End Date
Red Hat Enterprise Linux Server	6.3	Subscribed	05/26/2011	01/01/2022

Figure 15.2. Valid Until...

The Red Hat Subscription Manager provides a series of log and UI messages that indicate any changes to the valid certificates of any installed products for a system. In the Subscription Manager GUI, the status of the system subscriptions is color-coded, where *green* means all products are fully subscribed, *yellow* means that some products may not be subscribed but updates are still in effect, and *red* means that updates are disabled.



Certificate Status: System is properly subscribed through 01/02/2022.

Product	Version	Status	Start Date
Red Hat Enterprise Linux Server	6.3	Subscribed	05/26/201

Product's Subscription Details

Certificate Status:	Covered by contract(s) 2380042 through 01/01/2022
Product:	Red Hat Enterprise Linux Server
Arch:	x86_64

Next Update: Tue 23 Oct 2012 11:39:02 PM

Figure 15.3. Color-Coded Status Views

The command-line tools also indicate that status of the machine. The green, yellow, and red codes translate to text status messages of *subscribed*, *partially subscribed*, and *expired/not subscribed*, respectively.

```
[root@server ~]# subscription-manager list
+-------------------------------------------+
    Installed Product Status
+-------------------------------------------+

ProductName:            Red Hat Enterprise Linux Server
Status: Not Subscribed
Expires:
SerialNumber:
ContractNumber:
AccountNumber:
```

Whenever there is a warning about subscription changes, a small icon appears in the top menu bar, similar to a fuel gauge.

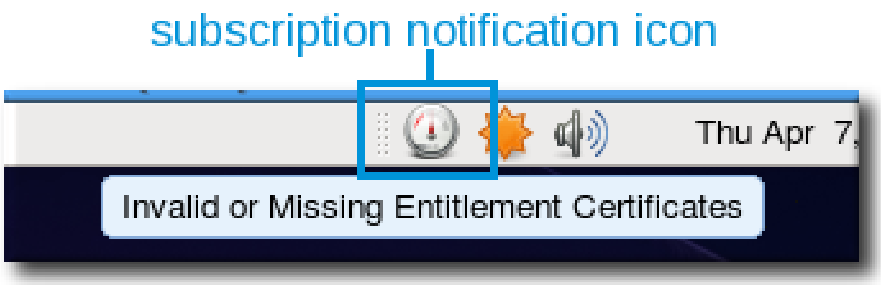
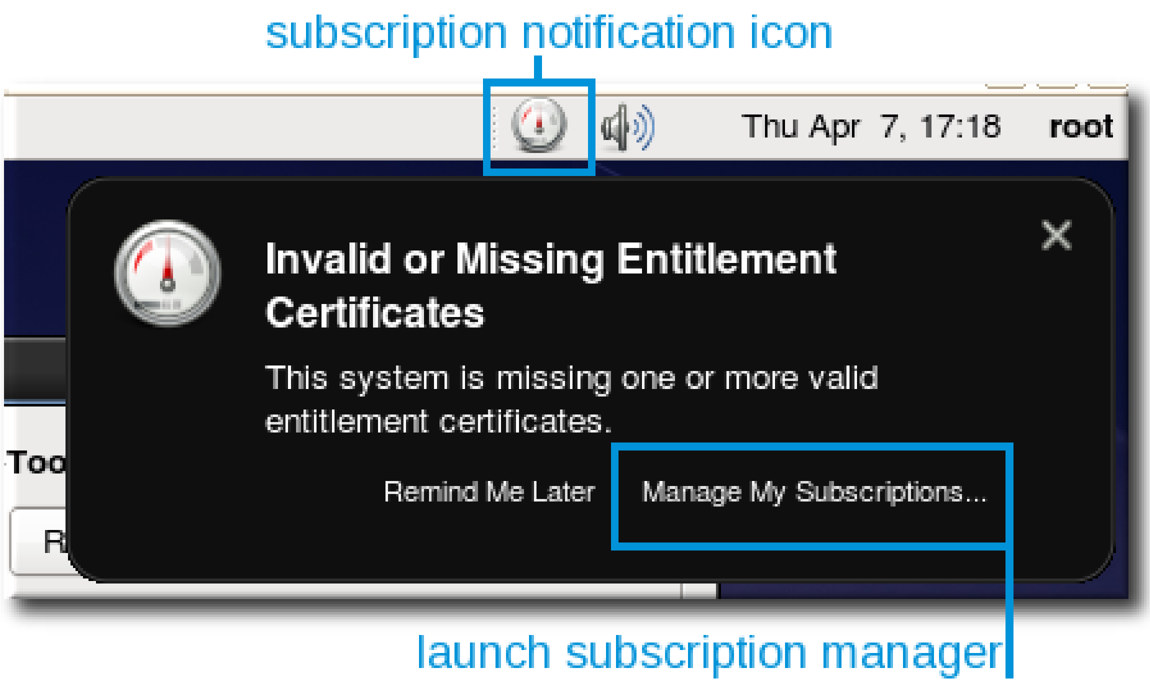

subscription notification icon
Thu Apr 7,
Invalid or Missing Entitlement Certificates

Figure 15.4. Subscription Notification Icon

As any installed product nears the expiration date of the subscription, the Subscription Manager daemon will issue a warning. A similar message is given when the system has products without a valid certificate, meaning either a subscription is not atached that covers that product or the product is installed past the expiration of the subscription. Clicking the **`Manage My Subscriptions...`** button in the subscription notification window opens the Red Hat Subscription Manager GUI to view and update subscriptions.

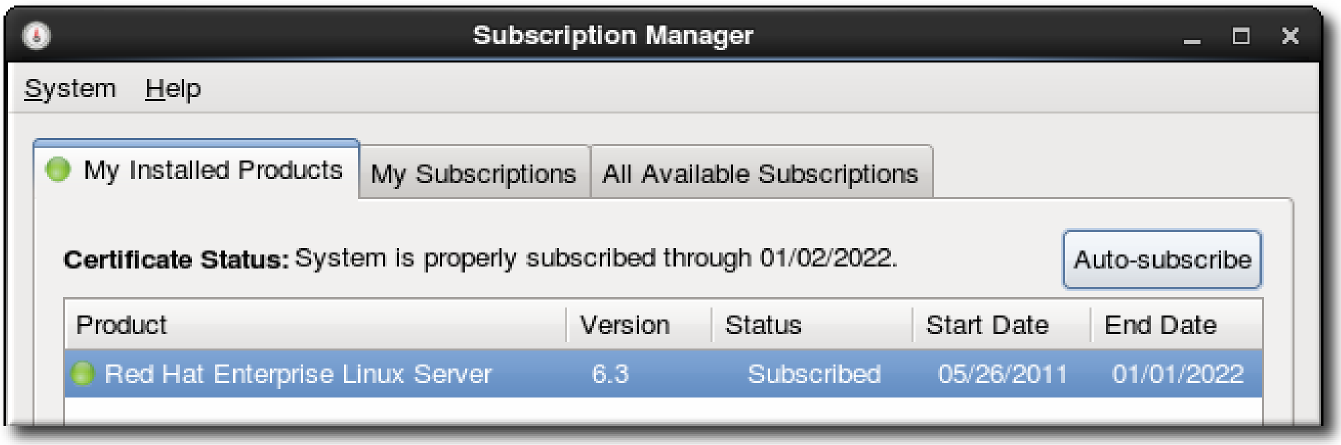

subscription notification icon
Thu Apr 7, 17:18
root
Too
R
Remind Me Later
Manage My Subscriptions...
launch subscription manager

Invalid or Missing Entitlement Certificates

This system is missing one or more valid entitlement certificates.

Figure 15.5. Subscription Warning Message

When the Subscription Manager UI opens, whether it was opened through a notification or just opened normally, there is an icon in the upper left corner that shows whether products lack a valid certificate. The easiest way to attach subscriptions which match invalidated products is to click the **`Autosubscribe`** button.

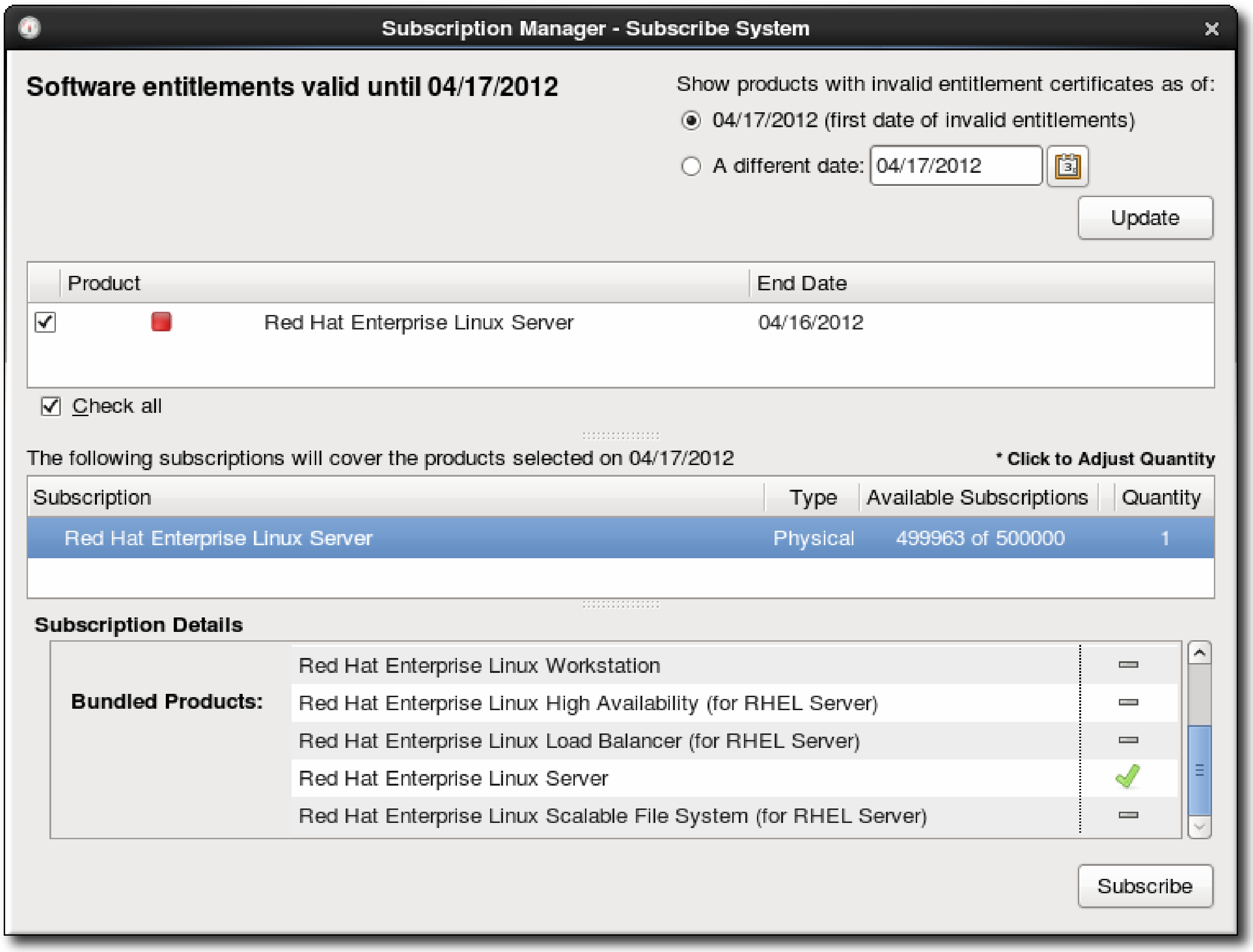


Certificate Status: System is properly subscribed through 01/02/2022.

Product	Version	Status	Start Date	End Date
Red Hat Enterprise Linux Server	6.3	Subscribed	05/26/2011	01/01/2022

Figure 15.6. Autosubscribe Button

The **`Subscribe System`** dialog shows a targeted list of available subscriptions that apply to the specific products that do not have valid certificates (assuming subscriptions are available).


Subscription Manager - Subscribe System
Update
Check all
* Click to Adjust Quantity
Subscribe

Software entitlements valid until 04/17/2012

Show products with invalid entitlement certificates as of:

(●) 04/17/2012 (first date of invalid entitlements)

( ) A different date: 04/17/2012

		Product	End Date
✓	🟥	Red Hat Enterprise Linux Server	04/16/2012

The following subscriptions will cover the products selected on 04/17/2012

Subscription	Type	Available Subscriptions	Quantity
Red Hat Enterprise Linux Server	Physical	499963 of 500000	1

Subscription Details

Bundled Products:	Red Hat Enterprise Linux Workstation	—
	Red Hat Enterprise Linux High Availability (for RHEL Server)	—
	Red Hat Enterprise Linux Load Balancer (for RHEL Server)	—
	Red Hat Enterprise Linux Server	✓
	Red Hat Enterprise Linux Scalable File System (for RHEL Server)	—

Figure 15.7. Subscribe System

Part III. Network-Related Configuration

After explaining how to configure the network, this part discusses topics related to networking such as how to allow remote logins, share files and directories over the network, and set up a Web server.

Chapter 16. Network Interfaces

Under Red Hat Enterprise Linux, all network communications occur between configured software *interfaces* and *physical networking devices* connected to the system.

The configuration files for network interfaces are located in the **`/etc/sysconfig/network-scripts/`** directory. The scripts used to activate and deactivate these network interfaces are also located here. Although the number and type of interface files can differ from system to system, there are three categories of files that exist in this directory:

1. *Interface configuration files*
2. *Interface control scripts*
3. *Network function files*

The files in each of these categories work together to enable various network devices.

This chapter explores the relationship between these files and how they are used.

16.1. Network Configuration Files

Before delving into the interface configuration files, let us first itemize the primary configuration files used in network configuration. Understanding the role these files play in setting up the network stack can be helpful when customizing a Red Hat Enterprise Linux system.

The primary network configuration files are as follows:

`/etc/hosts`

The main purpose of this file is to resolve host names that cannot be resolved any other way. It can also be used to resolve host names on small networks with no **`DNS`** server. Regardless of the type of network the computer is on, this file should contain a line specifying the **`IP`** address of the loopback device (**`127.0.0.1`**) as **`localhost.localdomain`**. For more information, refer to the **`hosts(5)`** man page.

`/etc/resolv.conf`

This file specifies the **`IP`** addresses of **`DNS`** servers and the search domain. Unless configured to do otherwise, the network initialization scripts populate this file. For more information about this file, refer to the **`resolv.conf(5)`** man page.

`/etc/sysconfig/network`

This file specifies routing and host information for all network interfaces. It is used to contain directives which are to have global effect and not to be interface specific. For more information about this file and the directives it accepts, refer to Section 32.1.21, “`/etc/sysconfig/network`”.

`/etc/sysconfig/network-scripts/ifcfg-<`*`interface-name`*`>`

For each network interface, there is a corresponding interface configuration script. Each of these files provide information specific to a particular network interface. Refer to Section 16.2, “Interface Configuration Files” for more information on this type of file and the directives it accepts.

Warning

The **/etc/sysconfig/networking/** directory is used by the **Network Administration Tool** (**system-config-network**) and its contents should **not** be edited manually. Using only one method for network configuration is strongly encouraged, due to the risk of configuration deletion.

For more information about configuring network interfaces using the **Network Administration Tool**, refer to Chapter 17, *Network Configuration*

16.2. Interface Configuration Files

Interface configuration files control the software interfaces for individual network devices. As the system boots, it uses these files to determine what interfaces to bring up and how to configure them. These files are usually named **ifcfg-*<name>*** , where *<name>* refers to the name of the device that the configuration file controls.

16.2.1. Ethernet Interfaces

One of the most common interface files is **/etc/sysconfig/network-scripts/ifcfg-eth0**, which controls the first Ethernet *network interface card* or NIC in the system. In a system with multiple NICs, there are multiple **ifcfg-eth*<X>*** files (where *<X>* is a unique number corresponding to a specific interface). Because each device has its own configuration file, an administrator can control how each interface functions individually.

The following is a sample **ifcfg-eth0** file for a system using a fixed **IP** address:

```
DEVICE=eth0
BOOTPROTO=none
ONBOOT=yes
NETMASK=255.255.255.0
IPADDR=10.0.1.27
USERCTL=no
```

The values required in an interface configuration file can change based on other values. For example, the **ifcfg-eth0** file for an interface using **DHCP** looks different because **IP** information is provided by the **DHCP** server:

```
DEVICE=eth0
BOOTPROTO=dhcp
ONBOOT=yes
```

The **Network Administration Tool** (**system-config-network**) is an easy way to make changes to the various network interface configuration files (refer to Chapter 17, *Network Configuration* for detailed instructions on using this tool).

However, it is also possible to manually edit the configuration files for a given network interface.

Below is a listing of the configurable parameters in an Ethernet interface configuration file:

BONDING_OPTS=*<parameters>*

sets the configuration parameters for the bonding device, and is used in

/etc/sysconfig/network-scripts/ifcfg-bond*<N>* (see Section 16.2.3, “Channel Bonding Interfaces”). These parameters are identical to those used for bonding devices in **/sys/class/net/*<bonding device>*/bonding**, and the module parameters for the bonding driver as described in ***bonding** Module Directives*.

This configuration method is used so that multiple bonding devices can have different configurations. If you use **BONDING_OPTS** in **ifcfg-*<name>*** , do *not* use **/etc/modprobe.conf** to specify options for the bonding device.

BOOTPROTO=*<protocol>*

where ***<protocol>*** is one of the following:

- **none** — No boot-time protocol should be used.
- **bootp** — The BOOTP protocol should be used.
- **dhcp** — The **DHCP** protocol should be used.

BROADCAST=*<address>*

where ***<address>*** is the broadcast address. This directive is deprecated, as the value is calculated automatically with **ipcalc**.

DEVICE=*<name>*

where ***<name>*** is the name of the physical device (except for dynamically-allocated PPP devices where it is the *logical name*).

DHCP_HOSTNAME=*<name>*

where ***<name>*** is a short host name to be sent to the **DHCP** server. Use this option only if the **DHCP** server requires the client to specify a host name before receiving an **IP** address.

DNS*{1,2}*=*<address>*

where ***<address>*** is a name server address to be placed in **/etc/resolv.conf** if the **PEERDNS** directive is set to **yes**.

ETHTOOL_OPTS=*<options>*

where ***<options>*** are any device-specific options supported by **ethtool**. For example, if you wanted to force 100Mb, full duplex:

```
ETHTOOL_OPTS="autoneg off speed 100 duplex full"
```

Instead of a custom initscript, use **ETHTOOL_OPTS** to set the interface speed and duplex settings. Custom initscripts run outside of the network init script lead to unpredictable results during a post-boot network service restart.

Note

Changing speed or duplex settings almost always requires disabling autonegotiation with the **autoneg off** option. This needs to be stated first, as the option entries are order-dependent.

`GATEWAY=<address>`

where *<address>* is the **`IP`** address of the network router or gateway device (if any).

`HOTPLUG=<answer>`

where *<answer>* is one of the following:

- **`yes`** — This device should be activated when it is hot-plugged (this is the default option).
- **no** — This device should not be activated when it is hot-plugged.

The **`HOTPLUG=no`** option can be used to prevent a channel bonding interface from being activated when a bonding kernel module is loaded.

Refer to Section 16.2.3, “Channel Bonding Interfaces” for more about channel bonding interfaces.

`HWADDR=<MAC-address>`

where *<MAC-address>* is the hardware address of the Ethernet device in the form *AA:BB:CC:DD:EE:FF*. This directive must be used in machines containing more than one NIC to ensure that the interfaces are assigned the correct device names regardless of the configured load order for each NIC's module. This directive should **not** be used in conjunction with **`MACADDR`**.

`IPADDR=<address>`

where ***`<address>`*** is the **`IP`** address.

`LINKDELAY=<time>`

where *<time>* is the number of seconds to wait for link negotiation before configuring the device.

`MACADDR=<MAC-address>`

where *<MAC-address>* is the hardware address of the Ethernet device in the form *AA:BB:CC:DD:EE:FF*. This directive is used to assign a MAC address to an interface, overriding the one assigned to the physical NIC. This directive should **not** be used in conjunction with **`HWADDR`**.

`MASTER=<bond-interface>`

where ***`<bond-interface>`*** is the channel bonding interface to which the Ethernet interface is linked.

This directive is used in conjunction with the **`SLAVE`** directive.

Refer to Section 16.2.3, “Channel Bonding Interfaces” for more information about channel bonding interfaces.

`NETMASK=<mask>`

where ***`<mask>`*** is the netmask value.

`NETWORK=<address>`

where ***`<address>`*** is the network address. This directive is deprecated, as the value is calculated automatically with **`ipcalc`**.

`ONBOOT=<answer>`

where ***<answer>*** is one of the following:

- **yes** — This device should be activated at boot-time.
- **no** — This device should not be activated at boot-time.

PEERDNS=*<answer>*

where ***<answer>*** is one of the following:

- **yes** — Modify **/etc/resolv.conf** if the DNS directive is set. If using **DHCP**, then **yes** is the default.
- **no** — Do not modify **/etc/resolv.conf**.

SLAVE=*<answer>*

where ***<answer>*** is one of the following:

- **yes** — This device is controlled by the channel bonding interface specified in the **MASTER** directive.
- **no** — This device is *not* controlled by the channel bonding interface specified in the **MASTER** directive.

This directive is used in conjunction with the **MASTER** directive.

Refer to Section 16.2.3, "Channel Bonding Interfaces" for more about channel bonding interfaces.

SRCADDR=*<address>*

where ***<address>*** is the specified source **IP** address for outgoing packets.

USERCTL=*<answer>*

where ***<answer>*** is one of the following:

- **yes** — Non-root users are allowed to control this device.
- **no** — Non-root users are not allowed to control this device.

16.2.2. IPsec Interfaces

The following example shows the **ifcfg** file for a network-to-network IPsec connection for LAN A. The unique name to identify the connection in this example is **ipsec1**, so the resulting file is named **/etc/sysconfig/network-scripts/ifcfg-ipsec1**.

```
TYPE=IPsec
ONBOOT=yes
IKE_METHOD=PSK
SRCNET=192.168.1.0/24
DSTNET=192.168.2.0/24
DST=X.X.X.X
```

In the example above, *X.X.X.X* is the publicly routable **IP** address of the destination IPsec router.

Below is a listing of the configurable parameters for an IPsec interface:

DST=<*address*>

where <*address*> is the **IP** address of the IPsec destination host or router. This is used for both host-to-host and network-to-network IPsec configurations.

DSTNET=<*network*>

where <*network*> is the network address of the IPsec destination network. This is only used for network-to-network IPsec configurations.

SRC=<*address*>

where <*address*> is the **IP** address of the IPsec source host or router. This setting is optional and is only used for host-to-host IPsec configurations.

SRCNET=<*network*>

where <*network*> is the network address of the IPsec source network. This is only used for network-to-network IPsec configurations.

TYPE=<*interface-type*>

where <*interface-type*> is **IPSEC**. Both applications are part of the **ipsec-tools** package.

If manual key encryption with IPsec is being used, refer to **/usr/share/doc/initscripts-<*version-number*>/sysconfig.txt** (replace <*version-number*> with the version of the **initscripts** package installed) for configuration parameters.

The **racoon** IKEv1 key management daemon negotiates and configures a set of parameters for IPSec. It can use preshared keys, RSA signatures, or GSS-API. If **racoon** is used to automatically manage key encryption, the following options are required:

IKE_METHOD=<*encryption-method*>

where <*encryption-method*> is either **PSK**, **X509**, or **GSSAPI**. If **PSK** is specified, the **IKE_PSK** parameter must also be set. If **X509** is specified, the **IKE_CERTFILE** parameter must also be set.

IKE_PSK=<*shared-key*>

where <*shared-key*> is the shared, secret value for the PSK (preshared keys) method.

IKE_CERTFILE=<*cert-file*>

where <*cert-file*> is a valid **X.509** certificate file for the host.

IKE_PEER_CERTFILE=<*cert-file*>

where <*cert-file*> is a valid **X.509** certificate file for the *remote* host.

IKE_DNSSEC=<*answer*>

where <*answer*> is **yes**. The **racoon** daemon retrieves the remote host's **X.509** certificate via DNS. If a **IKE_PEER_CERTFILE** is specified, do *not* include this parameter.

For more information about the encryption algorithms available for IPsec, refer to the **setkey** man page. For more information about **racoon**, refer to the **racoon** and **racoon.conf** man pages.

16.2.3. Channel Bonding Interfaces

Red Hat Enterprise Linux allows administrators to bind multiple network interfaces together into a single channel using the **`bonding`** kernel module and a special network interface called a *channel bonding interface*. Channel bonding enables two or more network interfaces to act as one, simultaneously increasing the bandwidth and providing redundancy.

To create a channel bonding interface, create a file in the **`/etc/sysconfig/network-scripts/`** directory called **`ifcfg-bond<N>`** , replacing <*N*> with the number for the interface, such as **`0`**.

The contents of the file can be identical to whatever type of interface is getting bonded, such as an Ethernet interface. The only difference is that the **`DEVICE=`** directive must be **`bond<N>`** , replacing <*N*> with the number for the interface.

The following is a sample channel bonding configuration file, **`ifcfg-bond0`**:

```
DEVICE=bond0
IPADDR=192.168.1.1
NETMASK=255.255.255.0
ONBOOT=yes
BOOTPROTO=none
USERCTL=no
BONDING_OPTS="<bonding parameters separated by spaces>"
```

After the channel bonding interface is created, the network interfaces to be bound together must be configured by adding the **`MASTER=`** and **`SLAVE=`** directives to their configuration files. The configuration files for each of the channel-bonded interfaces can be nearly identical.

For example, if two Ethernet interfaces are being channel bonded, both **`eth0`** and **`eth1`** may look like the following example:

```
DEVICE=eth<N>
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
```

In this example, replace <*N*> with the numerical value for the interface.

For a channel bonding interface to be valid, the kernel module must be loaded. To ensure that the module is loaded when the channel bonding interface is brought up, add the following line to **`/etc/modprobe.conf`**:

```
alias bond<N> bonding
```

Replace <*N*> with the number of the interface, such as **`0`**.

Important

In Red Hat Enterprise Linux 5.10, interface-specific parameters for the bonding kernel module must be specified as a space-separated list in the **`BONDING_OPTS="`*`bonding parameters`*`"`** directive in the **`ifcfg-bond`*`N`*** interface file. Do **not** specify options for the bonding device in the **`/etc/modprobe.conf`** file.

The **`debug`** and **`max_bonds`** parameters are not interface specific and therefore, if required, should be specified in **`/etc/modprobe.conf`** as follows:

```
options bonding debug=1 max_bonds=1
```

However, the **`max_bonds`** parameter should **not** be set when using **`ifcfg-bond`*`N`*** files with the **`BONDING_OPTS`** directive as this directive will cause the network scripts to create the bond interfaces as required.

Note that any changes to **`/etc/modprobe.conf`** will not take effect until the module is next loaded. A running module must first be unloaded. For further instructions and advice on configuring the bonding module, as well as to view the list of bonding parameters, refer to Section 45.5.1, “The Channel Bonding Module”.

16.2.4. Alias and Clone Files

Two lesser-used types of interface configuration files are *alias* and *clone* files.

Alias interface configuration files, which are used to bind multiple addresses to a single interface, use the **`ifcfg-<`*`if-name`*`>:<`*`alias-value`*`>`** naming scheme.

For example, an **`ifcfg-eth0:0`** file could be configured to specify **`DEVICE=eth0:0`** and a static **`IP`** address of **`10.0.0.2`**, serving as an alias of an Ethernet interface already configured to receive its **`IP`** information via **`DHCP`** in **`ifcfg-eth0`**. Under this configuration, **`eth0`** is bound to a dynamic **`IP`** address, but the same physical network card can receive requests via the fixed, **`10.0.0.2`** **`IP`** address.

Warning

Alias interfaces do not support **`DHCP`**.

A clone interface configuration file should use the following naming convention: **`ifcfg-<`*`if-name`*`>-<`*`clone-name`*`>`** . While an alias file allows multiple addresses for an existing interface, a clone file is used to specify additional options for an interface. For example, a standard **`DHCP`** Ethernet interface called **`eth0`**, may look similar to this:

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp
```

Since the default value for the **`USERCTL`** directive is **`no`** if it is not specified, users cannot bring this interface up and down. To give users the ability to control the interface, create a clone by copying **`ifcfg-eth0`** to **`ifcfg-eth0-user`** and add the following line to **`ifcfg-eth0-user`**:

```
USERCTL=yes
```

This way a user can bring up the **`eth0`** interface using the **`/sbin/ifup eth0-user`** command because the configuration options from **`ifcfg-eth0`** and **`ifcfg-eth0-user`** are combined. While this is a very basic example, this method can be used with a variety of options and interfaces.

The easiest way to create alias and clone interface configuration files is to use the graphical **Network Administration Tool**. For more information on using this tool, refer to Chapter 17, *Network Configuration*.

16.2.5. Dialup Interfaces

If you are connecting to the Internet via a dialup connection, a configuration file is necessary for the interface.

PPP interface files are named using the following format:

`ifcfg-ppp<X>`

where *<X>* is a unique number corresponding to a specific interface.

The PPP interface configuration file is created automatically when **`wvdial`**, the **Network Administration Tool** or **Kppp** is used to create a dialup account. It is also possible to create and edit this file manually.

The following is a typical **`ifcfg-ppp0`** file:

```
DEVICE=ppp0
NAME=test
WVDIALSECT=test
MODEMPORT=/dev/modem
LINESPEED=115200
PAPNAME=test
USERCTL=true
ONBOOT=no
PERSIST=no
DEFROUTE=yes
PEERDNS=yes
DEMAND=no
IDLETIMEOUT=600
```

Serial Line Internet Protocol (SLIP) is another dialup interface, although it is used less frequently. SLIP files have interface configuration file names such as **`ifcfg-sl0`**.

Other options that may be used in these files include:

`DEFROUTE=<answer>`

where **`<answer>`** is one of the following:

- **`yes`** — Set this interface as the default route.
- **`no`** — Do not set this interface as the default route.

`DEMAND=<answer>`

where **`<answer>`** is one of the following:

- **`yes`** — This interface allows **`pppd`** to initiate a connection when someone attempts to use it.
- **`no`** — A connection must be manually established for this interface.

`IDLETIMEOUT=<`*`value`*`>`

where **`<`*`value`*`>`** is the number of seconds of idle activity before the interface disconnects itself.

`INITSTRING=<`*`string`*`>`

where **`<`*`string`*`>`** is the initialization string passed to the modem device. This option is primarily used in conjunction with SLIP interfaces.

`LINESPEED=<`*`value`*`>`

where **`<`*`value`*`>`** is the baud rate of the device. Possible standard values include **`57600`**, **`38400`**, **`19200`**, and **`9600`**.

`MODEMPORT=<`*`device`*`>`

where **`<`*`device`*`>`** is the name of the serial device that is used to establish the connection for the interface.

`MTU=<`*`value`*`>`

where **`<`*`value`*`>`** is the *Maximum Transfer Unit (MTU)* setting for the interface. The MTU refers to the largest number of bytes of data a frame can carry, not counting its header information. In some dialup situations, setting this to a value of **`576`** results in fewer packets dropped and a slight improvement to the throughput for a connection.

`NAME=<`*`name`*`>`

where **`<`*`name`*`>`** is the reference to the title given to a collection of dialup connection configurations

`PAPNAME=<`*`name`*`>`

where **`<`*`name`*`>`** is the username given during the *Password Authentication Protocol (PAP)* exchange that occurs to allow connections to a remote system.

`PERSIST=<`*`answer`*`>`

where **`<`*`answer`*`>`** is one of the following:

- **`yes`** — This interface should be kept active at all times, even if deactivated after a modem hang up.
- **`no`** — This interface should not be kept active at all times.

`REMIP=<`*`address`*`>`

where **`<`*`address`*`>`** is the **`IP`** address of the remote system. This is usually left unspecified.

`WVDIALSECT=<`*`name`*`>`

where **`<`*`name`*`>`** associates this interface with a dialer configuration in **`/etc/wvdial.conf`**. This file contains the phone number to be dialed and other important information for the interface.

16.2.6. Other Interfaces

Other common interface configuration files include the following:

`ifcfg-lo`

A local *loopback interface* is often used in testing, as well as being used in a variety of applications that require an **IP** address pointing back to the same system. Any data sent to the loopback device is immediately returned to the host's network layer.

Warning

The loopback interface script, **/etc/sysconfig/network-scripts/ifcfg-lo**, should never be edited manually. Doing so can prevent the system from operating correctly.

ifcfg-irlan0

An *infrared interface* allows information between devices, such as a laptop and a printer, to flow over an infrared link. This works in a similar way to an Ethernet device except that it commonly occurs over a peer-to-peer connection.

ifcfg-plip0

A *Parallel Line Interface Protocol (PLIP)* connection works much the same way as an Ethernet device, except that it utilizes a parallel port.

ifcfg-tr0

Token Ring topologies are not as common on *Local Area Networks* (*LANs*) as they once were, having been eclipsed by Ethernet.

16.3. Interface Control Scripts

The interface control scripts activate and deactivate system interfaces. There are two primary interface control scripts that call on control scripts located in the **/etc/sysconfig/network-scripts/** directory: **/sbin/ifdown** and **/sbin/ifup**.

The **ifup** and **ifdown** interface scripts are symbolic links to scripts in the **/sbin/** directory. When either of these scripts are called, they require the value of the interface to be specified, such as:

```
ifup eth0
```

Warning

The **ifup** and **ifdown** interface scripts are the only scripts that the user should use to bring up and take down network interfaces.

The following scripts are described for reference purposes only.

Two files used to perform a variety of network initialization tasks during the process of bringing up a network interface are **/etc/rc.d/init.d/functions** and **/etc/sysconfig/network-scripts/network-functions**. Refer to Section 16.6, “Network Function Files” for more information.

After verifying that an interface has been specified and that the user executing the request is allowed to control the interface, the correct script brings the interface up or down. The following are common interface control scripts found within the **/etc/sysconfig/network-scripts/** directory:

`ifup-aliases`

Configures **`IP`** aliases from interface configuration files when more than one **`IP`** address is associated with an interface.

`ifup-ippp` and `ifdown-ippp`

Brings ISDN interfaces up and down.

`ifup-ipsec` and `ifdown-ipsec`

Brings IPsec interfaces up and down.

`ifup-ipv6` and `ifdown-ipv6`

Brings **`IPv6`** interfaces up and down.

`ifup-ipx`

Brings up an IPX interface.

`ifup-plip`

Brings up a PLIP interface.

`ifup-plusb`

Brings up a USB interface for network connections.

`ifup-post` and `ifdown-post`

Contains commands to be executed after an interface is brought up or down.

`ifup-ppp` and `ifdown-ppp`

Brings a PPP interface up or down.

`ifup-routes`

Adds static routes for a device as its interface is brought up.

`ifdown-sit` and `ifup-sit`

Contains function calls related to bringing up and down an **`IPv6`** tunnel within an **`IPv4`** connection.

`ifup-sl` and `ifdown-sl`

Brings a SLIP interface up or down.

`ifup-wireless`

Brings up a wireless interface.

Warning

Removing or modifying any scripts in the **`/etc/sysconfig/network-scripts/`** directory can cause interface connections to act irregularly or fail. Only advanced users should modify scripts related to a network interface.

The easiest way to manipulate all network scripts simultaneously is to use the **`/sbin/service`** command on the network service (**`/etc/rc.d/init.d/network`**), as illustrated the following command:

```
service network <action>
```

Here, *<action>* can be either **`start`**, **`stop`**, or **`restart`**.

To view a list of configured devices and currently active network interfaces, use the following command:

```
service network status
```

16.4. Static Routes and the Default Gateway

Static routes are for traffic that must not, or should not, go through the default gateway. Routing is often handled by devices on the network dedicated to routing (although any device can be configured to perform routing). Therefore, it is often not necessary to configure static routes on Red Hat Enterprise Linux servers or clients. Exceptions include traffic that must pass through an encrypted VPN tunnel or traffic that should take a specific route for reasons of cost or security. The default gateway is for any and all traffic which is not destined for the local network and for which no preferred route is specified in the routing table. The default gateway is traditionally a dedicated network router.

Configuring Static Routes Using the Command Line

If static routes are required, they can be added to the routing table by means of the **`ip route add`** command and removed using the **`ip route del`** command. The more frequently used **`ip route`** commands take the following form:

```
ip route [ add | del | change | append | replace ] destination-address
```

See the **`ip-route(8)`** man page for more details on the options and formats.

Use the **`ip route`** command without options to display the **`IP`** routing table. For example:

```
~]$ ip route
default via 192.168.122.1 dev eth0  proto static  metric 1024
192.168.122.0/24 dev ens9  proto kernel  scope link  src 192.168.122.107
192.168.122.0/24 dev eth0  proto kernel  scope link  src 192.168.122.126
```

To add a static route to a host address, in other words to a single **`IP`** address, issue a command as **`root`**:

```
~]# ip route add 192.0.2.1 via 10.0.0.1 [dev ifname]
```

Where *192.0.2.1* is the **`IP`** address of the host in dotted decimal notation, *10.0.0.1* is the next hop address and *ifname* is the exit interface leading to the next hop.

To add a static route to a network, in other words to an **`IP`** address representing a range of **`IP`** addresses, issue the following command as **`root`**:

```
~]# ip route add 192.0.2.0/24 via 10.0.0.1 [dev ifname]
```

where *192.0.2.0* is the **IP** address of the destination network in dotted decimal notation and */24* is the network prefix. The network prefix is the number of enabled bits in the subnet mask. This format of network address slash network prefix length is sometimes referred to as *classless inter-domain routing* (CIDR) notation.

Static route configuration can be stored per-interface in a **/etc/sysconfig/network-scripts/route-*interface*** file. For example, static routes for the eth0 interface would be stored in the **/etc/sysconfig/network-scripts/route-eth0** file. The **route-*interface*** file has two formats: **ip** command arguments and network/netmask directives. These are described below.

See the **ip-route(8)** man page for more information on the **ip route** command.

Configuring The Default Gateway

The default gateway is determined by the network scripts which parse the **/etc/sysconfig/network** file first and then the network interface **ifcfg** files for interfaces that are "up". The **ifcfg** files are parsed in numerically ascending order, and the last GATEWAY directive to be read is used to compose a default route in the routing table.

The default route can thus be indicated by means of the GATEWAY directive and can be specified either globally or in interface-specific configuration files. Specifying the gateway globally has certain advantages in static networking environments, especially if more than one network interface is present. It can make fault finding simpler if applied consistently. There is also the GATEWAYDEV directive, which is a global option. If multiple devices specify GATEWAY, and one interface uses the GATEWAYDEV directive, that directive will take precedence. This option is not recommend as it can have unexpected consequences if an interface goes down and it can complicate fault finding.

Global default gateway configuration is stored in the **/etc/sysconfig/network** file. This file specifies gateway and host information for all network interfaces. For more information about this file and the directives it accepts, refer to Section 32.1.21, "**/etc/sysconfig/network**".

16.5. Configuring Static Routes in ifcfg files

Static routes set using **ip** commands at the command prompt will be lost if the system is shutdown or restarted. To configure static routes to be persistent after a system restart, they must be placed in per-interface configuration files in the **/etc/sysconfig/network-scripts/** directory. The file name should be of the format **route-*ifname***. There are two types of commands to use in the configuration files; **ip** commands as explained in Section 16.5.1, "Static Routes Using the IP Command Arguments Format" and the *Network/Netmask* format as explained in Section 16.5.2, "Network/Netmask Directives Format".

16.5.1. Static Routes Using the IP Command Arguments Format

If required in a per-interface configuration file, for example **/etc/sysconfig/network-scripts/route-eth0**, define a route to a default gateway on the first line. This is only required if the gateway is not set via **DHCP** and is not set globally in the **/etc/sysconfig/network** file:

```
default via 192.168.1.1 dev interface
```

where *192.168.1.1* is the **IP** address of the default gateway. The *interface* is the interface that is connected to, or can reach, the default gateway. The **dev** option can be omitted, it is optional. Note that this setting takes precedence over a setting in the **/etc/sysconfig/network** file.

If a route to a remote network is required, a static route can be specified as follows. Each line is parsed as an individual route:

```
10.10.10.0/24 via 192.168.1.1 [dev interface]
```

where *10.10.10.0/24* is the network address and prefix length of the remote or destination network. The address *192.168.1.1* is the **IP** address leading to the remote network. It is preferably the *next hop address* but the address of the exit interface will work. The "next hop" means the remote end of a link, for example a gateway or router. The **dev** option can be used to specify the exit interface *interface* but it is not required. Add as many static routes as required.

The following is an example of a **route-*interface*** file using the **ip** command arguments format. The default gateway is **192.168.0.1**, interface eth0 and a leased line or WAN connection is available at **192.168.0.10**. The two static routes are for reaching the **10.10.10.0/24** network and the **172.16.1.10/32** host:

```
default via 192.168.0.1 dev eth0
10.10.10.0/24 via 192.168.0.10 dev eth0
172.16.1.10/32 via 192.168.0.10 dev eth0
```

In the above example, packets going to the local **192.168.0.0/24** network will be directed out the interface attached to that network. Packets going to the **10.10.10.0/24** network and **172.16.1.10/32** host will be directed to **192.168.0.10**. Packets to unknown, remote, networks will use the default gateway therefore static routes should only be configured for remote networks or hosts if the default route is not suitable. Remote in this context means any networks or hosts that are not directly attached to the system.

Specifying an exit interface is optional. It can be useful if you want to force traffic out of a specific interface. For example, in the case of a VPN, you can force traffic to a remote network to pass through a tun0 interface even when the interface is in a different subnet to the destination network.

Important

If the default gateway is already assigned from **DHCP**, the **IP** command arguments format can cause one of two errors during start-up, or when bringing up an interface from the down state using the **ifup** command: "RTNETLINK answers: File exists" or 'Error: either "to" is a duplicate, or "*X.X.X.X*" is a garbage.', where *X.X.X.X* is the gateway, or a different **IP** address. These errors can also occur if you have another route to another network using the default gateway. Both of these errors are safe to ignore.

16.5.2. Network/Netmask Directives Format

You can also use the network/netmask directives format for **route-*interface*** files. The following is a template for the network/netmask format, with instructions following afterwards:

```
ADDRESS0=10.10.10.0
NETMASK0=255.255.255.0
GATEWAY0=192.168.1.1
```

- **ADDRESS0=*10.10.10.0*** is the network address of the remote network or host to be reached.
- **NETMASK0=*255.255.255.0*** is the netmask for the network address defined with **ADDRESS0=*10.10.10.0***.
- **GATEWAY0=*192.168.1.1*** is the default gateway, or an **IP** address that can be used to reach **ADDRESS0=*10.10.10.0***

The following is an example of a **`route-*interface*`** file using the network/netmask directives format. The default gateway is **`192.168.0.1`** but a leased line or WAN connection is available at **`192.168.0.10`**. The two static routes are for reaching the **`10.10.10.0/24`** and **`172.16.1.0/24`** networks:

```
ADDRESS0=10.10.10.0
NETMASK0=255.255.255.0
GATEWAY0=192.168.0.10
ADDRESS1=172.16.1.10
NETMASK1=255.255.255.0
GATEWAY1=192.168.0.10
```

Subsequent static routes must be numbered sequentially, and must not skip any values. For example, **`ADDRESS0`**, **`ADDRESS1`**, **`ADDRESS2`**, and so on.

16.6. Network Function Files

Red Hat Enterprise Linux makes use of several files that contain important common functions used to bring interfaces up and down. Rather than forcing each interface control file to contain these functions, they are grouped together in a few files that are called upon when necessary.

The **`/etc/sysconfig/network-scripts/network-functions`** file contains the most commonly used **`IPv4`** functions, which are useful to many interface control scripts. These functions include contacting running programs that have requested information about changes in the status of an interface, setting host names, finding a gateway device, verifying whether or not a particular device is down, and adding a default route.

As the functions required for **`IPv6`** interfaces are different from **`IPv4`** interfaces, a **`/etc/sysconfig/network-scripts/network-functions-ipv6`** file exists specifically to hold this information. The functions in this file configure and delete static **`IPv6`** routes, create and remove tunnels, add and remove **`IPv6`** addresses to an interface, and test for the existence of an **`IPv6`** address on an interface.

16.7. Additional Resources

The following are resources which explain more about network interfaces.

16.7.1. Installed Documentation

`/usr/share/doc/initscripts-<*version*>/sysconfig.txt`

A guide to available options for network configuration files, including **`IPv6`** options not covered in this chapter.

`/usr/share/doc/iproute-<*version*>/ip-cref.ps`

This file contains a wealth of information about the **`ip`** command, which can be used to manipulate routing tables, among other things. Use the **ggv** or **kghostview** application to view this file.

Chapter 17. Network Configuration

To communicate with each other, computers must have a network connection. This is accomplished by having the operating system recognize an interface card (such as Ethernet, ISDN modem, or token ring) and configuring the interface to connect to the network.

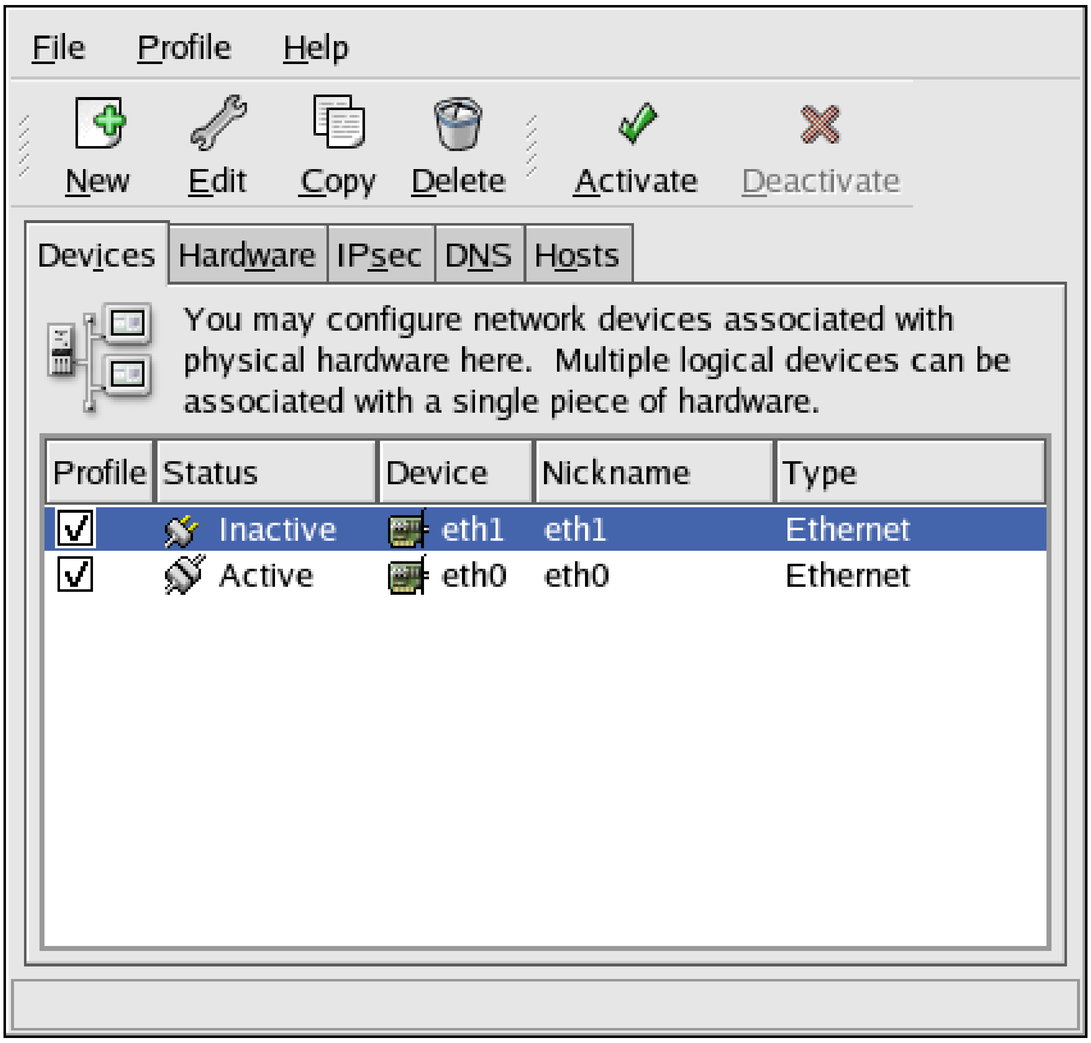
The **Network Administration Tool** can be used to configure the following types of network interfaces:

- Ethernet
- ISDN
- modem
- xDSL
- token ring
- CIPE
- wireless devices

It can also be used to configure IPsec connections, manage DNS settings, and manage the **`/etc/hosts`** file used to store additional hostnames and IP address combinations.

To use the **Network Administration Tool**, you must have root privileges. To start the application, go to the Applications (the main menu on the panel) > **System Settings** > **Network**, or type the command **`system-config-network`** at a shell prompt (for example, in an **XTerm** or a **GNOME terminal**). If you type the command, the graphical version is displayed if **X** is running; otherwise, the text-based version is displayed.

To use the command line version, execute the command **`system-config-network-cmd --help`** as root to view all of the options.



You may configure network devices associated with physical hardware here. Multiple logical devices can be associated with a single piece of hardware.

Profile	Status	Device	Nickname	Type
✓	Inactive	eth1	eth1	Ethernet
✓	Active	eth0	eth0	Ethernet

Figure 17.1. Network Administration Tool

Note

Use the Red Hat Hardware Compatibility List (http://hardware.redhat.com/hcl/) to determine if Red Hat Enterprise Linux supports your hardware device.

17.1. Overview

To configure a network connection with the **Network Administration Tool**, perform the following steps:

1. Add a network device associated with the physical hardware device.
2. Add the physical hardware device to the hardware list, if it does not already exist.
3. Configure the hostname and DNS settings.
4. Configure any hosts that cannot be looked up through DNS.

This chapter discusses each of these steps for each type of network connection.

17.2. Establishing an Ethernet Connection

To establish an Ethernet connection, you need a network interface card (NIC), a network cable (usually a CAT5 cable), and a network to connect to. Different networks are configured to use different network speeds; make sure your NIC is compatible with the network to which you want to connect.

To add an Ethernet connection, follow these steps:

1. Click the **`Devices`** tab.

2. Click the **`New`** button on the toolbar.

3. Select **`Ethernet connection`** from the **`Device Type`** list, and click **`Forward`**.

4. If you have already added the network interface card to the hardware list, select it from the **`Ethernet card`** list. Otherwise, select **`Other Ethernet Card`** to add the hardware device.

   Note

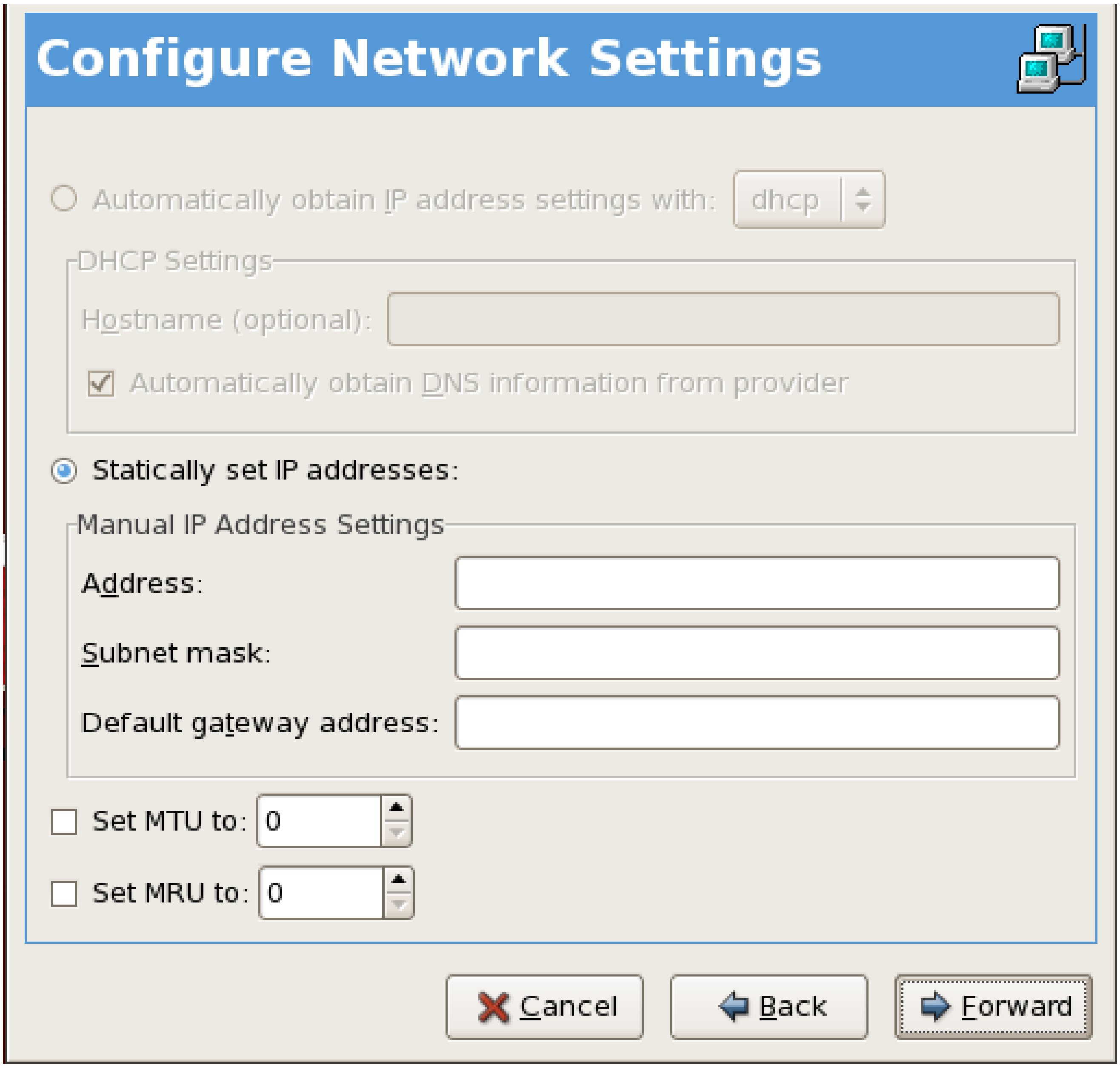
   The installation program detects supported Ethernet devices and prompts you to configure them. If you configured any Ethernet devices during the installation, they are displayed in the hardware list on the **`Hardware`** tab.

5. If you selected **`Other Ethernet Card`**, the **`Select Ethernet Adapter`** window appears. Select the manufacturer and model of the Ethernet card. Select the device name. If this is the system's first Ethernet card, select **`eth0`** as the device name; if this is the second Ethernet card, select **`eth1`** (and so on). The **Network Administration Tool** also allows you to configure the resources for the NIC. Click **`Forward`** to continue.

6. In the **`Configure Network Settings`** window shown in Figure 17.2, “Ethernet Settings”, choose between DHCP and a static IP address. If the device receives a different IP address each time the network is started, do not specify a hostname.

7. Do not specify a value for the **`Set MTU to`** or **`Set MRU to`** fields. *MTU* stands for Maximum Transmission Unit and *MRU* for Maximum Receive Unit; the network configuration tool will choose appropriate values for both of these parameters. Click **`Forward`** to continue.

8. Click **`Apply`** on the **`Create Ethernet Device`** page.



Configure Network Settings

( ) Automatically obtain IP address settings with: dhcp

DHCP Settings

Hostname (optional):

[✓] Automatically obtain DNS information from provider

(●) Statically set IP addresses:

Manual IP Address Settings

Address:

Subnet mask:

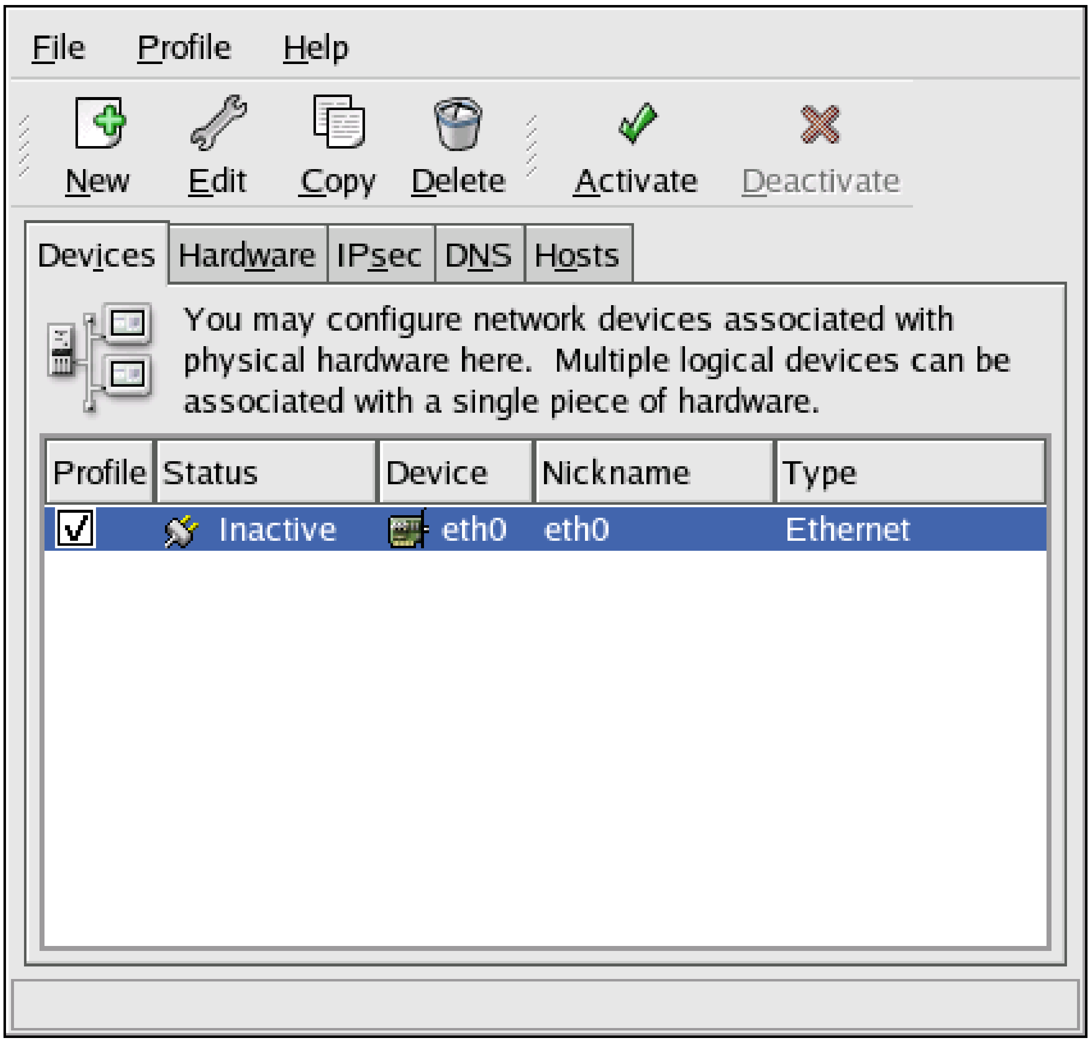
Default gateway address:

[ ] Set MTU to: 0

[ ] Set MRU to: 0

Figure 17.2. Ethernet Settings

After configuring the Ethernet device, it appears in the device list as shown in Figure 17.3, “Ethernet Device”.



You may configure network devices associated with physical hardware here. Multiple logical devices can be associated with a single piece of hardware.

Profile	Status	Device	Nickname	Type
✓	Inactive	eth0	eth0	Ethernet

Figure 17.3. Ethernet Device

Be sure to select **File** > **Save** to save the changes.

After adding the Ethernet device, you can edit its configuration by selecting the device from the device list and clicking **`Edit`**. For example, when the device is added, it is configured to start at boot time by default. To change this setting, select to edit the device, modify the **`Activate device when computer starts`** value, and save the changes.

When the device is added, it is not activated immediately, as seen by its **`Inactive`** status. To activate the device, select it from the device list, and click the **`Activate`** button. If the system is configured to activate the device when the computer starts (the default), this step does not have to be performed again.

If you associate more than one device with an Ethernet card, the subsequent devices are *device aliases*. A device alias allows you to setup multiple virtual devices for one physical device, thus giving the one physical device more than one IP address. For example, you can configure an eth1 device and an eth1:1 device. For details, refer to Section 17.11, “Device Aliases”.

17.3. Establishing an ISDN Connection

An ISDN connection is an Internet connection established with a ISDN modem card through a special phone line installed by the phone company. ISDN connections are popular in Europe.

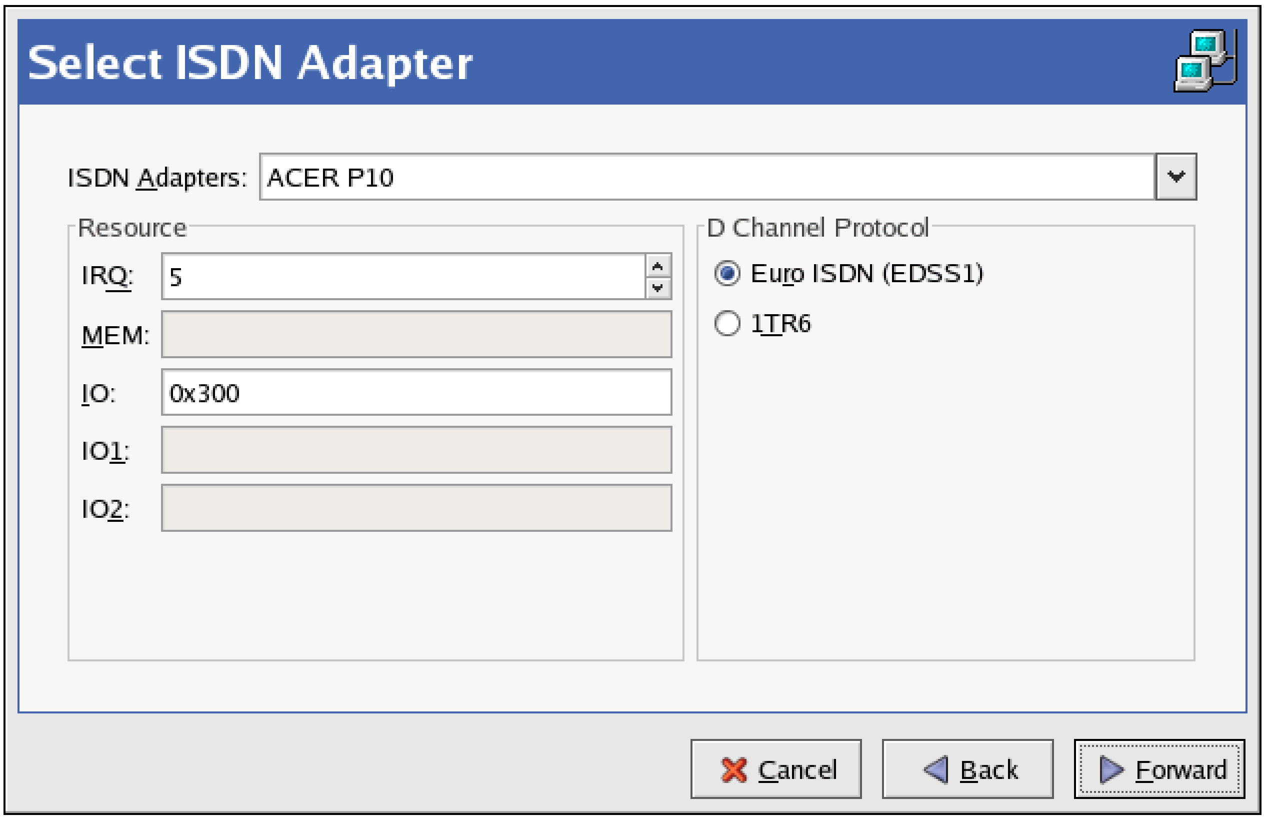
To add an ISDN connection, follow these steps:

1. Click the **`Devices`** tab.

2. Click the **New** button on the toolbar.

3. Select **ISDN connection** from the **Device Type** list, and click **Forward**.

4. Select the ISDN adapter from the pulldown menu. Then configure the resources and D channel protocol for the adapter. Click **Forward** to continue.


Select ISDN Adapter

ISDN Adapters: ACER P10

Resource

IRQ: 5

MEM:

IO: 0x300

IO1:

IO2:

D Channel Protocol

[✓ Euro ISDN (EDSS1)] 1TR6

Figure 17.4. ISDN Settings

5. If your Internet Service Provider (ISP) is in the pre-configured list, select it. Otherwise, enter the required information about your ISP account. If you do not know the values, contact your ISP. Click **Forward**.

6. In the **IP Settings** window, select the **Encapsulation Mode** and whether to obtain an IP address automatically or to set a static IP instead. Click **Forward** when finished.

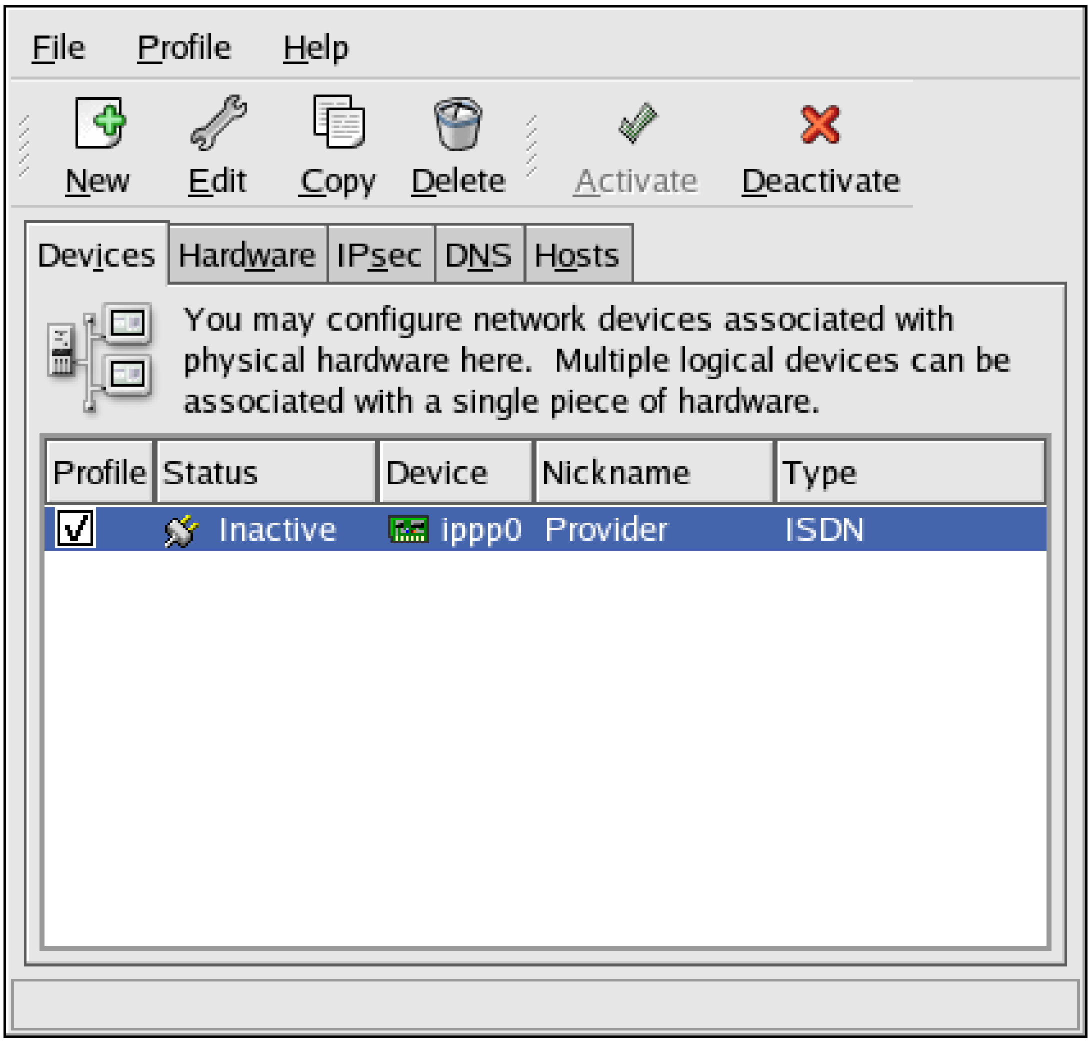
7. On the **Create Dialup Connection** page, click **Apply**.

After configuring the ISDN device, it appears in the device list as a device with type **ISDN** as shown in Figure 17.5, “ISDN Device”.

Be sure to select **File** > **Save** to save the changes.

After adding the ISDN device, you can edit its configuration by selecting the device from the device list and clicking **Edit**. For example, when the device is added, it is configured not to start at boot time by default. Edit its configuration to modify this setting. Compression, PPP options, login name, password, and more can be changed.

When the device is added, it is not activated immediately, as seen by its **Inactive** status. To activate the device, select it from the device list, and click the **Activate** button. If the system is configured to activate the device when the computer starts (the default), this step does not have to be performed again.



You may configure network devices associated with physical hardware here. Multiple logical devices can be associated with a single piece of hardware.

Profile	Status	Device	Nickname	Type
✓	Inactive	ippp0	Provider	ISDN

Figure 17.5. ISDN Device

17.4. Establishing a Modem Connection

A modem can be used to configure an Internet connection over an active phone line. An Internet Service Provider (ISP) account (also called a dial-up account) is required.

To add a modem connection, follow these steps:

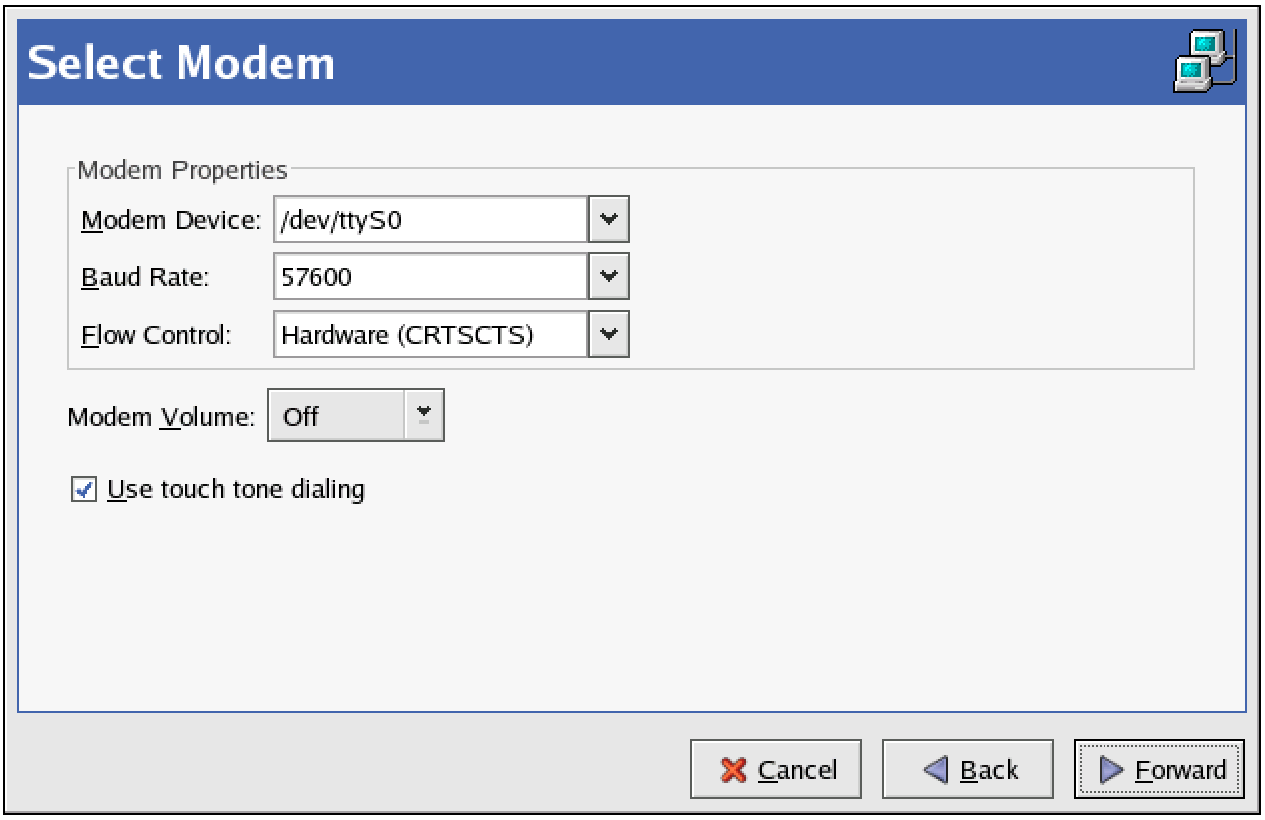
1. Click the **`Devices`** tab.

2. Click the **`New`** button on the toolbar.

3. Select **`Modem connection`** from the **`Device Type`** list, and click **`Forward`**.

4. If there is a modem already configured in the hardware list (on the **`Hardware`** tab), the **Network Administration Tool** assumes you want to use it to establish a modem connection. If there are no modems already configured, it tries to detect any modems in the system. This probe might take a while. If a modem is not found, a message is displayed to warn you that the settings shown are not values found from the probe.

5. After probing, the window in Figure 17.6, “Modem Settings” appears.



Modem Device: /dev/ttyS0
Baud Rate: 57600
Flow Control: Hardware (CRTSCTS)
Modem Volume: Off
✓ Use touch tone dialing

Figure 17.6. Modem Settings

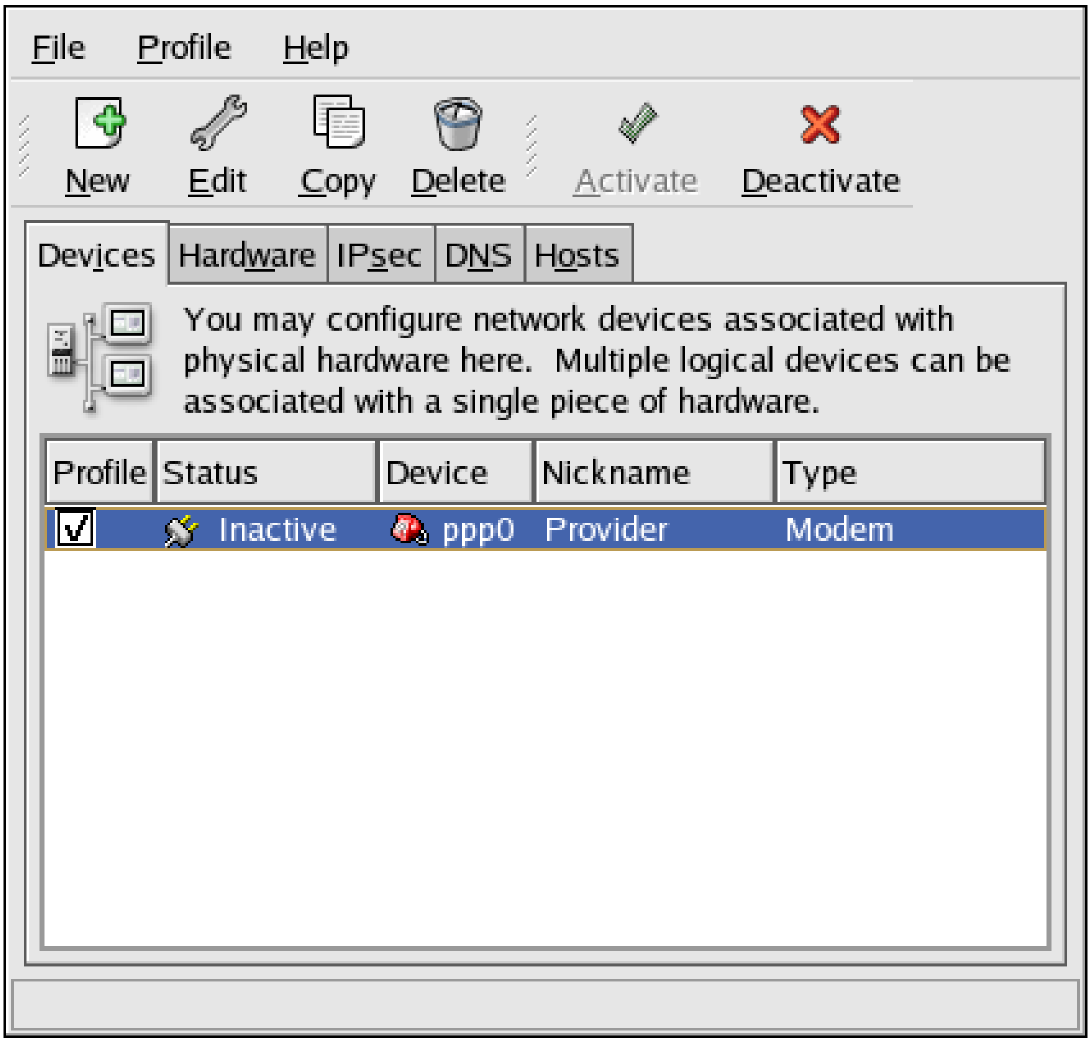
6. Configure the modem device, baud rate, flow control, and modem volume. If you do not know these values, accept the defaults if the modem was probed successfully. If you do not have touch tone dialing, uncheck the corresponding checkbox. Click **`Forward`**.

7. If your ISP is in the pre-configured list, select it. Otherwise, enter the required information about your ISP account. If you do not know these values, contact your ISP. Click **`Forward`**.

8. On the **`IP Settings`** page, select whether to obtain an IP address automatically or whether to set one statically. Click **`Forward`** when finished.

9. On the **`Create Dialup Connection`** page, click **`Apply`**.

After configuring the modem device, it appears in the device list with the type **`Modem`** as shown in Figure 17.7, "Modem Device".



You may configure network devices associated with physical hardware here. Multiple logical devices can be associated with a single piece of hardware.

Profile	Status	Device	Nickname	Type
✓	Inactive	ppp0	Provider	Modem

Figure 17.7. Modem Device

Be sure to select **File** > **Save** to save the changes.

After adding the modem device, you can edit its configuration by selecting the device from the device list and clicking **`Edit`**. For example, when the device is added, it is configured not to start at boot time by default. Edit its configuration to modify this setting. Compression, PPP options, login name, password, and more can also be changed.

When the device is added, it is not activated immediately, as seen by its **`Inactive`** status. To activate the device, select it from the device list, and click the **`Activate`** button. If the system is configured to activate the device when the computer starts (the default), this step does not have to be performed again.

17.5. Establishing an xDSL Connection

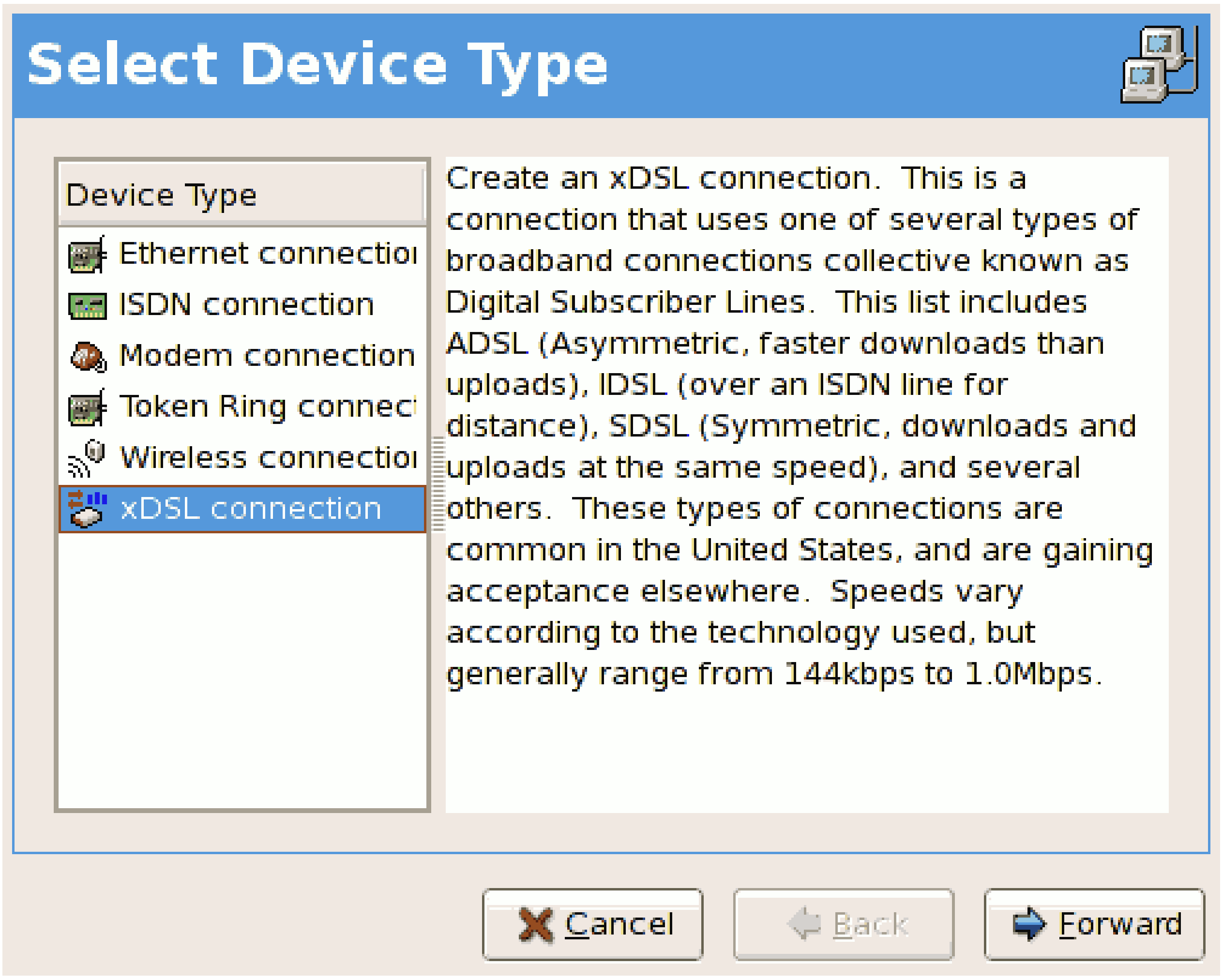
DSL stands for *Digital Subscriber Lines*. There are different types of DSL such as ADSL, IDSL, and SDSL. The **Network Administration Tool** uses the term *xDSL* to mean all types of DSL connections.

Some DSL providers require that the system is configured to obtain an IP address through DHCP with an Ethernet card. Some DSL providers require you to configure a PPPoE (Point-to-Point Protocol over Ethernet) connection with an Ethernet card. Ask your DSL provider which method to use.

If you are required to use DHCP, refer to Section 17.2, “Establishing an Ethernet Connection” to configure your Ethernet card.

If you are required to use PPPoE, follow these steps:

1. Click the **Devices** tab.
2. Click the **New** button.
3. Select **xDSL connection** from the **Device Type** list, and click **Forward** as shown in Figure 17.8, “Select Device Type”.



Select Device Type

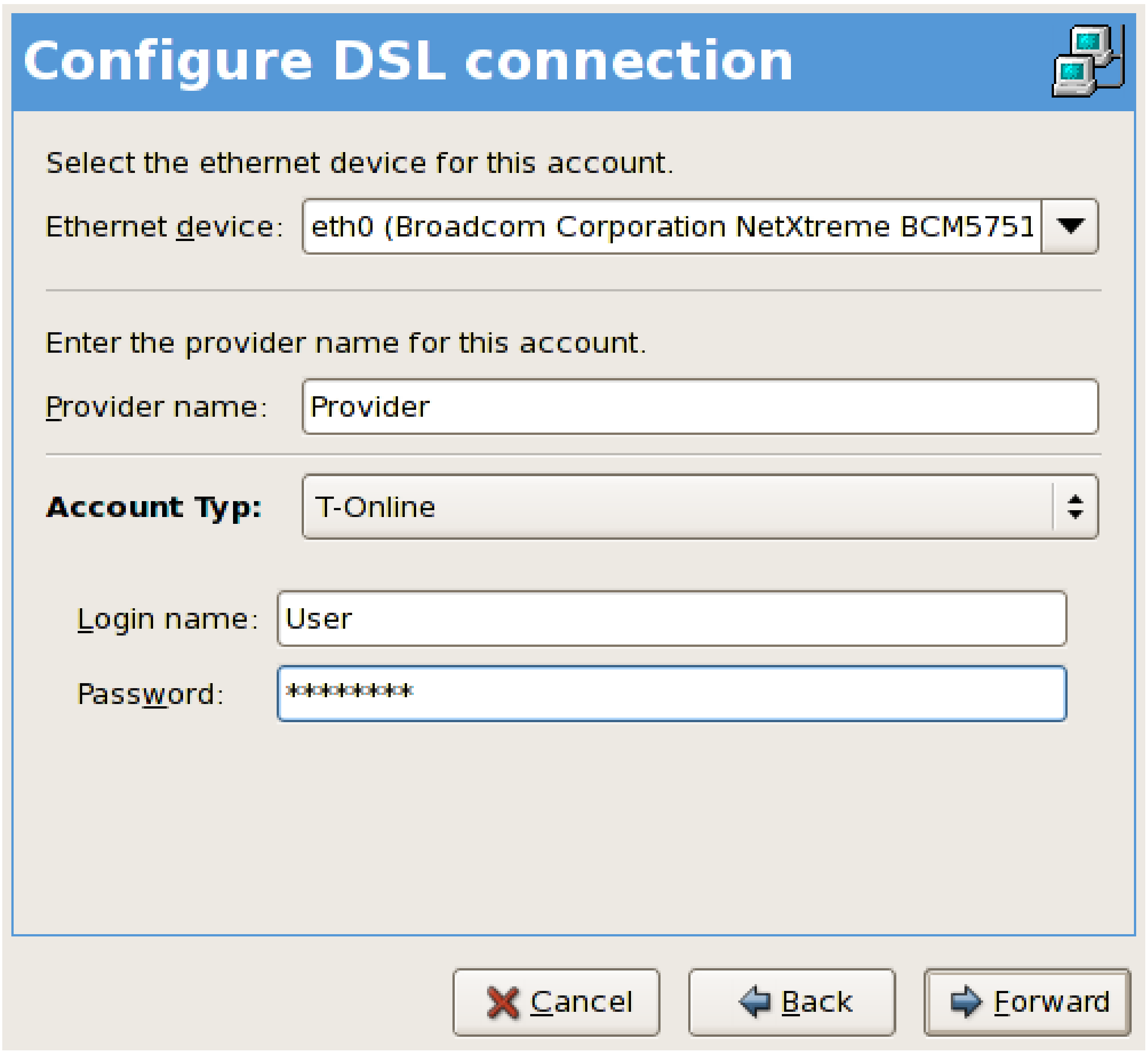
Create an xDSL connection. This is a connection that uses one of several types of broadband connections collective known as Digital Subscriber Lines. This list includes ADSL (Asymmetric, faster downloads than uploads), IDSL (over an ISDN line for distance), SDSL (Symmetric, downloads and uploads at the same speed), and several others. These types of connections are common in the United States, and are gaining acceptance elsewhere. Speeds vary according to the technology used, but generally range from 144kbps to 1.0Mbps.

Figure 17.8. Select Device Type

4. If your Ethernet card is in the hardware list, select the **Ethernet Device** from the pulldown menu from the page shown in Figure 17.9, “xDSL Settings”. Otherwise, the **Select Ethernet Adapter** window appears.

Note

The installation program detects supported Ethernet devices and prompts you to configure them. If you configured any Ethernet devices during the installation, they are displayed in the hardware list on the **Hardware** tab.


Configure DSL connection

Select the ethernet device for this account.

Ethernet device: eth0 (Broadcom Corporation NetXtreme BCM5751

Enter the provider name for this account.

Provider name: Provider

Account Typ: T-Online

Login name: User

Password: *********

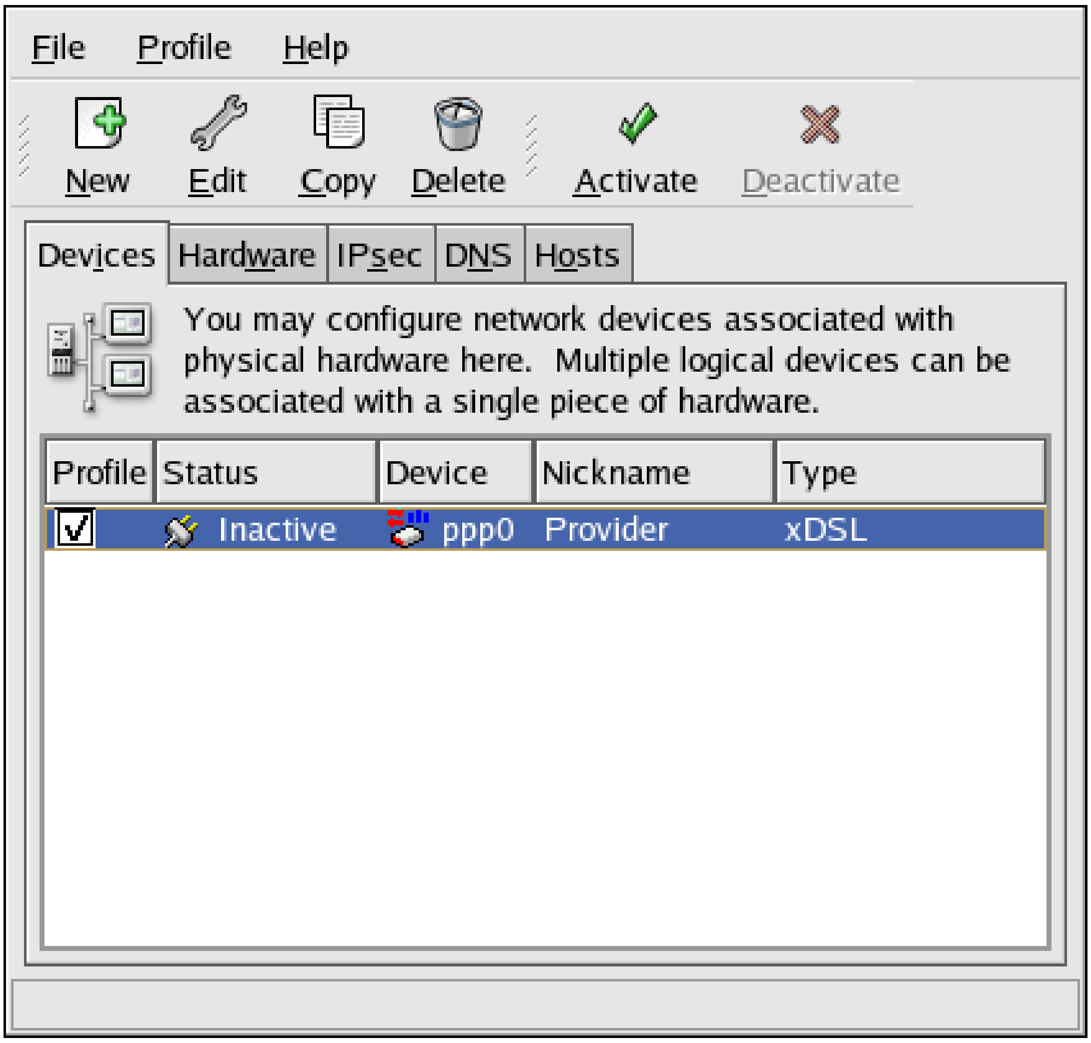
Figure 17.9. xDSL Settings

5. Enter the **`Provider Name`**, **`Login Name`**, and **`Password`**. If you are not setting up a T-Online account, select **`Normal`** from the **`Account Type`** pulldown menu.

   If you are setting up a T-Online account, select **`T-Online`** from the **`Account Type`** pulldown menu and enter any values in the **`Login name`** and **`Password`** field. You can further configure your T-Online account settings once the DSL connection has been fully configured (refer to Setting Up a T-Online Account).

6. Click the **`Forward`** to go to the **Create DSL Connection** menu. Check your settings and click **`Apply`** to finish.

7. After configuring the DSL connection, it appears in the device list as shown in Figure 17.10, “xDSL Device”.

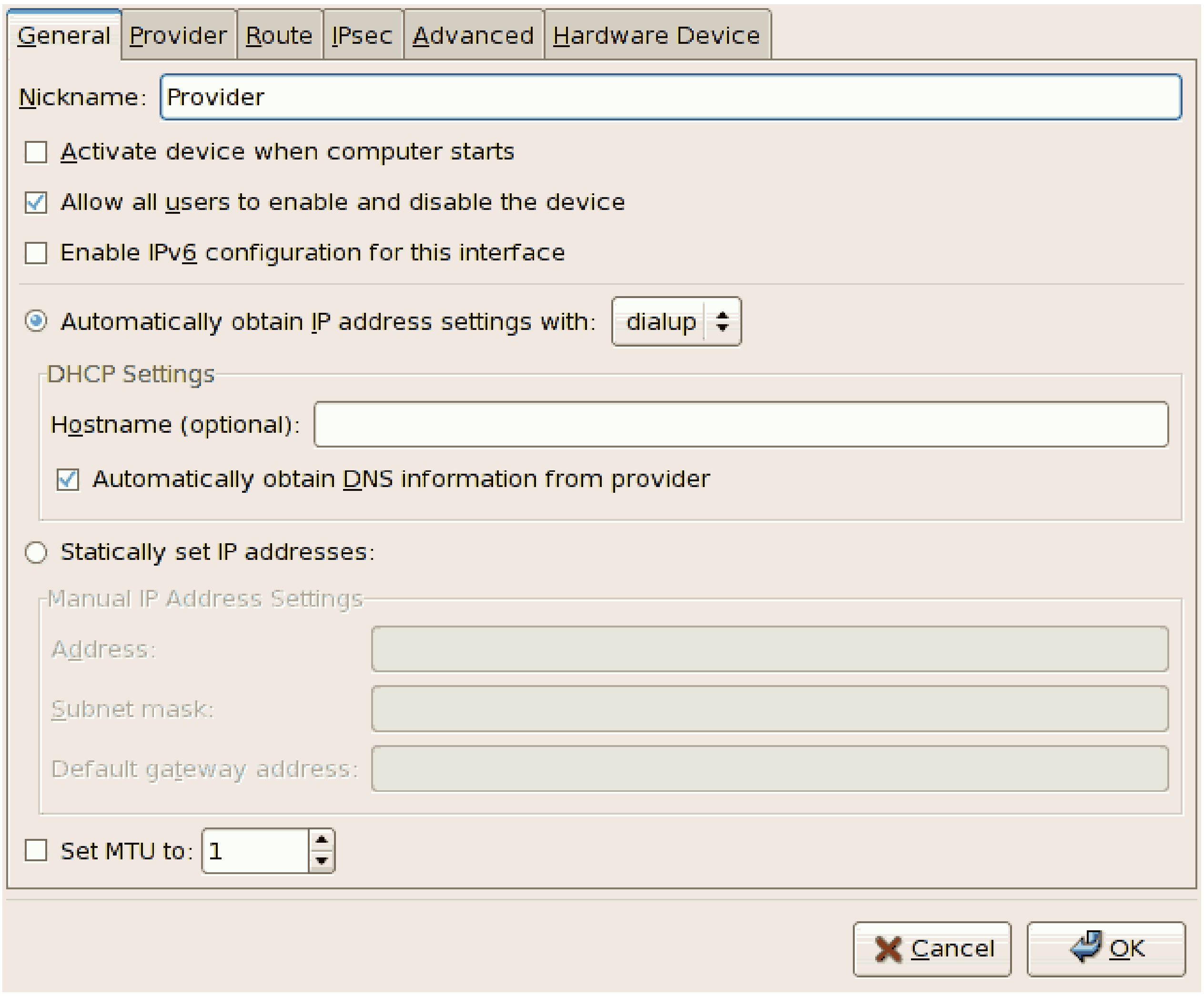


You may configure network devices associated with physical hardware here. Multiple logical devices can be associated with a single piece of hardware.

Profile	Status	Device	Nickname	Type
✓	Inactive	ppp0	Provider	xDSL

Figure 17.10. xDSL Device

8. After adding the xDSL connection, you can edit its configuration by selecting the device from the device list and clicking **`Edit`**.



Nickname: Provider

☐ Activate device when computer starts

☑ Allow all users to enable and disable the device

☐ Enable IPv6 configuration for this interface

◉ Automatically obtain IP address settings with: dialup

DHCP Settings

Hostname (optional):

☑ Automatically obtain DNS information from provider

○ Statically set IP addresses:

Manual IP Address Settings

Address:

Subnet mask:

Default gateway address:

☐ Set MTU to: 1

Figure 17.11. xDSL Configuration

For example, when the device is added, it is configured not to start at boot time by default. Edit its configuration to modify this setting. Click **`OK`** when finished.

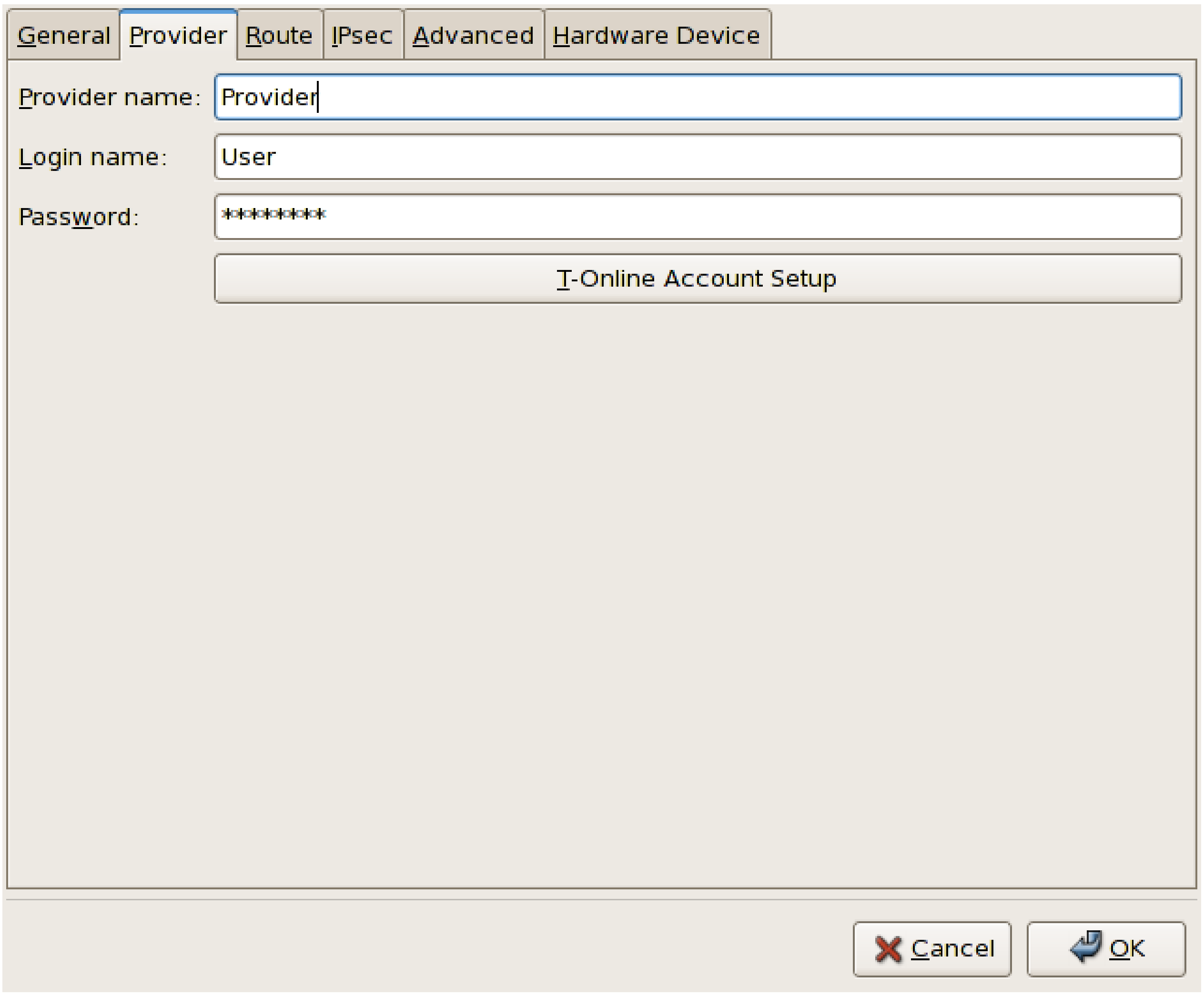
9. Once you are satisfied with your xDSL connection settings, select **File** > **Save** to save the changes.

Setting Up a T-Online Account

If you are setting up a T-Online Account, follow these additional steps:

1. Select the device from the device list and click **`Edit`**.

2. Select the **`Provider`** tab from the **xDSL Configuration** menu as shown in Figure 17.12, “xDSL Configuration - Provider Tab”.


General | Provider | Route | IPsec | Advanced | Hardware Device

Provider name: Provider

Login name: User

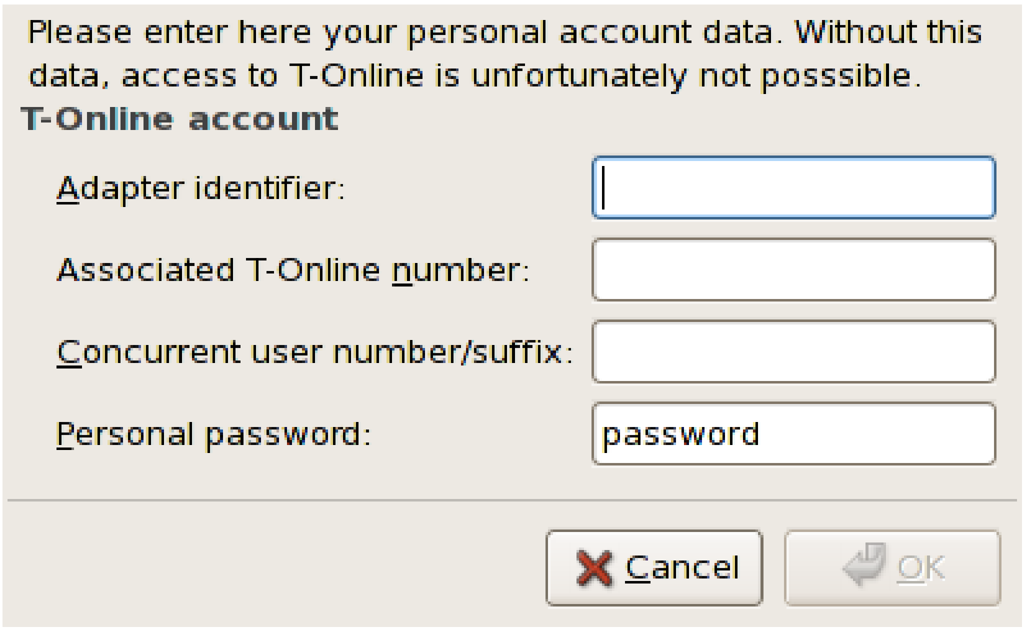
Password: *********

T-Online Account Setup

Cancel | OK

Figure 17.12. xDSL Configuration - Provider Tab

3. Click the **`T-Online Account Setup`** button. This will open the **Account Setup** window for your T-Online account as shown in Figure 17.13, “Account Setup”.


Please enter here your personal account data. Without this data, access to T-Online is unfortunately not posssible.

T-Online account

Adapter identifier:

Associated T-Online number:

Concurrent user number/suffix:

Personal password: password

Cancel | OK

Figure 17.13. Account Setup

4. Enter your **`Adapter identifier`**, **`Associated T-Online number`**, **`Concurrent user number/suffix`**, and **`Personal password.`**. Click **`OK`** when finished to close the **Account Setup** window.

5. On the **xDSL Configuration** window, click **`OK`**. Be sure to select **File** > **Save** from the **Network Administration Tool** to save the changes.

When the device is added, it is not activated immediately, as seen by its **`Inactive`** status. To activate the device, select it from the device list, and click the **`Activate`** button. If the system is configured to activate the device when the computer starts (the default), this step does not have to be performed again.

17.6. Establishing a Token Ring Connection

A *token ring network* is a network in which all the computers are connected in a circular pattern. A *token*, or a special network packet, travels around the token ring and allows computers to send information to each other.

Note

For more information on using token rings under Linux, refer to the *Linux Token Ring Project* website available at http://www.linuxtr.net/.

To add a token ring connection, follow these steps:

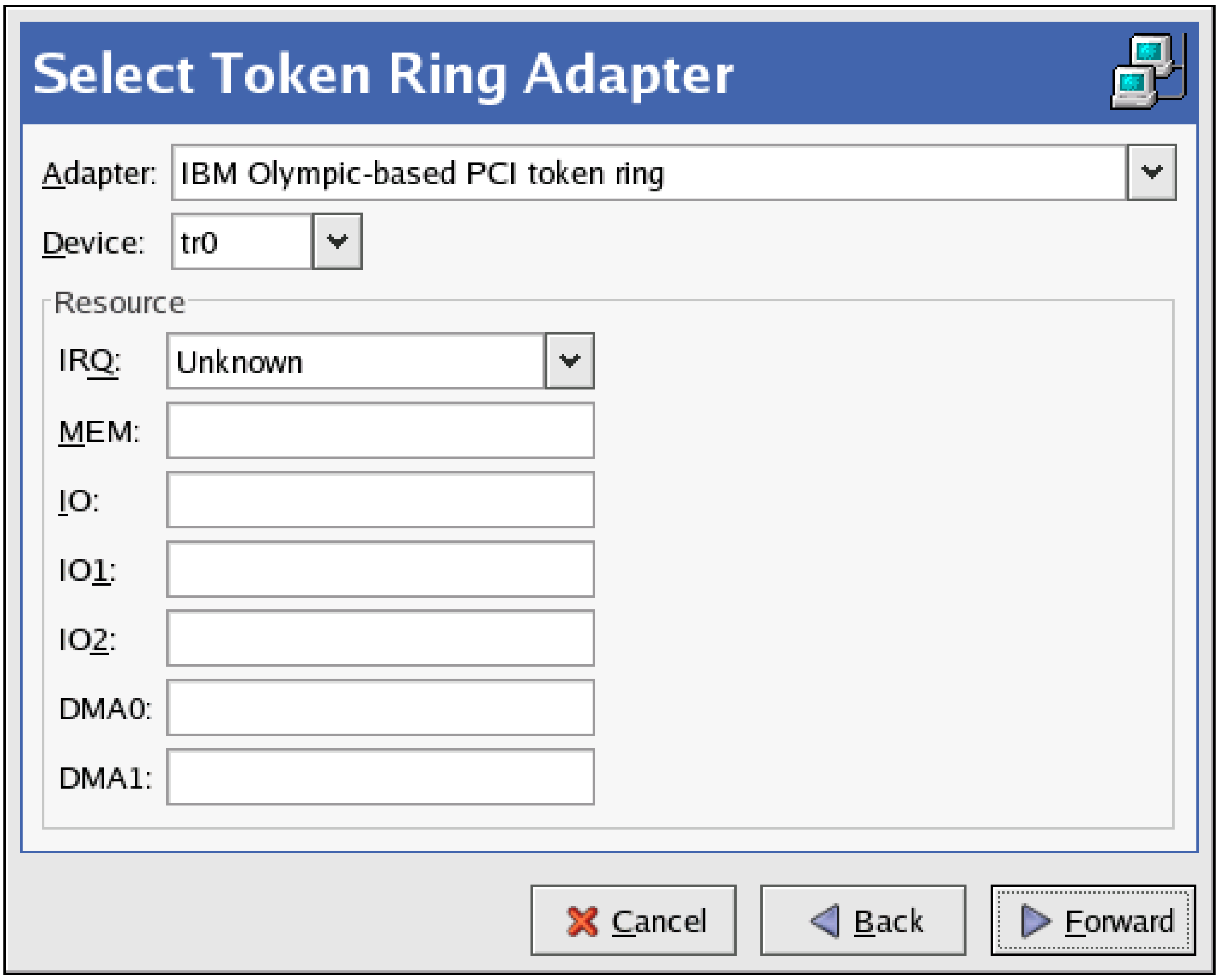
1. Click the **`Devices`** tab.

2. Click the **`New`** button on the toolbar.

3. Select **`Token Ring connection`** from the **`Device Type`** list and click **`Forward`**.

4. If you have already added the token ring card to the hardware list, select it from the **`Tokenring card`** list. Otherwise, select **`Other Tokenring Card`** to add the hardware device.

5. If you selected **`Other Tokenring Card`**, the **`Select Token Ring Adapter`** window as shown in Figure 17.14, “Token Ring Settings” appears. Select the manufacturer and model of the adapter. Select the device name. If this is the system's first token ring card, select **`tr0`**; if this is the second token ring card, select **`tr1`** (and so on). The **Network Administration Tool** also allows the user to configure the resources for the adapter. Click **`Forward`** to continue.


Select Token Ring Adapter

Adapter: IBM Olympic-based PCI token ring

Device: tr0

Resource

IRQ: Unknown

MEM:

IO:

IO1:

IO2:

DMA0:

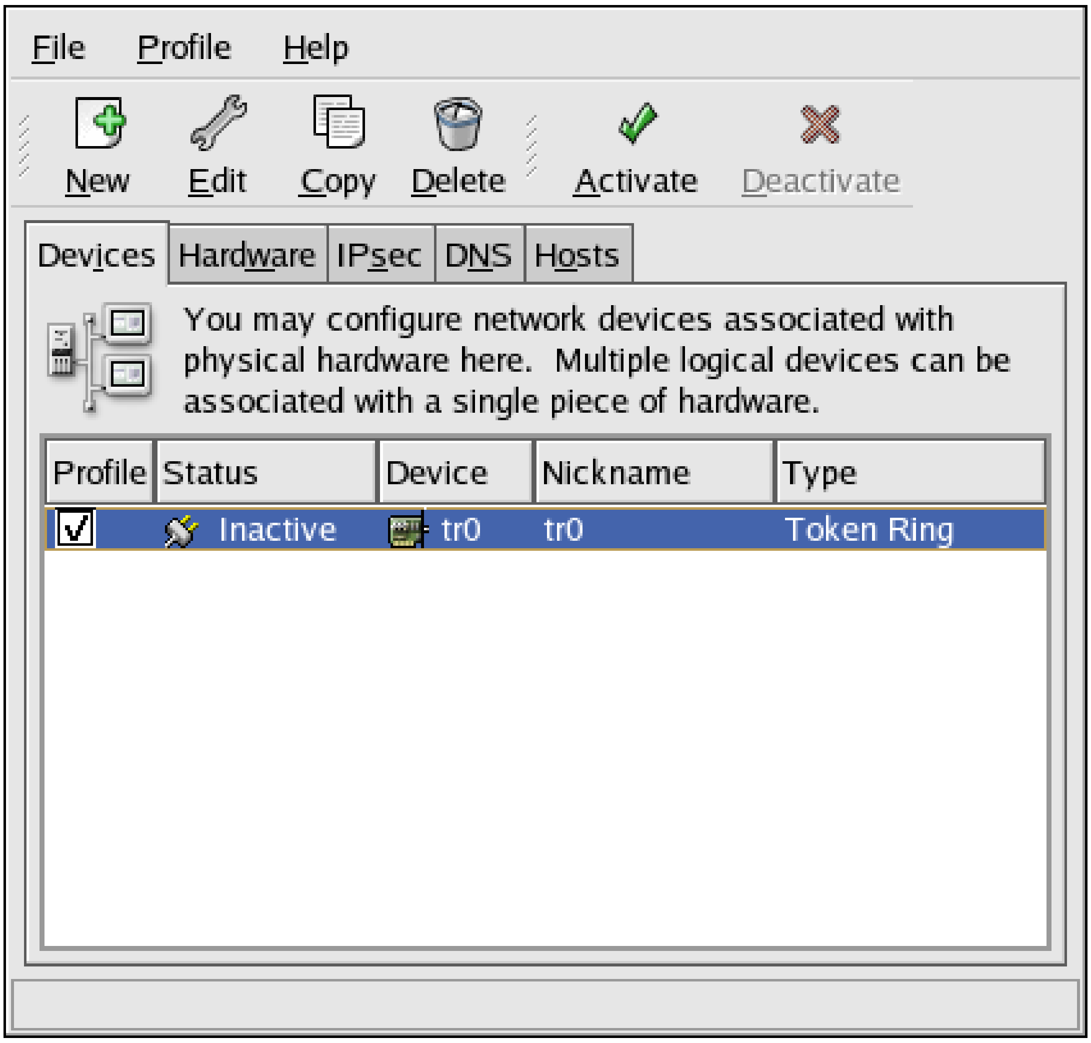
DMA1:

Figure 17.14. Token Ring Settings

6. On the **`Configure Network Settings`** page, choose between DHCP and static IP address. You may specify a hostname for the device. If the device receives a dynamic IP address each time the network is started, do not specify a hostname. Click **`Forward`** to continue.

7. Click **`Apply`** on the **`Create Tokenring Device`** page.

After configuring the token ring device, it appears in the device list as shown in Figure 17.15, “Token Ring Device”.



You may configure network devices associated with physical hardware here. Multiple logical devices can be associated with a single piece of hardware.

Profile	Status	Device	Nickname	Type
✓	Inactive	tr0	tr0	Token Ring

Figure 17.15. Token Ring Device

Be sure to select **File** > **Save** to save the changes.

After adding the device, you can edit its configuration by selecting the device from the device list and clicking **`Edit`**. For example, you can configure whether the device is started at boot time.

When the device is added, it is not activated immediately, as seen by its **`Inactive`** status. To activate the device, select it from the device list, and click the **`Activate`** button. If the system is configured to activate the device when the computer starts (the default), this step does not have to be performed again.

17.7. Establishing a Wireless Connection

Wireless Ethernet devices are becoming increasingly popular. The configuration is similar to the Ethernet configuration except that it allows you to configure settings such as the SSID and key for the wireless device.

To add a wireless Ethernet connection, follow these steps:

1. Click the **`Devices`** tab.
2. Click the **`New`** button on the toolbar.
3. Select **`Wireless connection`** from the **`Device Type`** list and click **`Forward`**.
4. If you have already added the wireless network interface card to the hardware list, select it from the **`Wireless card`** list. Otherwise, select **`Other Wireless Card`** to add the hardware device.

Note

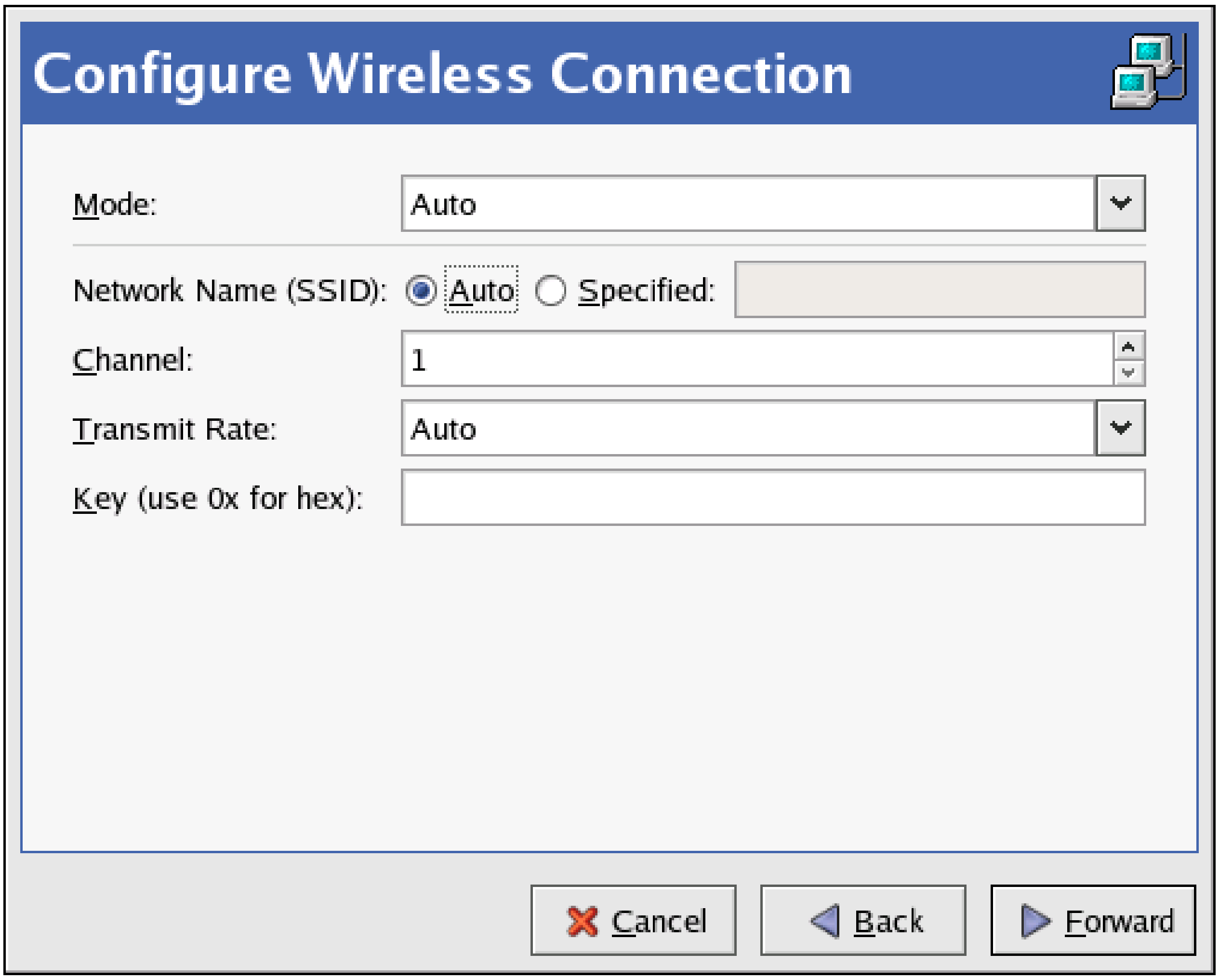
The installation program usually detects supported wireless Ethernet devices and prompts you to configure them. If you configured them during the installation, they are displayed in the hardware list on the **`Hardware`** tab.

5. If you selected **`Other Wireless Card`**, the **`Select Ethernet Adapter`** window appears. Select the manufacturer and model of the Ethernet card and the device. If this is the first Ethernet card for the system, select **`eth0`**; if this is the second Ethernet card for the system, select **`eth1`** (and so on). The **Network Administration Tool** also allows the user to configure the resources for the wireless network interface card. Click **`Forward`** to continue.

6. On the **`Configure Wireless Connection`** page as shown in Figure 17.16, "Wireless Settings", configure the settings for the wireless device.

Note

For the **Authentication** dropdown, note that wireless access points using WEP encryption have a choice between using open system and shared key authentication. Shared key authentication requires an exchange between the client and the access point during the association process that proves that the client has the correct WEP key. Open system authentication allows all wireless clients to connect. Counterintuitively, shared key authentication is less secure than open system, and thus is less widely deployed. It is therefore recommended to select **`Open System (open)`** as the authentication method when you do not know which method the access point requires. If connecting to the access point using open system fails, then try switching to shared key authentication.



Mode: Auto

Network Name (SSID): [✓ Auto] Specified:

Channel: 1

Transmit Rate: Auto

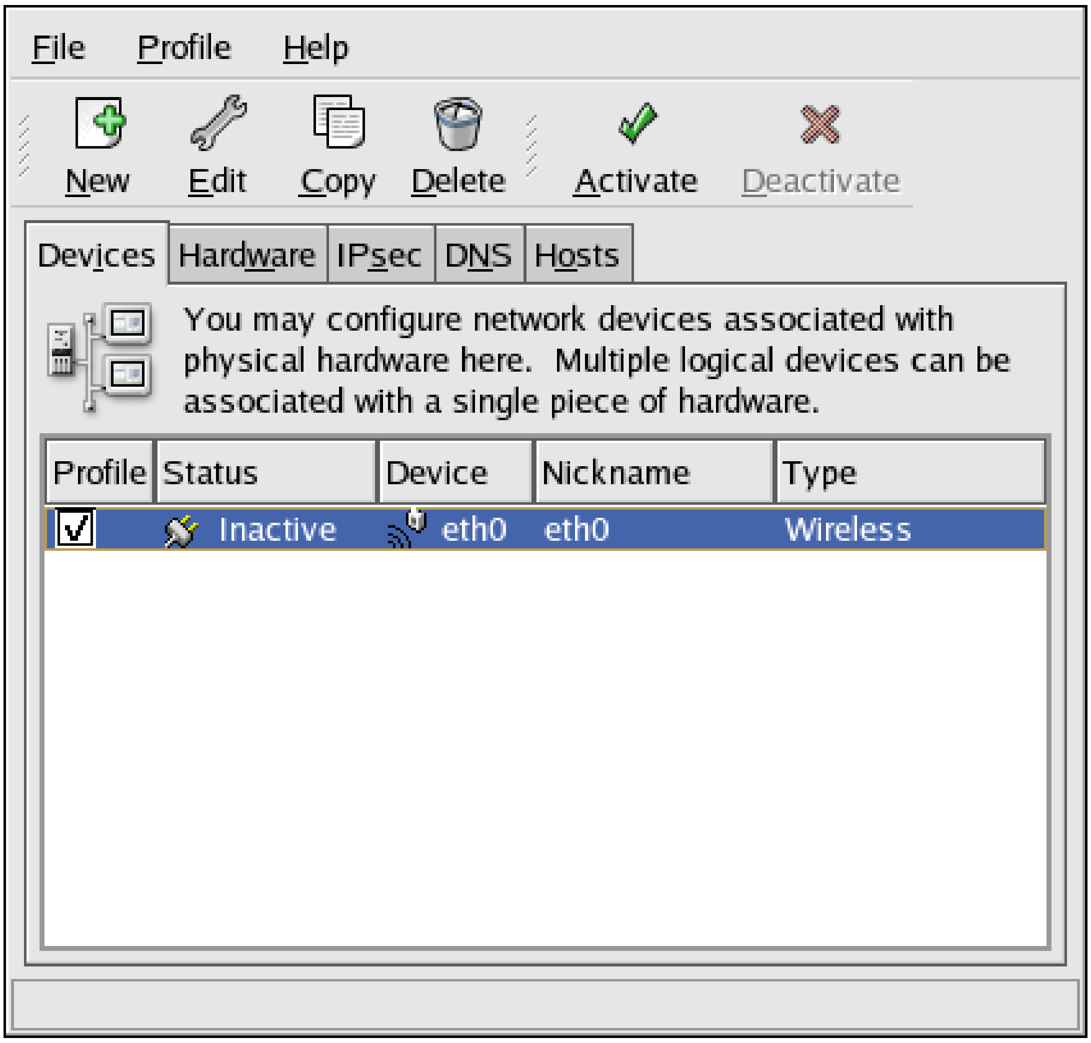
Key (use 0x for hex):

Figure 17.16. Wireless Settings

7. On the **`Configure Network Settings`** page, choose between DHCP and static IP address. You may specify a hostname for the device. If the device receives a dynamic IP address each time the network is started, do not specify a hostname. Click **`Forward`** to continue.

8. Click **`Apply`** on the **`Create Wireless Device`** page.

After configuring the wireless device, it appears in the device list as shown in Figure 17.17, “Wireless Device”.



You may configure network devices associated with physical hardware here. Multiple logical devices can be associated with a single piece of hardware.

Profile	Status	Device	Nickname	Type
✓	Inactive	eth0	eth0	Wireless

Figure 17.17. Wireless Device

Be sure to select **File** > **Save** to save the changes.

After adding the wireless device, you can edit its configuration by selecting the device from the device list and clicking **`Edit`**. For example, you can configure the device to activate at boot time.

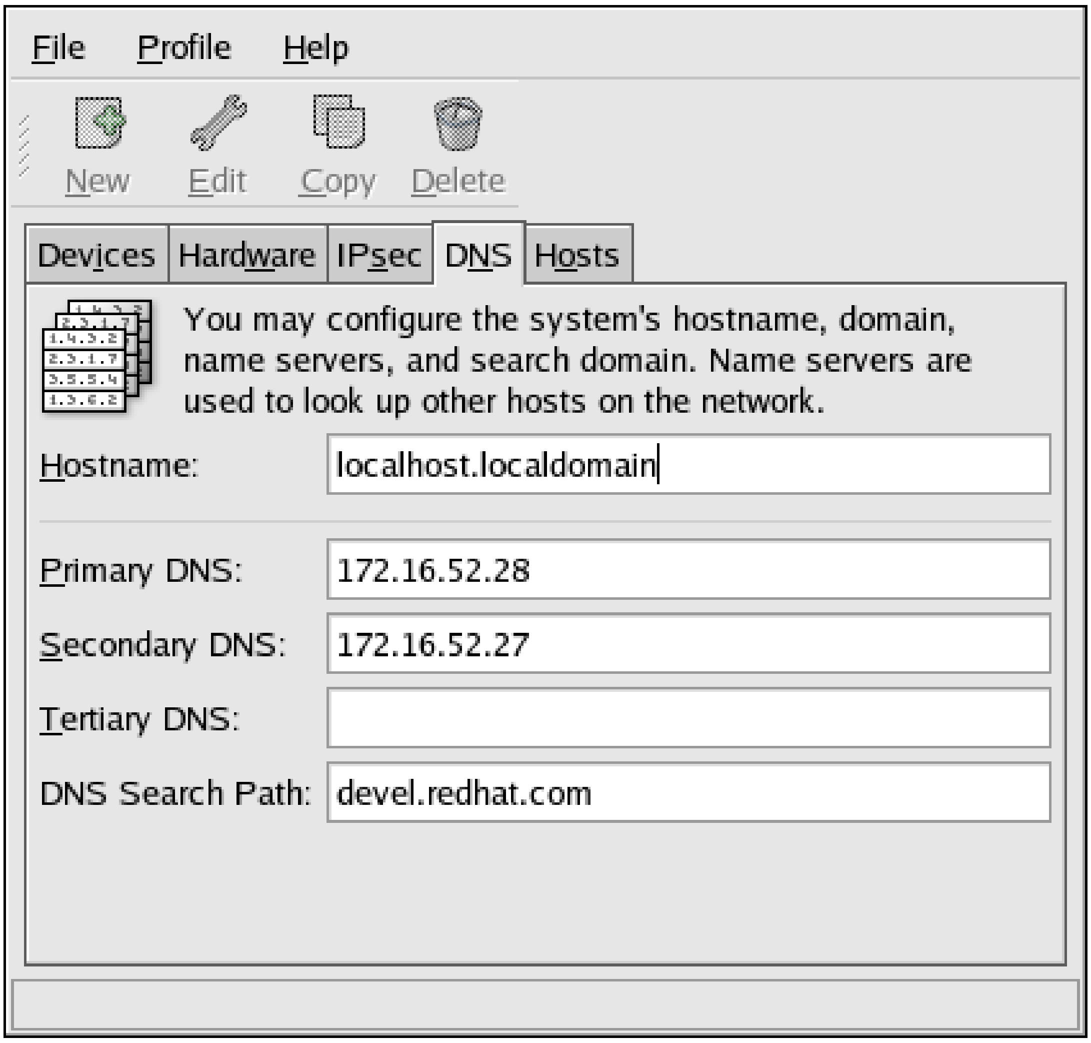
When the device is added, it is not activated immediately, as seen by its **`Inactive`** status. To activate the device, select it from the device list, and click the **`Activate`** button. If the system is configured to activate the device when the computer starts (the default), this step does not have to be performed again.

17.8. Managing DNS Settings

The **`DNS`** tab allows you to configure the system's hostname, domain, name servers, and search domain. Name servers are used to look up other hosts on the network.

If the DNS server names are retrieved from DHCP or PPPoE (or retrieved from the ISP), do not add primary, secondary, or tertiary DNS servers.

If the hostname is retrieved dynamically from DHCP or PPPoE (or retrieved from the ISP), do not change it.



You may configure the system's hostname, domain, name servers, and search domain. Name servers are used to look up other hosts on the network.

Hostname:	localhost.localdomain
Primary DNS:	172.16.52.28
Secondary DNS:	172.16.52.27
Tertiary DNS:	
DNS Search Path:	devel.redhat.com

Figure 17.18. DNS Configuration

Note

The name servers section does not configure the system to be a name server. Instead, it configures which name servers to use when resolving IP addresses to hostnames and vice-versa.

Warning

If the hostname is changed and **`system-config-network`** is started on the local host, you may not be able to start another **X11** application. As such, you may have to re-login to a new desktop session.

17.9. Managing Hosts

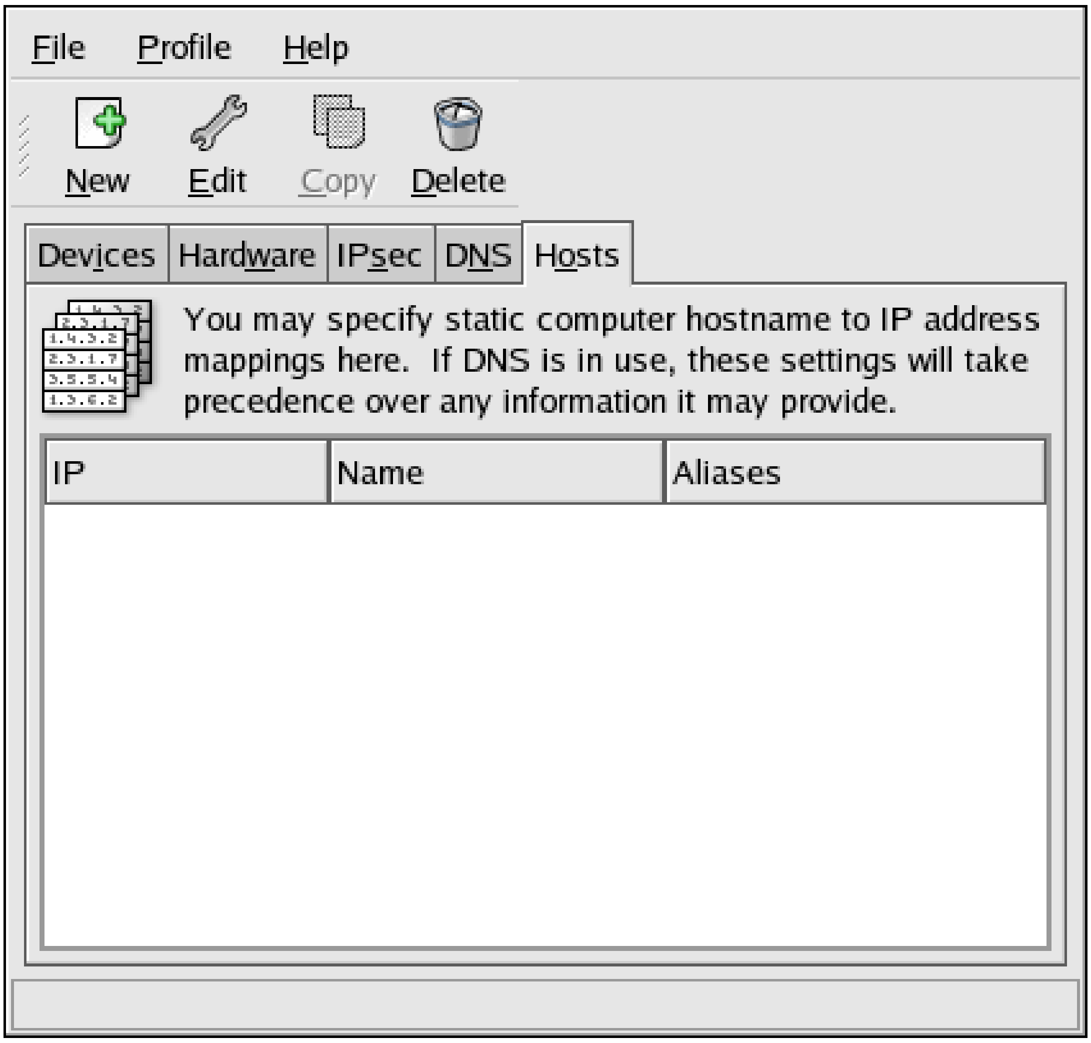
The **`Hosts`** tab allows you to add, edit, or remove hosts from the **`/etc/hosts`** file. This file contains IP addresses and their corresponding hostnames.

When your system tries to resolve a hostname to an IP address or tries to determine the hostname for an IP address, it refers to the **`/etc/hosts`** file before using the name servers (if you are using the default Red Hat Enterprise Linux configuration). If the IP address is listed in the **`/etc/hosts`** file, the name servers are not used. If your network contains computers whose IP addresses are not listed in DNS, it is recommended that you add them to the `/etc/hosts` file.

To add an entry to the `/etc/hosts` file, go to the **`Hosts`** tab, click the **`New`** button on the toolbar, provide the requested information, and click **`OK`**. Select **File** > **Save** or press **`Ctrl`**+**`S`** to save the changes to the `/etc/hosts` file. The network or network services do not need to be restarted since the current version of the file is referred to each time an address is resolved.

Warning

Do not remove the **`localhost`** entry. Even if the system does not have a network connection or have a network connection running constantly, some programs need to connect to the system via the localhost loopback interface.



You may specify static computer hostname to IP address mappings here. If DNS is in use, these settings will take precedence over any information it may provide.

IP	Name	Aliases

Figure 17.19. Hosts Configuration

Note

To change lookup order, edit the **`/etc/host.conf`** file. The line **`order hosts, bind`** specifies that **`/etc/hosts`** takes precedence over the name servers. Changing the line to**`order bind, hosts`** configures the system to resolve hostnames and IP addresses using the name servers first. If the IP address cannot be resolved through the name servers, the system then looks for the IP address in the **`/etc/hosts`** file.

17.10. Working with Profiles

Multiple logical network devices can be created for each physical hardware device. For example, if you have one Ethernet card in your system (eth0), you can create logical network devices with different nicknames and different configuration options, all to be specifically associated with eth0.

Logical network devices are different from device aliases. Logical network devices associated with the same physical device must exist in different profiles and cannot be activated simultaneously. Device aliases are also associated with the same physical hardware device, but device aliases associated with the same physical hardware can be activated at the same time. Refer to Section 17.11, “Device Aliases” for details about creating device aliases.

Profiles can be used to create multiple configuration sets for different networks. A configuration set can include logical devices as well as hosts and DNS settings. After configuring the profiles, you can use the **Network Administration Tool** to switch back and forth between them.

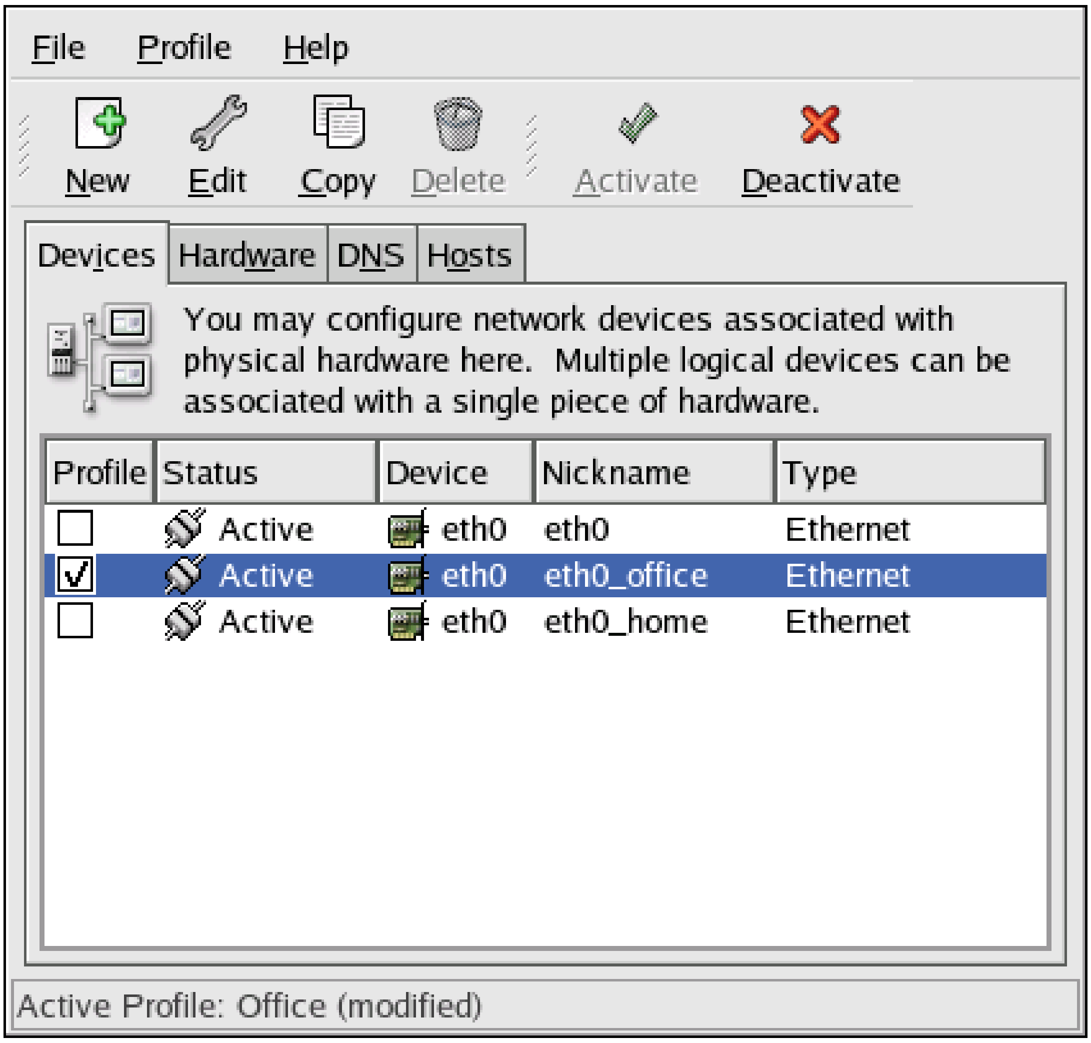
By default, there is one profile called **`Common`**. To create a new profile, select **Profile** > **New** from the pull-down menu, and enter a unique name for the profile.

You are now modifying the new profile as indicated by the status bar at the bottom of the main window.

Click on an existing device already in the list and click the **`Copy`** button to copy the existing device to a logical network device. If you use the **`New`** button, a network alias is created, which is incorrect. To change the properties of the logical device, select it from the list and click **`Edit`**. For example, the **`Nickname`** can be changed to a more descriptive name, such as **`eth0_office`**, so that it can be recognized more easily. Once you have finished editing your new profile, make sure to save it by clicking **Save** from the **File** menu. If you forget to save after creating a profile, that profile will be lost.

In the list of devices, there is a column of checkboxes labeled **`Profile`**. For each profile, you can check or uncheck devices. Only the checked devices are included for the currently selected profile. For example, if you create a logical device named **`eth0_office`** in a profile called **`Office`** and want to activate the logical device if the profile is selected, uncheck the **`eth0`** device and check the **`eth0_office`** device.

For example, Figure 17.20, “Office Profile” shows a profile called **`Office`** with the logical device **`eth0_office`**. It is configured to activate the first Ethernet card using DHCP.



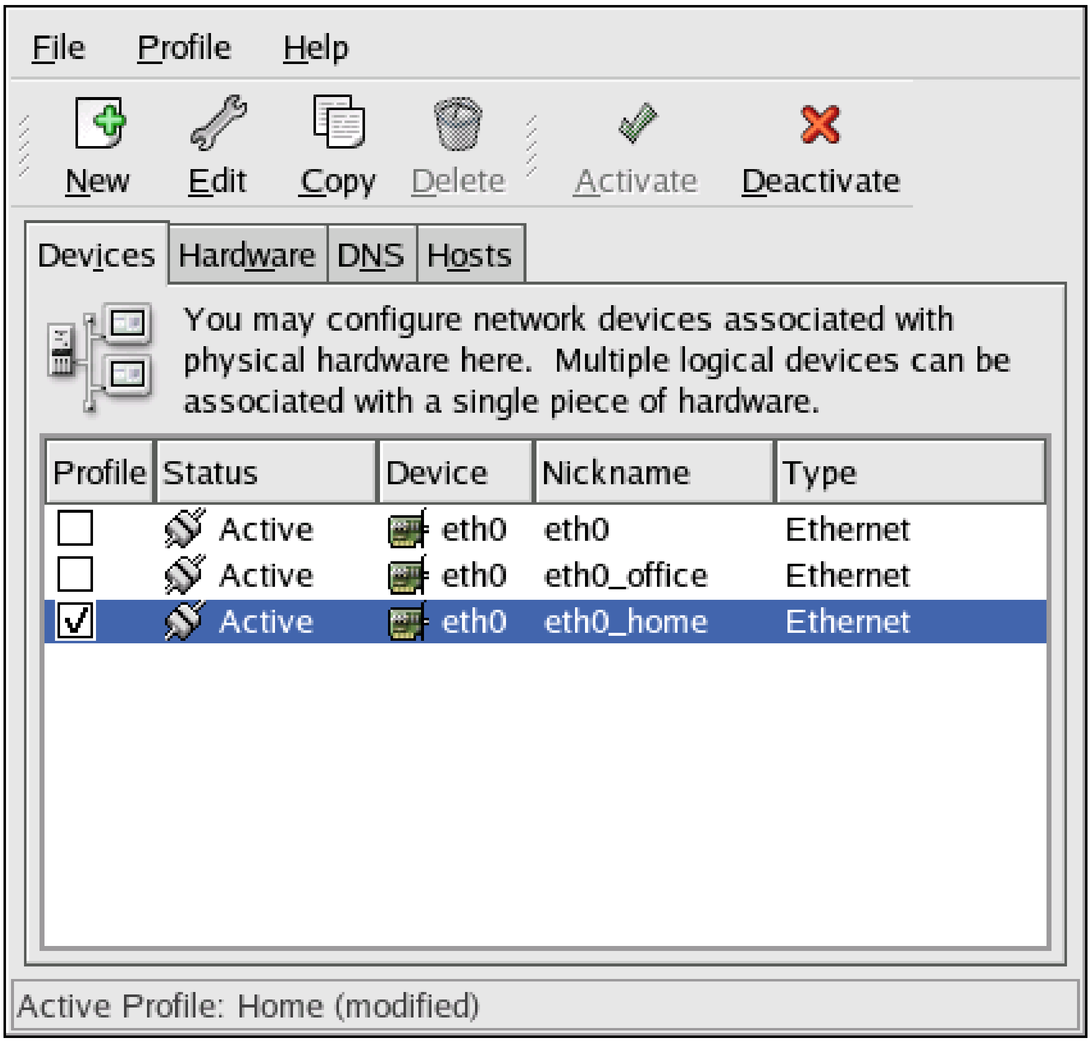
You may configure network devices associated with physical hardware here. Multiple logical devices can be associated with a single piece of hardware.

Profile	Status	Device	Nickname	Type
☐	Active	eth0	eth0	Ethernet
✓	Active	eth0	eth0_office	Ethernet
☐	Active	eth0	eth0_home	Ethernet

Active Profile: Office (modified)

Figure 17.20. Office Profile

Notice that the **Home** profile as shown in Figure 17.21, “Home Profile” activates the **eth0_home** logical device, which is associated with **eth0**.



You may configure network devices associated with physical hardware here. Multiple logical devices can be associated with a single piece of hardware.

Profile	Status	Device	Nickname	Type
☐	Active	eth0	eth0	Ethernet
☐	Active	eth0	eth0_office	Ethernet
✓	Active	eth0	eth0_home	Ethernet

Active Profile: Home (modified)

Figure 17.21. Home Profile

You can also configure **`eth0`** to activate in the **`Office`** profile only and to activate a PPP (modem) device in the **`Home`** profile only. Another example is to have the **`Common`** profile activate **`eth0`** and an **`Away`** profile activate a PPP device for use while traveling.

To activate a profile at boot time, modify the boot loader configuration file to include the **`netprofile=<`*`profilename`*`>`** option. For example, if the system uses GRUB as the boot loader and **`/boot/grub/grub.conf`** contains:

```
title Red Hat Enterprise Linux (2.6.9-5.EL)
         root (hd0,0)
  kernel /vmlinuz-2.6.9-5.EL ro root=/dev/VolGroup00/LogVol00 rhgb quiet
  initrd /initrd-2.6.9-5.EL.img
```

Modify it to the following (where <*profilename*> is the name of the profile to be activated at boot time):

```
title Red Hat Enterprise Linux (2.6.9-5.EL)
         root (hd0,0)
  kernel /vmlinuz-2.6.9-5.EL ro root=/dev/VolGroup00/LogVol00 \
   netprofile=<profilename>  \    rhgb quiet
  initrd /initrd-2.6.9-5.EL.img
```

To switch profiles after the system has booted, go to Applications (the main menu on the panel) > **System Tools** > **Network Device Control** (or type the command **`system-control-network`**) to select a profile and activate it. The activate profile section only appears in the **Network Device Control** interface if more than the default **`Common`** interface exists.

Alternatively, execute the following command to enable a profile (replace *<profilename>* with the name of the profile):

```
system-config-network-cmd --profile <profilename> --activate
```

17.11. Device Aliases

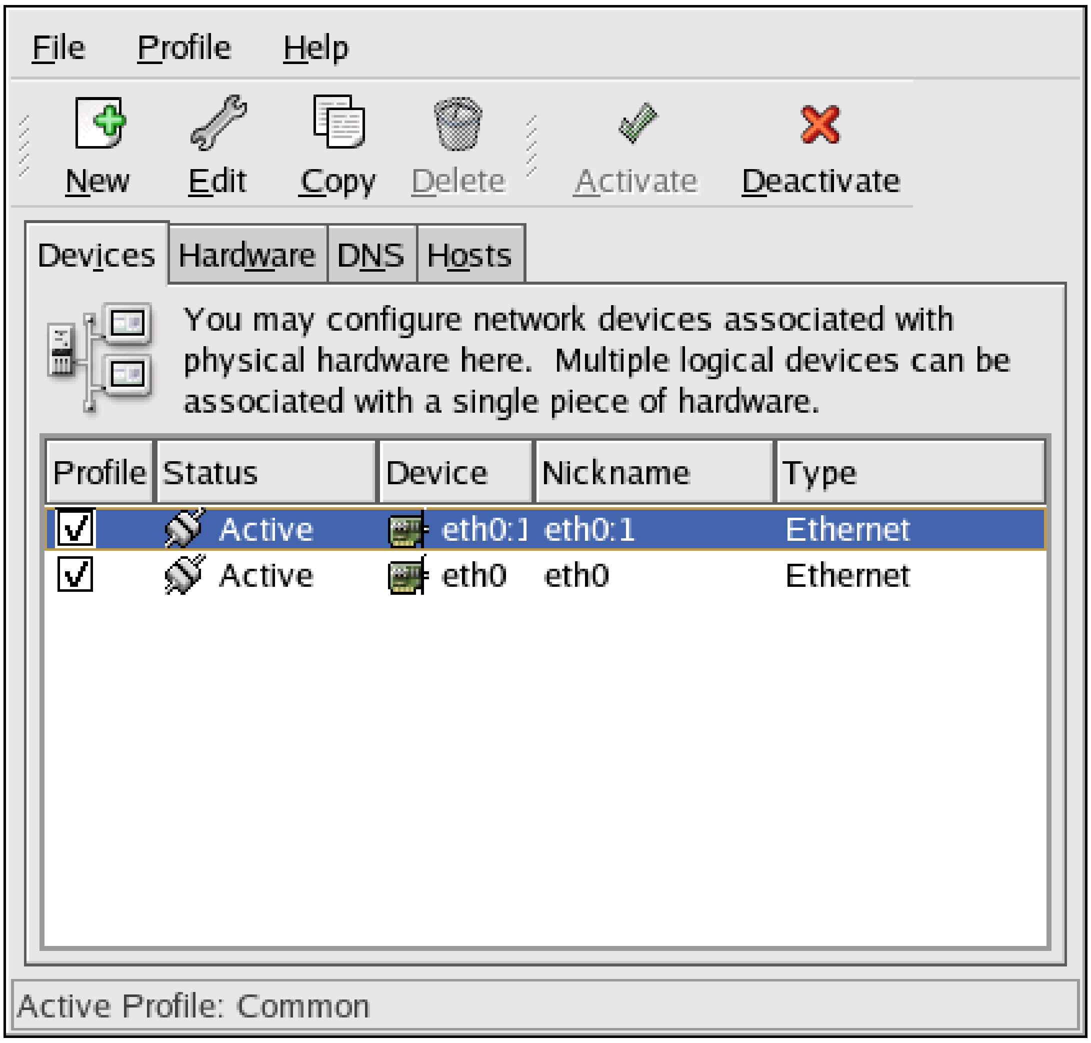
Device aliases are virtual devices associated with the same physical hardware, but they can be activated at the same time to have different IP addresses. They are commonly represented as the device name followed by a colon and a number (for example, eth0:1). They are useful if you want to have multiple IP addresses for a system that only has one network card.

After configuring the Ethernet device —such as **`eth0`** —to use a static IP address (DHCP does not work with aliases), go to the **`Devices`** tab and click **`New`**. Select the Ethernet card to configure with an alias, set the static IP address for the alias, and click **`Apply`** to create it. Since a device already exists for the Ethernet card, the one just created is the alias, such as **`eth0:1`**.

Warning

If you are configuring an Ethernet device to have an alias, neither the device nor the alias can be configured to use DHCP. You must configure the IP addresses manually.

Figure 17.22, “Network Device Alias Example” shows an example of one alias for the **`eth0`** device. Notice the **`eth0:1`** device — the first alias for **`eth0`**. The second alias for **`eth0`** would have the device name **`eth0:2`**, and so on. To modify the settings for the device alias, such as whether to activate it at boot time and the alias number, select it from the list and click the **`Edit`** button.



You may configure network devices associated with physical hardware here. Multiple logical devices can be associated with a single piece of hardware.

Profile	Status	Device	Nickname	Type
✓	Active	eth0:1	eth0:1	Ethernet
✓	Active	eth0	eth0	Ethernet

Active Profile: Common

Figure 17.22. Network Device Alias Example

Select the alias and click the **`Activate`** button to activate the alias. If you have configured multiple profiles, select which profiles in which to include it.

To verify that the alias has been activated, use the command **`/sbin/ifconfig`**. The output should show the device and the device alias with different IP addresses:

```
eth0      Link encap:Ethernet
 HWaddr 00:A0:CC:60:B7:G4
 inet addr:192.168.100.5  Bcast:192.168.100.255  Mask:255.255.255.0
 UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
 RX packets:161930 errors:1 dropped:0 overruns:0 frame:0
 TX packets:244570 errors:0 dropped:0 overruns:0 carrier:0
 collisions:475 txqueuelen:100
 RX bytes:55075551 (52.5 Mb)  TX bytes:178108895 (169.8 Mb)
 Interrupt:10 Base address:0x9000  eth0:1    Link encap:Ethernet  HWaddr
00:A0:CC:60:B7:G4
 inet addr:192.168.100.42  Bcast:192.168.100.255  Mask:255.255.255.0
 UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
 Interrupt:10 Base address:0x9000  lo
 Link encap:Local Loopback
 inet addr:127.0.0.1  Mask:255.0.0.0
 UP LOOPBACK RUNNING  MTU:16436  Metric:1
```

```
RX packets:5998 errors:0 dropped:0 overruns:0 frame:0
TX packets:5998 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:1627579 (1.5 Mb)  TX bytes:1627579 (1.5 Mb)
```

17.12. Saving and Restoring the Network Configuration

The command line version of **Network Administration Tool** can be used to save the system's network configuration to a file. This file can then be used to restore the network settings to a Red Hat Enterprise Linux system.

This feature can be used as part of an automated backup script, to save the configuration before upgrading or reinstalling, or to copy the configuration to a different Red Hat Enterprise Linux system.

To save, or *export*, the network configuration of a system to the file **`/tmp/network-config`**, execute the following command as root:

```
system-config-network-cmd -e > /tmp/network-config
```

To restore, or *import*, the network configuration from the file created from the previous command, execute the following command as root:

```
system-config-network-cmd -i -c -f /tmp/network-config
```

The **`-i`** option means to import the data, the **`-c`** option means to clear the existing configuration prior to importing, and the **`-f`** option specifies that the file to import is as follows.

Chapter 18. Controlling Access to Services

Maintaining security on your system is extremely important, and one approach for this task is to manage access to system services carefully. Your system may need to provide open access to particular services (for example, **`httpd`** if you are running a Web server). However, if you do not need to provide a service, you should turn it off to minimize your exposure to possible bug exploits.

There are several different methods for managing access to system services. Choose which method of management to use based on the service, your system's configuration, and your level of Linux expertise.

The easiest way to deny access to a service is to turn it off. Both the services managed by **`xinetd`** and the services in the **`/etc/rc.d/init.d`** hierarchy (also known as SysV services) can be configured to start or stop using three different applications:

Services Configuration Tool

> This is a graphical application that displays a description of each service, displays whether each service is started at boot time (for runlevels 3, 4, and 5), and allows services to be started, stopped, and restarted.

ntsysv

> This is a text-based application that allows you to configure which services are started at boot time for each runlevel. Non-**`xinetd`** services can not be started, stopped, or restarted using this program.

`chkconfig`

> This is a command line utility that allows you to turn services on and off for the different runlevels. Non-**`xinetd`** services can not be started, stopped, or restarted using this utility.

You may find that these tools are easier to use than the alternatives — editing the numerous symbolic links located in the directories below **`/etc/rc.d`** by hand or editing the **`xinetd`** configuration files in **`/etc/xinetd.d`**.

Another way to manage access to system services is by using **`iptables`** to configure an IP firewall. If you are a new Linux user, note that **`iptables`** may not be the best solution for you. Setting up **`iptables`** can be complicated, and is best tackled by experienced Linux system administrators.

On the other hand, the benefit of using **`iptables`** is flexibility. For example, if you need a customized solution which provides certain hosts access to certain services, **`iptables`** can provide it for you. Refer to Section 48.8.1, “Netfilter and IPTables” and Section 48.8.3, “Using IPTables” for more information about **`iptables`**.

Alternatively, if you are looking for a utility to set general access rules for your home machine, and/or if you are new to Linux, try the **Security Level Configuration Tool** (**`system-config-securitylevel`**), which allows you to select the security level for your system, similar to the **`Firewall Configuration`** screen in the installation program.

Refer to Section 48.8, “Firewalls” for more information.

Important

When you allow access for new services, always remember that both the firewall and SELinux need to be configured as well. One of the most common mistakes committed when configuring a new service is neglecting to implement the necessary firewall configuration and SELinux policies to allow access for it. Refer to Section 48.8.2, “Basic Firewall Configuration” for more information.

18.1. Runlevels

Before you can configure access to services, you must understand Linux runlevels. A runlevel is a state, or *mode*, that is defined by the services listed in the directory **`/etc/rc.d/rc<x>.d`**, where <*x*> is the number of the runlevel.

The following runlevels exist:

- 0 — Halt
- 1 — Single-user mode
- 2 — Not used (user-definable)
- 3 — Full multi-user mode
- 4 — Not used (user-definable)
- 5 — Full multi-user mode (with an X-based login screen)
- 6 — Reboot

If you use a text login screen, you are operating in runlevel 3. If you use a graphical login screen, you are operating in runlevel 5.

The default runlevel can be changed by modifying the **`/etc/inittab`** file, which contains a line near the top of the file similar to the following:

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

Change the number in this line to the desired runlevel. The change does not take effect until you reboot the system.

18.2. TCP Wrappers

Many UNIX system administrators are accustomed to using TCP wrappers to manage access to certain network services. Any network services managed by **`xinetd`** (as well as any program with built-in support for **`libwrap`**) can use TCP wrappers to manage access. **`xinetd`** can use the **`/etc/hosts.allow`** and **`/etc/hosts.deny`** files to configure access to system services. As the names imply, **`hosts.allow`** contains a list of rules that allow clients to access the network services controlled by **`xinetd`**, and **`hosts.deny`** contains rules to deny access. The **`hosts.allow`** file takes precedence over the **`hosts.deny`** file. Permissions to grant or deny access can be based on individual IP address (or hostnames) or on a pattern of clients. Refer to **`hosts_access`** in section 5 of the man pages (**`man 5 hosts_access`**) for details.

18.2.1. `xinetd`

To control access to Internet services, use **`xinetd`**, which is a secure replacement for **`inetd`**. The **`xinetd`** daemon conserves system resources, provides access control and logging, and can be used to start special-purpose servers. **`xinetd`** can also be used to grant or deny access to particular hosts, provide service access at specific times, limit the rate of incoming connections, limit the load created by connections, and more.

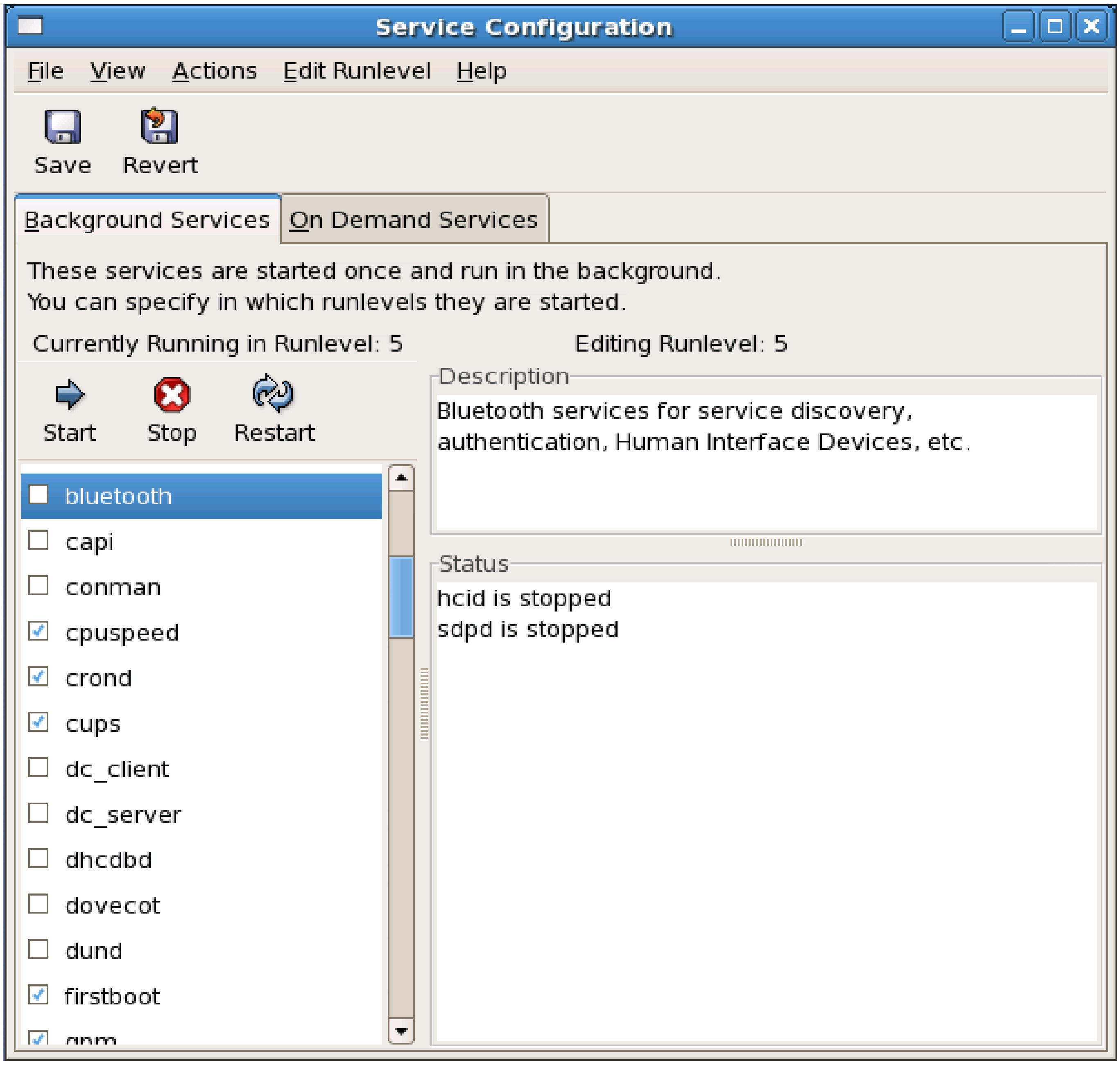
`xinetd` runs constantly and listens on all ports for the services it manages. When a connection request arrives for one of its managed services, **`xinetd`** starts up the appropriate server for that service.

The configuration file for **`xinetd`** is **`/etc/xinetd.conf`**, but the file only contains a few defaults and an instruction to include the **`/etc/xinetd.d`** directory. To enable or disable an **`xinetd`** service, edit its configuration file in the **`/etc/xinetd.d`** directory. If the **`disable`** attribute is set to **`yes`**, the service is disabled. If the **`disable`** attribute is set to **`no`**, the service is enabled. You can edit any of the **`xinetd`** configuration files or change its enabled status using the **Services Configuration Tool**, **ntsysv**, or **`chkconfig`**. For a list of network services controlled by **`xinetd`**, review the contents of the **`/etc/xinetd.d`** directory with the command **`ls /etc/xinetd.d`**.

18.3. Services Configuration Tool

The **Services Configuration Tool** is a graphical application developed by Red Hat to configure which SysV services in the **`/etc/rc.d/init.d`** directory are started at boot time (for runlevels 3, 4, and 5) and which **`xinetd`** services are enabled. It also allows you to start, stop, and restart SysV services as well as reload **`xinetd`**.

To start the **Services Configuration Tool** from the desktop, go to the Applications (the main menu on the panel) > **System Settings** > **Server Settings** > **Services** or type the command **`system-config-services`** at a shell prompt (for example, in an **XTerm** or a **GNOME terminal**).



These services are started once and run in the background.
You can specify in which runlevels they are started.

Currently Running in Runlevel: 5

Editing Runlevel: 5

- [ ] bluetooth
- [ ] capi
- [ ] conman
- [x] cpuspeed
- [x] crond
- [x] cups
- [ ] dc_client
- [ ] dc_server
- [ ] dhcdbd
- [ ] dovecot
- [ ] dund
- [x] firstboot
- [x] gpm

Description: Bluetooth services for service discovery, authentication, Human Interface Devices, etc.

Status:

```
hcid is stopped
sdpd is stopped
```

Figure 18.1. Services Configuration Tool

The **Services Configuration Tool** displays the current runlevel as well as the runlevel you are currently editing. To edit a different runlevel, select **Edit Runlevel** from the pulldown menu and select runlevel 3, 4, or 5. Refer to Section 18.1, “Runlevels” for a description of runlevels.

The **Services Configuration Tool** lists the services from the **`/etc/rc.d/init.d`** directory as well as the services controlled by **`xinetd`**. Click on the name of the service from the list on the left-hand side of the application to display a brief description of that service as well as the status of the service. If the service is not an **`xinetd`** service, the status window shows whether the service is currently running. If the service is controlled by **`xinetd`**, the status window displays the phrase **`xinetd service`**.

To start, stop, or restart a service immediately, select the service from the list and click the appropriate button on the toolbar (or choose the action from the **Actions** pulldown menu). If the service is an **`xinetd`** service, the action buttons are disabled because they cannot be started or stopped individually.

If you enable/disable an **`xinetd`** service by checking or unchecking the checkbox next to the service name, you must select **File** > **Save Changes** from the pulldown menu (or the **Save** button above the tabs) to reload **`xinetd`** and immediately enable/disable the **`xinetd`** service that you changed. **`xinetd`** is also configured to remember the setting. You can enable/disable multiple **`xinetd`** services at a time and save the changes when you are finished.

For example, assume you check **`rsync`** to enable it in runlevel 3 and then save the changes. The **`rsync`** service is immediately enabled. The next time **`xinetd`** is started, **`rsync`** is still enabled.

Note

When you save changes to **`xinetd`** services, **`xinetd`** is reloaded, and the changes take place immediately. When you save changes to other services, the runlevel is reconfigured, but the changes do not take effect immediately.

To enable a non-**`xinetd`** service to start at boot time for the currently selected runlevel, check the box beside the name of the service in the list. After configuring the runlevel, apply the changes by selecting **File** > **Save Changes** from the pulldown menu. The runlevel configuration is changed, but the runlevel is not restarted; thus, the changes do not take place immediately.

For example, assume you are configuring runlevel 3. If you change the value for the **`httpd`** service from checked to unchecked and then select **Save Changes**, the runlevel 3 configuration changes so that **`httpd`** is not started at boot time. However, runlevel 3 is not reinitialized, so **`httpd`** is still running. Select one of following options at this point:

1. Stop the **`httpd`** service — Stop the service by selecting it from the list and clicking the **`Stop`** button. A message appears stating that the service was stopped successfully.

2. Reinitialize the runlevel — Reinitialize the runlevel by going to a shell prompt and typing the command **`telinit`** ***`x`*** (where *x* is the runlevel number; in this example, 3.). This option is recommended if you change the **`Start at Boot`** value of multiple services and want to activate the changes immediately.

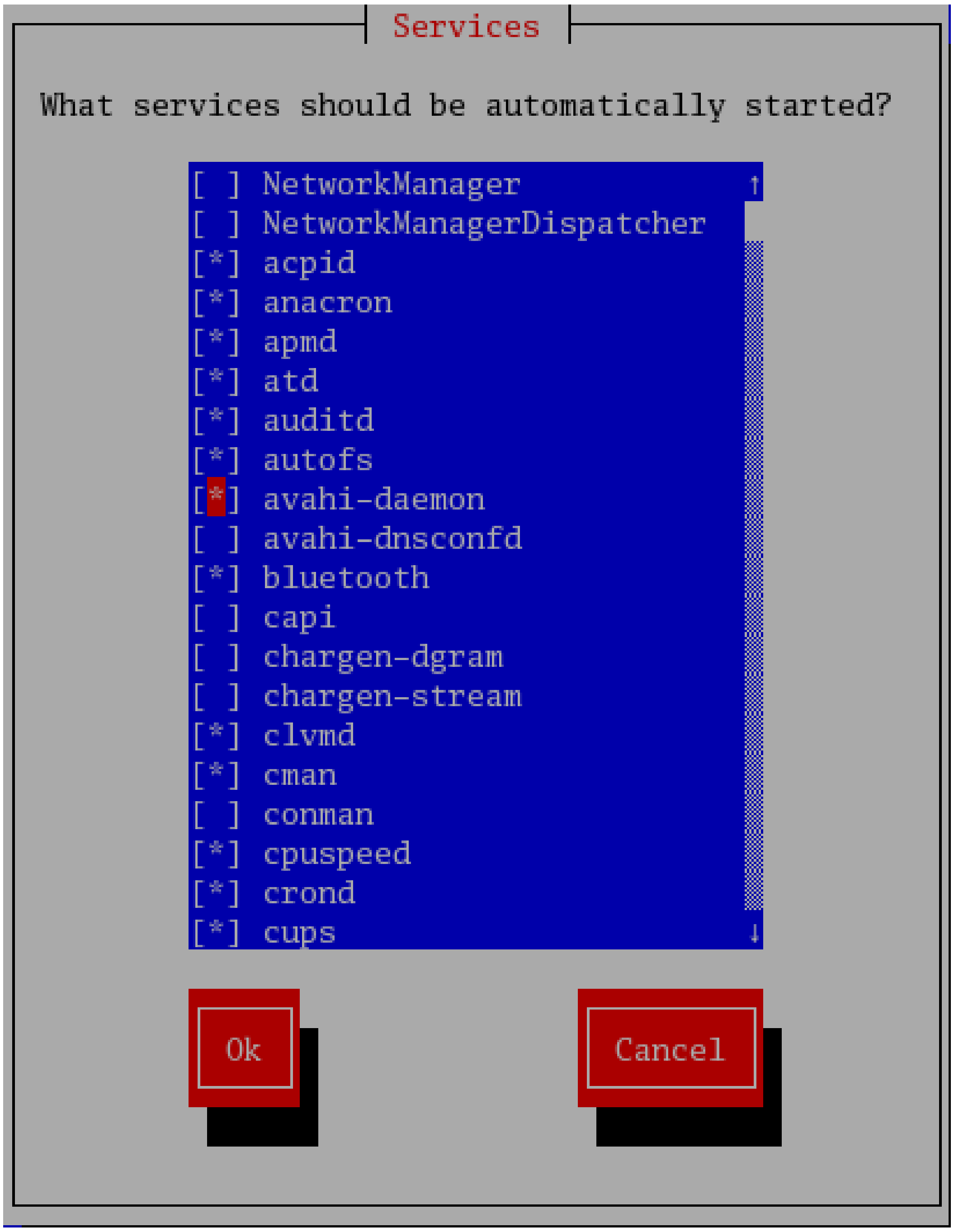
3. Do nothing else — You do not have to stop the **`httpd`** service. You can wait until the system is rebooted for the service to stop. The next time the system is booted, the runlevel is initialized without the **`httpd`** service running.

To add a service to a runlevel, select the runlevel from the **Edit Runlevel** pulldown menu, and then select **Actions** > **Add Service**. To delete a service from a runlevel, select the runlevel from the **Edit Runlevel** pulldown menu, select the service to be deleted from the list on the left, and select **Actions** > **Delete Service**.

18.4. ntsysv

The **ntsysv** utility provides a simple interface for activating or deactivating services. You can use **ntsysv** to turn an **`xinetd`**-managed service on or off. You can also use **ntsysv** to configure runlevels. By default, only the current runlevel is configured. To configure a different runlevel, specify one or more runlevels with the **`--level`** option. For example, the command **`ntsysv --level 345`** configures runlevels 3, 4, and 5.

The **ntsysv** interface works like the text mode installation program. Use the up and down arrows to navigate up and down the list. The space bar selects/unselects services and is also used to "press" the **`Ok`** and **`Cancel`** buttons. To move between the list of services and the **`Ok`** and **`Cancel`** buttons, use the **`Tab`** key. An asterisk (**`*`**) signifies that a service is set to on. Pressing the **`F1`** key displays a short description of the selected service.



```
Services

What services should be automatically started?

[ ] NetworkManager
[ ] NetworkManagerDispatcher
[*] acpid
[*] anacron
[*] apmd
[*] atd
[*] auditd
[*] autofs
[*] avahi-daemon
[ ] avahi-dnsconfd
[*] bluetooth
[ ] capi
[ ] chargen-dgram
[ ] chargen-stream
[*] clvmd
[*] cman
[ ] conman
[*] cpuspeed
[*] crond
[*] cups
```

Figure 18.2. The ntsysv utility

Warning

Services managed by **`xinetd`** are immediately affected by **ntsysv**. For all other services, changes do not take effect immediately. You must stop or start the individual service with the command **`service <daemon> stop`** (where *<daemon>* is the name of the service you want to stop; for example, **`httpd`**). Replace **`stop`** with **`start`** or **`restart`** to start or restart the service.

18.5. `chkconfig`

The **chkconfig** command can also be used to activate and deactivate services. The **chkconfig --list** command displays a list of system services and whether they are started (**on**) or stopped (**off**) in runlevels 0-6. At the end of the list is a section for the services managed by **xinetd**.

If the **chkconfig --list** command is used to query a service managed by **xinetd**, it displays whether the **xinetd** service is enabled (**on**) or disabled (**off**). For example, the command **chkconfig --list rsync** returns the following output:

```
rsync           on
```

As shown, **rsync** is enabled as an **xinetd** service. If **xinetd** is running, **rsync** is enabled.

If you use **chkconfig --list** to query a service in **/etc/rc.d**, that service's settings for each runlevel are displayed. For example, the command **chkconfig --list httpd** returns the following output:

```
httpd           0:off    1:off    2:on     3:on     4:on     5:on     6:off
```

chkconfig can also be used to configure a service to be started (or not) in a specific runlevel. For example, to turn **nscd** off in runlevels 3, 4, and 5, use the following command:

```
chkconfig --level 345 nscd off
```

Warning

Services managed by **xinetd** are immediately affected by **chkconfig**. For example, if **xinetd** is running while **rsync** is disabled, and the command **chkconfig rsync on** is executed, then **rsync** is immediately enabled without having to restart **xinetd** manually. Changes for other services do not take effect immediately after using **chkconfig**. You must stop or start the individual service with the command **service *<daemon>* stop** (where *<daemon>* is the name of the service you want to stop; for example, **httpd**). Replace **stop** with **start** or **restart** to start or restart the service.

18.6. Additional Resources

For more information, refer to the following resources.

18.6.1. Installed Documentation

- The man pages for **ntsysv**, **chkconfig**, **xinetd**, and **xinetd.conf**.
- **man 5 hosts_access** — The man page for the format of host access control files (in section 5 of the man pages).

18.6.2. Useful Websites

- http://www.xinetd.org — The **xinetd** webpage. It contains sample configuration files and a more detailed list of features.

Chapter 19. Berkeley Internet Name Domain (BIND)

On most modern networks, including the Internet, users locate other computers by name. This frees users from the daunting task of remembering the numerical network address of network resources. The most effective way to configure a network to allow such name-based connections is to set up a *Domain Name Service* (*DNS*) or a *nameserver*, which resolves hostnames on the network to numerical addresses and vice versa.

This chapter reviews the nameserver included in Red Hat Enterprise Linux and the *Berkeley Internet Name Domain* (*BIND*) DNS server, with an emphasis on the structure of its configuration files and how it may be administered both locally and remotely.

Note

BIND is also known as the service **`named`** in Red Hat Enterprise Linux. You can manage it via the Services Configuration Tool (**`system-config-service`**).

19.1. Introduction to DNS

DNS associates hostnames with their respective IP addresses, so that when users want to connect to other machines on the network, they can refer to them by name, without having to remember IP addresses.

Use of DNS and FQDNs also has advantages for system administrators, allowing the flexibility to change the IP address for a host without affecting name-based queries to the machine. Conversely, administrators can shuffle which machines handle a name-based query.

DNS is normally implemented using centralized servers that are authoritative for some domains and refer to other DNS servers for other domains.

When a client host requests information from a nameserver, it usually connects to port 53. The nameserver then attempts to resolve the FQDN based on its resolver library, which may contain authoritative information about the host requested or cached data from an earlier query. If the nameserver does not already have the answer in its resolver library, it queries other nameservers, called *root nameservers*, to determine which nameservers are authoritative for the FQDN in question. Then, with that information, it queries the authoritative nameservers to determine the IP address of the requested host. If a reverse lookup is performed, the same procedure is used, except that the query is made with an unknown IP address rather than a name.

19.1.1. Nameserver Zones

On the Internet, the FQDN of a host can be broken down into different sections. These sections are organized into a hierarchy (much like a tree), with a main trunk, primary branches, secondary branches, and so forth. Consider the following FQDN:

```
bob.sales.example.com
```

When looking at how an FQDN is resolved to find the IP address that relates to a particular system, read the name from right to left, with each level of the hierarchy divided by periods (**`.`**). In this example, **`com`** defines the *top level domain* for this FQDN. The name **`example`** is a sub-domain under **`com`**, while **`sales`** is a sub-domain under **`example`**. The name furthest to the left, **`bob`**, identifies a specific machine hostname.

Except for the hostname, each section is called a *zone*, which defines a specific *namespace*. A namespace controls the naming of the sub-domains to its left. While this example only contains two sub-domains, an FQDN must contain at least one sub-domain but may include many more, depending upon how the namespace is organized.

Zones are defined on authoritative nameservers through the use of *zone files* (which describe the namespace of that zone), the mail servers to be used for a particular domain or sub-domain, and more. Zone files are stored on *primary nameservers* (also called *master nameservers*), which are truly authoritative and where changes are made to the files, and *secondary nameservers* (also called *slave nameservers*), which receive their zone files from the primary nameservers. Any nameserver can be a primary and secondary nameserver for different zones at the same time, and they may also be considered authoritative for multiple zones. It all depends on how the nameserver is configured.

19.1.2. Nameserver Types

There are four primary nameserver configuration types:

master

Stores original and authoritative zone records for a namespace, and answers queries about the namespace from other nameservers.

slave

Answers queries from other nameservers concerning namespaces for which it is considered an authority. However, slave nameservers get their namespace information from master nameservers.

caching-only

Offers name-to-IP resolution services, but is not authoritative for any zones. Answers for all resolutions are cached in memory for a fixed period of time, which is specified by the retrieved zone record.

forwarding

Forwards requests to a specific list of nameservers for name resolution. If none of the specified nameservers can perform the resolution, the resolution fails.

A nameserver may be one or more of these types. For example, a nameserver can be a master for some zones, a slave for others, and only offer forwarding resolutions for others.

19.1.3. BIND as a Nameserver

BIND performs name resolution services through the **`/usr/sbin/named`** daemon. BIND also includes an administration utility called **`/usr/sbin/rndc`**. More information about **`rndc`** can be found in Section 19.4, “Using **`rndc`** ”.

BIND stores its configuration files in the following locations:

`/etc/named.conf`

The configuration file for the **`named`** daemon

`/var/named/` directory

The **`named`** working directory which stores zone, statistic, and cache files

Note

If you have installed the **`bind-chroot`** package, the BIND service will run in the **`/var/named/chroot`** environment. All configuration files will be moved there. As such, **`named.conf`** will be located in **`/var/named/chroot/etc/named.conf`**, and so on.

Note

If you have installed the **`caching-nameserver`** package, the default configuration file is **`/etc/named.caching-nameserver.conf`**. To override this default configuration, you can create your own custom configuration file in **`/etc/named.conf`**. BIND will use the **`/etc/named.conf`** custom file instead of the default configuration file after you restart.

The next few sections review the BIND configuration files in more detail.

19.2. `/etc/named.conf`

The **`named.conf`** file is a collection of statements using nested options surrounded by opening and closing ellipse characters, **`{ }`**. Administrators must be careful when editing **`named.conf`** to avoid syntax errors as many seemingly minor errors prevent the **`named`** service from starting.

A typical **`named.conf`** file is organized similar to the following example:

```
<statement-1> ["<statement-1-name>"] [<statement-1-class>] {
 <option-1>;
 <option-2>;
 <option-N>;
};
<statement-2> ["<statement-2-name>"] [<statement-2-class>] {
 <option-1>;
 <option-2>;
 <option-N>;
};
<statement-N> ["<statement-N-name>"] [<statement-N-class>] {
 <option-1>;
 <option-2>;
 <option-N>;
};
```

19.2.1. Common Statement Types

The following types of statements are commonly used in **`/etc/named.conf`**:

19.2.1.1. `acl` Statement

The **`acl`** statement (or access control statement) defines groups of hosts which can then be permitted or denied access to the nameserver.

An **`acl`** statement takes the following form:

```
acl <acl-name> {
 <match-element>;
 [<match-element>; ...]
};
```

In this statement, replace *<acl-name>* with the name of the access control list and replace*<match-element>* with a semi-colon separated list of IP addresses. Most of the time, an individual IP address or IP network notation (such as **`10.0.1.0/24`**) is used to identify the IP addresses within the **`acl`** statement.

The following access control lists are already defined as keywords to simplify configuration:

- **any** — Matches every IP address
- **`localhost`** — Matches any IP address in use by the local system
- **`localnets`** — Matches any IP address on any network to which the local system is connected
- **none** — Matches no IP addresses

When used in conjunction with other statements (such as the **`options`** statement), **`acl`** statements can be very useful in preventing the misuse of a BIND nameserver.

The following example defines two access control lists and uses an **`options`** statement to define how they are treated by the nameserver:

```
acl black-hats {
 10.0.2.0/24;
 192.168.0.0/24;
};
acl red-hats {
 10.0.1.0/24;
};
options {
 blackhole { black-hats; };
 allow-query { red-hats; };
 allow-recursion { red-hats; };
};
```

This example contains two access control lists, **`black-hats`** and **`red-hats`**. Hosts in the **`black-hats`** list are denied access to the nameserver, while hosts in the **`red-hats`** list are given normal access.

19.2.1.2. `include` Statement

The **`include`** statement allows files to be included in a **`named.conf`** file. In this way, sensitive configuration data (such as **`keys`**) can be placed in a separate file with restrictive permissions.

An **`include`** statement takes the following form:

```
include "<file-name>"
```

In this statement, *<file-name>* is replaced with an absolute path to a file.

19.2.1.3. `options` Statement

The **`options`** statement defines global server configuration options and sets defaults for other statements. It can be used to specify the location of the **named** working directory, the types of queries allowed, and much more.

The **`options`** statement takes the following form:

```
options {
 <option>;
 [<option>; ...]
};
```

In this statement, the *<option>* directives are replaced with a valid option.

The following are commonly used options:

`allow-query`

Specifies which hosts are allowed to query this nameserver. By default, all hosts are allowed to query. An access control list, or collection of IP addresses or networks, may be used here to allow only particular hosts to query the nameserver.

`allow-recursion`

Similar to **`allow-query`**, this option applies to recursive queries. By default, all hosts are allowed to perform recursive queries on the nameserver.

`blackhole`

Specifies which hosts are not allowed to query the server.

`directory`

Specifies the **named** working directory if different from the default value, **`/var/named/`**.

`forwarders`

Specifies a list of valid IP addresses for nameservers where requests should be forwarded for resolution.

`forward`

Specifies the forwarding behavior of a **`forwarders`** directive.

The following options are accepted:

- **`first`** — Specifies that the nameservers listed in the **`forwarders`** directive be queried before **named** attempts to resolve the name itself.
- **`only`** — Specifies that **named** does not attempt name resolution itself in the event that queries to nameservers specified in the **`forwarders`** directive fail.

`listen-on`

Specifies the network interface on which **named** listens for queries. By default, all interfaces are used.

Using this directive on a DNS server which also acts a gateway, BIND can be configured to only answer queries that originate from one of the networks.

The following is an example of a **listen-on** directive:

```
options {
 listen-on { 10.0.1.1; };
};
```

In this example, only requests that arrive from the network interface serving the private network (**10.0.1.1**) are accepted.

notify

Controls whether **named** notifies the slave servers when a zone is updated. It accepts the following options:

- **yes** — Notifies slave servers.
- **no** — Does not notify slave servers.
- **explicit** — Only notifies slave servers specified in an **also-notify** list within a zone statement.

pid-file

Specifies the location of the process ID file created by **named**.

root-delegation-only

Turns on the enforcement of delegation properties in top-level domains (TLDs) and root zones with an optional exclude list. *Delegation* is the process of dividing a single zone into multiple subzones. In order to create a delegated zone, items known as *NS records* are used. NameServer records (delegation records) announce the authoritative nameservers for a particular zone.

The following **root-delegation-only** example specifies an exclude list of TLDs from whom undelegated responses are expected and trusted:

```
options {
 root-delegation-only exclude { "ad"; "ar"; "biz"; "cr"; "cu"; "de";
"dm"; "id";
   "lu"; "lv"; "md"; "ms"; "museum"; "name"; "no"; "pa";
   "pf"; "se"; "sr"; "to"; "tw"; "us"; "uy"; };
};
```

statistics-file

Specifies an alternate location for statistics files. By default, **named** statistics are saved to the **/var/named/named.stats** file.

There are several other options also available, many of which rely upon one another to work properly. Refer to the *BIND 9 Administrator Reference Manual* referenced in Section 19.7.1, “Installed Documentation” and the **bind.conf** man page for more details.

19.2.1.4. zone Statement

A **zone** statement defines the characteristics of a zone, such as the location of its configuration file and zone-specific options. This statement can be used to override the global **options** statements.

A **zone** statement takes the following form:

```
zone <zone-name> <zone-class> {
 <zone-options>;
 [<zone-options>; ...]
};
```

In this statement, *<zone-name>* is the name of the zone, *<zone-class>* is the optional class of the zone, and *<zone-options>* is a list of options characterizing the zone.

The *<zone-name>* attribute for the zone statement is particularly important. It is the default value assigned for the **$ORIGIN** directive used within the corresponding zone file located in the **/var/named/** directory. The **named** daemon appends the name of the zone to any non-fully qualified domain name listed in the zone file.

Note

If you have installed the **caching-nameserver** package, the default configuration file will be in **/etc/named.rfc1912.zones**.

For example, if a **zone** statement defines the namespace for **example.com**, use **example.com** as the *<zone-name>* so it is placed at the end of hostnames within the **example.com** zone file.

For more information about zone files, refer to Section 19.3, “Zone Files”.

The most common **zone** statement options include the following:

allow-query

Specifies the clients that are allowed to request information about this zone. The default is to allow all query requests.

allow-transfer

Specifies the slave servers that are allowed to request a transfer of the zone's information. The default is to allow all transfer requests.

allow-update

Specifies the hosts that are allowed to dynamically update information in their zone. The default is to deny all dynamic update requests.

Be careful when allowing hosts to update information about their zone. Do not enable this option unless the host specified is completely trusted. In general, it is better to have an administrator manually update the records for a zone and reload the **named** service.

file

Specifies the name of the file in the **named** working directory that contains the zone's configuration data.

masters

Specifies the IP addresses from which to request authoritative zone information and is used only if the zone is defined as **type slave**.

notify

Specifies whether or not **named** notifies the slave servers when a zone is updated. This directive accepts the following options:

- **`yes`** — Notifies slave servers.
- **`no`** — Does not notify slave servers.
- **`explicit`** — Only notifies slave servers specified in an **`also-notify`** list within a zone statement.

`type`

Defines the type of zone.

Below is a list of valid options:

- **`delegation-only`** — Enforces the delegation status of infrastructure zones such as COM, NET, or ORG. Any answer that is received without an explicit or implicit delegation is treated as **`NXDOMAIN`**. This option is only applicable in TLDs or root zone files used in recursive or caching implementations.
- **`forward`** — Forwards all requests for information about this zone to other nameservers.
- **`hint`** — A special type of zone used to point to the root nameservers which resolve queries when a zone is not otherwise known. No configuration beyond the default is necessary with a **`hint`** zone.
- **`master`** — Designates the nameserver as authoritative for this zone. A zone should be set as the **`master`** if the zone's configuration files reside on the system.
- **`slave`** — Designates the nameserver as a slave server for this zone. Also specifies the IP address of the master nameserver for the zone.

`zone-statistics`

Configures **named** to keep statistics concerning this zone, writing them to either the default location (**`/var/named/named.stats`**) or the file listed in the **`statistics-file`** option in the **`server`** statement. Refer to Section 19.2.2, “Other Statement Types” for more information about the **`server`** statement.

19.2.1.5. Sample `zone` Statements

Most changes to the **`/etc/named.conf`** file of a master or slave nameserver involves adding, modifying, or deleting **`zone`** statements. While these **`zone`** statements can contain many options, most nameservers require only a small subset to function efficiently. The following **`zone`** statements are very basic examples illustrating a master-slave nameserver relationship.

The following is an example of a **`zone`** statement for the primary nameserver hosting **`example.com`** (**`192.168.0.1`**):

```
zone "example.com" IN {
 type master;
 file "example.com.zone";
 allow-update { none; };
};
```

In the statement, the zone is identified as **example.com**, the type is set to **master**, and the **named** service is instructed to read the **/var/named/example.com.zone** file. It also tells **named** not to allow any other hosts to update.

A slave server's **zone** statement for **example.com** is slightly different from the previous example. For a slave server, the type is set to **slave** and in place of the **allow-update** line is a directive telling **named** the IP address of the master server.

The following is an example slave server **zone** statement for **example.com** zone:

```
zone "example.com" {
 type slave;
 file "example.com.zone";
 masters { 192.168.0.1; };
};
```

This **zone** statement configures **named** on the slave server to query the master server at the **192.168.0.1** IP address for information about the **example.com** zone. The information that the slave server receives from the master server is saved to the **/var/named/example.com.zone** file.

19.2.2. Other Statement Types

The following is a list of lesser used statement types available within **named.conf**:

controls

Configures various security requirements necessary to use the **rndc** command to administer the **named** service.

Refer to Section 19.4.1, "Configuring **/etc/named.conf** " to learn more about how the **controls** statement is structured and what options are available.

key "*<key-name>*"

Defines a particular key by name. Keys are used to authenticate various actions, such as secure updates or the use of the **rndc** command. Two options are used with **key**:

- **algorithm *<algorithm-name>*** — The type of algorithm used, such as **dsa** or **hmac-md5**.
- **secret "*<key-value>*"** — The encrypted key.

Refer to Section 19.4.2, "Configuring **/etc/rndc.conf** " for instructions on how to write a **key** statement.

logging

Allows for the use of multiple types of logs, called *channels*. By using the **channel** option within the **logging** statement, a customized type of log can be constructed — with its own file name (**file**), size limit (**size**), versioning (**version**), and level of importance (**severity**). Once a customized channel is defined, a **category** option is used to categorize the channel and begin logging when **named** is restarted.

By default, **named** logs standard messages to the **syslog** daemon, which places them in **/var/log/messages**. This occurs because several standard channels are built into BIND with various severity levels, such as **default_syslog** (which handles informational logging messages) and **`default_debug`** (which specifically handles debugging messages). A default category, called **`default`**, uses the built-in channels to do normal logging without any special configuration.

Customizing the logging process can be a very detailed process and is beyond the scope of this chapter. For information on creating custom BIND logs, refer to the *BIND 9 Administrator Reference Manual* referenced in Section 19.7.1, "Installed Documentation".

`server`

Specifies options that affect how **named** should respond to remote nameservers, especially with regard to notifications and zone transfers.

The **`transfer-format`** option controls whether one resource record is sent with each message (**`one-answer`**) or multiple resource records are sent with each message (**`many-answers`**). While **`many-answers`** is more efficient, only newer BIND nameservers understand it.

`trusted-keys`

Contains assorted public keys used for secure DNS (DNSSEC). Refer to Section 19.5.3, "Security" for more information concerning BIND security.

`view "<`*`view-name`*`>"`

Creates special views depending upon which network the host querying the nameserver is on. This allows some hosts to receive one answer regarding a zone while other hosts receive totally different information. Alternatively, certain zones may only be made available to particular trusted hosts while non-trusted hosts can only make queries for other zones.

Multiple views may be used, but their names must be unique. The **`match-clients`** option specifies the IP addresses that apply to a particular view. Any **`options`** statement may also be used within a view, overriding the global options already configured for **named**. Most **`view`** statements contain multiple **`zone`** statements that apply to the **`match-clients`** list. The order in which **`view`** statements are listed is important, as the first **`view`** statement that matches a particular client's IP address is used.

Refer to Section 19.5.2, "Multiple Views" for more information about the **`view`** statement.

19.2.3. Comment Tags

The following is a list of valid comment tags used within **`named.conf`**:

- **`//`** — When placed at the beginning of a line, that line is ignored by **named**.
- **`#`** — When placed at the beginning of a line, that line is ignored by **named**.
- **`/*`** and **`*/`** — When text is enclosed in these tags, the block of text is ignored by **named**.

19.3. Zone Files

Zone files contain information about a namespace and are stored in the **named** working directory (**`/var/named/`**) by default. Each zone file is named according to the **`file`** option data in the **`zone`** statement, usually in a way that relates to the domain in question and identifies the file as containing zone data, such as **`example.com.zone`**.

Note

If you have installed the **`bind-chroot`** package, the BIND service will run in the **`/var/named/chroot`** environment. All configuration files will be moved there. As such, you can find the zone files in **`/var/named/chroot/var/named`**.

Each zone file may contain *directives* and *resource records*. Directives tell the nameserver to perform tasks or apply special settings to the zone. Resource records define the parameters of the zone and assign identities to individual hosts. Directives are optional, but resource records are required to provide name service to a zone.

All directives and resource records should be entered on individual lines.

Comments can be placed after semicolon characters (**`;`**) in zone files.

19.3.1. Zone File Directives

Directives begin with the dollar sign character (**`$`**) followed by the name of the directive. They usually appear at the top of the zone file.

The following are commonly used directives:

`$INCLUDE`

Configures **`named`** to include another zone file in this zone file at the place where the directive appears. This allows additional zone settings to be stored apart from the main zone file.

`$ORIGIN`

Appends the domain name to unqualified records, such as those with the hostname and nothing more.

For example, a zone file may contain the following line:

```
$ORIGIN example.com.
```

Any names used in resource records that do not end in a trailing period (**`.`**) are appended with **`example.com`**.

Note

The use of the **`$ORIGIN`** directive is unnecessary if the zone is specified in **`/etc/named.conf`** because the zone name is used as the value for the **`$ORIGIN`** directive by default.

`$TTL`

Sets the default *Time to Live (TTL)* value for the zone. This is the length of time, in seconds, that a zone resource record is valid. Each resource record can contain its own TTL value, which overrides this directive.

Increasing this value allows remote nameservers to cache the zone information for a longer period of time, reducing the number of queries for the zone and lengthening the amount of time required tc proliferate resource record changes.

19.3.2. Zone File Resource Records

The primary component of a zone file is its resource records.

There are many types of zone file resource records. The following are used most frequently:

`A`

This refers to the Address record, which specifies an IP address to assign to a name, as in this example:

```
<host> IN A <IP-address>
```

If the *<host>* value is omitted, then an **`A`** record points to a default IP address for the top of the namespace. This system is the target for all non-FQDN requests.

Consider the following **`A`** record examples for the **`example.com`** zone file:

```
server1  IN A 10.0.1.3
  IN A 10.0.1.5
```

Requests for **`example.com`** are pointed to 10.0.1.3 or 10.0.1.5.

`CNAME`

This refers to the Canonical Name record, which maps one name to another. This type of record can also be referred to as an *alias record*.

The next example tells **`named`** that any requests sent to the *<alias-name>* should point to the host, *<real-name>*. **`CNAME`** records are most commonly used to point to services that use a common naming scheme, such as **`www`** for Web servers.

```
<alias-name> IN CNAME <real-name>
```

In the following example, an **`A`** record binds a hostname to an IP address, while a**`CNAME`** record points the commonly used **`www`** hostname to it.

```
server1  IN A 10.0.1.5
www  IN CNAME server1
```

`MX`

This refers to the Mail eXchange record, which tells where mail sent to a particular namespace controlled by this zone should go.

```
 IN MX <preference-value> <email-server-name>
```

Here, the *<preference-value>* allows numerical ranking of the email servers for a namespace, giving preference to some email systems over others. The **`MX`** resource record with the lowest *<preference-value>* is preferred over the others. However, multiple email servers can possess the same value to distribute email traffic evenly among them.

The *<email-server-name>* may be a hostname or FQDN.

```
IN     MX     10     mail.example.com.
IN     MX     20     mail2.example.com.
```

In this example, the first **`mail.example.com`** email server is preferred to the **`mail2.example.com`** email server when receiving email destined for the **`example.com`** domain.

`NS`

This refers to the NameServer record, which announces the authoritative nameservers for a particular zone.

The following illustrates the layout of an **`NS`** record:

```
IN NS <nameserver-name>
```

Here, *<nameserver-name>* should be an FQDN.

Next, two nameservers are listed as authoritative for the domain. It is not important whether these nameservers are slaves or if one is a master; they are both still considered authoritative.

```
IN     NS     dns1.example.com.
IN     NS     dns2.example.com.
```

`PTR`

This refers to the PoinTeR record, which is designed to point to another part of the namespace.

`PTR` records are primarily used for reverse name resolution, as they point IP addresses back to a particular name. Refer to Section 19.3.4, “Reverse Name Resolution Zone Files” for more examples of **`PTR`** records in use.

`SOA`

This refers to the Start Of Authority resource record, which proclaims important authoritative information about a namespace to the nameserver.

Located after the directives, an **`SOA`** resource record is the first resource record in a zone file.

The following shows the basic structure of an **`SOA`** resource record:

```
@  IN SOA  <primary-name-server>  <hostmaster-email> (
 <serial-number>
 <time-to-refresh>
 <time-to-retry>
 <time-to-expire>
 <minimum-TTL> )
```

The @ symbol places the **`$ORIGIN`** directive (or the zone's name, if the **`$ORIGIN`** directive is not set) as the namespace being defined by this **`SOA`** resource record. The hostname of the primary nameserver that is authoritative for this domain is the *<primary-name-server>* directive, and the email of the person to contact about this namespace is the *<hostmaster-email>* directive.

The *<serial-number>* directive is a numerical value incremented every time the zone file is altered to indicate it is time for **named** to reload the zone. The *<time-to-refresh>* directive is the numerical value slave servers use to determine how long to wait before asking the master nameserver if any changes have been made to the zone. The *<serial-number>* directive is a numerical value used by the slave servers to determine if it is using outdated zone data and should therefore refresh it.

The *<time-to-retry>* directive is a numerical value used by slave servers to determine the length of time to wait before issuing a refresh request in the event that the master nameserver is not answering. If the master has not replied to a refresh request before the amount of time specified in the *<time-to-expire>* directive elapses, the slave servers stop responding as an authority for requests concerning that namespace.

In BIND 4 and 8, the *<minimum-TTL>* directive is the amount of time other nameservers cache the zone's information. However, in BIND 9, the *<minimum-TTL>* directive defines how long negative answers are cached for. Caching of negative answers can be set to a maximum of 3 hours (**`3H`**).

When configuring BIND, all times are specified in seconds. However, it is possible to use abbreviations when specifying units of time other than seconds, such as minutes (**`M`**), hours (**`H`**), days (**`D`**), and weeks (**`W`**). The table in Table 19.1, "Seconds compared to other time units" shows an amount of time in seconds and the equivalent time in another format.

Table 19.1. Seconds compared to other time units

Seconds	Other Time Units
`60`	`1M`
`1800`	`30M`
`3600`	`1H`
`10800`	`3H`
`21600`	`6H`
`43200`	`12H`
`86400`	`1D`
`259200`	`3D`
`604800`	`1W`
`31536000`	`365D`

The following example illustrates the form an **`SOA`** resource record might take when it is populated with real values.

```
@ IN SOA dns1.example.com. hostmaster.example.com. (
  2001062501 ; serial
  21600      ; refresh after 6 hours
  3600       ; retry after 1 hour
  604800     ; expire after 1 week
  86400 )    ; minimum TTL of 1 day
```

19.3.3. Example Zone File

Seen individually, directives and resource records can be difficult to grasp. However, when placed together in a single file, they become easier to understand.

The following example shows a very basic zone file.

```
$ORIGIN example.com.
$TTL 86400
@  IN SOA dns1.example.com. hostmaster.example.com. (
   2001062501 ; serial
   21600      ; refresh after 6 hours
```

```
    3600       ; retry after 1 hour
    604800     ; expire after 1 week
    86400 )    ; minimum TTL of 1 day
;
;
  IN NS dns1.example.com.
  IN NS dns2.example.com.
dns1  IN A 10.0.1.1
  IN AAAA aaaa:bbbb::1
dns2  IN A 10.0.1.2
  IN AAAA aaaa:bbbb::2
;
;
@  IN MX 10 mail.example.com.
  IN MX 20 mail2.example.com.
mail  IN A 10.0.1.5
  IN AAAA aaaa:bbbb::5
mail2  IN A 10.0.1.6
  IN AAAA aaaa:bbbb::6
;
;
; This sample zone file illustrates sharing the same IP addresses
; for multiple services:
;
services IN A 10.0.1.10
  IN AAAA aaaa:bbbb::10
  IN A 10.0.1.11
  IN AAAA aaaa:bbbb::11
ftp  IN CNAME services.example.com.
www  IN CNAME services.example.com.
;
;
```

In this example, standard directives and **SOA** values are used. The authoritative nameservers are set as **dns1.example.com** and **dns2.example.com**, which have **A** records that tie them to **10.0.1.1** and **10.0.1.2**, respectively.

The email servers configured with the **MX** records point to **mail** and **mail2** via **A** records. Since the **mail** and **mail2** names do not end in a trailing period (**.**), the **$ORIGIN** domain is placed after them, expanding them to **mail.example.com** and **mail2.example.com**. Through the related **A** resource records, their IP addresses can be determined.

Services available at the standard names, such as **www.example.com** (WWW), are pointed at the appropriate servers using a **CNAME** record.

This zone file would be called into service with a **zone** statement in the **named.conf** similar to the following:

```
zone "example.com" IN {
 type master;
 file "example.com.zone";
 allow-update { none; };
};
```

19.3.4. Reverse Name Resolution Zone Files

A reverse name resolution zone file is used to translate an IP address in a particular namespace into an FQDN. It looks very similar to a standard zone file, except that **PTR** resource records are used to link the IP addresses to a fully qualified domain name.

The following illustrates the layout of a **PTR** record:

```
<last-IP-digit> IN PTR <FQDN-of-system>
```

The *<last-IP-digit>* is the last number in an IP address which points to a particular system's FQDN.

In the following example, IP addresses **10.0.1.1** through **10.0.1.6** are pointed to corresponding FQDNs. It can be located in **/var/named/example.com.rr.zone**.

```
$ORIGIN 1.0.10.in-addr.arpa.
$TTL 86400
@ IN SOA dns1.example.com. hostmaster.example.com. (
   2001062501 ; serial
   21600      ; refresh after 6 hours
   3600       ; retry after 1 hour
   604800     ; expire after 1 week
   86400 )    ; minimum TTL of 1 day
;
@ IN NS dns1.example.com.
;
1 IN PTR dns1.example.com.
2 IN PTR dns2.example.com.
;
5 IN PTR server1.example.com.
6 IN PTR server2.example.com.
;
3 IN PTR ftp.example.com.
4 IN PTR ftp.example.com.
```

This zone file would be called into service with a **zone** statement in the **named.conf** file similar to the following:

```
zone "1.0.10.in-addr.arpa" IN {
 type master;
 file "example.com.rr.zone";
 allow-update { none; };
};
```

There is very little difference between this example and a standard **zone** statement, except for the zone name. Note that a reverse name resolution zone requires the first three blocks of the IP address reversed followed by **.in-addr.arpa**. This allows the single block of IP numbers used in the reverse name resolution zone file to be associated with the zone.

19.4. Using `rndc`

BIND includes a utility called **rndc** which allows command line administration of the **named** daemon from the localhost or a remote host.

In order to prevent unauthorized access to the **named** daemon, BIND uses a shared secret key authentication method to grant privileges to hosts. This means an identical key must be present in both **/etc/named.conf** and the **rndc** configuration file, **/etc/rndc.conf**.

Note

If you have installed the **bind-chroot** package, the BIND service will run in the **/var/named/chroot** environment. All configuration files will be moved there. As such, the **rndc.conf** file is located in **/var/named/chroot/etc/rndc.conf**.

Note that since the **rndc** utility does not run in a **chroot** environment, **/etc/rndc.conf** is a symlink to **/var/named/chroot/etc/rndc.conf**.

19.4.1. Configuring /etc/named.conf

In order for **rndc** to connect to a **named** service, there must be a **controls** statement in the BIND server's **/etc/named.conf** file.

The **controls** statement, shown in the following example, allows **rndc** to connect from the localhost.

```
controls {
 inet 127.0.0.1
  allow { localhost; } keys { <key-name>; };
};
```

This statement tells **named** to listen on the default TCP port 953 of the loopback address and allow **rndc** commands coming from the localhost, if the proper key is given. The *<key-name>* specifies a name in the **key** statement within the **/etc/named.conf** file. The next example illustrates a sample **key** statement.

```
key "<key-name>" {
 algorithm hmac-md5;
 secret "<key-value>";
};
```

In this case, the *<key-value>* uses the HMAC-MD5 algorithm. Use the following command to generate keys using the HMAC-MD5 algorithm:

```
dnssec-keygen -a hmac-md5 -b <bit-length> -n HOST <key-file-name>
```

A key with at least a 256-bit length is a good idea. The actual key that should be placed in the *<key-value>* area can be found in the ***<key-file-name>*** file generated by this command.

Warning

Because **/etc/named.conf** is world-readable, it is advisable to place the **key** statement in a separate file, readable only by root, and then use an **include** statement to reference it. For example:

```
include "/etc/rndc.key";
```

19.4.1.1. Firewall Blocking Communication

If a firewall is blocking connections from the **named** daemon to other nameservers, the recommended best practice is to change the firewall settings whenever possible.

Warning

DNS resolvers, that are not configured to perform DNSSEC validation or that need to query DNS zones that are not protected by DNSSEC only, use a 16-bit transaction identifier (TXID) and the destination UDP port number to check whether the DNS reply was sent by the server they queried for DNS data.

Previously, BIND always used a fixed UDP source port when sending DNS queries. BIND used either a port configured using the **query-source** (and **query-source-v6**) directive, or one randomly chosen at startup. When a static query source port is used, TXID offers insufficient protection against spoofed replies and allows an attacker to efficiently perform *cache-poisoning* attacks. To address this issue, BIND was updated to allow the use of a randomly-selected source port for each DNS query, making it more difficult for an attacker to spoof replies, when the query packets cannot be detected. A security update [3] was released for all the affected Red Hat Enterprise Linux versions. Additionally, the default configuration provided by the *caching-nameserver* package was updated to no longer specify a fixed query source port.

When deploying BIND as a DNS resolver, ensure that BIND is not forced, by the aforementioned configuration directives, to use a fixed query source port. Your firewall configuration must also permit the use of random query source ports. Previously, it was common practice to configure BIND to use port **53** as a query source port, and only allow DNS queries from that port on the firewall.

19.4.2. Configuring /etc/rndc.conf

The **key** is the most important statement in **/etc/rndc.conf**.

```
key "<key-name>" {
 algorithm hmac-md5;
 secret "<key-value>";
};
```

The <*key-name*> and <*key-value*> should be exactly the same as their settings in **/etc/named.conf**.

To match the keys specified in the target server's **/etc/named.conf**, add the following lines to **/etc/rndc.conf**.

```
options {
 default-server  localhost;
 default-key     "<key-name>";
};
```

This directive sets a global default key. However, the **rndc** configuration file can also specify different keys for different servers, as in the following example:

```
server localhost {
 key  "<key-name>";
};
```

Important

Make sure that only the root user can read or write to the **/etc/rndc.conf** file.

For more information about the **/etc/rndc.conf** file, refer to the **rndc.conf** man page.

19.4.3. Command Line Options

An **rndc** command takes the following form:

```
rndc <options> <command> <command-options>
```

When executing **rndc** on a properly configured localhost, the following commands are available:

- **halt** — Stops the **named** service immediately.
- **querylog** — Logs all queries made to this nameserver.
- **refresh** — Refreshes the nameserver's database.
- **reload** — Reloads the zone files but keeps all other previously cached responses. This command also allows changes to zone files without losing all stored name resolutions.

  If changes made only affect a specific zone, reload only that specific zone by adding the name of the zone after the **reload** command.
- **stats** — Dumps the current **named** statistics to the **/var/named/named.stats** file.
- **stop** — Stops the server gracefully, saving any dynamic update and *Incremental Zone Transfers* (*IXFR*) data before exiting.

Occasionally, it may be necessary to override the default settings in the **/etc/rndc.conf** file. The following options are available:

- **-c *<configuration-file>*** — Specifies the alternate location of a configuration file.
- **-p *<port-number>*** — Specifies a port number to use for the **rndc** connection other than the default port 953.
- **-s *<server>*** — Specifies a server other than the **default-server** listed in **/etc/rndc.conf**.

- **-y *<key-name>*** — Specifies a key other than the **default-key** option in **/etc/rndc.conf**.

Additional information about these options can be found in the **rndc** man page.

19.5. Advanced Features of BIND

Most BIND implementations only use **named** to provide name resolution services or to act as an authority for a particular domain or sub-domain. However, BIND version 9 has a number of advanced features that allow for a more secure and efficient DNS service.

Warning

Some of these advanced features, such as DNSSEC, TSIG, and IXFR (which are defined in the following section), should only be used in network environments with nameservers that support the features. If the network environment includes non-BIND or older BIND nameservers, verify that each advanced feature is supported before attempting to use it.

All of the features mentioned are discussed in greater detail in the *BIND 9 Administrator Reference Manual* referenced in Section 19.7.1, "Installed Documentation".

19.5.1. DNS Protocol Enhancements

BIND supports Incremental Zone Transfers (IXFR), where a slave nameserver only downloads the updated portions of a zone modified on a master nameserver. The standard transfer process requires that the entire zone be transferred to each slave nameserver for even the smallest change. For very popular domains with very lengthy zone files and many slave nameservers, IXFR makes the notification and update process much less resource-intensive.

Note that IXFR is only available when using *dynamic updating* to make changes to master zone records. If manually editing zone files to make changes, Automatic Zone Transfer (AXFR) is used. More information on dynamic updating is available in the *BIND 9 Administrator Reference Manual* referenced in Section 19.7.1, "Installed Documentation".

19.5.2. Multiple Views

Through the use of the **view** statement in **named.conf**, BIND can present different information depending on which network a request originates from.

This is primarily used to deny sensitive DNS entries from clients outside of the local network, while allowing queries from clients inside the local network.

The **view** statement uses the **match-clients** option to match IP addresses or entire networks and give them special options and zone data.

19.5.3. Security

BIND supports a number of different methods to protect the updating and transfer of zones, on both master and slave nameservers:

DNSSEC

Short for *DNS SECurity*, this feature allows for zones to be cryptographically signed with a *zone key*.

In this way, the information about a specific zone can be verified as coming from a nameserver that has signed it with a particular private key, as long as the recipient has that nameserver's public key.

BIND version 9 also supports the SIG(0) public/private key method of message authentication.

TSIG

Short for *Transaction SIGnatures*, this feature allows a transfer from master to slave only after verifying that a shared secret key exists on both nameservers.

This feature strengthens the standard IP address-based method of transfer authorization. An attacker would not only need to have access to the IP address to transfer the zone, but they would also need to know the secret key.

BIND version 9 also supports *TKEY*, which is another shared secret key method of authorizing zone transfers.

19.5.4. IP version 6

BIND version 9 supports name service in IP version 6 (IPv6) environments through the use of **`A6`** zone records.

If the network environment includes both IPv4 and IPv6 hosts, use the **`lwresd`** lightweight resolver daemon on all network clients. This daemon is a very efficient, caching-only nameserver which understands the new **`A6`** and **`DNAME`** records used under IPv6. Refer to the **`lwresd`** man page for more information.

19.6. Common Mistakes to Avoid

It is very common for beginners to make mistakes when editing BIND configuration files. Be sure to avoid the following issues:

- *Take care to increment the serial number when editing a zone file.*

  If the serial number is not incremented, the master nameserver has the correct, new information, but the slave nameservers are never notified of the change and do not attempt to refresh their data of that zone.

- *Be careful to use ellipses and semi-colons correctly in the **`/etc/named.conf`** file.*

  An omitted semi-colon or unclosed ellipse section can cause **`named`** to refuse to start.

- *Remember to place periods (`.`) in zone files after all FQDNs and omit them on hostnames.*

  A period at the end of a domain name denotes a fully qualified domain name. If the period is omitted, then **`named`** appends the name of the zone or the **`$ORIGIN`** value to complete it.

- If a firewall is blocking connections from the **`named`** daemon to other nameservers, the recommended best practice is to change the firewall settings whenever possible. For important security information regarding fixed UDP source ports, refer to Section 19.4.1.1, “Firewall Blocking Communication”.

19.7. Additional Resources

The following sources of information provide additional resources regarding BIND.

19.7.1. Installed Documentation

BIND features a full range of installed documentation covering many different topics, each placed in its own subject directory. For each item below, replace <*version-number*> with the version of **`bind`** installed on the system:

`/usr/share/doc/bind-<`*`version-number`*`>/`

> This directory lists the most recent features.

`/usr/share/doc/bind-<`*`version-number`*`>/arm/`

> This directory contains the *BIND 9 Administrator Reference Manual* in HTML and SGML formats, which details BIND resource requirements, how to configure different types of nameservers, how to perform load balancing, and other advanced topics. For most new users of BIND, this is the best place to start.

`/usr/share/doc/bind-<`*`version-number`*`>/draft/`

> This directory contains assorted technical documents that review issues related to DNS service and propose some methods to address them.

`/usr/share/doc/bind-<`*`version-number`*`>/misc/`

> This directory contains documents designed to address specific advanced issues. Users of BIND version 8 should consult the **`migration`** document for specific changes they must make when moving to BIND 9. The **`options`** file lists all of the options implemented in BIND 9 that are used in **`/etc/named.conf`**.

`/usr/share/doc/bind-<`*`version-number`*`>/rfc/`

> This directory provides every RFC document related to BIND.

There are also a number of man pages for the various applications and configuration files involved with BIND. The following lists some of the more important man pages.

Administrative Applications

- **`man rndc`** — Explains the different options available when using the **`rndc`** command to control a BIND nameserver.

Server Applications

- **`man named`** — Explores assorted arguments that can be used to control the BIND nameserver daemon.
- **`man lwresd`** — Describes the purpose of and options available for the lightweight resolver daemon.

Configuration Files

- **`man named.conf`** — A comprehensive list of options available within the **`named`** configuration file.
- **`man rndc.conf`** — A comprehensive list of options available within the **`rndc`** configuration file.

19.7.2. Useful Websites

- http://www.isc.org/index.pl?/sw/bind/ — The home page of the BIND project containing information about current releases as well as a PDF version of the *BIND 9 Administrator Reference Manual*.

- http://www.redhat.com/mirrors/LDP/HOWTO/DNS-HOWTO.html — Covers the use of BIND as a resolving, caching nameserver and the configuration of various zone files necessary to serve as the primary nameserver for a domain.

19.7.3. Related Books

- *DNS and BIND* by Paul Albitz and Cricket Liu; O'Reilly & Associates — A popular reference that explains both common and esoteric BIND configuration options, as well as providing strategies for securing a DNS server.
- *The Concise Guide to DNS and BIND* by Nicolai Langfeldt; Que — Looks at the connection between multiple network services and BIND, with an emphasis on task-oriented, technical topics.

[3] The security update was RHSA-2008:0533.

Chapter 20. OpenSSH

SSH™ (or *S*ecure *SH*ell) is a protocol which facilitates secure communications between two systems using a client/server architecture and allows users to log into server host systems remotely. Unlike other remote communication protocols, such as FTP or Telnet, SSH encrypts the login session, rendering the connection difficult for intruders to collect unencrypted passwords.

SSH is designed to replace older, less secure terminal applications used to log into remote hosts, such as **`telnet`** or **`rsh`**. A related program called **`scp`** replaces older programs designed to copy files between hosts, such as **`rcp`**. Because these older applications do not encrypt passwords transmitted between the client and the server, avoid them whenever possible. Using secure methods to log into remote systems decreases the risks for both the client system and the remote host.

20.1. Features of SSH

The SSH protocol provides the following safeguards:

- After an initial connection, the client can verify that it is connecting to the same server it had connected to previously.
- The client transmits its authentication information to the server using strong, 128-bit encryption.
- All data sent and received during a session is transferred using 128-bit encryption, making intercepted transmissions extremely difficult to decrypt and read.
- The client can forward X11 [4] applications from the server. This technique, called *X11 forwarding*, provides a secure means to use graphical applications over a network.

Because the SSH protocol encrypts everything it sends and receives, it can be used to secure otherwise insecure protocols. Using a technique called *port forwarding*, an SSH server can become a conduit to securing otherwise insecure protocols, like POP, and increasing overall system and data security.

The OpenSSH server and client can also be configured to create a tunnel similar to a virtual private network for traffic between server and client machines.

Finally, OpenSSH servers and clients can be configured to authenticate using the GSSAPI implementation of the Kerberos network authentication protocol. For more information on configuring Kerberos authentication services, refer to Section 48.6, “Kerberos”.

Red Hat Enterprise Linux includes the general OpenSSH package (**`openssh`**) as well as the OpenSSH server (**`openssh-server`**) and client (**`openssh-clients`**) packages. Note, the OpenSSH packages require the OpenSSL package (**`openssl`**) which installs several important cryptographic libraries, enabling OpenSSH to provide encrypted communications.

20.1.1. Why Use SSH?

Nefarious computer users have a variety of tools at their disposal enabling them to disrupt, intercept, and re-route network traffic in an effort to gain access to a system. In general terms, these threats can be categorized as follows:

- *Interception of communication between two systems* — In this scenario, the attacker can be somewhere on the network between the communicating parties, copying any information passed between them. The attacker may intercept and keep the information, or alter the information and send it on to the intended recipient.

  This attack can be mounted through the use of a packet sniffer — a common network utility.

- *Impersonation of a particular host* — Using this strategy, an attacker's system is configured to pose as the intended recipient of a transmission. If this strategy works, the user's system remains unaware that it is communicating with the wrong host.

  This attack can be mounted through techniques known as DNS poisoning [5] or IP spoofing [6].

Both techniques intercept potentially sensitive information and, if the interception is made for hostile reasons, the results can be disastrous.

If SSH is used for remote shell login and file copying, these security threats can be greatly diminished. This is because the SSH client and server use digital signatures to verify their identity. Additionally, all communication between the client and server systems is encrypted. Attempts to spoof the identity of either side of a communication does not work, since each packet is encrypted using a key known only by the local and remote systems.

20.2. SSH Protocol Versions

The SSH protocol allows any client and server programs built to the protocol's specifications to communicate securely and to be used interchangeably.

Two varieties of SSH (version 1 and version 2) currently exist. The OpenSSH suite under Red Hat Enterprise Linux uses SSH version 2 which has an enhanced key exchange algorithm not vulnerable to the exploit in version 1. However, the OpenSSH suite does support version 1 connections.

Important

It is recommended that only SSH version 2-compatible servers and clients are used whenever possible.

20.3. Event Sequence of an SSH Connection

The following series of events help protect the integrity of SSH communication between two hosts.

1. A cryptographic handshake is made so that the client can verify that it is communicating with the correct server.
2. The transport layer of the connection between the client and remote host is encrypted using a symmetric cipher.
3. The client authenticates itself to the server.
4. The remote client interacts with the remote host over the encrypted connection.

20.3.1. Transport Layer

The primary role of the transport layer is to facilitate safe and secure communication between the two hosts a the time of authentication and during subsequent communication. The transport layer accomplishes this by handling the encryption and decryption of data, and by providing integrity protection of data packets as they are sent and received. The transport layer also provides compression, speeding the transfer of information.

Once an SSH client contacts a server, key information is exchanged so that the two systems can correctly construct the transport layer. The following steps occur during this exchange:

- Keys are exchanged
- The public key encryption algorithm is determined
- The symmetric encryption algorithm is determined
- The message authentication algorithm is determined
- The hash algorithm is determined

During the key exchange, the server identifies itself to the client with a unique *host key*. If the client has never communicated with this particular server before, the server's host key is unknown to the client and it does not connect. OpenSSH gets around this problem by accepting the server's host key. This is done after the user is notified and has both accepted and verified the new host key. In subsequent connections, the server's host key is checked against the saved version on the client, providing confidence that the client is indeed communicating with the intended server. If, in the future, the host key no longer matches, the user must remove the client's saved version before a connection can occur.

Warning

It is possible for an attacker to masquerade as an SSH server during the initial contact since the local system does not know the difference between the intended server and a false one set up by an attacker. To help prevent this, verify the integrity of a new SSH server by contacting the server administrator before connecting for the first time or in the event of a host key mismatch.

SSH is designed to work with almost any kind of public key algorithm or encoding format. After an initial key exchange creates a hash value used for exchanges and a shared secret value, the two systems immediately begin calculating new keys and algorithms to protect authentication and future data sent over the connection.

After a certain amount of data has been transmitted using a given key and algorithm (the exact amount depends on the SSH implementation), another key exchange occurs, generating another set of hash values and a new shared secret value. Even if an attacker is able to determine the hash and shared secret value, this information is only useful for a limited period of time.

20.3.2. Authentication

Once the transport layer has constructed a secure tunnel to pass information between the two systems, the server tells the client the different authentication methods supported, such as using a private key-encoded signature or typing a password. The client then tries to authenticate itself to the server using one of these supported methods.

SSH servers and clients can be configured to allow different types of authentication, which gives each side the optimal amount of control. The server can decide which encryption methods it supports based on its security model, and the client can choose the order of authentication methods to attempt from the available options.

20.3.3. Channels

After a successful authentication over the SSH transport layer, multiple channels are opened via a technique called *multiplexing* [7]. Each of these channels handles communication for different terminal sessions and for forwarded X11 sessions.

Both clients and servers can create a new channel. Each channel is then assigned a different number on each end of the connection. When the client attempts to open a new channel, the clients sends the channel number along with the request. This information is stored by the server and is used to direct communication to that channel. This is done so that different types of sessions do not affect one another and so that when a given session ends, its channel can be closed without disrupting the primary SSH connection.

Channels also support *flow-control*, which allows them to send and receive data in an orderly fashion. In this way, data is not sent over the channel until the client receives a message that the channel is open.

The client and server negotiate the characteristics of each channel automatically, depending on the type of service the client requests and the way the user is connected to the network. This allows great flexibility in handling different types of remote connections without having to change the basic infrastructure of the protocol.

20.4. Configuring an OpenSSH Server

To run an OpenSSH server, you must first make sure that you have the proper RPM packages installed. The **`openssh-server`** package is required and is dependent on the **`openssh`** package.

The OpenSSH daemon uses the configuration file **`/etc/ssh/sshd_config`**. The default configuration file should be sufficient for most purposes. If you want to configure the daemon in ways not provided by the default **`sshd_config`**, read the **`sshd`** man page for a list of the keywords that can be defined in the configuration file.

To start the OpenSSH service, use the command **`/sbin/service sshd start`**. To stop the OpenSSH server, use the command **`/sbin/service sshd stop`**. If you want the daemon to start automatically at boot time, refer to Chapter 18, *Controlling Access to Services* for information on how to manage services.

If you reinstall, the reinstalled system creates a new set of identification keys. Any clients who had connected to the system with any of the OpenSSH tools before the reinstall will see the following message:

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that the RSA host key has just been changed.
```

If you want to keep the host keys generated for the system, backup the **`/etc/ssh/ssh_host*key*`** files and restore them after the reinstall. This process retains the system's identity, and when clients try to connect to the system after the reinstall, they will not receive the warning message.

20.4.1. Requiring SSH for Remote Connections

For SSH to be truly effective, using insecure connection protocols, such as Telnet and FTP, should be prohibited. Otherwise, a user's password may be protected using SSH for one session, only to be captured later while logging in using Telnet.

Some services to disable include:

- **`telnet`**
- **`rsh`**
- **`rlogin`**
- **`vsftpd`**

To disable insecure connection methods to the system, use the command line program **`chkconfig`**, the ncurses-based program **/usr/sbin/ntsysv**, or the **Services Configuration Tool** (**`system-config-services`**) graphical application. All of these tools require root level access.

For more information on runlevels and configuring services with **`chkconfig`**, **/usr/sbin/ntsysv**, and the **Services Configuration Tool**, refer to Chapter 18, *Controlling Access to Services*.

20.5. OpenSSH Configuration Files

OpenSSH has two different sets of configuration files: one for client programs (**`ssh`**, **`scp`**, and **`sftp`**) and one for the server daemon (**`sshd`**).

System-wide SSH configuration information is stored in the **`/etc/ssh/`** directory:

- **`moduli`** — Contains Diffie-Hellman groups used for the Diffie-Hellman key exchange which is critical for constructing a secure transport layer. When keys are exchanged at the beginning of an SSH session, a shared, secret value is created which cannot be determined by either party alone. This value is then used to provide host authentication.
- **`ssh_config`** — The system-wide default SSH client configuration file. It is overridden if one is also present in the user's home directory (**`~/.ssh/config`**).
- **`sshd_config`** — The configuration file for the **`sshd`** daemon.
- **`ssh_host_dsa_key`** — The DSA private key used by the **`sshd`** daemon.
- **`ssh_host_dsa_key.pub`** — The DSA public key used by the **`sshd`** daemon.
- **`ssh_host_key`** — The RSA private key used by the **`sshd`** daemon for version 1 of the SSH protocol.
- **`ssh_host_key.pub`** — The RSA public key used by the **`sshd`** daemon for version 1 of the SSH protocol.
- **`ssh_host_rsa_key`** — The RSA private key used by the **`sshd`** daemon for version 2 of the SSH protocol.
- **`ssh_host_rsa_key.pub`** — The RSA public key used by the **`sshd`** for version 2 of the SSH protocol.

User-specific SSH configuration information is stored in the user's home directory within the **`~/.ssh/`** directory:

- **`authorized_keys`** — This file holds a list of authorized public keys for servers. When the client connects to a server, the server authenticates the client by checking its signed public key stored within this file.
- **`id_dsa`** — Contains the DSA private key of the user.
- **`id_dsa.pub`** — The DSA public key of the user.
- **`id_rsa`** — The RSA private key used by **`ssh`** for version 2 of the SSH protocol.
- **`id_rsa.pub`** — The RSA public key used by **`ssh`** for version 2 of the SSH protocol
- **`identity`** — The RSA private key used by **`ssh`** for version 1 of the SSH protocol.
- **`identity.pub`** — The RSA public key used by **`ssh`** for version 1 of the SSH protocol.

- **`known_hosts`** — This file contains DSA host keys of SSH servers accessed by the user. This file is very important for ensuring that the SSH client is connecting the correct SSH server.

> **Important**
>
> If an SSH server's host key has changed, the client notifies the user that the connection cannot proceed until the server's host key is deleted from the **`known_hosts`** file using a text editor. Before doing this, however, contact the system administrator of the SSH server to verify the server is not compromised.

Refer to the **`ssh_config`** and **`sshd_config`** man pages for information concerning the various directives available in the SSH configuration files.

20.6. Configuring an OpenSSH Client

To connect to an OpenSSH server from a client machine, you must have the **`openssh-clients`** and **`openssh`** packages installed on the client machine.

20.6.1. Using the `ssh` Command

The **`ssh`** command is a secure replacement for the **`rlogin`**, **`rsh`**, and **`telnet`** commands. It allows you to log in to a remote machine as well as execute commands on a remote machine.

Logging in to a remote machine with **`ssh`** is similar to using **`telnet`**. To log in to a remote machine named penguin.example.net, type the following command at a shell prompt:

```
ssh penguin.example.net
```

The first time you **`ssh`** to a remote machine, you will see a message similar to the following:

```
The authenticity of host 'penguin.example.net' can't be established.
DSA key fingerprint is 94:68:3a:3a:bc:f3:9a:9b:01:5d:b3:07:38:e2:11:0c.
Are you sure you want to continue connecting (yes/no)?
```

Type **`yes`** to continue. This will add the server to your list of known hosts (**`~/.ssh/known_hosts`**) as seen in the following message:

```
Warning: Permanently added 'penguin.example.net' (RSA) to the list of known
hosts.
```

Next, you will see a prompt asking for your password for the remote machine. After entering your password, you will be at a shell prompt for the remote machine. If you do not specify a username the username that you are logged in as on the local client machine is passed to the remote machine. If you want to specify a different username, use the following command:

```
ssh username@penguin.example.net
```

You can also use the syntax **`ssh -l`** ***`username`*** **`penguin.example.net`**.

The **`ssh`** command can be used to execute a command on the remote machine without logging in to a shell prompt. The syntax is **`ssh`** ***`hostname command`***. For example, if you want to execute the command **`ls`**

`/usr/share/doc` on the remote machine penguin.example.net, type the following command at a shell prompt:

```
ssh penguin.example.net ls /usr/share/doc
```

After you enter the correct password, the contents of the remote directory **`/usr/share/doc`** will be displayed, and you will return to your local shell prompt.

20.6.2. Using the `scp` Command

The **`scp`** command can be used to transfer files between machines over a secure, encrypted connection. It is similar to **`rcp`**.

The general syntax to transfer a local file to a remote system is as follows:

```
scp <localfile> username@tohostname:<remotefile>
```

The *<localfile>* specifies the source including path to the file, such as **`/var/log/maillog`**. The *<remotefile>* specifies the destination, which can be a new filename such as **`/tmp/hostname-maillog`**. For the remote system, if you do not have a preceding **`/`**, the path will be relative to the home directory of*username*, typically **`/home/username/`**.

To transfer the local file **`shadowman`** to the home directory of your account on penguin.example.net, type the following at a shell prompt (replace *username* with your username):

```
scp shadowman username@penguin.example.net:shadowman
```

This will transfer the local file **`shadowman`** to **`/home/`*`username`*`/shadowman`** on penguin.example.net. Alternately, you can leave off the final **`shadowman`** in the **`scp`** command.

The general syntax to transfer a remote file to the local system is as follows:

```
scp username@tohostname:<remotefile> <newlocalfile>
```

The *<remotefile>* specifies the source including path, and *<newlocalfile>* specifies the destination including path.

Multiple files can be specified as the source files. For example, to transfer the contents of the directory **`downloads/`** to an existing directory called **`uploads/`** on the remote machine penguin.example.net, type the following at a shell prompt:

```
scp downloads/* username@penguin.example.net:uploads/
```

20.6.3. Using the `sftp` Command

The **`sftp`** utility can be used to open a secure, interactive FTP session. It is similar to **`ftp`** except that it uses a secure, encrypted connection. The general syntax is **`sftp`** ***`username@hostname.com`***. Once authenticated, you can use a set of commands similar to those used by FTP. Refer to the **`sftp`** man page for a list of these commands. To read the man page, execute the command **`man sftp`** at a shell prompt. The **`sftp`** utility is only available in OpenSSH version 2.5.0p1 and higher.

20.7. More Than a Secure Shell

A secure command line interface is just the beginning of the many ways SSH can be used. Given the proper amount of bandwidth, X11 sessions can be directed over an SSH channel. Or, by using TCP/IP forwarding, previously insecure port connections between systems can be mapped to specific SSH channels.

20.7.1. X11 Forwarding

Opening an X11 session over an SSH connection is as easy as connecting to the SSH server using the **-Y** option and running an X program on a local machine.

```
ssh -Y <user>@example.com
```

When an X program is run from the secure shell prompt, the SSH client and server create a new secure channel, and the X program data is sent over that channel to the client machine transparently.

X11 forwarding can be very useful. For example, X11 forwarding can be used to create a secure, interactive session of the **Printer Configuration Tool**. To do this, connect to the server using **ssh** and type:

```
system-config-printer &
```

After supplying the root password for the server, the **Printer Configuration Tool** appears and allows the remote user to safely configure printing on the remote system.

20.7.2. Port Forwarding

SSH can secure otherwise insecure TCP/IP protocols via port forwarding. When using this technique, the SSH server becomes an encrypted conduit to the SSH client.

Port forwarding works by mapping a local port on the client to a remote port on the server. SSH can map any port from the server to any port on the client; port numbers do not need to match for this technique to work.

To create a TCP/IP port forwarding channel which listens for connections on the localhost, use the following command:

```
ssh -L local-port:remote-hostname:remote-port username@hostname
```

Note

Setting up port forwarding to listen on ports below 1024 requires root level access.

To check email on a server called **mail.example.com** using POP3 through an encrypted connection, use the following command:

```
ssh -L 1100:mail.example.com:110 mail.example.com
```

Once the port forwarding channel is in place between the client machine and the mail server, direct a POP3 mail client to use port 1100 on the localhost to check for new mail. Any requests sent to port 1100 on the client system are directed securely to the **mail.example.com** server.

If **mail.example.com** is not running an SSH server, but another machine on the same network is, SSH can still be used to secure part of the connection. However, a slightly different command is necessary:

```
ssh -L 1100:mail.example.com:110 other.example.com
```

In this example, POP3 requests from port 1100 on the client machine are forwarded through the SSH connection on port 22 to the SSH server, **`other.example.com`**. Then, **`other.example.com`** connects to port 110 on **`mail.example.com`** to check for new mail. Note, when using this technique only the connection between the client system and **`other.example.com`** SSH server is secure.

Port forwarding can also be used to get information securely through network firewalls. If the firewall is configured to allow SSH traffic via its standard port (22) but blocks access to other ports, a connection between two hosts using the blocked ports is still possible by redirecting their communication over an established SSH connection.

Note

Using port forwarding to forward connections in this manner allows any user on the client system to connect to that service. If the client system becomes compromised, the attacker also has access to forwarded services.

System administrators concerned about port forwarding can disable this functionality on the server by specifying a **`No`** parameter for the **`AllowTcpForwarding`** line in **`/etc/ssh/sshd_config`** and restarting the **`sshd`** service.

20.7.3. Generating Key Pairs

If you do not want to enter your password every time you use **`ssh`**, **`scp`**, or **`sftp`** to connect to a remote machine, you can generate an authorization key pair.

Keys must be generated for each user. To generate keys for a user, use the following steps as the user who wants to connect to remote machines. If you complete the steps as root, only root will be able to use the keys.

Starting with OpenSSH version 3.0, **`~/.ssh/authorized_keys2`**, **`~/.ssh/known_hosts2`**, and **`/etc/ssh_known_hosts2`** are obsolete. SSH Protocol 1 and 2 share the **`~/.ssh/authorized_keys`**, **`~/.ssh/known_hosts`**, and **`/etc/ssh/ssh_known_hosts`** files.

Red Hat Enterprise Linux 5.10 uses SSH Protocol 2 and RSA keys by default.

Note

If you reinstall and want to save your generated key pair, backup the **`.ssh`** directory in your home directory. After reinstalling, copy this directory back to your home directory. This process can be done for all users on your system, including root.

20.7.3.1. Generating an RSA Key Pair for Version 2

Use the following steps to generate an RSA key pair for version 2 of the SSH protocol. This is the default starting with OpenSSH 2.9.

1. To generate an RSA key pair to work with version 2 of the protocol, type the following command at a shell prompt:

   ```
   ssh-keygen -t rsa
   ```

Accept the default file location of **~/.ssh/id_rsa**. Enter a passphrase different from your account password and confirm it by entering it again.

The public key is written to **~/.ssh/id_rsa.pub**. The private key is written to **~/.ssh/id_rsa**. Never distribute your private key to anyone.

2. Change the permissions of the **.ssh** directory using the following command:

```
chmod 755 ~/.ssh
```

3. Copy the contents of **~/.ssh/id_rsa.pub** into the file **~/.ssh/authorized_keys** on the machine to which you want to connect. If the file **~/.ssh/authorized_keys** exist, append the contents of the file **~/.ssh/id_rsa.pub** to the file **~/.ssh/authorized_keys** on the other machine.

4. Change the permissions of the **authorized_keys** file using the following command:

```
chmod 644 ~/.ssh/authorized_keys
```

5. If you are running GNOME or are running in a graphical desktop with GTK2+ libraries installed, skip to Section 20.7.3.4, “Configuring **ssh-agent** with a GUI”. If you are not running the X Window System, skip to Section 20.7.3.5, “Configuring **ssh-agent**”.

20.7.3.2. Generating a DSA Key Pair for Version 2

Use the following steps to generate a DSA key pair for version 2 of the SSH Protocol.

1. To generate a DSA key pair to work with version 2 of the protocol, type the following command at a shell prompt:

```
ssh-keygen -t dsa
```

Accept the default file location of **~/.ssh/id_dsa**. Enter a passphrase different from your account password and confirm it by entering it again.

Note

A passphrase is a string of words and characters used to authenticate a user. Passphrases differ from passwords in that you can use spaces or tabs in the passphrase. Passphrases are generally longer than passwords because they are usually phrases instead of a single word.

The public key is written to **~/.ssh/id_dsa.pub**. The private key is written to **~/.ssh/id_dsa**. It is important never to give anyone the private key.

2. Change the permissions of the **.ssh** directory with the following command:

```
chmod 755 ~/.ssh
```

3. Copy the contents of **~/.ssh/id_dsa.pub** into the file **~/.ssh/authorized_keys** on the machine to which you want to connect. If the file **~/.ssh/authorized_keys** exist, append the contents of the file **~/.ssh/id_dsa.pub** to the file **~/.ssh/authorized_keys** on the other machine.

4. Change the permissions of the **authorized_keys** file using the following command:

   ```
   chmod 644 ~/.ssh/authorized_keys
   ```

5. If you are running GNOME or a graphical desktop environment with the GTK2+ libraries installed, skip to Section 20.7.3.4, “Configuring **ssh-agent** with a GUI”. If you are not running the X Window System, skip to Section 20.7.3.5, “Configuring **ssh-agent**”.

20.7.3.3. Generating an RSA Key Pair for Version 1.3 and 1.5

Use the following steps to generate an RSA key pair, which is used by version 1 of the SSH Protocol. If you are only connecting between systems that use DSA, you do not need an RSA version 1.3 or RSA version 1.5 key pair.

1. To generate an RSA (for version 1.3 and 1.5 protocol) key pair, type the following command at a shell prompt:

   ```
   ssh-keygen -t rsa1
   ```

   Accept the default file location (**~/.ssh/identity**). Enter a passphrase different from your account password. Confirm the passphrase by entering it again.

   The public key is written to **~/.ssh/identity.pub**. The private key is written to **~/.ssh/identity**. Do not give anyone the private key.

2. Change the permissions of your **.ssh** directory and your key with the commands **chmod 755 ~/.ssh** and **chmod 644 ~/.ssh/identity.pub**.

3. Copy the contents of **~/.ssh/identity.pub** into the file **~/.ssh/authorized_keys** on the machine to which you wish to connect. If the file **~/.ssh/authorized_keys** does not exist, you can copy the file **~/.ssh/identity.pub** to the file **~/.ssh/authorized_keys** on the remote machine.

4. If you are running GNOME, skip to Section 20.7.3.4, “Configuring **ssh-agent** with a GUI”. If you are not running GNOME, skip to Section 20.7.3.5, “Configuring **ssh-agent**”.

20.7.3.4. Configuring ssh-agent with a GUI

The **ssh-agent** utility can be used to save your passphrase so that you do not have to enter it each time you initiate an **ssh** or **scp** connection. If you are using GNOME, the **gnome-ssh-askpass** package contains the application used to prompt you for your passphrase when you log in to GNOME and save it until you log out of GNOME. You will not have to enter your password or passphrase for any **ssh** or **scp** connection made during that GNOME session. If you are not using GNOME, refer to Section 20.7.3.5, “Configuring **ssh-agent**”.

To save your passphrase during your GNOME session, follow the following steps:

1. You will need to have the package **gnome-ssh-askpass** installed; you can use the command **rpm -q openssh-askpass** to determine if it is installed or not. If it is not installed, install it from your Red Hat Enterprise Linux CD-ROM set, from a Red Hat FTP mirror site, or using Red Hat Network.

2. Select **Main Menu Button** (on the Panel) > **Preferences** > **More Preferences** > **`Sessions`**, and click on the **`Startup Programs`** tab. Click **Add** and enter **`/usr/bin/ssh-add`** in the **`Startup Command`** text area. Set it a priority to a number higher than any existing commands to ensure that it is executed last. A good priority number for **`ssh-add`** is 70 or higher. The higher the priority number, the lower the priority. If you have other programs listed, this one should have the lowest priority. Click **`Close`** to exit the program.

3. Log out and then log back into GNOME; in other words, restart X. After GNOME is started, a dialog box will appear prompting you for your passphrase(s). Enter the passphrase requested. If you have both DSA and RSA key pairs configured, you will be prompted for both. From this point on, you should not be prompted for a password by **`ssh`**, **`scp`**, or **`sftp`**.

20.7.3.5. Configuring `ssh-agent`

The **`ssh-agent`** can be used to store your passphrase so that you do not have to enter it each time you make a **`ssh`** or **`scp`** connection. If you are not running the X Window System, follow these steps from a shell prompt. If you are running GNOME but you do not want to configure it to prompt you for your passphrase when you log in (refer to Section 20.7.3.4, “Configuring **`ssh-agent`** with a GUI”), this procedure will work in a terminal window, such as an XTerm. If you are running X but not GNOME, this procedure will work in a terminal window. However, your passphrase will only be remembered for that terminal window; it is not a global setting.

1. At a shell prompt, type the following command:

```
exec /usr/bin/ssh-agent $SHELL
```

2. Then type the command:

```
ssh-add
```

   and enter your passphrase(s). If you have more than one key pair configured, you will be prompted for each one.

3. When you log out, your passphrase(s) will be forgotten. You must execute these two commands each time you log in to a virtual console or open a terminal window.

20.8. Additional Resources

The OpenSSH and OpenSSL projects are in constant development, and the most up-to-date information for them is available from their websites. The man pages for OpenSSH and OpenSSL tools are also good sources of detailed information.

20.8.1. Installed Documentation

- The **`ssh`**, **`scp`**, **`sftp`**, **`sshd`**, and **`ssh-keygen`** man pages — These man pages include information on how to use these commands as well as all the parameters that can be used with them.

20.8.2. Useful Websites

- http://www.openssh.com/ — The OpenSSH FAQ page, bug reports, mailing lists, project goals, and a more technical explanation of the security features.

- http://www.openssl.org/ — The OpenSSL FAQ page, mailing lists, and a description of the project goal.

- http://www.freesshd.com/ — SSH client software for other platforms.

---

[4] X11 refers to the X11R7 windowing display system, traditionally referred to as the X Window System or X. Red Hat Enterprise Linux includes X11R7, an open source X Window System.

[5] DNS poisoning occurs when an intruder cracks a DNS server, pointing client systems to a maliciously duplicated host.

[6] IP spoofing occurs when an intruder sends network packets which falsely appear to be from a trusted host on the network.

[7] A multiplexed connection consists of several signals being sent over a shared, common medium. With SSH, different channels are sent over a common secure connection.

Chapter 21. Network File System (NFS)

A *Network File System* (NFS) allows remote hosts to mount file systems over a network and interact with those file systems as though they are mounted locally. This enables system administrators to consolidate resources onto centralized servers on the network.

This chapter focuses on fundamental NFS concepts and supplemental information.

21.1. How It Works

Currently, there are three versions of NFS. NFS version 2 (NFSv2) is older and is widely supported. NFS version 3 (NFSv3) has more features, including 64bit file handles, Safe Async writes and more robust error handling. NFS version 4 (NFSv4) works through firewalls and on the Internet, no longer requires portmapper, supports ACLs, and utilizes stateful operations. Red Hat Enterprise Linux supports NFSv2, NFSv3, and NFSv4 clients, and when mounting a file system via NFS, Red Hat Enterprise Linux uses NFSv3 by default, if the server supports it.

All versions of NFS can use *Transmission Control Protocol* (TCP) running over an IP network, with NFSv4 requiring it. NFSv2 and NFSv3 can use the *User Datagram Protocol* (UDP) running over an IP network to provide a stateless network connection between the client and server.

When using NFSv2 or NFSv3 with UDP, the stateless UDP connection under normal conditions has less Protocol overhead than TCP which can translate into better performance on very clean, non-congested networks. The NFS server sends the client a file handle after the client is authorized to access the shared volume. This file handle is an opaque object stored on the server's side and is passed along with RPC requests from the client. The NFS server can be restarted without affecting the clients and the cookie remains intact. However, because UDP is stateless, if the server goes down unexpectedly, UDP clients continue to saturate the network with requests for the server. For this reason, TCP is the preferred protocol when connecting to an NFS server.

Because protocol support has been incorporated into the v4 protocol, NFSv4 has no interaction with the **`portmap`**, **`rpc.lockd`**, and **`rpc.statd`** daemons. NFSv4 listens on the well-known TCP port 2049, which eliminates the need for **`portmap`** interaction. The mounting and locking protocols have been incorporated into the V4 protocol which eliminates the need for interaction with **`rpc.lockd`** and **`rpc.statd`**. The **`rpc.mountd`** daemon is still required on the server, but is not involved in any over-the-wire operations.

Note

TCP is the default transport protocol for NFS under Red Hat Enterprise Linux. UDP can be used for compatibility purposes as needed, but is not recommended for wide usage.

All the RPC/NFS daemon have a **`-p`** command line option that can set the port, making firewall configuration easier.

After the client is granted access by TCP wrappers, the NFS server refers to its configuration file, **`/etc/exports`**, to determine whether the client is allowed to access any of the exported file systems. Once access is granted, all file and directory operations are available to the user.

Important

In order for NFS to work with a default installation of Red Hat Enterprise Linux with a firewall enabled, IPTables with the default TCP port 2049 must be configured. Without proper IPTables configuration, NFS does not function properly.

The NFS initialization script and **`rpc.nfsd`** process now allow binding to any specified port during system start up. However, this can be error prone if the port is unavailable or conflicts with another daemon.

21.1.1. Required Services

Red Hat Enterprise Linux uses a combination of kernel-level support and daemon processes to provide NFS file sharing. All NFS versions rely on *Remote Procedure Calls* (RPC) between clients and servers. RPC services under Linux are controlled by the **`portmap`** service. To share or mount NFS file systems, the following services work together, depending on which version of NFS is implemented:

- **`nfs`** — (**`/sbin/service nfs start`**) starts the NFS server and the appropriate RPC processes to service requests for shared NFS file systems.
- **`nfslock`** — (**`/sbin/service nfslock start`**) is a mandatory service that starts the appropriate RPC processes to allow NFS clients to lock files on the server.
- **`portmap`** — accepts port reservations from local RPC services. These ports are then made available (or advertised) so the corresponding remote RPC services access them. **`portmap`** responds to requests for RPC services and sets up connections to the requested RPC service.

The following RPC processes facilitate NFS services:

- **`rpc.mountd`** — This process receives mount requests from NFS clients and verifies the requested file system is currently exported. This process is started automatically by the **`nfs`** service and does not require user configuration.
- **`rpc.nfsd`** — Allows explicit NFS versions and protocols the server advertises to be defined. It works with the Linux kernel to meet the dynamic demands of NFS clients, such as providing server threads each time an NFS client connects. This process corresponds to the **`nfs`** service.
- **`rpc.lockd`** — allows NFS clients to lock files on the server. If **`rpc.lockd`** is not started, file locking will fail. **`rpc.lockd`** implements the *Network Lock Manager (NLM)* protocol. This process corresponds to the **`nfslock`** service. This is not used with NFSv4.
- **`rpc.statd`** — This process implements the *Network Status Monitor (NSM)* RPC protocol which notifies NFS clients when an NFS server is restarted without being gracefully brought down. This process is started automatically by the **`nfslock`** service and does not require user configuration. This is not used with NFSv4.
- **`rpc.rquotad`** — This process provides user quota information for remote users. This process is started automatically by the **`nfs`** service and does not require user configuration.
- **`rpc.idmapd`** — This process provides NFSv4 client and server upcalls which map between on-the-wire NFSv4 names (which are strings in the form of user@domain) and local UIDs and GIDs. For **`idmapd`** to function with NFSv4, the **`/etc/idmapd.conf`** must be configured. This service is required for use with NFSv4.

To use NFS on your system, make sure you have the *nfs-utils*, *nfs-utils-lib*, and *portmap* packages installed.

21.2. NFS Client Configuration

NFS shares are mounted on the client side using the **mount** command. The format of the command is as follows:

```
mount -t <nfs-type> -o <options> <host>:</remote/export> </local/directory>
```

Replace *<nfs-type>* with either **nfs** for NFSv2 or NFSv3 servers, or **nfs4** for NFSv4 servers. Replace *<options>* with a comma separated list of options for the NFS file system (refer to Section 21.4, "Common NFS Mount Options" for details). Replace *<host>* with the remote host, *</remote/export>* with the remote directory being mounted, and *</local/directory>* with the local directory where the remote file system is to be mounted.

Refer to the **mount** man page for more details.

If accessing an NFS share by manually issuing the **mount** command, the file system must be remounted manually after the system is rebooted. Red Hat Enterprise Linux offers two methods for mounting remote file systems automatically at boot time: the **/etc/fstab** file or the **autofs** service.

21.2.1. Mounting NFS File Systems using /etc/fstab

An alternate way to mount an NFS share from another machine is to add a line to the **/etc/fstab** file. The **/etc/fstab** file is referenced by the **netfs** service at boot time, so lines referencing NFS shares have the same effect as manually typing the **mount** command during the boot process. Each line in this file must state the hostname of the NFS server, the directory on the server being exported, and the directory on the local machine where the NFS share is to be mounted. You must be root to modify the **/etc/fstab** file.

The general syntax for a line in **/etc/fstab** is as follows:

```
<server>:</remote/export> </local/directory> <nfs-type> <options> 0 0
```

Replace ***<server>*** with the hostname, IP address, or fully qualified domain name of the server exporting the file system. Replace ***</remote/export>*** with the path to the exported directory, and ***</local/directory>*** with the local file system on which the exported directory is mounted. Replace ***<nfs-type>*** with either **nfs** for NFSv2 or NFSv3 servers, or **nfs4** for NFSv4 servers. Finally, replace *<options>* with a comma separated list of options for the NFS file system (see Section 21.4, "Common NFS Mount Options" for details). Note that the mount point must exist before **/etc/fstab** is read, otherwise the mount fails.

The following is a sample **/etc/fstab** line to mount an NFS export:

```
server:/usr/local/pub    /pub   nfs    defaults 0 0
```

After adding this line to **/etc/fstab** on the client system, type the command **mount /pub** at a shell prompt, and the mount point **/pub** is mounted from the server. The mount point **/pub** must exist on the client machine before this command can be executed.

For more information about the **/etc/fstab** configuration file and its contents, refer to the **fstab** manual page.

21.3. autofs

One drawback to using **`/etc/fstab`** is that, regardless of how infrequently a user accesses the NFS mounted file system, the system must dedicate resources to keep the mounted file system in place. This is not a problem with one or two mounts, but when the system is maintaining mounts to many systems at one time, overall system performance can be affected. An alternative to **`/etc/fstab`** is to use the kernel-based automount utility. An automounter consists of two components. One is a kernel module that implements a file system, while the other is a user-space daemon that performs all of the other functions. The automount utility can mount and unmount NFS file systems automatically (on demand mounting) therefore saving system resources. The automount utility can be used to mount other file systems including AFS, SMBFS, CIFS and local file systems.

`autofs` uses **`/etc/auto.master`** (master map) as its default primary configuration file. This can be changed to use another supported network source and name using the autofs configuration (in **`/etc/sysconfig/autofs`**) in conjunction with the Name Service Switch mechanism. An instance of the version 4 daemon was run for each mount point configured in the master map and so it could be run manually from the command line for any given mount point. This is not possible with version 5 because it uses a single daemon to manage all configured mount points, so all automounts must be configured in the master map. This is in line with the usual requirements of other industry standard automounters. Mount point, hostname, exported directory, and options can all be specified in a set of files (or other supported network sources) rather than configuring them manually for each host. Please ensure that you have the **`autofs`** package installed if you wish to use this service.

21.3.1. What's new in `autofs` version 5?

Direct map support

Autofs direct maps provide a mechanism to automatically mount file systems at arbitrary points in the file system hierarchy. A direct map is denoted by a mount point of "/-" in the master map. Entries in a direct map contain an absolute path name as a key (instead of the relative path names used in indirect maps).

Lazy mount and unmount support

Multimount map entries describe a hierarchy of mount points under a single key. A good example of this is the "-hosts" map, commonly used for automounting all exports from a host under "**`/net/<host>`**" as a multi-mount map entry. When using the "**`-hosts`**" map, an '**`ls`**' of "**`/net/<host>`**" will mount autofs trigger mounts for each export from **`<host>`** and mount and expire them as they are accessed. This can greatly reduce the number of active mounts needed when accessing a server with a large number of exports.

Enhanced LDAP support

The Lightweight Directory Access Protocol, or LDAP, support in autofs version 5 has been enhanced in several ways with respect to autofs version 4. The autofs configuration file (**`/etc/sysconfig/autofs`**) provides a mechanism to specify the autofs schema that a site implements, thus precluding the need to determine this via trial and error in the application itself. In addition, authenticated binds to the LDAP server are now supported, using most mechanisms supported by the common LDAP server implementations. A new configuration file has been added for this support: **`/etc/autofs_ldap_auth.conf`**. The default configuration file is self-documenting, and uses an XML format.

Proper use of the Name Service Switch (`nsswitch`) configuration.

The Name Service Switch configuration file exists to provide a means of determining from where specific configuration data comes. The reason for this configuration is to allow administrators the flexibility of using the back-end database of choice, while maintaining a uniform software interface to access the data. While the version 4 automounter is becoming increasingly better at handling the name service switch configuration, it is still not complete. Autofs version 5, on the other hand, is a complete implementation. See the manual page for nsswitch.conf for more information on the supported syntax of this file. Please note that not all nss databases are valid map sources and the parser will reject ones that are invalid. Valid sources are files, yp, nis, nisplus, ldap and hesiod.

Multiple master map entries per autofs mount point

One thing that is frequently used but not yet mentioned is the handling of multiple master map entries for the direct mount point "/-". The map keys for each entry are merged and behave as one map.

An example is seen in the connectathon test maps for the direct mounts below:

```
/- /tmp/auto_dcthon
/- /tmp/auto_test3_direct
/- /tmp/auto_test4_direct
```

21.3.2. `autofs` Configuration

The primary configuration file for the automounter is **`/etc/auto.master`**, also referred to as the master map which may be changed as described in the introduction section above. The master map lists autofs-controlled mount points on the system, and their corresponding configuration files or network sources known as automount maps. The format of the master map is as follows:

```
<mount-point> <map-name> <options>
```

where:

- **`mount-point`** is the autofs mount point such as **`/home`**.
- **`map-name`** is the name of a map source which contains a list of mount points, and the file system location from which those mount points should be mounted. The syntax for a map entry is described below.
- **`options`** if supplied, will apply to all entries in the given map provided they don't themselves have options specified. This behavior is different from autofs version 4 where the options where cumulative. This has been changed to meet our primary goal of mixed environment compatibility.

The following is a sample **`/etc/auto.master`** file:

```
~]$ cat /etc/auto.master
/home /etc/auto.misc
```

The general format of maps is similar to the master map, however the "options" appear between the mount point and the location instead of at the end of the entry as in the master map:

```
<mount-point>   [<options>]   <location>
```

where:

- **`<mount-point>`** is the autofs mount point. This can be a single directory name for an indirect mount or the full path of the mount point for direct mounts. Each direct and indirect map entry key (**`<mount-point>`** above) may be followed by a space separated list of offset directories (sub directory names each beginning with a "/") making them what is known as a mutli-mount entry.

- <options> if supplied, are the mount options for the map entries that do not specify their own options.
- <location> is the file system location such as a local file system path (preceded with the Sun map format escape character ":" for map names beginning with "/"), an NFS file system or other valid file system location.

The following is a sample map file:

```
~]$ cat /etc/auto.misc
payroll -fstype=nfs personnel:/dev/hda3
sales -fstype=ext3 :/dev/hda4
```

The first column in a map file indicates the autofs mount point (**`sales`** and **`payroll`** from the server called **`personnel`**). The second column indicates the options for the autofs mount while the third column indicates the source of the mount. Following the above configuration, the autofs mount points will be **`/home/payroll`** and **`/home/sales`**. The **`-fstype=`** option is often omitted and is generally not needed for correct operation.

The automounter will create the directories if they do not exist. If the directories exist before the automounter was started, the automounter will not remove them when it exits. You can start or restart the automount daemon by issuing the following command:

```
service autofs start
```

or

```
service autofs restart
```

Using the above configuration, if a process requires access to an autofs unmounted directory such as **`/home/payroll/2006/July.sxc`**, the automount daemon automatically mounts the directory. If a timeout is specified, the directory will automatically be unmounted if the directory is not accessed for the timeout period.

You can view the status of the automount daemon by issuing the following command in your terminal:

```
/sbin/service/autofs status
```

21.3.3. `autofs` Common Tasks

21.3.3.1. Overriding or augmenting site configuration files

It can be useful to override site defaults for a specific mount point on a client system. For example, assuming that the automounter maps are stored in NIS and the **`/etc/nsswitch.conf`** file has the following directive:

```
automount:  files nis
```

and the NIS **`auto.master`** map file contains the following:

```
/home auto.home
```

Also assume the NIS **`auto.home`** map contains the following:

```
beth     fileserver.example.com:/export/home/beth
joe      fileserver.example.com:/export/home/joe
*        fileserver.example.com:/export/home/&
```

and the file map **`/etc/auto.home`** does not exist.

For the above example, lets assume that the client system needs to mount home directories from a different server. In this case, the client will need to use the following **`/etc/auto.master`** map:

```
/home /etc/auto.home2
+auto.master
```

And the **`/etc/auto.home2`** map contains the entry:

```
*   labserver.example.com:/export/home/&
```

Because only the first occurrence of a mount point is processed, **`/home`** will contain the contents of **`/etc/auto.home2`** instead of the NIS **`auto.home`** map.

Alternatively, if you just want to augment the site-wide **`auto.home`** map with a few entries, create a **`/etc/auto.home`** file map, and in it put your new entries and at the end, include the NIS auto.home map. Then the **`/etc/auto.home`** file map might look similar to:

```
mydir someserver:/export/mydir
+auto.home
```

Given the NIS **`auto.home`** map listed above, an **`ls`** of **`/home`** would now give:

```
~]$ ls /home
beth  joe  mydir
```

This last example works as expected because **`autofs`** knows not to include the contents of a file map of the same name as the one it is reading and so moves on to the next map source in the **`nsswitch`** configuration.

21.3.3.2. Using LDAP to Store Automounter Maps

LDAP client libraries must be installed on all systems which are to retrieve automounter maps from LDAP. On RHEL 5, the **`openldap`** package should be installed automatically as a dependency of the **`automounter`**. To configure LDAP access, modify **`/etc/openldap/ldap.conf`**. Ensure that BASE and URI are set appropriately for your site. Please also ensure that the schema is set in the configuration.

The most recently established schema for storing automount maps in LDAP is described by **`rfc2307bis`**. To use this schema it is necessary to set it in the **`autofs`** configuration (**`/etc/sysconfig/autofs`**) by removing the comment characters from the schema definition. For example:

```
DEFAULT_MAP_OBJECT_CLASS="automountMap"
DEFAULT_ENTRY_OBJECT_CLASS="automount"
DEFAULT_MAP_ATTRIBUTE="automountMapName"
DEFAULT_ENTRY_ATTRIBUTE="automountKey"
DEFAULT_VALUE_ATTRIBUTE="automountInformation"
```

Ensure that these are the only schema entries not commented in the configuration. Please also note that the **`automountKey`** replaces the **`cn`** attribute in the **`rfc2307bis`** schema. An **`LDIF`** of a sample configuration is described below:

```
# extended LDIF
#
# LDAPv3
# base <> with scope subtree
# filter: (&(objectclass=automountMap)(automountMapName=auto.master))
# requesting: ALL
#

# auto.master, example.com
dn: automountMapName=auto.master,dc=example,dc=com
objectClass: top
objectClass: automountMap
automountMapName: auto.master

# extended LDIF
#
# LDAPv3
# base <automountMapName=auto.master,dc=example,dc=com> with scope subtree
# filter: (objectclass=automount)
# requesting: ALL
#

# /home, auto.master, example.com
dn: automountMapName=auto.master,dc=example,dc=com
objectClass: automount
cn: /home

automountKey: /home
automountInformation: auto.home

# extended LDIF
#
# LDAPv3
# base <> with scope subtree
# filter: (&(objectclass=automountMap)(automountMapName=auto.home))
# requesting: ALL
#

# auto.home, example.com
dn: automountMapName=auto.home,dc=example,dc=com
objectClass: automountMap
automountMapName: auto.home

# extended LDIF
#
# LDAPv3
# base <automountMapName=auto.home,dc=example,dc=com> with scope subtree
# filter: (objectclass=automount)
# requesting: ALL
#

# foo, auto.home, example.com
```

```
dn: automountKey=foo,automountMapName=auto.home,dc=example,dc=com
objectClass: automount
automountKey: foo
automountInformation: filer.example.com:/export/foo

# /, auto.home, example.com
dn: automountKey=/,automountMapName=auto.home,dc=example,dc=com
objectClass: automount
automountKey: /
automountInformation: filer.example.com:/export/&
```

21.3.3.3. Adapting Autofs v4 Maps To Autofs v5

v4 Multi-map entries

Autofs version 4 introduced the notion of a multi-map entry in the master map. A multi-map entry is of the form:

```
<mount-point> <maptype1> <mapname1> <options1> -- <maptype2> <mapname2>
<options2> -- ...
```

Any number of maps can be combined into a single map in this manner. This feature is no longer present in v5. This is because Version 5 supports included maps which can be used to attain the same results. Consider the following multi-map example:

```
/home file /etc/auto.home -- nis auto.home
```

This can be replaced by the following configuration for v5:

`/etc/nsswitch.conf` must list:

```
automount: files nis
```

`/etc/auto.master` should contain:

```
/home  auto.home
```

`/etc/auto.home` should contain:

```
<entries for the home directory>
+auto.home
```

In this way, the entries from **`/etc/auto.home`** and the nis **`auto.home`** map are combined.

Multiple master maps

In autofs version 4, it is possible to merge the contents of master maps from each source, such as files, nis, hesiod, and LDAP. The version 4 automounter looks for a master map for each of the sources listed in **`/etc/nsswitch.conf`**. The map is read if it exists and its contents are merged into one large **`auto.master`** map.

In version 5, this is no longer the behaviour. Only the first master map found from the list of sources in **`nsswitch.conf`** is consulted. If it is desirable to merge the contents of multiple master maps, included maps can be used. Consider the following example:

```
/etc/nsswitch.conf:
automount: files nis
```

```
/etc/auto.master:
/home  /etc/auto.home
+auto.master
```

The above configuration will merge the contents of the file-based **`auto.master`** and the NIS-based **`auto.master`**. However, because included map entries are only allowed in file maps, there is no way to include both an NIS **`auto.master`** and an LDAP **`auto.master`**.

This limitation can be overcome by creating a master maps that have a different name in the source. In the example above if we had an LDAP master map named **`auto.master.ldap`** we could also add **`"+auto.master.ldap"`** to the file based master map and provided that "**`ldap`**" is listed as a source in our nsswitch configuration it would also be included.

21.4. Common NFS Mount Options

Beyond mounting a file system via NFS on a remote host, other options can be specified at the time of the mount to make it easier to use. These options can be used with manual **`mount`** commands, **`/etc/fstab`** settings, and **`autofs`**.

The following are options commonly used for NFS mounts:

- **`hard`** or **`soft`** — Specifies whether the program using a file via an NFS connection should stop and wait (**`hard`**) for the server to come back online, if the host serving the exported file system is unavailable, or if it should report an error (**`soft`**).

  If **`hard`** is specified, the user cannot terminate the process waiting for the NFS communication to resume unless the **`intr`** option is also specified.

  If **`soft`** is specified, the user can set an additional **`timeo=<value>`** option, where **`<value>`** specifies the number of seconds to pass before the error is reported.

Note

Using soft mounts is not recommended as they can generate I/O errors in very congested networks or when using a very busy server.

- **`intr`** — Allows NFS requests to be interrupted if the server goes down or cannot be reached.

- **`nfsvers=2`** or **`nfsvers=3`** — Specifies which version of the NFS protocol to use. This is useful for hosts that run multiple NFS servers. If no version is specified, NFS uses the highest supported version by the kernel and **`mount`** command. This option is not supported with NFSv4 and should not be used.

- **`noacl`** — Turns off all ACL processing. This may be needed when interfacing with older versions of Red Hat Enterprise Linux, Red Hat Linux, or Solaris, since the most recent ACL technology is not compatible with older systems.

- **`nolock`** — Disables file locking. This setting is occasionally required when connecting to older NFS servers.

- **`noexec`** — Prevents execution of binaries on mounted file systems. This is useful if the system is mounting a non-Linux file system via NFS containing incompatible binaries.
- **`nosuid`** — Disables set-user-identifier or set-group-identifier bits. This prevents remote users from gaining higher privileges by running a setuid program.
- **`port=num`** — Specifies the numeric value of the NFS server port. If ***`num`*** is **`0`** (the default), then **`mount`** queries the remote host's portmapper for the port number to use. If the remote host's NFS daemon is not registered with its portmapper, the standard NFS port number of TCP 2049 is used instead.
- **`rsize=num`** and **`wsize=num`** — These settings speed up NFS communication for reads (**`rsize`**) and writes (**`wsize`**) by setting a larger data block size, in bytes, to be transferred at one time. Be careful when changing these values; some older Linux kernels and network cards do not work well with larger block sizes. For NFSv2 or NFSv3, the default values for both parameters is set to 8192. For NFSv4, the default values for both parameters is set to 32768.
- **`sec=mode`** — Specifies the type of security to utilize when authenticating an NFS connection.

  `sec=sys` is the default setting, which uses local UNIX UIDs and GIDs by means of AUTH_SYS to authenticate NFS operations.

  `sec=krb5` uses Kerberos V5 instead of local UNIX UIDs and GIDs to authenticate users.

  `sec=krb5i` uses Kerberos V5 for user authentication and performs integrity checking of NFS operations using secure checksums to prevent data tampering.

  `sec=krb5p` uses Kerberos V5 for user authentication, integrity checking, and encrypts NFS traffic to prevent traffic sniffing. This is the most secure setting, but it also has the most performance overhead involved.
- **`tcp`** — Specifies for the NFS mount to use the TCP protocol.
- **`udp`** — Specifies for the NFS mount to use the UDP protocol.

Many more options are listed on the **`mount`** and **`nfs`** man pages.

21.5. Starting and Stopping NFS

To run an NFS server, the **`portmap`** service must be running. To verify that **`portmap`** is active, type the following command as root:

```
service portmap status
```

If the **`portmap`** service is running, then the **`nfs`** service can be started. To start an NFS server, as root type:

```
service nfs start
```

Note

`nfslock` also has to be started for both the NFS client and server to function properly. To start NFS locking as root type: **`/sbin/service nfslock start`**. If NFS is set to start at boot, please ensure that **`nfslock`** also starts by running **`chkconfig --list nfslock`**. If **`nfslock`** is not set to **`on`**, this implies that you will need to manually run the **`/sbin/service nfslock start`** each time the computer starts. To set **`nfslock`** to automatically start on boot, type the following command in a terminal **`chkconfig nfslock on`**.

To stop the server, as root, type:

```
service nfs stop
```

The **`restart`** option is a shorthand way of stopping and then starting NFS. This is the most efficient way to make configuration changes take effect after editing the configuration file for NFS.

To restart the server, as root, type:

```
service nfs restart
```

The **`condrestart`** (*conditional restart*) option only starts **`nfs`** if it is currently running. This option is useful for scripts, because it does not start the daemon if it is not running.

To conditionally restart the server, as root, type:

```
service nfs condrestart
```

To reload the NFS server configuration file without restarting the service, as root, type:

```
service nfs reload
```

By default, the **`nfs`** service does *not* start automatically at boot time. To configure the NFS to start up at boot time, use an initscript utility, such as **`/sbin/chkconfig`**, **/usr/sbin/ntsysv**, or the **Services Configuration Tool** program. Refer to Chapter 18, *Controlling Access to Services* for more information regarding these tools.

21.6. NFS Server Configuration

There are three ways to configure an NFS server under Red Hat Enterprise Linux: using the **NFS Server Configuration Tool** (**`system-config-nfs`**), manually editing its configuration file (**`/etc/exports`**), or using the **`/usr/sbin/exportfs`** command.

To use the NFS Server Configuration Tool, you must be running X Windows, have root privileges, and have the system-config-nfs RPM package installed. To start the application, click on **System** > **Administration** > **Server Settings** > **NFS**. You can also type the command **`system-config-nfs`** in a terminal. The NFS Server Configuration tool window is illustrated below.

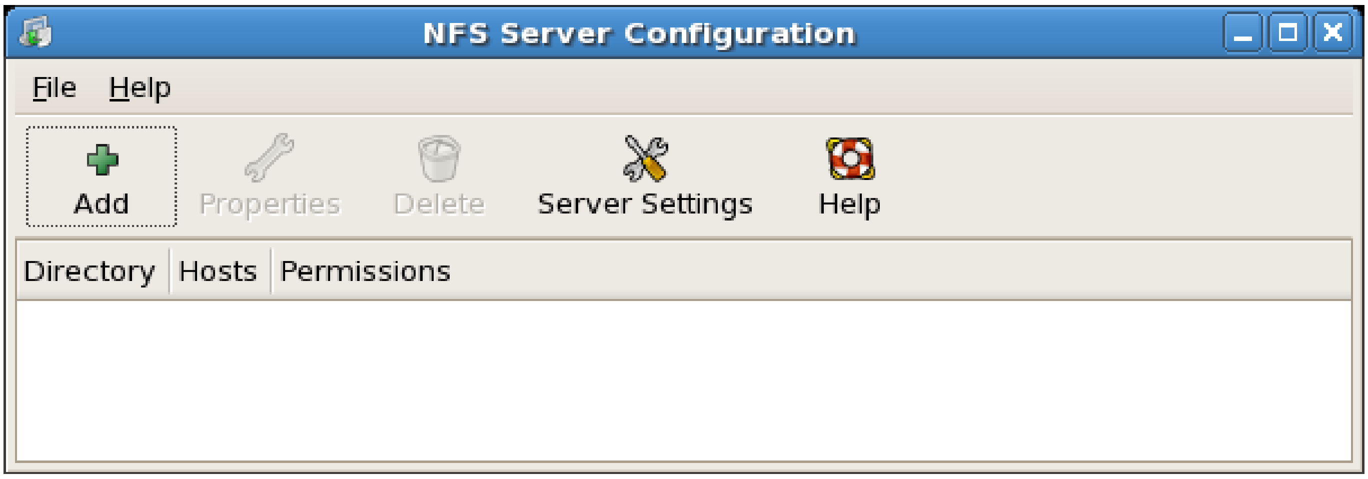
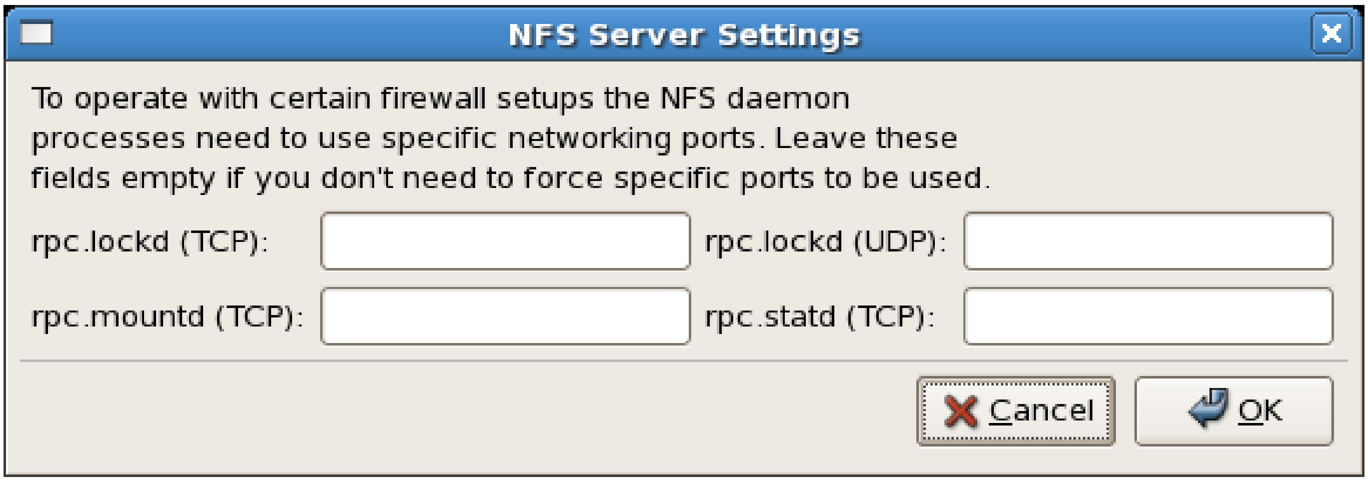


Figure 21.1. NFS Server Configuration Tool

Based on certain firewall settings, you may need to configure the NFS daemon processes to use specific networking ports. The NFS server settings allows you to specify the ports for each process instead of using the random ports assigned by the portmapper. You can set the NFS Server settings by clicking on the **`Server Settings`** button. The figure below illustrates the NFS Server Settings window.


NFS Server Settings

To operate with certain firewall setups the NFS daemon processes need to use specific networking ports. Leave these fields empty if you don't need to force specific ports to be used.

rpc.lockd (TCP):

rpc.lockd (UDP):

rpc.mountd (TCP):

rpc.statd (TCP):

Figure 21.2. NFS Server Settings

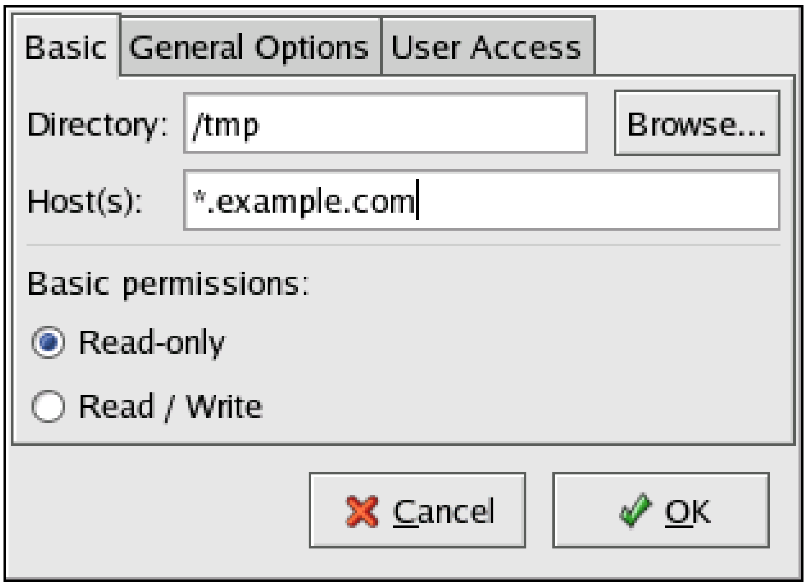
21.6.1. Exporting or Sharing NFS File Systems

Sharing or serving files from an NFS server is known as exporting the directories. The **NFS Server Configuration Tool** can be used to configure a system as an NFS server.

To add an NFS share, click the **Add** button. The dialog box shown in Figure 21.3, “Add Share” appears.

The **`Basic`** tab requires the following information:

- **`Directory`** — Specify the directory to share, such as **`/tmp`**.
- **`Host(s)`** — Specify the host(s) with which to share the directory. Refer to Section 21.6.4, “Hostname Formats” for an explanation of possible formats.
- **`Basic permissions`** — Specify whether the directory should have read-only or read/write permissions.



Basic | General Options | User Access

Directory: /tmp [Browse...]

Host(s): *.example.com

Basic permissions:

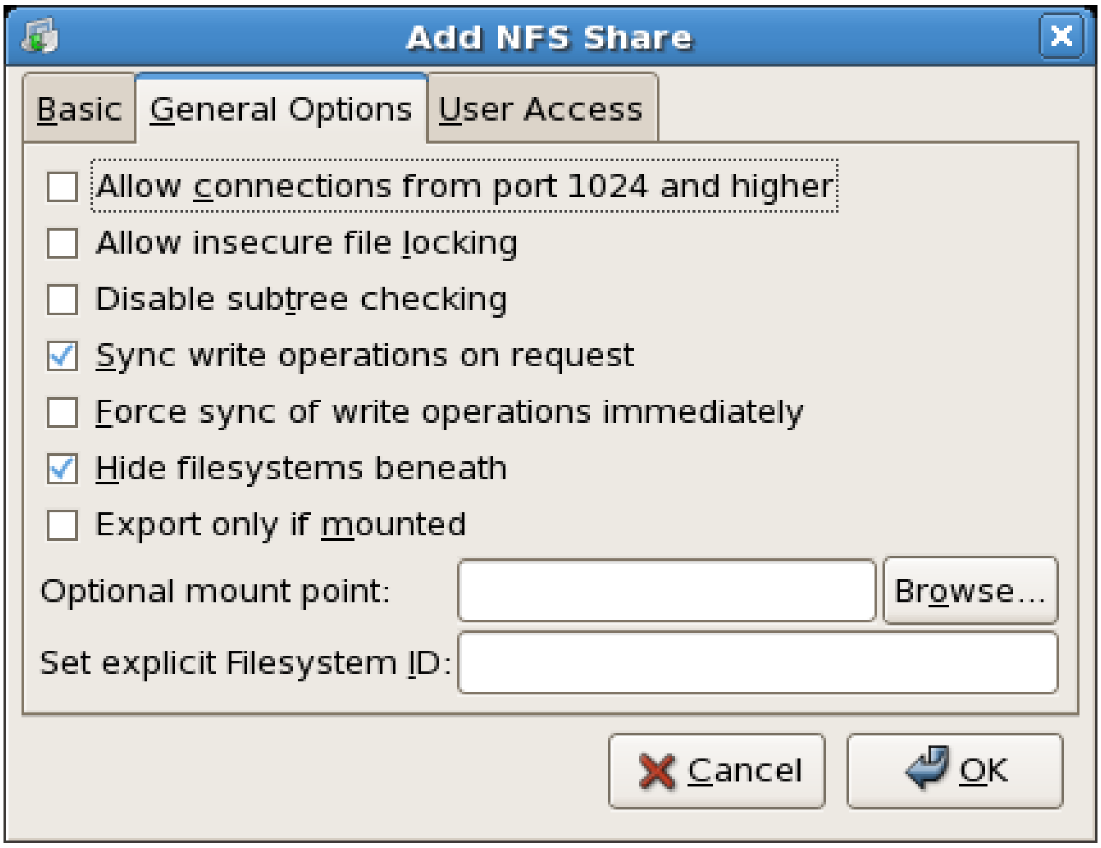
[✓ Read-only]

Read / Write

[Cancel] [OK]

Figure 21.3. Add Share

The **`General Options`** tab allows the following options to be configured:



Add NFS Share

Basic | General Options | User Access

- [ ] Allow connections from port 1024 and higher
- [ ] Allow insecure file locking
- [ ] Disable subtree checking
- [x] Sync write operations on request
- [ ] Force sync of write operations immediately
- [x] Hide filesystems beneath
- [ ] Export only if mounted

Optional mount point: [Browse...]

Set explicit Filesystem ID:

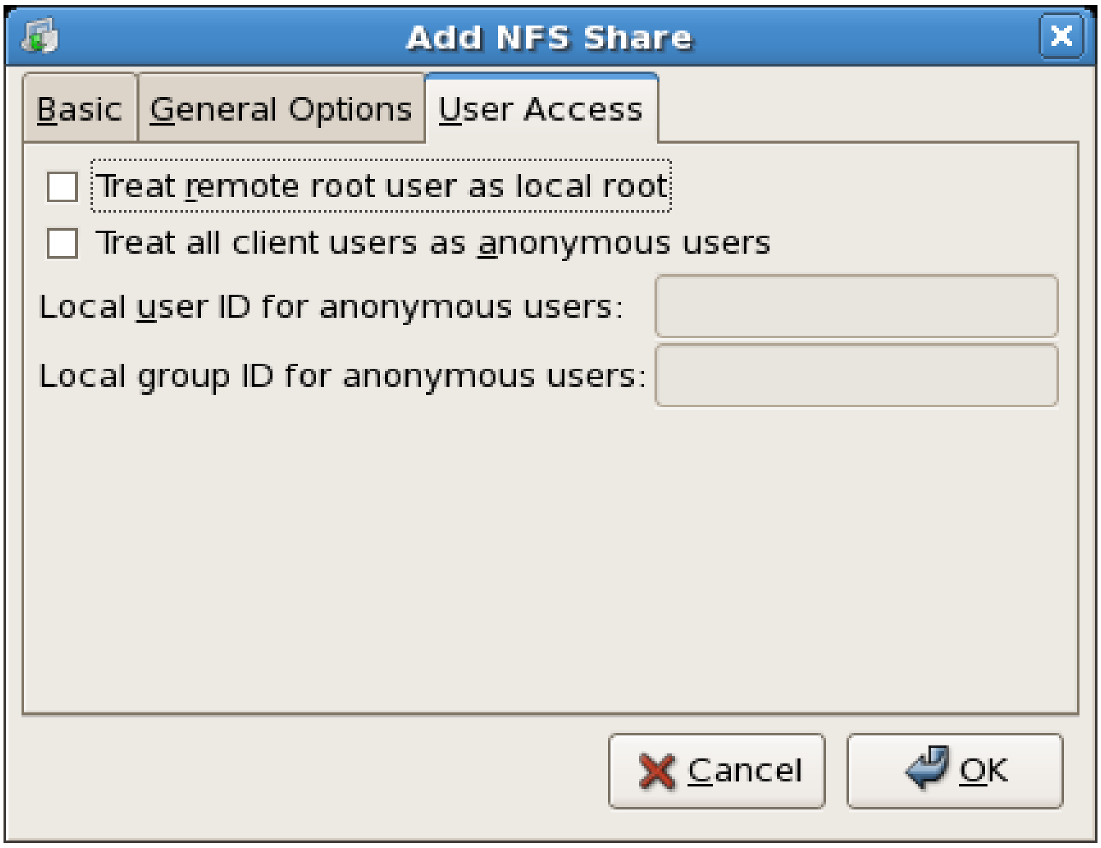
[Cancel] [OK]

Figure 21.4. NFS General Options

- **`Allow connections from port 1024 and higher`** — Services started on port numbers less than 1024 must be started as root. Select this option to allow the NFS service to be started by a user other than root. This option corresponds to **`insecure`**.
- **`Allow insecure file locking`** — Do not require a lock request. This option corresponds to **`insecure_locks`**.
- **`Disable subtree checking`** — If a subdirectory of a file system is exported, but the entire file system

is not exported, the server checks to see if the requested file is in the subdirectory exported. This check is called *subtree checking*. Select this option to disable subtree checking. If the entire file system is exported, selecting to disable subtree checking can increase the transfer rate. This option corresponds to **`no_subtree_check`**.

- **`Sync write operations on request`** — Enabled by default, this option does not allow the server to reply to requests before the changes made by the request are written to the disk. This option corresponds to **`sync`**. If this is not selected, the **`async`** option is used.
  - **`Force sync of write operations immediately`** — Do not delay writing to disk. This option corresponds to **`no_wdelay`**.
- **`Hide filesystems beneath`** turns the **`nohide`** option on or off. When the **`nohide`** option is off, nested directories are revealed. The clients can therefore navigate through a filesystem from the parent without noticing any changes.
- **`Export only if mounted`** sets the **`mountpoint`** option which allows a directory to be exported only if it has been mounted.
- **`Optional Mount Point`** specifies the path to an optional mount point. Click on the **`Browse`** to navigate to the preferred mount point or type the path if known.
- **`Set explicit Filesystem ID:`** sets the **`fsid=X`** option. This is mainly used in a clustered setup. Using a consistent filesystem ID in all clusters avoids having stale NFS filehandles.



☐ Treat remote root user as local root
☐ Treat all client users as anonymous users
Local user ID for anonymous users:
Local group ID for anonymous users:

Figure 21.5. NFS User Access

The **`User Access`** tab allows the following options to be configured:

- **`Treat remote root user as local root`** — By default, the user and group IDs of the root user are both 0. Root squashing maps the user ID 0 and the group ID 0 to the user and group IDs of anonymous so that root on the client does not have root privileges on the NFS server. If this option is selected, root is not mapped to anonymous, and root on a client has root privileges to exported directories. Selecting this option can greatly decrease the security of the system. Do not select it unless it is absolutely necessary. This option corresponds to **`no_root_squash`**.

- **`Treat all client users as anonymous users`** — If this option is selected, all user and group IDs are mapped to the anonymous user. This option corresponds to **`all_squash`**.

  - **`Specify local user ID for anonymous users`** — If **`Treat all client users as anonymous users`** is selected, this option lets you specify a user ID for the anonymous user. This option corresponds to **`anonuid`**.

  - **`Specify local group ID for anonymous users`** — If **`Treat all client users as anonymous users`** is selected, this option lets you specify a group ID for the anonymous user. This option corresponds to **`anongid`**.

To edit an existing NFS share, select the share from the list, and click the **`Properties`** button. To delete an existing NFS share, select the share from the list, and click the **`Delete`** button.

After clicking **`OK`** to add, edit, or delete an NFS share from the list, the changes take place immediately — the server daemon is restarted and the old configuration file is saved as **`/etc/exports.bak`**. The new configuration is written to **`/etc/exports`**.

The **NFS Server Configuration Tool** reads and writes directly to the **`/etc/exports`** configuration file. Thus, the file can be modified manually after using the tool, and the tool can be used after modifying the file manually (provided the file was modified with correct syntax).

The next this section discusses manually editing **`/etc/exports`** and using the **`/usr/sbin/exportfs`** command to export NFS file systems.

21.6.2. Command Line Configuration

If you prefer editing configuration files using a text editor or if you do not have the X Window System installed, you can modify the configuration file directly.

The **`/etc/exports`** file controls what directories the NFS server exports. Its format is as follows:

```
directory hostname(options)
```

The only option that needs to be specified is one of **`sync`** or **`async`** (**`sync`** is recommended). If **`sync`** is specified, the server does not reply to requests before the changes made by the request are written to the disk.

For example,

```
/misc/export speedy.example.com(sync)
```

would allow users from **`speedy.example.com`** to mount **`/misc/export`** with the default read-only permissions, but,

```
/misc/export speedy.example.com(rw,sync)
```

would allow users from **`speedy.example.com`** to mount **`/misc/export`** with read/write privileges.

Refer to Section 21.6.4, “Hostname Formats” for an explanation of possible hostname formats.

Warning

Be careful with spaces in the **`/etc/exports`** file. If there are no spaces between the hostname and the options in parentheses, the options apply only to the hostname. If there is a space between the hostname and the options, the options apply to the rest of the world. For example, examine the following lines:

```
/misc/export speedy.example.com(rw,sync) /misc/export
speedy.example.com (rw,sync)
```

The first line grants users from **`speedy.example.com`** read-write access and denies all other users. The second line grants users from **`speedy.example.com`** read-only access (the default) and allows the rest of the world read-write access.

Each time you change **`/etc/exports`**, you must inform the NFS daemon of the change, or reload the configuration file with the following command:

```
service nfs reload
```

21.6.3. Running NFS Behind a Firewall

Because NFS requires portmap, which dynamically assigns ports for RPC services and can cause problems for configuring firewall rules, you can edit the **`/etc/sysconfig/nfs`** configuration file to control which ports the required RPC services run on. Refer to and read Section 32.1.22, "**`/etc/sysconfig/nfs`**" for instructions on how to configure a firewall to allow NFS.

21.6.4. Hostname Formats

The host(s) can be in the following forms:

- Single machine — A fully qualified domain name (that can be resolved by the server), hostname (that can be resolved by the server), or an IP address.
- Series of machines specified with wildcards — Use the * or ? character to specify a string match. Wildcards are not to be used with IP addresses; however, they may accidentally work if reverse DNS lookups fail. When specifying wildcards in fully qualified domain names, dots (.) are not included in the wildcard. For example, **`*.example.com`** includes one.example.com but does not include one.two.example.com.
- IP networks — Use *a.b.c.d/z*, where *a.b.c.d* is the network and *z* is the number of bits in the netmask (for example 192.168.0.0/24). Another acceptable format is *a.b.c.d/netmask*, where *a.b.c.d* is the network and *netmask* is the netmask (for example, 192.168.100.8/255.255.255.0).
- Netgroups — In the format *@group-name*, where *group-name* is the NIS netgroup name.

21.7. The `/etc/exports` Configuration File

The **`/etc/exports`** file controls which file systems are exported to remote hosts and specifies options. Blank lines are ignored, comments can be made by starting a line with the hash mark (#), and long lines can be wrapped with a backslash (\). Each exported file system should be on its own individual line, and any lists of authorized hosts placed after an exported file system must be separated by space characters. Options for each of the hosts must be placed in parentheses directly after the host identifier, without any spaces separating the host and the first parenthesis. Valid host types are **gss/krb5**, **gss/krb5i**, and **gss/krb5p**.

A line for an exported file system has the following structure:

```
<export> <host1>(<options>) <hostN>(<options>)...
```

In this structure, replace *<export>* with the directory being exported, replace *<host1>* with the host or network to which the export is being shared, and replace *<options>* with the options for that host or network. Additional hosts can be specified in a space separated list.

The following methods can be used to specify host names:

- *single host* — Where one particular host is specified with a fully qualified domain name, hostname, or IP address.
- *wildcards* — Where a ***** or **?** character is used to take into account a grouping of fully qualified domain names that match a particular string of letters. Wildcards should not be used with IP addresses; however, it is possible for them to work accidentally if reverse DNS lookups fail.

  Be careful when using wildcards with fully qualified domain names, as they tend to be more exact than expected. For example, the use of ***.example.com** as a wildcard allows sales.example.com to access an exported file system, but not bob.sales.example.com. To match both possibilities both ***.example.com** and ***.*.example.com** must be specified.
- *IP networks* — Allows the matching of hosts based on their IP addresses within a larger network. For example, **192.168.0.0/28** allows the first 16 IP addresses, from 192.168.0.0 to 192.168.0.15, to access the exported file system, but not 192.168.0.16 and higher.
- *netgroups* — Permits an NIS netgroup name, written as **@*<group-name>***, to be used. This effectively puts the NIS server in charge of access control for this exported file system, where users can be added and removed from an NIS group without affecting **/etc/exports**.

In its simplest form, the **/etc/exports** file only specifies the exported directory and the hosts permitted to access it, as in the following example:

```
/exported/directory bob.example.com
```

In the example, **bob.example.com** can mount **/exported/directory/**. Because no options are specified in this example, the following default NFS options take effect:

- **ro** — Mounts of the exported file system are read-only. Remote hosts are not able to make changes to the data shared on the file system. To allow hosts to make changes to the file system, the read/write (**rw**) option must be specified.
- **wdelay** — Causes the NFS server to delay writing to the disk if it suspects another write request is imminent. This can improve performance by reducing the number of times the disk must be accessed by separate write commands, reducing write overhead. The **no_wdelay** option turns off this feature, but is only available when using the **sync** option.
- **root_squash** — Prevents root users connected remotely from having root privileges and assigns them the user ID for the user **nfsnobody**. This effectively "squashes" the power of the remote root user to the lowest local user, preventing unauthorized alteration of files on the remote server. Alternatively, the **no_root_squash** option turns off root squashing. To squash every remote user, including root, use the

all_squash option. To specify the user and group IDs to use with remote users from a particular host, use the **anonuid** and **anongid** options, respectively. In this case, a special user account can be created for remote NFS users to share and specify **(anonuid=*<uid-value>*,anongid=*<gid-value>*)**, where ***<uid-value>*** is the user ID number and ***<gid-value>*** is the group ID number.

Important

By default, *access control lists* (ACLs) are supported by NFS under Red Hat Enterprise Linux. To disable this feature, specify the **no_acl** option when exporting the file system.

Each default for every exported file system must be explicitly overridden. For example, if the **rw** option is not specified, then the exported file system is shared as read-only. The following is a sample line from **/etc/exports** which overrides two default options:

```
/another/exported/directory 192.168.0.3(rw,sync)
```

In this example **192.168.0.3** can mount **/another/exported/directory/** read/write and all transfers to disk are committed to the disk before the write request by the client is completed.

Additionally, other options are available where no default value is specified. These include the ability to disable sub-tree checking, allow access from insecure ports, and allow insecure file locks (necessary for certain early NFS client implementations). Refer to the **exports** man page for details on these lesser used options.

Warning

The format of the **/etc/exports** file is very precise, particularly in regards to use of the space character. Remember to always separate exported file systems from hosts and hosts from one another with a space character. However, there should be no other space characters in the file except on comment lines.

For example, the following two lines do not mean the same thing:

```
/home bob.example.com(rw)
/home bob.example.com (rw)
```

The first line allows only users from **bob.example.com** read/write access to the **/home** directory. The second line allows users from **bob.example.com** to mount the directory as read-only (the default), while the rest of the world can mount it read/write.

21.7.1. The `exportfs` Command

Every file system being exported to remote users via NFS, as well as the access level for those file systems, are listed in the **/etc/exports** file. When the **nfs** service starts, the **/usr/sbin/exportfs** command launches and reads this file, passes control to **rpc.mountd** (if NFSv2 or NFSv3) for the actual mounting process, then to **rpc.nfsd** where the file systems are then available to remote users.

When issued manually, the **/usr/sbin/exportfs** command allows the root user to selectively export or unexport directories without restarting the NFS service. When given the proper options, the

/usr/sbin/exportfs command writes the exported file systems to **/var/lib/nfs/xtab**. Since **rpc.mountd** refers to the **xtab** file when deciding access privileges to a file system, changes to the list of exported file systems take effect immediately.

The following is a list of commonly used options available for **/usr/sbin/exportfs**:

- **-r** — Causes all directories listed in **/etc/exports** to be exported by constructing a new export list in **/etc/lib/nfs/xtab**. This option effectively refreshes the export list with any changes that have been made to **/etc/exports**.
- **-a** — Causes all directories to be exported or unexported, depending on what other options are passed to **/usr/sbin/exportfs**. If no other options are specified, **/usr/sbin/exportfs** exports all file systems specified in **/etc/exports**.
- **-o *file-systems*** — Specifies directories to be exported that are not listed in **/etc/exports**. Replace *file-systems* with additional file systems to be exported. These file systems must be formatted in the same way they are specified in **/etc/exports**. Refer to Section 21.7, "The **/etc/exports** Configuration File" for more information on **/etc/exports** syntax. This option is often used to test an exported file system before adding it permanently to the list of file systems to be exported.
- **-i** — Ignores **/etc/exports**; only options given from the command line are used to define exported file systems.
- **-u** — Unexports all shared directories. The command **/usr/sbin/exportfs -ua** suspends NFS file sharing while keeping all NFS daemons up. To re-enable NFS sharing, type **exportfs -r**.
- **-v** — Verbose operation, where the file systems being exported or unexported are displayed in greater detail when the **exportfs** command is executed.

If no options are passed to the **/usr/sbin/exportfs** command, it displays a list of currently exported file systems.

For more information about the **/usr/sbin/exportfs** command, refer to the **exportfs** man page.

21.7.1.1. Using exportfs with NFSv4

The **exportfs** command is used in maintaining the NFS table of exported file systems. When typed in a terminal with no arguments, the **exportfs** command shows all the exported directories.

Since NFSv4 no longer utilizes the **MOUNT** protocol, which was used with the NFSv2 and NFSv3 protocols, the mounting of file systems has changed.

An NFSv4 client now has the ability to see all of the exports served by the NFSv4 server as a single file system, called the NFSv4 pseudo-file system. On Red Hat Enterprise Linux, the pseudo-file system is identified as a single, real file system, identified at export with the **fsid=0** option.

For example, the following commands could be executed on an NFSv4 server:

```
mkdir /exports
mkdir /exports/opt
mkdir /exports/etc
mount --bind /usr/local/opt /exports/opt
mount --bind /usr/local/etc /exports/etc
exportfs -o fsid=0,insecure,no_subtree_check gss/krb5p:/exports
exportfs -o rw,nohide,insecure,no_subtree_check gss/krb5p:/exports/opt
exportfs -o rw,nohide,insecure,no_subtree_check gss/krb5p:/exports/etc
```

In this example, clients are provided with multiple file systems to mount, by using the **--bind** option which creates unbreakable links.

Because of the pseudo-file systems feature, NFS version 2, 3 and 4 export configurations are not always compatible. For example, given the following directory tree:

```
/home
/home/sam
/home/john
/home/joe
```

and the export:

```
/home *(rw,fsid=0,sync)
```

Using NFS version 2,3 and 4 the following would work:

```
mount server:/home /mnt/home
ls /mnt/home/joe
```

Using v4 the following would work:

```
mount -t nfs4 server:/ /mnt/home
ls /mnt/home/joe
```

The difference being "**server:/home**" and "**server:/**". To make the exports configurations compatible for all version, one needs to export (read only) the root filesystem with an **fsid=0**. The **fsid=0** signals the NFS server that this export is the root.

```
/ *(ro,fsid=0)
/home *(rw,sync,nohide)
```

Now with these exports, both "**mount server:/home /mnt/home**" and "**mount -t nfs server:/home /mnt/home**" will work as expected.

21.8. Securing NFS

NFS is well suited for sharing entire file systems with a large number of known hosts in a transparent manner. However, with ease of use comes a variety of potential security problems.

The following points should be considered when exporting NFS file systems on a server or mounting them on a client. Doing so minimizes NFS security risks and better protects data on the server.

21.8.1. Host Access

Depending on which version of NFS you plan to implement, depends on your existing network environment, and your security concerns. The following sections explain the differences between implementing security measures with NFSv2, NFSv3, and NFSv4. If at all possible, use of NFSv4 is recommended over other versions of NFS.

21.8.1.1. Using NFSv2 or NFSv3

NFS controls who can mount an exported file system based on the host making the mount request, not the user that actually uses the file system. Hosts must be given explicit rights to mount the exported file system. Access control is not possible for users, other than through file and directory permissions. In other words, once a file system is exported via NFS, any user on any remote host connected to the NFS server can access the shared data. To limit the potential risks, administrators often allow read-only access or squash user permissions to a common user and group ID. Unfortunately, these solutions prevent the NFS share from being used in the way it was originally intended.

Additionally, if an attacker gains control of the DNS server used by the system exporting the NFS file system, the system associated with a particular hostname or fully qualified domain name can be pointed to an unauthorized machine. At this point, the unauthorized machine *is* the system permitted to mount the NFS share, since no username or password information is exchanged to provide additional security for the NFS mount.

Wildcards should be used sparingly when exporting directories via NFS as it is possible for the scope of the wildcard to encompass more systems than intended.

It is also possible to restrict access to the **`portmap`** service via TCP wrappers. Access to ports used by **`portmap`**, **`rpc.mountd`**, and **`rpc.nfsd`** can also be limited by creating firewall rules with **`iptables`**.

For more information on securing NFS and **`portmap`**, refer to Section 48.9, “IPTables”.

21.8.1.2. Using NFSv4

The release of NFSv4 brought a revolution to authentication and security to NFS exports. NFSv4 mandates the implementation of the RPCSEC_GSS kernel module and the Kerberos version 5 GSS-API mechanism. With NFSv4, the mandatory security mechanisms are oriented towards authenticating individual users, and not client machines as used in NFSv2 and NFSv3.

Note

It is assumed that a Kerberos ticket-granting server (KDC) is installed and configured correctly, prior to configuring an NFSv4 server. Kerberos is a network authentication system which allows clients and servers to authenticate to each other through use of symmetric encryption and a trusted third party, the KDC.

Important

When the NFSv4 server is configured to use the Kerberos version 5 GSS-API mechanism, the use of NFS over UDP is not supported and an attempt to mount the NFS-exported file system on the client system may fail. Consequently, users are advised to use TCP in this situation. Refer to Section 21.10, “Using NFS over TCP” for more information.

NFSv4 includes ACL support based on the Microsoft Windows NT model, not the POSIX model, because of its features and because it is widely deployed. NFSv2 and NFSv3 do not have support for native ACL attributes.

Another important security feature of NFSv4 is the removal of the use of the MOUNT protocol for mounting file systems. This protocol presented possible security holes because of the way that it handled file handles.

For more information on the RPCSEC_GSS framework, including how **`rpc.svcgssd`** and **`rpc.gssd`** inter operate, refer to http://www.citi.umich.edu/projects/nfsv4/gssd/.

21.8.2. File Permissions

Once the NFS file system is mounted read/write by a remote host, the only protection each shared file has is its permissions. If two users that share the same user ID value mount the same NFS file system, they can modify each others files. Additionally, anyone logged in as root on the client system can use the **`su -`** command to become a user who could access particular files via the NFS share.

By default, access control lists (ACLs) are supported by NFS under Red Hat Enterprise Linux. It is not recommended that this feature be disabled.

The default behavior when exporting a file system via NFS is to use *root squashing*. This sets the user ID of anyone accessing the NFS share as the root user on their local machine to a value of the server's **`nfsnobody`** account. Never turn off root squashing.

If exporting an NFS share as read-only, consider using the **`all_squash`** option, which makes every user accessing the exported file system take the user ID of the **`nfsnobody`** user.

21.9. NFS and `portmap`

Note

The following section only applies to NFSv2 or NFSv3 implementations that require the **`portmap`** service for backward compatibility.

The **`portmapper`** maps RPC services to the ports they are listening on. RPC processes notify**`portmap`** when they start, registering the ports they are listening on and the RPC program numbers they expect to serve. The client system then contacts **`portmap`** on the server with a particular RPC program number. The **`portmap`** service redirects the client to the proper port number so it can communicate with the requested service.

Because RPC-based services rely on **`portmap`** to make all connections with incoming client requests, **`portmap`** must be available before any of these services start.

The **`portmap`** service uses TCP wrappers for access control, and access control rules for **`portmap`** affect *all* RPC-based services. Alternatively, it is possible to specify access control rules for each of the NFS RPC daemons. The man pages for **`rpc.mountd`** and **`rpc.statd`** contain information regarding the precise syntax for these rules.

21.9.1. Troubleshooting NFS and `portmap`

Because **`portmap`** provides coordination between RPC services and the port numbers used to communicate with them, it is useful to view the status of current RPC services using **`portmap`** when troubleshooting. The **`rpcinfo`** command shows each RPC-based service with port numbers, an RPC program number, a version number, and an IP protocol type (TCP or UDP).

To make sure the proper NFS RPC-based services are enabled for **`portmap`**, issue the following command as root:

```
rpcinfo -p
```

The following is sample output from this command:

```
program vers proto   port
```

```
 100000    2   tcp    111  portmapper
 100000    2   udp    111  portmapper
 100021    1   udp  32774  nlockmgr
 100021    3   udp  32774  nlockmgr
 100021    4   udp  32774  nlockmgr
 100021    1   tcp  34437  nlockmgr
 100021    3   tcp  34437  nlockmgr
 100021    4   tcp  34437  nlockmgr
 100011    1   udp    819  rquotad
 100011    2   udp    819  rquotad
 100011    1   tcp    822  rquotad
 100011    2   tcp    822  rquotad
 100003    2   udp   2049  nfs
 100003    3   udp   2049  nfs
 100003    2   tcp   2049  nfs
 100003    3   tcp   2049  nfs
 100005    1   udp    836  mountd
 100005    1   tcp    839  mountd
 100005    2   udp    836  mountd
 100005    2   tcp    839  mountd
 100005    3   udp    836  mountd
 100005    3   tcp    839  mountd
```

If one of the NFS services does not start up correctly, **`portmap`** is unable to map RPC requests from clients for that service to the correct port. In many cases, if NFS is not present in **`rpcinfo`** output, restarting NFS causes the service to correctly register with **`portmap`** and begin working. For instructions on starting NFS, refer to Section 21.5, “Starting and Stopping NFS”.

Other useful options are available for the **`rpcinfo`** command. Refer to the **`rpcinfo`** man page for more information.

21.10. Using NFS over TCP

The default transport protocol for NFSv4 is TCP. The advantages of using TCP include the following:

- Improved connection durability, thus less **`NFS stale file handles`** messages.
- Performance gain on heavily loaded networks because TCP acknowledges every packet, unlike UDP which only acknowledges completion.
- TCP has better congestion control than UDP. On a very congested network, UDP packets are the first packets that are dropped. This means that if NFS is writing data (in 8K chunks) all of that 8K must be retransmitted over UDP. Because of TCP's reliability, only parts of that 8K data are transmitted at a time.
- Error detection. When a TCP connection breaks (due to the server being unavailable) the client stops sending data and restarts the connection process once the server becomes available. With UDP, since it's connection-less, the client continues to pound the network with data until the server reestablishes a connection.

The main disadvantage is that there is a very small performance hit due to the overhead associated with the TCP protocol.

21.11. Additional Resources

Administering an NFS server can be a challenge. Many options, including quite a few not mentioned in this chapter, are available for exporting or mounting NFS shares. Consult the following sources for more information.

21.11.1. Installed Documentation

- **`/usr/share/doc/nfs-utils-<version-number>/`** — Replace <*version-number*> with the version number of the NFS package installed. This directory contains a wealth of information about the NFS implementation for Linux, including a look at various NFS configurations and their impact on file transfer performance.
- **`man mount`** — Contains a comprehensive look at mount options for both NFS server and client configurations.
- **`man fstab`** — Gives details for the format of the **`/etc/fstab`** file used to mount file systems at boot-time.
- **`man nfs`** — Provides details on NFS-specific file system export and mount options.
- **`man exports`** — Shows common options used in the **`/etc/exports`** file when exporting NFS file systems.

21.11.2. Useful Websites

- http://nfs.sourceforge.net/ — The home of the Linux NFS project and a great place for project status updates.
- http://www.citi.umich.edu/projects/nfsv4/linux/ — An NFSv4 for Linux 2.6 kernel resource.
- http://www.nfsv4.org — The home of NFS version 4 and all related standards.
- http://www.vanemery.com/Linux/NFSv4/NFSv4-no-rpcsec.html — Describes the details of NFSv4 with Fedora Core 2, which includes the 2.6 kernel.
- http://www.sane.nl/events/sane2000/papers/pawlowski.pdf — An excellent whitepaper on the features and enhancements of the NFS Version 4 protocol.
- http://wiki.autofs.net — The Autofs wiki, discussions, documentation and enhancements.

21.11.3. Related Books

- *Managing NFS and NIS* by Hal Stern, Mike Eisler, and Ricardo Labiaga; O'Reilly & Associates — Makes an excellent reference guide for the many different NFS export and mount options available.
- *NFS Illustrated* by Brent Callaghan; Addison-Wesley Publishing Company — Provides comparisons of NFS to other network file systems and shows, in detail, how NFS communication occurs.

Chapter 22. Samba

Samba is an open source implementation of the Server Message Block (SMB) protocol. It allows the networking of Microsoft Windows®, Linux, UNIX, and other operating systems together, enabling access to Windows-based file and printer shares. Samba's use of SMB allows it to appear as a Windows server to Windows clients.

22.1. Introduction to Samba

The third major release of Samba, version 3.0.0, introduced numerous improvements from prior versions, including:

- The ability to join an Active Directory domain by means of LDAP and Kerberos
- Built in Unicode support for internationalization
- Support for Microsoft Windows XP Professional client connections to Samba servers without needing local registry hacking
- Two new documents developed by the Samba.org team, which include a 400+ page reference manual, and a 300+ page implementation and integration manual. For more information about these published titles, refer to Section 22.12.2, “Related Books”.

22.1.1. Samba Features

Samba is a powerful and versatile server application. Even seasoned system administrators must know its abilities and limitations before attempting installation and configuration.

What Samba can do:

- Serve directory trees and printers to Linux, UNIX, and Windows clients
- Assist in network browsing (with or without NetBIOS)
- Authenticate Windows domain logins
- Provide Windows Internet Name Service (WINS) name server resolution
- Act as a Windows NT®-style Primary Domain Controller (PDC)
- Act as a Backup Domain Controller (BDC) for a Samba-based PDC
- Act as an Active Directory domain member server
- Join a Windows NT/2000/2003 PDC

What Samba cannot do:

- Act as a BDC for a Windows PDC (and vice versa)
- Act as an Active Directory domain controller

22.2. Samba Daemons and Related Services

The following is a brief introduction to the individual Samba daemons and services.

22.2.1. Samba Daemons

Samba is comprised of three daemons (**`smbd`**, **`nmbd`**, and **`winbindd`**). Two services (**`smb`** and **`windbind`**) control how the daemons are started, stopped, and other service-related features. Each daemon is listed in detail, as well as which specific service has control over it.

`smbd`

The **`smbd`** server daemon provides file sharing and printing services to Windows clients. In addition, it is responsible for user authentication, resource locking, and data sharing through the SMB protocol. The default ports on which the server listens for SMB traffic are TCP ports 139 and 445.

The **`smbd`** daemon is controlled by the **`smb`** service.

`nmbd`

The **`nmbd`** server daemon understands and replies to NetBIOS name service requests such as those produced by SMB/CIFS in Windows-based systems. These systems include Windows 95/98/ME, Windows NT, Windows 2000, Windows XP, and LanManager clients. It also participates in the browsing protocols that make up the Windows **`Network Neighborhood`** view. The default port that the server listens to for NMB traffic is UDP port 137.

The **`nmbd`** daemon is controlled by the **`smb`** service.

`winbindd`

The **`winbind`** service resolves user and group information on a server running Windows NT 2000 or Windows Server 2003. This makes Windows user / group information understandable by UNIX platforms. This is achieved by using Microsoft RPC calls, Pluggable Authentication Modules (PAM), and the Name Service Switch (NSS). This allows Windows NT domain users to appear and operate as UNIX users on a UNIX machine. Though bundled with the Samba distribution, the **`winbind`** service is controlled separately from the **`smb`** service.

The **`winbindd`** daemon is controlled by the **`winbind`** service and does not require the **`smb`** service to be started in order to operate. Winbindd is also used when Samba is an Active Directory member, and may also be used on a Samba domain controller (to implement nested groups and/or interdomain trust). Because **`winbind`** is a client-side service used to connect to Windows NT-based servers, further discussion of **`winbind`** is beyond the scope of this manual.

Note

You may refer to Section 22.11, “Samba Distribution Programs” for a list of utilities included in the Samba distribution.

22.3. Connecting to a Samba Share

You can use **Nautilus** to view available Samba shares on your network. Select **Places** (on the Panel) > **Network Servers** to view a list of Samba workgroups on your network. You can also type **`smb:`** in the **`File`** > **`Open Location`** bar of Nautilus to view the workgroups.

As shown in Figure 22.1, “SMB Workgroups in Nautilus”, an icon appears for each available SMB workgroup on the network.

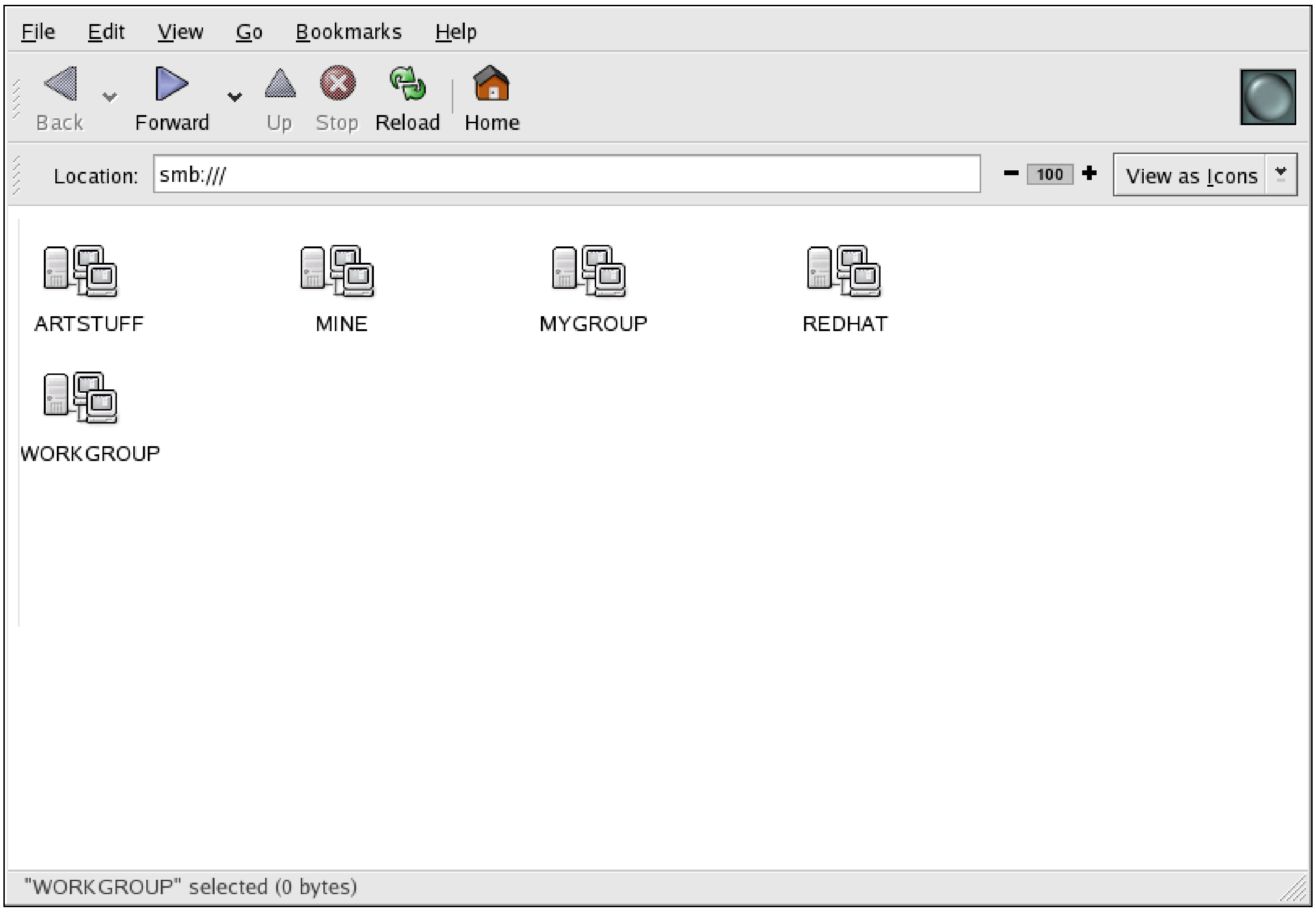


Figure 22.1. SMB Workgroups in Nautilus

Double-click one of the workgroup icons to view a list of computers within the workgroup.

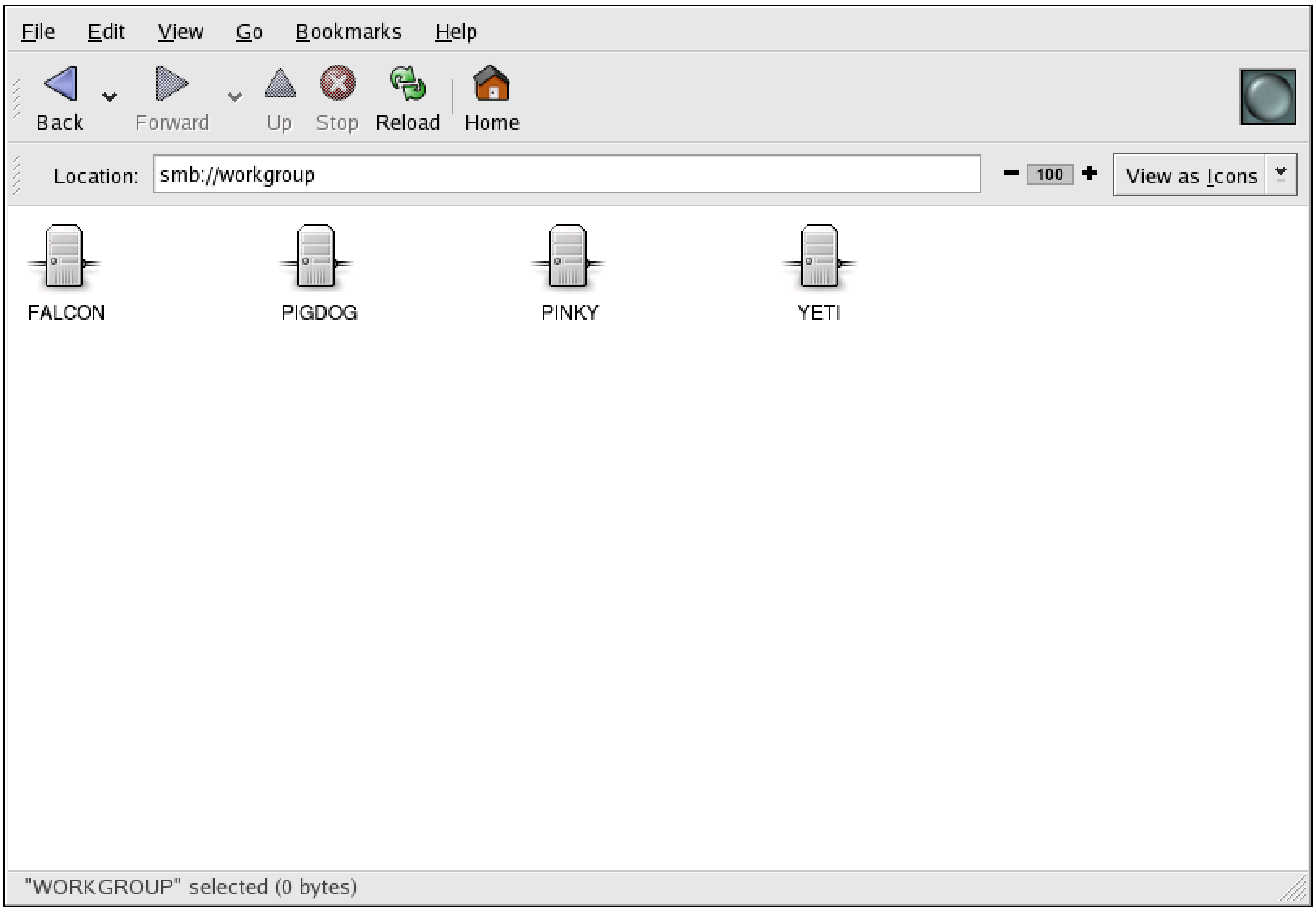


Figure 22.2. SMB Machines in Nautilus

As you can see from Figure 22.2, “SMB Machines in Nautilus”, there is an icon for each machine within the workgroup. Double-click on an icon to view the Samba shares on the machine. If a username and password combination is required, you are prompted for them.

Alternately, you can also specify the Samba server and sharename in the **`Location:`** bar for **Nautilus** using the following syntax (replace <*servername*> and <*sharename*> with the appropriate values):

```
smb://<servername>/<sharename>
```

22.3.1. Command Line

To query the network for Samba servers, use the **`findsmb`** command. For each server found, it displays its IP address, NetBIOS name, workgroup name, operating system, and SMB server version.

To connect to a Samba share from a shell prompt, type the following command:

```
smbclient //<hostname>/<sharename> -U <username>
```

Replace <*hostname*> with the hostname or IP address of the Samba server you want to connect to, <*sharename*> with the name of the shared directory you want to browse, and <*username*> with the Samba username for the system. Enter the correct password or press **`Enter`** if no password is required for the user.

If you see the **`smb:\>`** prompt, you have successfully logged in. Once you are logged in, type **`help`** for a list of commands. If you wish to browse the contents of your home directory, replace *sharename* with your username. If the **`-U`** switch is not used, the username of the current user is passed to the Samba server.

To exit **`smbclient`**, type **`exit`** at the `smb:\>` prompt.

22.3.2. Mounting the Share

Sometimes it is useful to mount a Samba share to a directory so that the files in the directory can be treated as if they are part of the local file system.

To mount a Samba share to a directory, create create a directory to mount it to (if it does not already exist), and execute the following command as root:

```
mount -t cifs -o <username>,<password> //<servername>/<sharename>
/mnt/point/
```

This command mounts *<sharename>* from *<servername>* in the local directory */mnt/point/*. For more information about mounting a samba share, refer to **`man mount.cifs`**.

22.4. Configuring a Samba Server

The default configuration file (**`/etc/samba/smb.conf`**) allows users to view their home directories as a Samba share. It also shares all printers configured for the system as Samba shared printers. In other words, you can attach a printer to the system and print to it from the Windows machines on your network.

22.4.1. Graphical Configuration

To configure Samba using a graphical interface, use the **Samba Server Configuration Tool**. For command line configuration, skip to Section 22.4.2, “Command Line Configuration”.

The **Samba Server Configuration Tool** is a graphical interface for managing Samba shares, users, and basic server settings. It modifies the configuration files in the **`/etc/samba/`** directory. Any changes to these files not made using the application are preserved.

To use this application, you must be running the X Window System, have root privileges, and have the **`system-config-samba`** RPM package installed. To start the **Samba Server Configuration Tool** from the desktop, go to the **System** (on the Panel) > **Administration** > **Server Settings** > **Samba** or type the command **`system-config-samba`** at a shell prompt (for example, in an XTerm or a GNOME terminal).

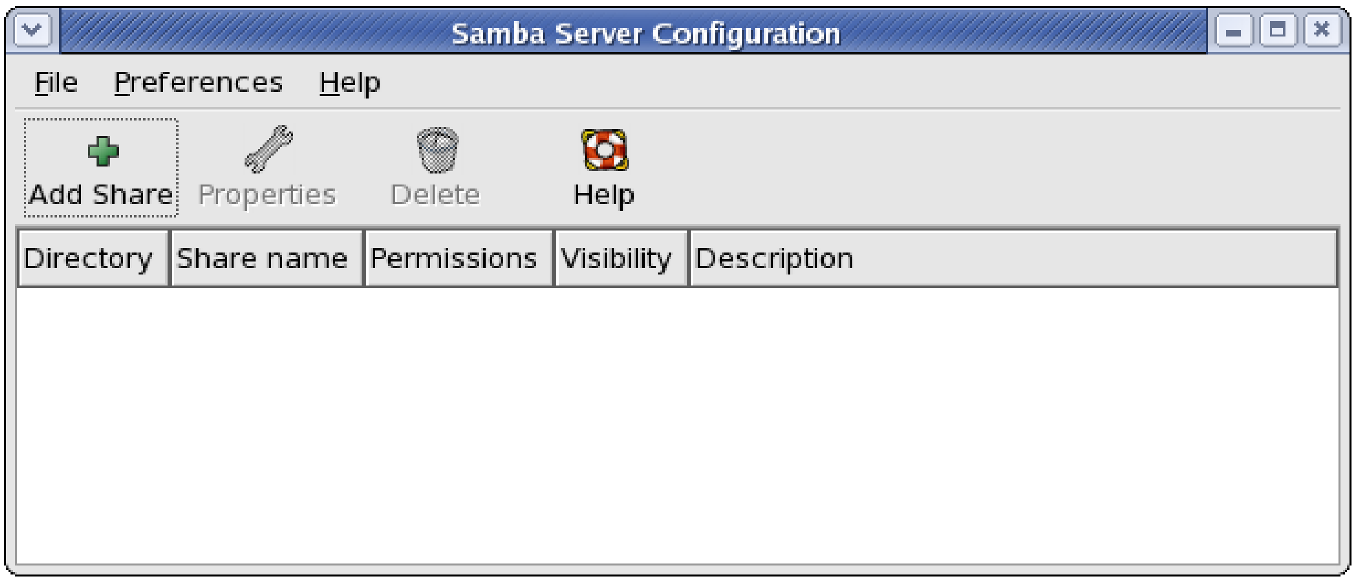


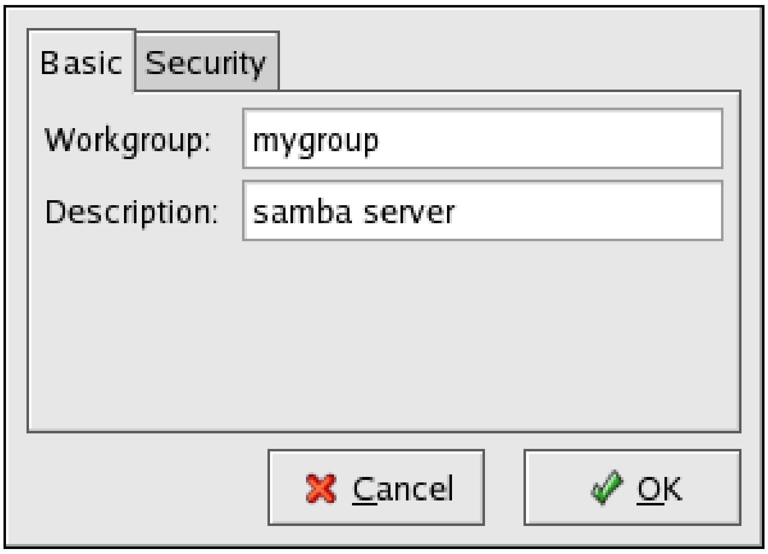
Figure 22.3. Samba Server Configuration Tool

Note

The **Samba Server Configuration Tool** does not display shared printers or the default stanza that allows users to view their own home directories on the Samba server.

22.4.1.1. Configuring Server Settings

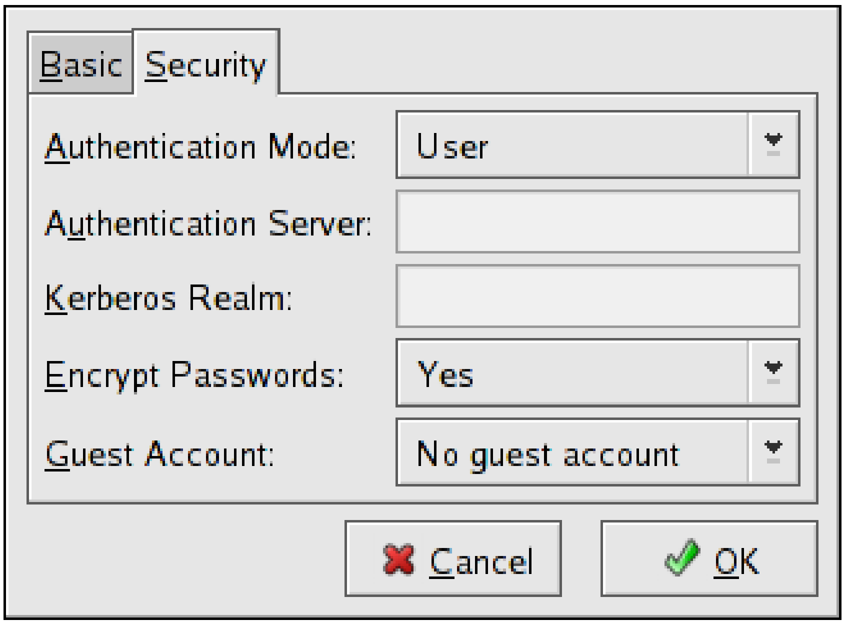
The first step in configuring a Samba server is to configure the basic settings for the server and a few security options. After starting the application, select **Preferences** > **Server Settings** from the pulldown menu. The **`Basic`** tab is displayed as shown in Figure 22.4, "Configuring Basic Server Settings".



Workgroup: mygroup
Description: samba server

Figure 22.4. Configuring Basic Server Settings

On the **`Basic`** tab, specify which workgroup the computer should be in as well as a brief description of the computer. They correspond to the **`workgroup`** and **`server string`** options in **`smb.conf`**.



Authentication Mode: User
Authentication Server:
Kerberos Realm:
Encrypt Passwords: Yes
Guest Account: No guest account

Figure 22.5. Configuring Security Server Settings

The **`Security`** tab contains the following options:

- **`Authentication Mode`** — This corresponds to the **`security`** option. Select one of the following types of authentication.
  - **`ADS`** — The Samba server acts as a domain member in an Active Directory Domain (ADS) realm. For this option, Kerberos must be installed and configured on the server, and Samba must become a member of the ADS realm using the **`net`** utility, which is part of the **`samba-client`** package. Refer to the **`net`** man page for details. This option does not configure Samba to be an ADS Controller. Specify the realm of the Kerberos server in the **`Kerberos Realm`** field.

    Note

    The **`Kerberos Realm`** field must be supplied in all uppercase letters, such as **`EXAMPLE.COM`**.

    Using a Samba server as a domain member in an ADS realm assumes proper configuration of Kerberos, including the **`/etc/krb5.conf`** file.

  - **`Domain`** — The Samba server relies on a Windows NT Primary or Backup Domain Controller to verify the user. The server passes the username and password to the Controller and waits for it to return. Specify the NetBIOS name of the Primary or Backup Domain Controller in the **`Authentication Server`** field.

    The **`Encrypted Passwords`** option must be set to **`Yes`** if this is selected.

  - **`Server`** — The Samba server tries to verify the username and password combination by passing them to another Samba server. If it can not, the server tries to verify using the user authentication mode. Specify the NetBIOS name of the other Samba server in the **`Authentication Server`** field.
  - **`Share`** — Samba users do not have to enter a username and password combination on a per Samba server basis. They are not prompted for a username and password until they try to connect to a specific shared directory from a Samba server.
  - **`User`** — (Default) Samba users must provide a valid username and password on a per Samba server basis. Select this option if you want the **`Windows Username`** option to work. Refer to Section 22.4.1.2, “Managing Samba Users” for details.
- **`Encrypt Passwords`** — This option must be enabled if the clients are connecting from a system with Windows 98, Windows NT 4.0 with Service Pack 3, or other more recent versions of Microsoft Windows. The passwords are transferred between the server and the client in an encrypted format instead of as a plain-text word that can be intercepted. This corresponds to the **`encrypted passwords`** option. Refer to Section 22.4.3, “Encrypted Passwords” for more information about encrypted Samba passwords.
- **`Guest Account`** — When users or guest users log into a Samba server, they must be mapped to a valid user on the server. Select one of the existing usernames on the system to be the guest Samba account. When guests log in to the Samba server, they have the same privileges as this user. This corresponds to the **`guest account`** option.

After clicking **`OK`**, the changes are written to the configuration file and the daemon is restarted; thus, the changes take effect immediately.

22.4.1.2. Managing Samba Users

The **Samba Server Configuration Tool** requires that an existing user account be active on the system acting as the Samba server before a Samba user can be added. The Samba user is associated with the existing user account.

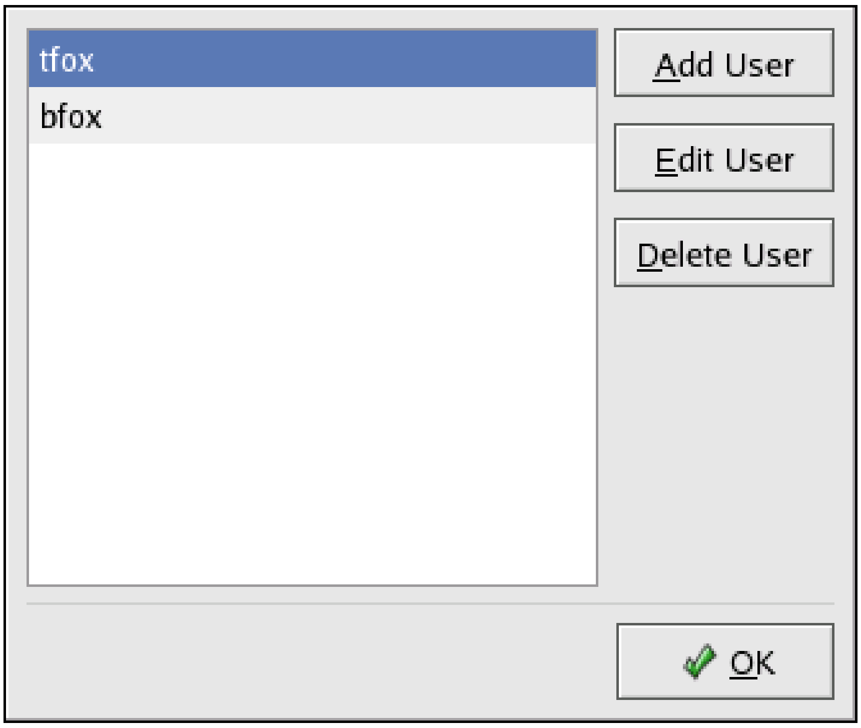


Figure 22.6. Managing Samba Users

To add a Samba user, select **Preferences** > **Samba Users** from the pulldown menu, and click the **`Add User`** button. In the **`Create New Samba User`** window select a **`Unix Username`** from the list of existing users on the local system.

If the user has a different username on a Windows machine and needs to log into the Samba server from the Windows machine, specify that Windows username in the **`Windows Username`** field. The **`Authentication Mode`** on the **`Security`** tab of the **`Server Settings`** preferences must be set to **`User`** for this option to work.

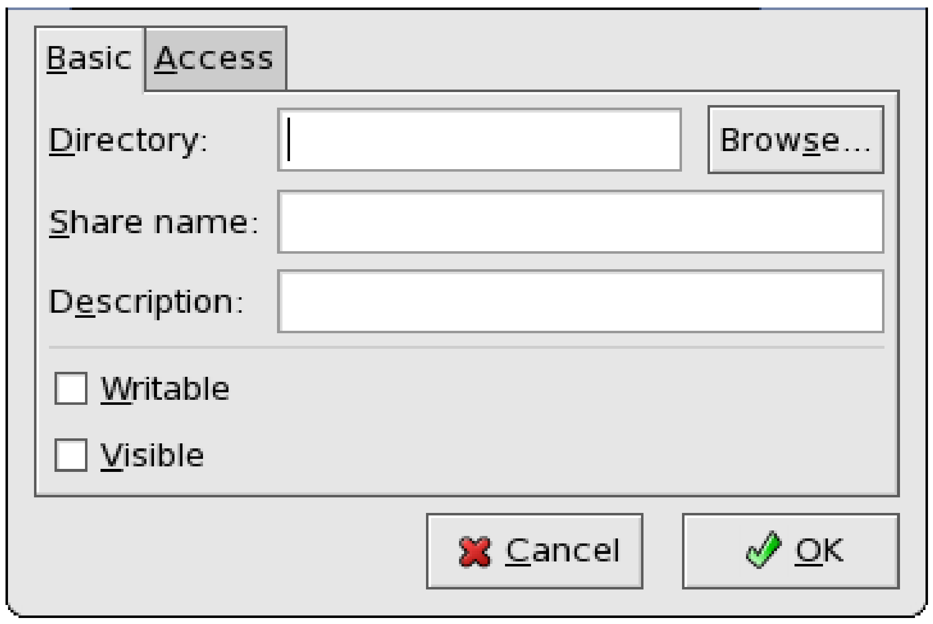
Also, configure a **`Samba Password`** for the Samba User and confirm it by typing it again. Even if you opt to use encrypted passwords for Samba, it is recommended that the Samba passwords for all users are different from their system passwords.

To edit an existing user, select the user from the list, and click **`Edit User`**. To delete an existing Samba user, select the user, and click the **`Delete User`** button. Deleting a Samba user does not delete the associated system user account.

The users are modified immediately after clicking the **`OK`** button.

22.4.1.3. Adding a Share

To create a Samba share, click the **Add** button from the main Samba configuration window.



Directory:

Share name:

Description:

☐ Writable

☐ Visible

Figure 22.7. Adding a Share

The **`Basic`** tab configures the following options:

- **`Directory`** — The directory to share via Samba. The directory must exist before it can be entered here.
- **`Share name`** — The actual name of the share that is seen from remote machines. By default, it is the same value as **`Directory`**, but can be configured.
- **`Descriptions`** — A brief description of the share.
- **`Writable`** — Enables users to read and write to the shared directory
- **`Visible`** — Grants read-only rights to users for the shared directory.

On the **`Access`** tab, select whether to allow only specified users to access the share or whether to allow all Samba users to access the share. If you select to allow access to specific users, select the users from the list of available Samba users.

The share is added immediately after clicking **`OK`**.

22.4.2. Command Line Configuration

Samba uses **`/etc/samba/smb.conf`** as its configuration file. If you change this configuration file, the changes do not take effect until you restart the Samba daemon with the command **`service smb restart`**.

To specify the Windows workgroup and a brief description of the Samba server, edit the following lines in your **`smb.conf`** file:

```
workgroup = WORKGROUPNAME
server string = BRIEF COMMENT ABOUT SERVER
```

Replace *WORKGROUPNAME* with the name of the Windows workgroup to which this machine should belong. The *BRIEF COMMENT ABOUT SERVER* is optional and is used as the Windows comment about the Samba system.

To create a Samba share directory on your Linux system, add the following section to your **`smb.conf`** file (after modifying it to reflect your needs and your system):

```
[sharename]
comment = Insert a comment here
path = /home/share/
valid users = tfox carole
public = no
writable = yes
printable = no
create mask = 0765
```

The above example allows the users **`tfox`** and **`carole`** to read and write to the directory **`/home/share`**, on the Samba server, from a Samba client.

22.4.3. Encrypted Passwords

Encrypted passwords are enabled by default because it is more secure to do so. To create a user with an encrypted password, use the command **`smbpasswd -a <username>`** .

22.5. Starting and Stopping Samba

To start a Samba server, type the following command in a shell prompt while logged in as root:

```
service smb start
```

Important

To set up a domain member server, you must first join the domain or Active Directory using the **`net join`** command *before* starting the **`smb`** service.

To stop the server, type the following command in a shell prompt while logged in as root:

```
service smb stop
```

The **`restart`** option is a quick way of stopping and then starting Samba. This is the most reliable way to make configuration changes take effect after editing the configuration file for Samba. Note that the restart option starts the daemon even if it was not running originally.

To restart the server, type the following command in a shell prompt while logged in as root:

```
service smb restart
```

The **`condrestart`** (*conditional restart*) option only starts **`smb`** on the condition that it is currently running. This option is useful for scripts, because it does not start the daemon if it is not running.

Note

When the **`smb.conf`** file is changed, Samba automatically reloads it after a few minutes. Issuing a manual **`restart`** or **`reload`** is just as effective.

To conditionally restart the server, type the following command as root:

```
service smb condrestart
```

A manual reload of the **smb.conf** file can be useful in case of a failed automatic reload by the **smb** service. To ensure that the Samba server configuration file is reloaded without restarting the service, type the following command as root:

```
service smb reload
```

By default, the **smb** service does *not* start automatically at boot time. To configure Samba to start at boot time, use an initscript utility, such as **/sbin/chkconfig**, **/usr/sbin/ntsysv**, or the **Services Configuration Tool** program. Refer to Chapter 18, *Controlling Access to Services* for more information regarding these tools.

22.6. Samba Server Types and the smb.conf File

Samba configuration is straightforward. All modifications to Samba are done in the **/etc/samba/smb.conf** configuration file. Although the default **smb.conf** file is well documented, it does not address complex topics such as LDAP, Active Directory, and the numerous domain controller implementations.

The following sections describe the different ways a Samba server can be configured. Keep in mind your needs and the changes required to the **smb.conf** file for a successful configuration.

22.6.1. Stand-alone Server

A stand-alone server can be a workgroup server or a member of a workgroup environment. A stand-alone server is not a domain controller and does not participate in a domain in any way. The following examples include several anonymous share-level security configurations and one user-level security configuration. For more information on share-level and user-level security modes, refer to Section 22.7, “Samba Security Modes”.

22.6.1.1. Anonymous Read-Only

The following **smb.conf** file shows a sample configuration needed to implement anonymous read-only file sharing. The **security = share** parameter makes a share anonymous. Note, security levels for a single Samba server cannot be mixed. The **security** directive is a global Samba parameter located in the **[global]** configuration section of the **smb.conf** file.

```
[global]
workgroup = DOCS
netbios name = DOCS_SRV
security = share
[data]
comment = Documentation Samba Server
path = /export
read only = Yes
guest only = Yes
```

22.6.1.2. Anonymous Read/Write

The following **smb.conf** file shows a sample configuration needed to implement anonymous read/write file sharing. To enable anonymous read/write file sharing, set the **read only** directive to **no**. The **force user** and **force group** directives are also added to enforce the ownership of any newly placed files specified in the share.

Note

Although having an anonymous read/write server is possible, it is not recommended. Any files placed in the share space, regardless of user, are assigned the user/group combination as specified by a generic user (**force user**) and group (**force group**) in the **smb.conf** file.

```
[global]
workgroup = DOCS
netbios name = DOCS_SRV
security = share
[data]
comment = Data
path = /export
force user = docsbot
force group = users
read only = No
guest ok = Yes
```

22.6.1.3. Anonymous Print Server

The following **smb.conf** file shows a sample configuration needed to implement an anonymous print server. Setting **browseable** to **no** as shown does not list the printer in Windows **Network Neighborhood**. Although hidden from browsing, configuring the printer explicitly is possible. By connecting to **DOCS_SRV** using NetBIOS, the client can have access to the printer if the client is also part of the **DOCS** workgroup. It is also assumed that the client has the correct local printer driver installed, as the **use client driver** directive is set to **Yes**. In this case, the Samba server has no responsibility for sharing printer drivers to the client.

```
[global]
workgroup = DOCS
netbios name = DOCS_SRV
security = share
printcap name = cups
disable spools= Yes
show add printer wizard = No
printing = cups
[printers]
comment = All Printers
path = /var/spool/samba
guest ok = Yes
printable = Yes
use client driver = Yes
browseable = Yes
```

22.6.1.4. Secure Read/Write File and Print Server

The following **smb.conf** file shows a sample configuration needed to implement a secure read/write print server. Setting the **`security`** directive to **`user`** forces Samba to authenticate client connections. Notice the **`[homes]`** share does not have a **`force user`** or **`force group`** directive as the **`[public]`** share does. The **`[homes]`** share uses the authenticated user details for any files created as opposed to the **`force user`** and **`force group`** in **`[public]`**.

```
[global]
workgroup = DOCS
netbios name = DOCS_SRV
security = user
printcap name = cups
disable spools = Yes
show add printer wizard = No
printing = cups
[homes]
comment = Home Directories
valid users = %S
read only = No
browseable = No
[public]
comment = Data
path = /export
force user = docsbot
force group = users
guest ok = Yes
[printers]
comment = All Printers
path = /var/spool/samba
printer admin = john, ed, @admins
create mask = 0600
guest ok = Yes
printable = Yes
use client driver = Yes
browseable = Yes
```

22.6.2. Domain Member Server

A domain member, while similar to a stand-alone server, is logged into a domain controller (either Windows or Samba) and is subject to the domain's security rules. An example of a domain member server would be a departmental server running Samba that has a machine account on the Primary Domain Controller (PDC). All of the department's clients still authenticate with the PDC, and desktop profiles and all network policy files are included. The difference is that the departmental server has the ability to control printer and network shares.

22.6.2.1. Active Directory Domain Member Server

The following **`smb.conf`** file shows a sample configuration needed to implement an Active Directory domain member server. In this example, Samba authenticates users for services being run locally but is also a client of the Active Directory. Ensure that your kerberos **`realm`** parameter is shown in all caps (for example **`realm = EXAMPLE.COM`**). Since Windows 2000/2003 requires Kerberos for Active Directory authentication, the **`realm`** directive is required. If Active Directory and Kerberos are running on different servers, the **`password server`** directive may be required to help the distinction.

```
[global]
realm = EXAMPLE.COM
```

```
security = ADS
encrypt passwords = yes
# Optional. Use only if Samba cannot determine the Kerberos server
automatically.
password server = kerberos.example.com
```

In order to join a member server to an Active Directory domain, the following steps must be completed:

- Configuration of the **`smb.conf`** file on the member server
- Configuration of Kerberos, including the **`/etc/krb5.conf`** file, on the member server
- Creation of the machine account on the Active Directory domain server
- Association of the member server to the Active Directory domain

To create the machine account and join the Windows 2000/2003 Active Directory, Kerberos must first be initialized for the member server wishing to join the Active Directory domain. To create an administrative Kerberos ticket, type the following command as root on the member server:

```
kinit administrator@EXAMPLE.COM
```

The **`kinit`** command is a Kerberos initialization script that references the Active Directory administrator account and Kerberos realm. Since Active Directory requires Kerberos tickets, **`kinit`** obtains and caches a Kerberos ticket-granting ticket for client/server authentication. For more information on Kerberos, the **`/etc/krb5.conf`** file, and the **`kinit`** command, refer to Section 48.6, “Kerberos”.

To join an Active Directory server (windows1.example.com), type the following command as root on the member server:

```
net ads join -S windows1.example.com -U administrator%password
```

Since the machine **`windows1`** was automatically found in the corresponding Kerberos realm (the **`kinit`** command succeeded), the **`net`** command connects to the Active Directory server using its required administrator account and password. This creates the appropriate machine account on the Active Directory and grants permissions to the Samba domain member server to join the domain.

Note

Since **`security = ads`** and not **`security = user`** is used, a local password backend such as **`smbpasswd`** is not needed. Older clients that do not support **`security = ads`** are authenticated as if **`security = domain`** had been set. This change does not affect functionality and allows local users not previously in the domain.

22.6.2.2. Windows NT4-based Domain Member Server

The following **`smb.conf`** file shows a sample configuration needed to implement a Windows NT4-based domain member server. Becoming a member server of an NT4-based domain is similar to connecting to an Active Directory. The main difference is NT4-based domains do not use Kerberos in their authentication method, making the **`smb.conf`** file simpler. In this instance, the Samba member server functions as a pass through to the NT4-based domain server.

```
[global]
```

```
workgroup = DOCS
netbios name = DOCS_SRV
security = domain
[homes]
comment = Home Directories
valid users = %S
read only = No
browseable = No
[public]
comment = Data
path = /export
force user = docsbot
force group = users
guest ok = Yes
```

Having Samba as a domain member server can be useful in many situations. There are times where the Samba server can have other uses besides file and printer sharing. It may be beneficial to make Samba a domain member server in instances where Linux-only applications are required for use in the domain environment. Administrators appreciate keeping track of all machines in the domain, even if not Windows-based. In the event the Windows-based server hardware is deprecated, it is quite easy to modify the **`smb.conf`** file to convert the server to a Samba-based PDC. If Windows NT-based servers are upgraded to Windows 2000/2003, the **`smb.conf`** file is easily modifiable to incorporate the infrastructure change to Active Directory if needed.

Important

After configuring the **`smb.conf`** file, join the domain *before* starting Samba by typing the following command as root:

```
net rpc join -U administrator%password
```

Note that the **`-S`** option, which specifies the domain server hostname, does not need to be stated in the **`net rpc join`** command. Samba uses the hostname specified by the **`workgroup`** directive in the **`smb.conf`** file instead of it being stated explicitly.

22.6.3. Domain Controller

A domain controller in Windows NT is functionally similar to a Network Information Service (NIS) server in a Linux environment. Domain controllers and NIS servers both host user/group information databases as well as related services. Domain controllers are mainly used for security, including the authentication of users accessing domain resources. The service that maintains the user/group database integrity is called the *Security Account Manager* (SAM). The SAM database is stored differently between Windows and Linux Samba-based systems, therefore SAM replication cannot be achieved and platforms cannot be mixed in a PDC/BDC environment.

In a Samba environment, there can be only one PDC and zero or more BDCs.

Important

Samba cannot exist in a mixed Samba/Windows domain controller environment (Samba cannot be a BDC of a Windows PDC or vice versa). Alternatively, Samba PDCs and BDCs *can* coexist.

22.6.3.1. Primary Domain Controller (PDC) using `tdbsam`

The simplest and most common implementation of a Samba PDC uses the **`tdbsam`** password database backend. Planned to replace the aging **`smbpasswd`** backend, **`tdbsam`** has numerous improvements that are explained in more detail in Section 22.8, “Samba Account Information Databases”. The **`passdb backend`** directive controls which backend is to be used for the PDC.

```
[global]
workgroup = DOCS
netbios name = DOCS_SRV
passdb backend = tdbsam
security = user
add user script = /usr/sbin/useradd -m "%u"
delete user script = /usr/sbin/userdel -r "%u"
add group script = /usr/sbin/groupadd "%g"
delete group script = /usr/sbin/groupdel "%g"
add user to group script = /usr/sbin/usermod -G "%g" "%u"
add machine script = /usr/sbin/useradd -s /bin/false -d /dev/null  -g
machines "%u"
# The following specifies the default logon script
# Per user logon scripts can be specified in the user
# account using pdbedit logon script = logon.bat
# This sets the default profile path.
# Set per user paths with pdbedit
logon drive = H:
domain logons = Yes
os level = 35
preferred master = Yes
domain master = Yes
[homes]
 comment = Home Directories
 valid users = %S
 read only = No
[netlogon]
 comment = Network Logon Service
 path = /var/lib/samba/netlogon/scripts
 browseable = No
 read only = No
# For profiles to work, create a user directory under the
# path shown.
mkdir -p /var/lib/samba/profiles/john
[Profiles]
 comment = Roaming Profile Share
 path = /var/lib/samba/profiles
 read only = No
```

```
  browseable = No
  guest ok = Yes
  profile acls = Yes
# Other resource shares ... ...
```

To provide a functional PDC system which uses the **`tdbsam`** follow these steps:

1. Use a configuration of the **`smb.conf`** file as shown in the example above.

2. Add the root user to the Samba password database.

   ```
   smbpasswd -a root
   Provide the password here.
   ```

3. Start the **`smb`** service.

4. Make sure all profile, user, and netlogon directories are created.

5. Add groups that users can be members of.

   ```
   groupadd -f users
   groupadd -f nobody
   groupadd -f ntadmins
   ```

6. Associate the UNIX groups with their respective Windows groups.

   ```
   net groupmap add ntgroup="Domain Users" unixgroup=users
   net groupmap add ntgroup="Domain Guests" unixgroup=nobody
   net groupmap add ntgroup="Domain Admins" unixgroup=ntadmins
   ```

7. Grant access rights to a user or a group. For example, to grant the right to add client machines to the domain on a Samba domain controller, to the members to the Domain Admins group, execute the following command:

   ```
   net rpc rights grant 'DOCS\Domain Admins' SetMachineAccountPrivilege -
   S PDC -U root
   ```

Keep in mind that Windows systems prefer to have a primary group which is mapped to a domain group such as Domain Users.

Windows groups and users use the same namespace thus not allowing the existence of a group and a user with the same name like in UNIX.

Note

If you need more than one domain controller or have more than 250 users, do *not* use a **`tdbsam`** authentication backend. LDAP is recommended in these cases.

22.6.3.2. Primary Domain Controller (PDC) with Active Directory

Although it is possible for Samba to be a member of an Active Directory, it is not possible for Samba to operate as an Active Directory domain controller.

22.7. Samba Security Modes

There are only two types of security modes for Samba, *share-level* and *user-level*, which are collectively known as *security levels* . Share-level security can only be implemented in one way, while user-level security can be implemented in one of four different ways. The different ways of implementing a security level are called *security modes* .

22.7.1. User-Level Security

User-level security is the default setting for Samba. Even if the **`security = user`** directive is not listed in the **`smb.conf`** file, it is used by Samba. If the server accepts the client's username/password, the client can then mount multiple shares without specifying a password for each instance. Samba can also accept session-based username/password requests. The client maintains multiple authentication contexts by using a unique UID for each logon.

In **`smb.conf`**, the **`security = user`** directive that sets user-level security is:

```
[GLOBAL]
...
security = user
...
```

The following sections describe other implementations of user-level security.

22.7.1.1. Domain Security Mode (User-Level Security)

In domain security mode, the Samba server has a machine account (domain security trust account) and causes all authentication requests to be passed through to the domain controllers. The Samba server is made into a domain member server by using the following directives in **`smb.conf`**:

```
[GLOBAL]
...
security = domain
workgroup = MARKETING
...
```

22.7.1.2. Active Directory Security Mode (User-Level Security)

If you have an Active Directory environment, it is possible to join the domain as a native Active Directory member. Even if a security policy restricts the use of NT-compatible authentication protocols, the Samba server can join an ADS using Kerberos. Samba in Active Directory member mode can accept Kerberos tickets.

In **`smb.conf`**, the following directives make Samba an Active Directory member server:

```
[GLOBAL]
...
security = ADS
realm = EXAMPLE.COM
password server = kerberos.example.com
...
```

22.7.1.3. Server Security Mode (User-Level Security)

Server security mode was previously used when Samba was not capable of acting as a domain member server.

Note

It is highly recommended to *not* use this mode since there are numerous security drawbacks.

In **`smb.conf`**, the following directives enable Samba to operate in server security mode:

```
[GLOBAL]
...
encrypt passwords = Yes
security = server
password server = "NetBIOS_of_Domain_Controller"
...
```

22.7.2. Share-Level Security

With share-level security, the server accepts only a password without an explicit username from the client. The server expects a password for each share, independent of the username. There have been recent reports that Microsoft Windows clients have compatibility issues with share-level security servers. Samba developers strongly discourage use of share-level security.

In **`smb.conf`**, the **`security = share`** directive that sets share-level security is:

```
[GLOBAL]
...
security = share
...
```

22.8. Samba Account Information Databases

The latest release of Samba offers many new features including new password database backends not previously available. Samba version 3.0.0 fully supports all databases used in previous versions of Samba. However, although supported, many backends may not be suitable for production use.

The following is a list different backends you can use with Samba. Other backends not listed here may also be available.

Plain Text

Plain text backends are nothing more than the **`/etc/passwd`** type backends. With a plain text backend, all usernames and passwords are sent unencrypted between the client and the Samba server. This method is very unsecure and is not recommended for use by any means. It is possible that different Windows clients connecting to the Samba server with plain text passwords cannot support such an authentication method.

`smbpasswd`

A popular backend used in previous Samba packages, the **`smbpasswd`** backend utilizes a plain ASCII text layout that includes the MS Windows LanMan and NT account, and encrypted password information. The **`smbpasswd`** backend lacks the storage of the Windows NT/2000/2003 SAM extended controls. The **smbpasswd** backend is not recommended because it does not scale well or hold any Windows information, such as RIDs for NT-based groups. The **tdbsam** backend solves these issues for use in a smaller database (250 users), but is still not an enterprise-class solution.

ldapsam_compat

The **ldapsam_compat** backend allows continued OpenLDAP support for use with upgraded versions of Samba. This option normally used when migrating to Samba 3.0.

tdbsam

The **tdbsam** backend provides an ideal database backend for local servers, servers that do not need built-in database replication, and servers that do not require the scalability or complexity of LDAP. The **tdbsam** backend includes all of the **smbpasswd** database information as well as the previously-excluded SAM information. The inclusion of the extended SAM data allows Samba to implement the same account and system access controls as seen with Windows NT/2000/2003-based systems.

The **tdbsam** backend is recommended for 250 users at most. Larger organizations should require Active Directory or LDAP integration due to scalability and possible network infrastructure concerns.

ldapsam

The **ldapsam** backend provides an optimal distributed account installation method for Samba. LDAP is optimal because of its ability to replicate its database to any number of servers using the OpenLDAP **slurpd** daemon. LDAP databases are light-weight and scalable, and as such are preferred by large enterprises.

If you are upgrading from a previous version of Samba to 3.0, note that the **/usr/share/doc/samba-<*version*>/LDAP/samba.schema** has changed. This file contains the *attribute syntax definitions* and *objectclass definitions* that the **ldapsam** backend will need in order to function properly.

As such, if you are using the **ldapsam** backend for your Samba server, you will need to configure **slapd** to include this schema file. Refer to Section 28.5, "The **/etc/openldap/schema/** Directory" for directions on how to do this.

Note

You will need to have the **openldap-server** package installed if you want to use the **ldapsam** backend.

mysqlsam

The **mysqlsam** backend uses a MySQL-based database backend. This is useful for sites that already implement MySQL. At present, **mysqlsam** is now packed in a module separate from Samba, and as such is not officially supported by Samba.

22.9. Samba Network Browsing

Network browsing enables Windows and Samba servers to appear in the Windows **`Network Neighborhood`**. Inside the **`Network Neighborhood`**, icons are represented as servers and if opened, the server's shares and printers that are available are displayed.

Network browsing capabilities require NetBIOS over TCP/IP. NetBIOS-based networking uses broadcast (UDP) messaging to accomplish browse list management. Without NetBIOS and WINS as the primary method for TCP/IP hostname resolution, other methods such as static files (**`/etc/hosts`**) or DNS, must be used.

A domain master browser collates the browse lists from local master browsers on all subnets so that browsing can occur between workgroups and subnets. Also, the domain master browser should preferably be the local master browser for its own subnet.

22.9.1. Domain Browsing

By default, a Windows server PDC for a domain is also the domain master browser for that domain. A Samba server must *not* be set up as a domain master server in this type of situation

For subnets that do not include the Windows server PDC, a Samba server can be implemented as a local master browser. Configuring the **`smb.conf`** for a local master browser (or no browsing at all) in a domain controller environment is the same as workgroup configuration.

22.9.2. WINS (Windows Internetworking Name Server)

Either a Samba server or a Windows NT server can function as a WINS server. When a WINS server is used with NetBIOS enabled, UDP unicasts can be routed which allows name resolution across networks. Without a WINS server, the UDP broadcast is limited to the local subnet and therefore cannot be routed to other subnets, workgroups, or domains. If WINS replication is necessary, do not use Samba as your primary WINS server, as Samba does not currently support WINS replication.

In a mixed NT/2000/2003 server and Samba environment, it is recommended that you use the Microsoft WINS capabilities. In a Samba-only environment, it is recommended that you use *only one* Samba server for WINS.

The following is an example of the **`smb.conf`** file in which the Samba server is serving as a WINS server:

```
[global]
wins support = Yes
```

Note

All servers (including Samba) should connect to a WINS server to resolve NetBIOS names. Without WINS, browsing only occurs on the local subnet. Furthermore, even if a domain-wide list is somehow obtained, hosts cannot be resolved for the client without WINS.

22.10. Samba with CUPS Printing Support

Samba allows client machines to share printers connected to the Samba server. In addition, Samba also allows client machines to send documents built in Linux to Windows printer shares. Although there are other printing systems that function with Red Hat Enterprise Linux, CUPS (Common UNIX Print System) is the recommended printing system due to its close integration with Samba.

22.10.1. Simple `smb.conf` Settings

The following example shows a very basic **`smb.conf`** configuration for CUPS support:

```
[global]
load printers = Yes
printing = cups
printcap name = cups
[printers]
comment = All Printers
path = /var/spool/samba/print
printer = IBMInfoP
browseable = No
public = Yes
guest ok = Yes
writable = No
printable = Yes
printer admin = @ntadmins
[print$]
comment = Printer Drivers Share
path = /var/lib/samba/drivers
write list = ed, john
printer admin = ed, john
```

Other printing configurations are also possible. To add additional security and privacy for printing confidential documents, users can have their own print spooler not located in a public path. If a job fails, other users would not have access to the file.

The **`print$`** share contains printer drivers for clients to access if not available locally. The**`print$`** share is optional and may not be required depending on the organization.

Setting **`browseable`** to **`Yes`** enables the printer to be viewed in the Windows Network Neighborhood, provided the Samba server is set up correctly in the domain/workgroup.

22.11. Samba Distribution Programs

`findsmb`

`findsmb <`*`subnet_broadcast_address`*`>`

The **`findsmb`** program is a Perl script which reports information about SMB-aware systems on a specific subnet. If no subnet is specified the local subnet is used. Items displayed include IP address, NetBIOS name, workgroup or domain name, operating system, and version.

The following example shows the output of executing **`findsmb`** as any valid user on a system:

```
~]# findsmb
IP ADDR       NETBIOS NAME  WORKGROUP/OS/VERSION
------------------------------------------------------------------
10.1.59.25    VERVE          [MYGROUP] [Unix] [Samba 3.0.0-15]
10.1.59.26    STATION22      [MYGROUP] [Unix] [Samba 3.0.2-7.FC1]
10.1.56.45    TREK          +[WORKGROUP] [Windows 5.0] [Windows 2000 LAN
Manager]
10.1.57.94    PIXEL          [MYGROUP] [Unix] [Samba 3.0.0-15]
10.1.57.137   MOBILE001      [WORKGROUP] [Windows 5.0] [Windows 2000 LAN
```

```
Manager]
10.1.57.141   JAWS          +[KWIKIMART] [Unix] [Samba 2.2.7a-security-
rollup-fix]
10.1.56.159   FRED          +[MYGROUP] [Unix] [Samba 3.0.0-14.3E]
10.1.59.192   LEGION        *[MYGROUP] [Unix] [Samba 2.2.7-security-rollup-
fix]
10.1.56.205   NANCYN        +[MYGROUP] [Unix] [Samba 2.2.7a-security-rollup-
fix]
```

net

net *<protocol>* *<function>* *<misc_options>* *<target_options>*

The **net** utility is similar to the **net** utility used for Windows and MS-DOS. The first argument is used to specify the protocol to use when executing a command. The ***<protocol>*** option can be **ads**, **rap**, or **rpc** for specifying the type of server connection. Active Directory uses **ads**, Win9x/NT3 uses **rap**, and Windows NT4/2000/2003 uses **rpc**. If the protocol is omitted, **net** automatically tries to determine it.

The following example displays a list the available shares for a host named **wakko**:

```
~]# net -l share -S wakko
Password:

Enumerating shared resources (exports) on remote server:

Share name   Type     Description
----------   ----     -----------
data         Disk     Wakko data share
tmp          Disk     Wakko tmp share
IPC$         IPC      IPC Service (Samba Server)
ADMIN$       IPC      IPC Service (Samba Server)
```

The following example displays a list of Samba users for a host named **wakko**:

```
~]# net -l user -S wakko
root password:
User name             Comment
-----------------------------
andriusb              Documentation
joe                   Marketing
lisa                  Sales
```

nmblookup

nmblookup *<options>* *<netbios_name>*

The **nmblookup** program resolves NetBIOS names into IP addresses. The program broadcasts its query on the local subnet until the target machine replies.

Here is an example:

```
~]# nmblookup trek
querying trek on 10.1.59.255
10.1.56.45 trek<00>
```

pdbedit

pdbedit *<options>*

The **pdbedit** program manages accounts located in the SAM database. All backends are supported including **smbpasswd**, LDAP, NIS+, and the **tdb** database library.

The following are examples of adding, deleting, and listing users:

```
~]# pdbedit -a kristin
new password:
retype new password:
Unix username:        kristin
NT username:
Account Flags:        [U          ]
User SID:             S-1-5-21-1210235352-3804200048-1474496110-2012
Primary Group SID:    S-1-5-21-1210235352-3804200048-1474496110-2077
Full Name: Home Directory:       \\wakko\kristin
HomeDir Drive:
Logon Script:
Profile Path:         \\wakko\kristin\profile
Domain:               WAKKO
Account desc:
Workstations: Munged
dial:
Logon time:           0
Logoff time:          Mon, 18 Jan 2038 22:14:07 GMT
Kickoff time:         Mon, 18 Jan 2038 22:14:07 GMT
Password last set:    Thu, 29 Jan 2004 08:29:28
GMT Password can change:  Thu, 29 Jan 2004 08:29:28 GMT
Password must change: Mon, 18 Jan 2038 22:14:07 GMT

~]# pdbedit -v -L kristin
Unix username:        kristin
NT username:
Account Flags:        [U          ]
User SID:             S-1-5-21-1210235352-3804200048-1474496110-2012
Primary Group SID:    S-1-5-21-1210235352-3804200048-1474496110-2077
Full Name:
Home Directory:       \\wakko\kristin
HomeDir Drive:
Logon Script:
Profile Path:         \\wakko\kristin\profile
Domain:               WAKKO
Account desc:
Workstations: Munged
dial:
Logon time:           0
Logoff time:          Mon, 18 Jan 2038 22:14:07 GMT
Kickoff time:         Mon, 18 Jan 2038 22:14:07 GMT
Password last set:    Thu, 29 Jan 2004 08:29:28 GMT
Password can change:  Thu, 29 Jan 2004 08:29:28 GMT
Password must change: Mon, 18 Jan 2038 22:14:07 GMT

~]# pdbedit -L
andriusb:505:
```

```
joe:503:
lisa:504:
kristin:506:

~]# pdbedit -x joe
~]# pdbedit -L

andriusb:505: lisa:504: kristin:506:
```

`rpcclient`

`rpcclient <`*`server`*`> <`*`options`*`>`

The **`rpcclient`** program issues administrative commands using Microsoft RPCs, which provide access to the Windows administration graphical user interfaces (GUIs) for systems management. This is most often used by advanced users that understand the full complexity of Microsoft RPCs.

`smbcacls`

`smbcacls <`*`//server/share`*`> <`*`filename`*`> <`*`options`*`>`

The **`smbcacls`** program modifies Windows ACLs on files and directories shared by the Samba server.

`smbclient`

`smbclient <`*`//server/share`*`> <`*`password`*`> <`*`options`*`>`

The **`smbclient`** program is a versatile UNIX client which provides functionality similar to **`ftp`**.

`smbcontrol`

`smbcontrol -i <`*`options`*`>`

`smbcontrol <`*`options`*`> <`*`destination`*`> <`*`messagetype`*`> <`*`parameters`*`>`

The **`smbcontrol`** program sends control messages to running **`smbd`** or **`nmbd`** daemons. Executing **`smbcontrol -i`** runs commands interactively until a blank line or a 'q' is entered.

`smbpasswd`

`smbpasswd <`*`options`*`> <`*`username`*`> <`*`password`*`>`

The **`smbpasswd`** program manages encrypted passwords. This program can be run by a superuser to change any user's password as well as by an ordinary user to change their own Samba password.

`smbspool`

`smbspool <`*`job`*`> <`*`user`*`> <`*`title`*`> <`*`copies`*`> <`*`options`*`> <`*`filename`*`>`

The **`smbspool`** program is a CUPS-compatible printing interface to Samba. Although designed for use with CUPS printers, **`smbspool`** can work with non-CUPS printers as well.

`smbstatus`

`smbstatus <`*`options`*`>`

The **smbstatus** program displays the status of current connections to a Samba server.

smbtar

smbtar *<options>*

The **smbtar** program performs backup and restores of Windows-based share files and directories to a local tape archive. Though similar to the **tar** command, the two are not compatible.

testparm

testparm *<options> <filename> <hostname IP_address>*

The **testparm** program checks the syntax of the **smb.conf** file. If your **smb.conf** file is in the default location (**/etc/samba/smb.conf**) you do not need to specify the location. Specifying the hostname and IP address to the **testparm** program verifies that the **hosts.allow** and **host.deny** files are configured correctly. The **testparm** program also displays a summary of your **smb.conf** file and the server's role (stand-alone, domain, etc.) after testing. This is convenient when debugging as it excludes comments and concisely presents information for experienced administrators to read.

For example:

```
~]# testparm
Load smb config files from /etc/samba/smb.conf
Processing section "[homes]"
Processing section "[printers]"
Processing section "[tmp]"
Processing section "[html]"
Loaded services file OK.
Server role: ROLE_STANDALONE
Press enter to see a dump of your service definitions

<enter>
# Global parameters
[global]
 workgroup = MYGROUP
 server string = Samba Server
 security = SHARE
 log file = /var/log/samba/%m.log
 max log size = 50
 socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
 dns proxy = No
[homes]
 comment = Home Directories
 read only = No
 browseable = No
[printers]
 comment = All Printers
 path = /var/spool/samba
 printable = Yes
 browseable = No
[tmp]
 comment = Wakko tmp
 path = /tmp
 guest only = Yes
[html]
```

```
comment = Wakko www
path = /var/www/html
force user = andriusb
force group = users
read only = No
guest only = Yes
```

`wbinfo`

`wbinfo <options>`

The **`wbinfo`** program displays information from the **`winbindd`** daemon. The **`winbindd`** daemon must be running for **`wbinfo`** to work.

22.12. Additional Resources

The following sections give you the means to explore Samba in greater detail.

22.12.1. Installed Documentation

- **`/usr/share/doc/samba-<version-number>/`** — All additional files included with the Samba distribution. This includes all helper scripts, sample configuration files, and documentation.

  This directory also contains online versions of *The Official Samba-3 HOWTO-Collection* and *Samba-3 by Example*, both of which are cited below.

22.12.2. Related Books

- *The Official Samba-3 HOWTO-Collection* by John H. Terpstra and Jelmer R. Vernooij; Prentice Hall — The official Samba-3 documentation as issued by the Samba development team. This is more of a reference guide than a step-by-step guide.
- *Samba-3 by Example* by John H. Terpstra; Prentice Hall — This is another official release issued by the Samba development team which discusses detailed examples of OpenLDAP, DNS, DHCP, and printing configuration files. This has step-by-step related information that helps in real-world implementations.
- *Using Samba, 2nd Edition* by Jay T's, Robert Eckstein, and David Collier-Brown; O'Reilly — A good resource for novice to advanced users, which includes comprehensive reference material.

22.12.3. Useful Websites

- http://www.samba.org/ — Homepage for the Samba distribution and all official documentation created by the Samba development team. Many resources are available in HTML and PDF formats, while others are only available for purchase. Although many of these links are not Red Hat Enterprise Linux specific, some concepts may apply.
- http://samba.org/samba/archives.html — Active email lists for the Samba community. Enabling digest mode is recommended due to high levels of list activity.
- Samba newsgroups — Samba threaded newsgroups, such as gmane.org, that use the NNTP protocol are also available. This an alternative to receiving mailing list emails.

Chapter 23. Dynamic Host Configuration Protocol (DHCP)

Dynamic Host Configuration Protocol (DHCP) is a network protocol that automatically assigns TCP/IP information to client machines. Each DHCP client connects to the centrally located DHCP server, which returns that client's network configuration (including the IP address, gateway, and DNS servers).

23.1. Why Use DHCP?

DHCP is useful for automatic configuration of client network interfaces. When configuring the client system, the administrator chooses DHCP instead of specifying an IP address, netmask, gateway, or DNS servers. The client retrieves this information from the DHCP server. DHCP is also useful if an administrator wants to change the IP addresses of a large number of systems. Instead of reconfiguring all the systems, he can just edit one DHCP configuration file on the server for the new set of IP addresses. If the DNS servers for an organization changes, the changes are made on the DHCP server, not on the DHCP clients. When the administrator restarts the network or reboots the clients, the changes will go into effect.

If an organization has a functional DHCP server properly connected to a network, laptops and other mobile computer users can move these devices from office to office.

23.2. Configuring a DHCP Server

The **dhcp** package contains an ISC DHCP server. First, install the package as the superuser:

```
~]# yum install dhcp
```

Installing the **dhcp** package creates a file, **/etc/dhcpd.conf**, which is merely an empty configuration file:

```
~]# cat /etc/dhcpd.conf
#
# DHCP Server Configuration file.
#   see /usr/share/doc/dhcp*/dhcpd.conf.sample
```

The sample configuration file can be found at **/usr/share/doc/dhcp-<*version*>/dhcpd.conf.sample**. You should use this file to help you configure **/etc/dhcpd.conf**, which is explained in detail below.

DHCP also uses the file **/var/lib/dhcpd/dhcpd.leases** to store the client lease database. Refer to Section 23.2.2, “Lease Database” for more information.

23.2.1. Configuration File

The first step in configuring a DHCP server is to create the configuration file that stores the network information for the clients. Use this file to declare options and global options for client systems.

The configuration file can contain extra tabs or blank lines for easier formatting. Keywords are case-insensitive and lines beginning with a hash mark (#) are considered comments.

Two DNS update schemes are currently implemented — the ad-hoc DNS update mode and the interim DHCP-DNS interaction draft update mode. If and when these two are accepted as part of the Internet Engineering Task Force (IETF) standards process, there will be a third mode — the standard DNS update method. You must configure the DNS server for compatibility with these schemes. Version 3.0b2pl11 and previous versions used the ad-hoc mode; however, it has been deprecated. To keep the same behavior, add the following line to the top of the configuration file:

```
ddns-update-style ad-hoc;
```

To use the recommended mode, add the following line to the top of the configuration file:

```
ddns-update-style interim;
```

Refer to the **dhcpd.conf** man page for details about the different modes.

There are two types of statements in the configuration file:

- Parameters — State how to perform a task, whether to perform a task, or what network configuration options to send to the client.
- Declarations — Describe the topology of the network, describe the clients, provide addresses for the clients, or apply a group of parameters to a group of declarations.

The parameters that start with the keyword option are referred to as *options*. These options control DHCP options; whereas, parameters configure values that are not optional or control how the DHCP server behaves.

Parameters (including options) declared before a section enclosed in curly brackets ({ }) are considered global parameters. Global parameters apply to all the sections below it.

Important

If the configuration file is changed, the changes do not take effect until the DHCP daemon is restarted with the command **service dhcpd restart**.

Note

Instead of changing a DHCP configuration file and restarting the service each time, using the **omshell** command provides an interactive way to connect to, query, and change the configuration of a DHCP server. By using **omshell**, all changes can be made while the server is running. For more information on **omshell**, refer to the **omshell** man page.

In Example 23.1, “Subnet Declaration”, the **routers**, **subnet-mask**, **domain-name**, **domain-name-servers**, and **time-offset** options are used for any **host** statements declared below it.

Additionally, a **subnet** can be declared, a **subnet** declaration must be included for every subnet in the network. If it is not, the DHCP server fails to start.

In this example, there are global options for every DHCP client in the subnet and a **range** declared. Clients are assigned an IP address within the **range**.

Example 23.1. Subnet Declaration

```
subnet 192.168.1.0 netmask 255.255.255.0 {
        option routers                  192.168.1.254;
        option subnet-mask              255.255.255.0;
        option domain-name              "example.com";
        option domain-name-servers       192.168.1.1;

        option time-offset              -18000;     # Eastern Standard
Time

 range 192.168.1.10 192.168.1.100;
}
```

All subnets that share the same physical network should be declared within a **`shared-network`** declaration as shown in Example 23.2, “Shared-network Declaration”. Parameters within the **`shared-network`**, but outside the enclosed **`subnet`** declarations, are considered to be global parameters. The name of the **`shared-network`** must be a descriptive title for the network, such as using the title 'test-lab' to describe all the subnets in a test lab environment.

Example 23.2. Shared-network Declaration

```
shared-network name {
    option domain-name              "test.redhat.com";
    option domain-name-servers      ns1.redhat.com, ns2.redhat.com;
    option routers                  192.168.0.254;
    more parameters for EXAMPLE shared-network
    subnet 192.168.1.0 netmask 255.255.252.0 {
        parameters for subnet
        range 192.168.1.1 192.168.1.254;
    }
    subnet 192.168.2.0 netmask 255.255.252.0 {
        parameters for subnet
        range 192.168.2.1 192.168.2.254;
    }
}
```

As demonstrated in Example 23.3, “Group Declaration”, the **`group`** declaration is used to apply global parameters to a group of declarations. For example, shared networks, subnets, and hosts can be grouped.

Example 23.3. Group Declaration

```
group {
   option routers                  192.168.1.254;
   option subnet-mask              255.255.255.0;

   option domain-name              "example.com";
   option domain-name-servers       192.168.1.1;

   option time-offset              -18000;     # Eastern Standard Time

   host apex {
      option host-name "apex.example.com";
      hardware ethernet 00:A0:78:8E:9E:AA;
      fixed-address 192.168.1.4;
   }
   host raleigh {
      option host-name "raleigh.example.com";
      hardware ethernet 00:A1:DD:74:C3:F2;
      fixed-address 192.168.1.6;
   }
}
```

To configure a DHCP server that leases a dynamic IP address to a system within a subnet, modify Example 23.4, “Range Parameter” with your values. It declares a default lease time, maximum lease time, and network configuration values for the clients. This example assigns IP addresses in the **`range`** 192.168.1.10 and 192.168.1.100 to client systems.

Example 23.4. Range Parameter

```
default-lease-time 600;
max-lease-time 7200;
option subnet-mask 255.255.255.0;
option broadcast-address 192.168.1.255;
option routers 192.168.1.254;
option domain-name-servers 192.168.1.1, 192.168.1.2;
option domain-name "example.com";

subnet 192.168.1.0 netmask 255.255.255.0 {
   range 192.168.1.10 192.168.1.100;
}
```

To assign an IP address to a client based on the MAC address of the network interface card, use the **`hardware ethernet`** parameter within a **`host`** declaration. As demonstrated in Example 23.5, “Static IP Address using DHCP”, the **`host apex`** declaration specifies that the network interface card with the MAC address 00:A0:78:8E:9E:AA always receives the IP address 192.168.1.4.

Note that the optional parameter **`host-name`** can also be used to assign a host name to the client.

Example 23.5. Static IP Address using DHCP

```
host apex {
   option host-name "apex.example.com";
   hardware ethernet 00:A0:78:8E:9E:AA;
   fixed-address 192.168.1.4;
}
```

Note

The sample configuration file provided can be used as a starting point and custom configuration options can be added to it. To copy it to the proper location, use the following command:

```
cp /usr/share/doc/dhcp-<version-number>/dhcpd.conf.sample
/etc/dhcpd.conf
```

(where <*version-number*> is the DHCP version number).

For a complete list of option statements and what they do, refer to the **`dhcp-options`** man page.

23.2.2. Lease Database

On the DHCP server, the file **`/var/lib/dhcpd/dhcpd.leases`** stores the DHCP client lease database. Do not change this file. DHCP lease information for each recently assigned IP address is automatically stored in the lease database. The information includes the length of the lease, to whom the IP address has been assigned, the start and end dates for the lease, and the MAC address of the network interface card that was used to retrieve the lease.

All times in the lease database are in Coordinated Universal Time (UTC), not local time.

The lease database is recreated from time to time so that it is not too large. First, all known leases are saved in a temporary lease database. The **`dhcpd.leases`** file is renamed **`dhcpd.leases~`** and the temporary lease database is written to **`dhcpd.leases`**.

The DHCP daemon could be killed or the system could crash after the lease database has been renamed to the backup file but before the new file has been written. If this happens, the **`dhcpd.leases`** file does not exist, but it is required to start the service. Do not create a new lease file. If you do, all old leases are lost which causes many problems. The correct solution is to rename the **`dhcpd.leases~`** backup file to **`dhcpd.leases`** and then start the daemon.

23.2.3. Starting and Stopping the Server

Important

When the DHCP server is started for the first time, it fails unless the **`dhcpd.leases`** file exists. Use the command **`touch /var/lib/dhcpd/dhcpd.leases`** to create the file if it does not exist.

If the same server is also running BIND as a DNS server, this step is not necessary, as starting the **`named`** service automatically checks for a **`dhcpd.leases`** file.

To start the DHCP service, use the command **`/sbin/service dhcpd start`**. To stop the DHCP server, use the command **`/sbin/service dhcpd stop`**.

By default, the DHCP service does not start at boot time. To configure the daemon to start automatically at boot time, refer to Chapter 18, *Controlling Access to Services*.

If more than one network interface is attached to the system, but the DHCP server should only be started on one of the interfaces, configure the DHCP server to start only on that device. In **`/etc/sysconfig/dhcpd`**, add the name of the interface to the list of **`DHCPDARGS`**:

```
# Command line options here
DHCPDARGS=eth0
```

This is useful for a firewall machine with two network cards. One network card can be configured as a DHCP client to retrieve an IP address to the Internet. The other network card can be used as a DHCP server for the internal network behind the firewall. Specifying only the network card connected to the internal network makes the system more secure because users can not connect to the daemon via the Internet.

Other command line options that can be specified in **`/etc/sysconfig/dhcpd`** include:

- **`-p <`*`portnum`*`>`** — Specifies the UDP port number on which **`dhcpd`** should listen. The default is port 67. The DHCP server transmits responses to the DHCP clients at a port number one greater than the UDP port specified. For example, if the default port 67 is used, the server listens on port 67 for requests and responses to the client on port 68. If a port is specified here and the DHCP relay agent is used, the same port on which the DHCP relay agent should listen must be specified. Refer to Section 23.2.4, "DHCP Relay Agent" for details.

- **`-f`** — Runs the daemon as a foreground process. This is mostly used for debugging.

- **`-d`** — Logs the DHCP server daemon to the standard error descriptor. This is mostly used for debugging. If this is not specified, the log is written to **`/var/log/messages`**.

- **`-cf <`*`filename`*`>`** — Specifies the location of the configuration file. The default location is **`/etc/dhcpd.conf`**.

- **`-lf <`*`filename`*`>`** — Specifies the location of the lease database file. If a lease database file already exists, it is very important that the same file be used every time the DHCP server is started. It is strongly recommended that this option only be used for debugging purposes on non-production machines. The default location is **`/var/lib/dhcpd/dhcpd.leases`**.

- **`-q`** — Do not print the entire copyright message when starting the daemon.

23.2.4. DHCP Relay Agent

The DHCP Relay Agent (**`dhcrelay`**) allows for the relay of DHCP and BOOTP requests from a subnet with no DHCP server on it to one or more DHCP servers on other subnets.

When a DHCP client requests information, the DHCP Relay Agent forwards the request to the list of DHCP servers specified when the DHCP Relay Agent is started. When a DHCP server returns a reply, the reply is broadcast or unicast on the network that sent the original request.

The DHCP Relay Agent listens for DHCP requests on all interfaces unless the interfaces are specified in **`/etc/sysconfig/dhcrelay`** with the **`INTERFACES`** directive.

To start the DHCP Relay Agent, use the command **`service dhcrelay start`**.

23.3. Configuring a DHCP Client

The first step for configuring a DHCP client is to make sure the kernel recognizes the network interface card. Most cards are recognized during the installation process and the system is configured to use the correct kernel module for the card. If a card is added after installation, **Kudzu** [8] will recognize it and prompt you for the proper kernel module (Be sure to check the Hardware Compatibility List at http://hardware.redhat.com/hcl/). If either the installation program or kudzu does not recognize the network card, you can load the correct kernel module (refer to Chapter 45, *General Parameters and Modules* for details).

To configure a DHCP client manually, modify the **/etc/sysconfig/network** file to enable networking and the configuration file for each network device in the **/etc/sysconfig/network-scripts** directory. In this directory, each device should have a configuration file named **ifcfg-eth0**, where **eth0** is the network device name.

The **/etc/sysconfig/network** file should contain the following line:

```
NETWORKING=yes
```

The **NETWORKING** variable must be set to **yes** if you want networking to start at boot time.

The **/etc/sysconfig/network-scripts/ifcfg-eth0** file should contain the following lines:

```
DEVICE=eth0
BOOTPROTO=dhcp
ONBOOT=yes
```

A configuration file is needed for each device to be configured to use DHCP.

Other options for the network script includes:

- **DHCP_HOSTNAME** — Only use this option if the DHCP server requires the client to specify a hostname before receiving an IP address. (The DHCP server daemon in Red Hat Enterprise Linux does not support this feature.)
- **PEERDNS=*<answer>*** , where ***<answer>*** is one of the following:
  - **yes** — Modify **/etc/resolv.conf** with information from the server. If using DHCP, then **yes** is the default.
  - **no** — Do not modify **/etc/resolv.conf**.
- **SRCADDR=*<address>*** , where ***<address>*** is the specified source IP address for outgoing packets.
- **USERCTL=*<answer>*** , where ***<answer>*** is one of the following:
  - **yes** — Non-root users are allowed to control this device.
  - **no** — Non-root users are not allowed to control this device.

If you prefer using a graphical interface, refer to Chapter 17, *Network Configuration* for instructions on using the **Network Administration Tool** to configure a network interface to use DHCP.

Note

For advanced configurations of client DHCP options such as protocol timing, lease requirements and requests, dynamic DNS support, aliases, as well as a wide variety of values to override, prepend, or append to client-side configurations, refer to the **dhclient** and **dhclient.conf** man pages.

23.4. Configuring a Multihomed DHCP Server

23.4. Configuring a Multihomed DHCP Server

A multihomed DHCP server serves multiple networks, that is, multiple subnets. The examples in these sections detail how to configure a DHCP server to serve multiple networks, select which network interfaces to listen on, and how to define network settings for systems that move networks.

Before making any changes, back up the existing **`/etc/sysconfig/dhcpd`** and **`/etc/dhcpd.conf`** files.

The DHCP daemon listens on all network interfaces unless otherwise specified. Use the **`/etc/sysconfig/dhcpd`** file to specify which network interfaces the DHCP daemon listens on. The following **`/etc/sysconfig/dhcpd`** example specifies that the DHCP daemon listens on the **`eth0`** and **`eth1`** interfaces:

```
DHCPDARGS="eth0 eth1";
```

If a system has three network interfaces cards -- **`eth0`**, **`eth1`**, and **`eth2`** -- and it is only desired that the DHCP daemon listens on **`eth0`**, then only specify **`eth0`** in **`/etc/sysconfig/dhcpd`**:

```
DHCPDARGS="eth0";
```

The following is a basic **`/etc/dhcpd.conf`** file, for a server that has two network interfaces, **`eth0`** in a 10.0.0.0/24 network, and **`eth1`** in a 172.16.0.0/24 network. Multiple **`subnet`** declarations allow different settings to be defined for multiple networks:

```
ddns-update-style interim;
default-lease-time 600;
max-lease-time 7200;

subnet 10.0.0.0 netmask 255.255.255.0 {
 option subnet-mask 255.255.255.0;
 option routers 10.0.0.1;
 range 10.0.0.5 10.0.0.15;
}

subnet 172.16.0.0 netmask 255.255.255.0 {
 option subnet-mask 255.255.255.0;
 option routers 172.16.0.1;
 range 172.16.0.5 172.16.0.15;

}
```

`subnet` ***`10.0.0.0`*** **`netmask`** ***`255.255.255.0`***

A **`subnet`** declaration is required for every network your DHCP server is serving. Multiple subnets require multiple **`subnet`** declarations. If the DHCP server does not have a network interface in a range of a **`subnet`** declaration, the DHCP server does not serve that network.

If there is only one **`subnet`** declaration, and no network interfaces are in the range of that subnet, the DHCP daemon fails to start, and an error such as the following is logged to **`/var/log/messages`**:

```
dhcpd: No subnet declaration for eth0 (0.0.0.0).
dhcpd: ** Ignoring requests on eth0.  If this is not what
dhcpd:    you want, please write a subnet declaration
dhcpd:    in your dhcpd.conf file for the network segment
```

```
dhcpd:    to which interface eth1 is attached. **
dhcpd:
dhcpd:
dhcpd: Not configured to listen on any interfaces!
```

option subnet-mask *255.255.255.0*;

The **option subnet-mask** option defines a subnet mask, and overrides the **netmask** value in the **subnet** declaration. In simple cases, the subnet and netmask values are the same.

option routers *10.0.0.1*;

The **option routers** option defines the default gateway for the subnet. This is required for systems to reach internal networks on a different subnet, as well as external networks.

range *10.0.0.5 10.0.0.15*;

The **range** option specifies the pool of available IP addresses. Systems are assigned an address from the range of specified IP addresses.

For further information, refer to the **dhcpd.conf(5)** man page.

Warning

Alias interfaces are not supported by DHCP. If an alias interface is the only interface, in the only subnet specified in **/etc/dhcpd.conf**, the DHCP daemon fails to start.

23.4.1. Host Configuration

Before making any changes, back up the existing **/etc/sysconfig/dhcpd** and **/etc/dhcpd.conf** files.

Configuring a single system for multiple networks

The following **/etc/dhcpd.conf** example creates two subnets, and configures an IP address for the same system, depending on which network it connects to:

```
ddns-update-style interim;
default-lease-time 600;
max-lease-time 7200;

subnet 10.0.0.0 netmask 255.255.255.0 {
 option subnet-mask 255.255.255.0;
 option routers 10.0.0.1;
 range 10.0.0.5 10.0.0.15;
}

subnet 172.16.0.0 netmask 255.255.255.0 {
 option subnet-mask 255.255.255.0;
 option routers 172.16.0.1;
 range 172.16.0.5 172.16.0.15;

}

host example0 {
 hardware ethernet 00:1A:6B:6A:2E:0B;
 fixed-address 10.0.0.20;
}

host example1 {
 hardware ethernet 00:1A:6B:6A:2E:0B;
 fixed-address 172.16.0.20;
}
```

`host` ***`example0`***

The **`host`** declaration defines specific parameters for a single system, such as an IP address. To configure specific parameters for multiple hosts, use multiple **`host`** declarations.

Most DHCP clients ignore the name in **`host`** declarations, and as such, this name can anything, as long as it is unique to other **`host`** declarations. To configure the same system for multiple networks, use a different name for each **`host`** declaration, otherwise the DHCP daemon fails to start. Systems are identified by the **`hardware ethernet`** option, not the name in the **`host`** declaration.

`hardware ethernet` ***`00:1A:6B:6A:2E:0B`*****`;`**

The **`hardware ethernet`** option identifies the system. To find this address, run the **`ifconfig`** command on the desired system, and look for the **`HWaddr`** address.

`fixed-address` ***`10.0.0.20`*****`;`**

The **`fixed-address`** option assigns a valid IP address to the system specified by the **`hardware ethernet`** option. This address must be outside the IP address pool specified with the **`range`** option.

If **`option`** statements do not end with a semicolon, the DHCP daemon fails to start, and an error such as the following is logged to **`/var/log/messages`**:

```
/etc/dhcpd.conf line 20: semicolon expected.
dhcpd: }
dhcpd: ^
dhcpd: /etc/dhcpd.conf line 38: unexpected end of file
dhcpd:
dhcpd: ^
dhcpd: Configuration file errors encountered -- exiting
```

Configuring systems with multiple network interfaces

The following **`host`** declarations configure a single system, that has multiple network interfaces, so that each interface receives the same IP address. This configuration will not work if both network interfaces are connected to the same network at the same time:

```
host interface0 {
 hardware ethernet 00:1a:6b:6a:2e:0b;
 fixed-address 10.0.0.18;
}
```

```
host interface1 {
 hardware ethernet 00:1A:6B:6A:27:3A;
 fixed-address 10.0.0.18;
}
```

For this example, **`interface0`** is the first network interface, and **`interface1`** is the second interface. The different **`hardware ethernet`** options identify each interface.

If such a system connects to another network, add more **`host`** declarations, remembering to:

- assign a valid **`fixed-address`** for the network the host is connecting to.
- make the name in the **`host`** declaration unique.

When a name given in a **`host`** declaration is not unique, the DHCP daemon fails to start, and an error such as the following is logged to **`/var/log/messages`**:

```
dhcpd: /etc/dhcpd.conf line 31: host interface0: already exists
dhcpd: }
dhcpd: ^
dhcpd: Configuration file errors encountered -- exiting
```

This error was caused by having multiple **`host interface0`** declarations defined in **`/etc/dhcpd.conf`**.

23.5. Additional Resources

For additional configuration options, refer to the following resources.

23.5.1. Installed Documentation

- **dhcpd** man page — Describes how the DHCP daemon works.
- **`dhcpd.conf`** man page — Explains how to configure the DHCP configuration file; includes some examples.
- **`dhcpd.leases`** man page — Explains how to configure the DHCP leases file; includes some examples.
- **`dhcp-options`** man page — Explains the syntax for declaring DHCP options in **`dhcpd.conf`**; includes some examples.
- **`dhcrelay`** man page — Explains the DHCP Relay Agent and its configuration options.
- **`/usr/share/doc/dhcp-<`*`version`*`>/`** — Contains sample files, README files, and release notes for current versions of the DHCP service.

[8] **Kudzu** is a hardware probing tool run at system boot time to determine what hardware has been added or removed from the system.

Chapter 24. Migrating from MySQL 5.0 to MySQL 5.5

Before migrating from MySQL 5.0 to MySQL 5.5, back up all your data, including any MySQL databases. For more information about MySQL 5.1 and MySQL 5.5, refer to the release notes available at http://dev.mysql.com/doc/relnotes/mysql/5.1/en/ and http://dev.mysql.com/doc/relnotes/mysql/5.5/en/.

Note

Red Hat will not issue any more security advisories for the MySQL 5.0 packages (*mysql-5.0.** and related packages). Security advisories will be provided only for MySQL 5.5.

24.1. Upgrading from MySQL 5.0 to MySQL 5.5

The only supported way to upgrade from MySQL 5.0 to MySQL 5.5 is by using MySQL 5.1 as an intermediate step. This is why the *mysql51** Software Collection packages are provided. Note that the MySQL 5.1 packages are not supported and are provided only for the purposes of migrating to MySQL 5.5. You should not use the *mysql51** packages on any of your production systems.

Because the *mysql51* and *mysql55* Software Collections do not conflict with each other or any *mysql* packages, users can install *mysql51* and *mysql55* Software Collections together with *mysql* packages. It is also possible to run all versions of MySQL at the same time. However, the port and socket number must be changed in the `my.cnf` configuration files to prevent specific resources from conflicting.

There are two ways of upgrading from MySQL 5.0 to MySQL 5.5:

- Using the **mysqldump** and **mysql** utilities — the dump and restore upgrade generates an entirely new dump of all databases from one database. Next, the MySQL command line interface is run with the dump file as an input (alternatively, use the **mysqlimport** utility or the `LOAD DATA INFILE` SQL command) within the other database. Appropriate daemons have to be running during both dumping and restoring. Use the `--all-databases` option when executing the **mysqldump** command to include all databases in the resulting dump. The `--routines`, `--triggers`, and `--events` (valid only for MySQL 5.1 and above) options can be used if needed.

  Example 24.1, "Dump and Restore Upgrade" shows the specific commands used to upgrade using the dump and restore method.

- In-place upgrade — consists of copying data files from one database directory to another database directory while both MySQL daemons are stopped. Appropriate permissions and SELinux context must be set for the copied files. The in-place upgrade is usually faster and easier for large databases, but there are some risks and known problems. For more information, refer to the release notes for MySQL 5.1 and MySQL 5.5, linked to at the beginning of this chapter.

  Example 24.2, "In-place Upgrade" shows the specific commands used to perform an in-place upgrade.

After upgrading, either using the dump and restore method or performing an in-place upgrade, start the MySQL server and run the `mysql_upgrade` command. Running the `mysql_upgrade` is necessary to check and repair internal tables. Note that all scripts that interact with a MySQL server from a Software Collection, especially the `mysql_upgrade` script, should be run inside an `scl enable` environment.

Note

While running the **`mysql_upgrade`** command, you may encounter the following errors:

```
You can't use locks with log tables.
```

This is a known issue (reported at http://bugs.mysql.com/bug.php?id=30487), which has no effect on the upgrade process.

Except when data upgrading, consider changing appropriate settings in your **`my.cnf`** configuration file to reflect the new environment.

Example 24.1. Dump and Restore Upgrade

Example of dump and restore upgrading from MySQL 5.0 to MySQL 5.5 Software Collection:

```
~]# service mysqld start
Starting mysqld:                                           [  OK  ]
~]# mysqldump --all-databases --routines > dump.sql
~]# service mysqld stop
Stopping mysqld:                                           [  OK  ]
~]# service mysql51-mysqld start
Starting mysql51-mysqld:                                   [  OK  ]
~]# scl enable mysql51 'mysql' < dump.sql
~]# scl enable mysql51 'mysql_upgrade'
Looking for 'mysql' as: mysql
Looking for 'mysqlcheck' as: mysqlcheck
Running 'mysqlcheck with default connection arguments
Running 'mysqlcheck with default connection arguments
a.t1                                              OK
mysql.columns_priv                                OK
 ⋮
mysql.user                                        OK
Running 'mysql_fix_privilege_tables'...
OK
~]# scl enable mysql51 'mysqldump --all-databases --routines --events' >
dump2.sql
~]# service mysql51-mysqld stop
Stopping mysqld:                                           [  OK  ]
~]# service mysql55-mysqld start
Starting mysql55-mysqld:                                   [  OK  ]
~]# scl enable mysql55 'mysql' < dump2.sql
~]# scl enable mysql55 'mysql_upgrade'
Looking for 'mysql' as: mysql
Looking for 'mysqlcheck' as: mysqlcheck
Running 'mysqlcheck with default connection arguments
Running 'mysqlcheck with default connection arguments
a.t1                                              OK
mysql.columns_priv                                OK
```

```
 ⋮
mysql.user                                          OK
Running 'mysql_fix_privilege_tables'...
OK
```

Example 24.2. In-place Upgrade

Example of an in-place upgrade from MySQL 5.0 to MySQL 5.5 Software Collection:

```
~]# service mysqld stop
Stopping mysqld:                                            [  OK  ]
~]# service mysql51-mysqld stop
Stopping mysql51-mysqld:                                    [  OK  ]
~]# rm -rf /opt/rh/mysql51/root/var/lib/mysql/
~]# cp -r /var/lib/mysql/ /opt/rh/mysql51/root/var/lib/mysql/
~]# chown -R mysql:mysql /opt/rh/mysql51/root/var/lib/mysql/
~]# restorecon -R /opt/rh/mysql51/root/var/lib/mysql/
~]# service mysql51-mysqld start
Starting mysql51-mysqld:                                    [  OK  ]
~]# scl enable mysql51 'mysql_upgrade'
Looking for 'mysql' as: mysql
Looking for 'mysqlcheck' as: mysqlcheck
Running 'mysqlcheck with default connection arguments
Running 'mysqlcheck with default connection arguments
a.t1                                                OK
mysql.columns_priv                                  OK
 ⋮
mysql.user                                          OK
Running 'mysql_fix_privilege_tables'...
OK
~]# service mysql51-mysqld stop
Stopping mysql51-mysqld:                                    [  OK  ]
~]# service mysql55-mysqld stop
Stopping mysql55-mysqld:                                    [  OK  ]
~]# rm -rf /opt/rh/mysql55/root/var/lib/mysql/
~]# cp -r /opt/rh/mysql51/root/var/lib/mysql/
/opt/rh/mysql55/root/var/lib/mysql/
~]# chown -R mysql:mysql /opt/rh/mysql55/root/var/lib/mysql/
~]# restorecon -R /opt/rh/mysql55/root/var/lib/mysql/
~]# service mysql55-mysqld start
Starting mysql55-mysqld:                                    [  OK  ]
~]# scl enable mysql55 'mysql_upgrade'
Looking for 'mysql' as: mysql
Looking for 'mysqlcheck' as: mysqlcheck
Running 'mysqlcheck with default connection arguments
Running 'mysqlcheck with default connection arguments
a.t1                                                OK
mysql.columns_priv                                  OK
 ⋮
mysql.user                                          OK
Running 'mysql_fix_privilege_tables'...
OK
```

Chapter 25. Apache HTTP Server

The Apache HTTP Server is a robust, commercial-grade open source Web server developed by the Apache Software Foundation (http://www.apache.org/). Red Hat Enterprise Linux includes the Apache HTTP Server 2.2 as well as a number of server modules designed to enhance its functionality.

The default configuration file installed with the Apache HTTP Server works without alteration for most situations. This chapter outlines many of the directives found within its configuration file (**`/etc/httpd/conf/httpd.conf`**) to aid those who require a custom configuration or need to convert a configuration file from the older Apache HTTP Server 1.3 format.

Warning

If using the graphical **HTTP Configuration Tool** (*system-config-httpd*), *do not* hand edit the Apache HTTP Server's configuration file as the **HTTP Configuration Tool** regenerates this file whenever it is used.

25.1. Apache HTTP Server 2.2

There are important differences between the Apache HTTP Server 2.2 and version 2.0 (version 2.0 shipped with Red Hat Enterprise Linux 4 and earlier). This section reviews some of the features of Apache HTTP Server 2.2 and outlines important changes. If you are upgrading from version 1.3, you should also read the instructions on migrating from version 1.3 to version 2.0. For instructions on migrating a version 1.3 configuration file to the 2.0 format, refer to Section 25.2.2, “Migrating Apache HTTP Server 1.3 Configuration Files to 2.0”.

25.1.1. Features of Apache HTTP Server 2.2

Apache HTTP Server 2.2 features the following improvements over version 2.0 :

- Improved caching modules (mod_cache, mod_disk_cache, mod_mem_cache).
- A new structure for authentication and authorization support, replacing the authentication modules provided in previous versions.
- Support for proxy load balancing (mod_proxy_balancer)
- support for handling large files (namely, greater than 2GB) on 32-bit platforms

The following changes have been made to the default httpd configuration:

- The mod_cern_meta and mod_asis modules are no longer loaded by default.
- The mod_ext_filter module is now loaded by default.

If upgrading from a previous release of Red Hat Enterprise Linux, the httpd configuration will need to be updated for httpd 2.2. For more information, refer to http://httpd.apache.org/docs/2.2/upgrading.html

25.2. Migrating Apache HTTP Server Configuration Files

25.2.1. Migrating Apache HTTP Server 2.0 Configuration Files

This section outlines migration from version 2.0 to 2.2. If you are migrating from version 1.3, please refer to Section 25.2.2, “Migrating Apache HTTP Server 1.3 Configuration Files to 2.0”.

- Configuration files and startup scripts from version 2.0 need minor adjustments particularly in module names which may have changed. Third party modules which worked in version 2.0 can also work in version 2.2 but need to be recompiled before you load them. Key modules that need to be noted are authentication and authorization modules. For each of the modules which has been renamed the **`LoadModule`** line will need to be updated.
- The **`mod_userdir`** module will only act on requests if you provide a **`UserDir`** directive indicating a directory name. If you wish to maintain the procedures used in version 2.0, add the directive **`UserDir public_html`** in your configuration file.
- To enable SSL, edit the **`httpd.conf`** file adding the necessary **`mod_ssl`** directives. Use **`apachectl start`** as **`apachectl startssl`** is unavailable in version 2.2. You can view an example of SSL configuration for httpd in **`conf/extra/httpd-ssl.conf`**.
- To test your configuration it is advisable to use **`service httpd configtest`** which will detect configuration errors.

More information on upgrading from version 2.0 to 2.2 can be found on http://httpd.apache.org/docs/2.2/upgrading.html.

Important

Due to the vulnerability described in *POODLE: SSLv3 vulnerability (CVE-2014-3566)*, Red Hat recommends disabling **`SSL`** and using only **`TLSv1.1`** or **`TLSv1.2`**. Backwards compatibility can be achieved using **`TLSv1.0`**. Many products Red Hat supports have the ability to use **`SSLv2`** or **`SSLv3`** protocols, or enable them by default. However, the use of **`SSLv2`** or **`SSLv3`** is now strongly recommended against.

25.2.2. Migrating Apache HTTP Server 1.3 Configuration Files to 2.0

This section details migrating an Apache HTTP Server 1.3 configuration file to be utilized by Apache HTTP Server 2.0.

If upgrading to Red Hat Enterprise Linux 5 from Red Hat Enterprise Linux 2.1, note that the new stock configuration file for the Apache HTTP Server 2.0 package is installed as **`/etc/httpd/conf/httpd.conf.rpmnew`** and the original version 1.3 **`httpd.conf`** is left untouched. It is entirely up to you whether to use the new configuration file and migrate the old settings to it, or use the existing file as a base and modify it to suit; however, some parts of the file have changed more than others and a mixed approach is generally the best. The stock configuration files for both version 1.3 and 2.0 are divided into three sections.

If the **`/etc/httpd/conf/httpd.conf`** file is a modified version of the newly installed default and a saved a copy of the original configuration file is available, it may be easiest to invoke the **`diff`** command, as in the following example (logged in as root):

```
diff -u httpd.conf.orig httpd.conf | less
```

This command highlights any modifications made. If a copy of the original file is not available, extract it from an RPM package using the **`rpm2cpio`** and **`cpio`** commands, as in the following example:

```
rpm2cpio apache-<version-number>.i386.rpm | cpio -i --make
```

In the above command, replace <*version-number*> with the version number for the **apache** package.

Finally, it is useful to know that the Apache HTTP Server has a testing mode to check for configuration errors. To use access it, type the following command:

```
apachectl configtest
```

25.2.2.1. Global Environment Configuration

The global environment section of the configuration file contains directives which affect the overall operation of the Apache HTTP Server, such as the number of concurrent requests it can handle and the locations of the various files. This section requires a large number of changes and should be based on the Apache HTTP Server 2.0 configuration file, while migrating the old settings into it.

25.2.2.1.1. Interface and Port Binding

The **BindAddress** and **Port** directives no longer exist; their functionality is now provided by a more flexible **Listen** directive.

If **Port 80** was set in the 1.3 version configuration file, change it to **Listen 80** in the 2.0 configuration file. If **Port** was set to some value *other than 80*, then append the port number to the contents of the **ServerName** directive.

For example, the following is a sample Apache HTTP Server 1.3 directive:

```
Port 123
ServerName www.example.com
```

To migrate this setting to Apache HTTP Server 2.0, use the following structure:

```
Listen 123
ServerName www.example.com:123
```

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

- http://httpd.apache.org/docs-2.0/mod/mpm_common.html#listen
- http://httpd.apache.org/docs-2.0/mod/core.html#servername

25.2.2.1.2. Server-Pool Size Regulation

When the Apache HTTP Server accepts requests, it dispatches child processes or threads to handle them. This group of child processes or threads is known as a *server-pool*. Under Apache HTTP Server 2.0, the responsibility for creating and maintaining these server-pools has been abstracted to a group of modules called *Multi-Processing Modules* (*MPMs*). Unlike other modules, only one module from the MPM group can be loaded by the Apache HTTP Server. There are three MPM modules that ship with 2.0: **prefork**, **worker**, and **perchild**. Currently only the **prefork** and **worker** MPMs are available, although the **perchild** MPM may be available at a later date.

The original Apache HTTP Server 1.3 behavior has been moved into the **prefork** MPM. The **prefork** MPM accepts the same directives as Apache HTTP Server 1.3, so the following directives may be migrated directly:

- **`StartServers`**
- **`MinSpareServers`**
- **`MaxSpareServers`**
- **`MaxClients`**
- **`MaxRequestsPerChild`**

The **`worker`** MPM implements a multi-process, multi-threaded server providing greater scalability. When using this MPM, requests are handled by threads, conserving system resources and allowing large numbers of requests to be served efficiently. Although some of the directives accepted by the **`worker`** MPM are the same as those accepted by the **`prefork`** MPM, the values for those directives should not be transferred directly from an Apache HTTP Server 1.3 installation. It is best to instead use the default values as a guide, then experiment to determine what values work best.

Important

To use the **`worker`** MPM, create the file **`/etc/sysconfig/httpd`** and add the following directive:

```
HTTPD=/usr/sbin/httpd.worker
```

For more on the topic of MPMs, refer to the following documentation on the Apache Software Foundation's website:

- http://httpd.apache.org/docs-2.0/mpm.html

25.2.2.1.3. Dynamic Shared Object (DSO) Support

There are many changes required here, and it is highly recommended that anyone trying to modify an Apache HTTP Server 1.3 configuration to suit version 2.0 (as opposed to migrating the changes into the version 2.0 configuration) copy this section from the stock Apache HTTP Server 2.0 configuration file.

Those who do not want to copy the section from the stock Apache HTTP Server 2.0 configuration should note the following:

- The **`AddModule`** and **`ClearModuleList`** directives no longer exist. These directives where used to ensure that modules could be enabled in the correct order. The Apache HTTP Server 2.0 API allows modules to specify their ordering, eliminating the need for these two directives.
- The order of the **`LoadModule`** lines are no longer relevant in most cases.
- Many modules have been added, removed, renamed, split up, or incorporated into others.
- **`LoadModule`** lines for modules packaged in their own RPMs (**`mod_ssl`**, **`php`**, **`mod_perl`**, and the like) are no longer necessary as they can be found in their relevant files within the **`/etc/httpd/conf.d/`** directory.
- The various **`HAVE_XXX`** definitions are no longer defined.

Important

If modifying the original file, note that it is of paramount importance that the **`httpd.conf`** contains the following directive:

```
Include conf.d/*.conf
```

Omission of this directive results in the failure of all modules packaged in their own RPMs (such as **`mod_perl`**, **`php`**, and **`mod_ssl`**).

25.2.2.1.4. Other Global Environment Changes

The following directives have been removed from Apache HTTP Server 2.0's configuration:

- ***`ServerType`*** — The Apache HTTP Server can only be run as **`ServerType standalone`** making this directive irrelevant.
- ***`AccessConfig`*** and ***`ResourceConfig`*** — These directives have been removed as they mirror the functionality of the **`Include`** directive. If the **`AccessConfig`** and **`ResourceConfig`** directives are set, replace them with **`Include`** directives.

  To ensure that the files are read in the order implied by the older directives, the **`Include`** directives should be placed at the end of the **`httpd.conf`**, with the one corresponding to **`ResourceConfig`** preceding the one corresponding to **`AccessConfig`**. If using the default values, include them explicitly as **`conf/srm.conf`** and **`conf/access.conf`** files.

25.2.2.2. Main Server Configuration

The main server configuration section of the configuration file sets up the main server, which responds to any requests that are not handled by a virtual host defined within a **`<VirtualHost>`** container. Values here also provide defaults for any **`<VirtualHost>`** containers defined.

The directives used in this section have changed little between Apache HTTP Server 1.3 and version 2.0. If the main server configuration is heavily customized, it may be easier to modify the existing configuration file to suit Apache HTTP Server 2.0. Users with only lightly customized main server sections should migrate their changes into the default 2.0 configuration.

25.2.2.2.1. `UserDir` Mapping

The **`UserDir`** directive is used to enable URLs such as **`http://example.com/~bob/`** to map to a subdirectory within the home directory of the user **`bob`**, such as **`/home/bob/public_html/`**. A side-effect of this feature allows a potential attacker to determine whether a given username is present on the system. For this reason, the default configuration for Apache HTTP Server 2.0 disables this directive.

To enable **`UserDir`** mapping, change the directive in **`httpd.conf`** from:

```
UserDir disable
```

to the following:

```
UserDir public_html
```

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

- http://httpd.apache.org/docs-2.0/mod/mod_userdir.html#userdir

25.2.2.2.2. Logging

The following logging directives have been removed:

- **`AgentLog`**
- **`RefererLog`**
- **`RefererIgnore`**

However, agent and referrer logs are still available using the **`CustomLog`** and **`LogFormat`** directives.

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

- http://httpd.apache.org/docs-2.0/mod/mod_log_config.html#customlog
- http://httpd.apache.org/docs-2.0/mod/mod_log_config.html#logformat

25.2.2.2.3. Directory Indexing

The deprecated **`FancyIndexing`** directive has now been removed. The same functionality is available through the **`FancyIndexing`** *option* within the **`IndexOptions`** directive.

The **`VersionSort`** option to the **`IndexOptions`** directive causes files containing version numbers to be sorted in a more natural way. For example, **`httpd-2.0.6.tar`** appears before **`httpd-2.0.36.tar`** in a directory index page.

The defaults for the **`ReadmeName`** and **`HeaderName`** directives have changed from **`README`** and **`HEADER`** to **`README.html`** and **`HEADER.html`**.

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

- http://httpd.apache.org/docs-2.0/mod/mod_autoindex.html#indexoptions
- http://httpd.apache.org/docs-2.0/mod/mod_autoindex.html#readmename
- http://httpd.apache.org/docs-2.0/mod/mod_autoindex.html#headername

25.2.2.2.4. Content Negotiation

The **`CacheNegotiatedDocs`** directive now takes the argument **`on`** or **`off`**. Existing instances of **`CacheNegotiatedDocs`** should be replaced with **`CacheNegotiatedDocs on`**.

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

- http://httpd.apache.org/docs-2.0/mod/mod_negotiation.html#cachenegotiateddocs

25.2.2.2.5. Error Documents

To use a hard-coded message with the **`ErrorDocument`** directive, the message should be enclosed in a pair of double quotation marks **`"`**, rather than just preceded by a double quotation mark as required in Apache HTTP Server 1.3.

For example, the following is a sample Apache HTTP Server 1.3 directive:

```
ErrorDocument 404 "The document was not found
```

To migrate an **ErrorDocument** setting to Apache HTTP Server 2.0, use the following structure:

```
ErrorDocument 404 "The document was not found"
```

Note the trailing double quote in the previous **ErrorDocument** directive example.

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

- http://httpd.apache.org/docs-2.0/mod/core.html#errordocument

25.2.2.3. Virtual Host Configuration

The contents of all **<VirtualHost>** containers should be migrated in the same way as the main server section as described in Section 25.2.2.2, “Main Server Configuration”.

Important

Note that SSL/TLS virtual host configuration has been moved out of the main server configuration file and into **/etc/httpd/conf.d/ssl.conf**.

- http://httpd.apache.org/docs-2.0/vhosts/

25.2.2.4. Modules and Apache HTTP Server 2.0

In Apache HTTP Server 2.0, the module system has been changed to allow modules to be chained together or combined in new and interesting ways. *Common Gateway Interface* (*CGI*) scripts, for example, can generate server-parsed HTML documents which can then be processed by **mod_include**. This opens up a tremendous number of possibilities with regards to how modules can be combined to achieve a specific goal.

The way this works is that each request is served by exactly one *handler* module followed by zero or more *filter* modules.

Under Apache HTTP Server 1.3, for example, a Perl script would be handled in its entirety by the Perl module (**mod_perl**). Under Apache HTTP Server 2.0, the request is initially *handled* by the core module — which serves static files — and is then *filtered* by **mod_perl**.

Exactly how to use this, and all other new features of Apache HTTP Server 2.0, is beyond the scope of this document; however, the change has ramifications if the **PATH_INFO** directive is used for a document which is handled by a module that is now implemented as a filter, as each contains trailing path information after the true file name. The core module, which initially handles the request, does not by default understand **PATH_INFO** and returns **404 Not Found** errors for requests that contain such information. As an alternative, use the **AcceptPathInfo** directive to coerce the core module into accepting requests with **PATH_INFO**.

The following is an example of this directive:

```
AcceptPathInfo on
```

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

- http://httpd.apache.org/docs-2.0/mod/core.html#acceptpathinfo

- http://httpd.apache.org/docs-2.0/handler.html
- http://httpd.apache.org/docs-2.0/filter.html

25.2.2.4.1. The suexec Module

In Apache HTTP Server 2.0, the **`mod_suexec`** module uses the **`SuexecUserGroup`** directive, rather than the **`User`** and **`Group`** directives, which is used for configuring virtual hosts. The **`User`** and **`Group`** directives can still be used in general, but are deprecated for configuring virtual hosts.

For example, the following is a sample Apache HTTP Server 1.3 directive:

```
<VirtualHost vhost.example.com:80>
  User someone
  Group somegroup
</VirtualHost>
```

To migrate this setting to Apache HTTP Server 2.0, use the following structure:

```
<VirtualHost vhost.example.com:80>
  SuexecUserGroup someone somegroup
</VirtualHost>
```

25.2.2.4.2. The `mod_ssl` Module

The configuration for **`mod_ssl`** has been moved from the **`httpd.conf`** file into the **`/etc/httpd/conf.d/ssl.conf`** file. For this file to be loaded, and for **`mod_ssl`** to work, the statement **`Include conf.d/*.conf`** must be in the **`httpd.conf`** file as described in Section 25.2.2.1.3, "Dynamic Shared Object (DSO) Support".

`ServerName` directives in SSL virtual hosts must explicitly specify the port number.

For example, the following is a sample Apache HTTP Server 1.3 directive:

```
<VirtualHost _default_:443>
  # General setup for the virtual host
  ServerName ssl.example.name
  ...
</VirtualHost>
```

To migrate this setting to Apache HTTP Server 2.0, use the following structure:

```
<VirtualHost _default_:443>
  # General setup for the virtual host
  ServerName ssl.host.name:443
  ...
</VirtualHost>
```

It is also important to note that both the **`SSLLog`** and **`SSLLogLevel`** directives have been removed. The **`mod_ssl`** module now obeys the **`ErrorLog`** and **`LogLevel`** directives. Refer to ErrorLog and LogLevel for more information about these directives.

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

- http://httpd.apache.org/docs-2.0/mod/mod_ssl.html

- http://httpd.apache.org/docs-2.0/vhosts/

Important

Due to the vulnerability described in *POODLE: SSLv3 vulnerability (CVE-2014-3566)*, Red Hat recommends disabling **SSL** and using only **TLSv1.1** or **TLSv1.2**. Backwards compatibility can be achieved using **TLSv1.0**. Many products Red Hat supports have the ability to use **SSLv2** or **SSLv3** protocols, or enable them by default. However, the use of **SSLv2** or **SSLv3** is now strongly recommended against.

25.2.2.4.3. The mod_proxy Module

Proxy access control statements are now placed inside a **<Proxy>** block rather than a **<Directory proxy:>**.

The caching functionality of the old **mod_proxy** has been split out into the following three modules:

- **mod_cache**
- **mod_disk_cache**
- **mod_mem_cache**

These generally use directives similar to the older versions of the **mod_proxy** module, but it is advisable to verify each directive before migrating any cache settings.

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

- http://httpd.apache.org/docs-2.0/mod/mod_proxy.html

25.2.2.4.4. The mod_include Module

The **mod_include** module is now implemented as a filter and is therefore enabled differently. Refer to Section 25.2.2.4, "Modules and Apache HTTP Server 2.0" for more about filters.

For example, the following is a sample Apache HTTP Server 1.3 directive:

```
AddType text/html .shtml
AddHandler server-parsed .shtml
```

To migrate this setting to Apache HTTP Server 2.0, use the following structure:

```
AddType text/html .shtml
AddOutputFilter INCLUDES .shtml
```

Note that the **Options +Includes** directive is still required for the **<Directory>** container or in a **.htaccess** file.

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

- http://httpd.apache.org/docs-2.0/mod/mod_include.html

25.2.2.4.5. The mod_auth_dbm and mod_auth_db Modules

Apache HTTP Server 1.3 supported two authentication modules, **mod_auth_db** and **mod_auth_dbm**, which used Berkeley Databases and DBM databases respectively. These modules have been combined into a single module named **mod_auth_dbm** in Apache HTTP Server 2.0, which can access several different database formats. To migrate from **mod_auth_db**, configuration files should be adjusted by replacing **AuthDBUserFile** and **AuthDBGroupFile** with the **mod_auth_dbm** equivalents, **AuthDBMUserFile** and **AuthDBMGroupFile**. Also, the directive **AuthDBMType DB** must be added to indicate the type of database file in use.

The following example shows a sample **mod_auth_db** configuration for Apache HTTP Server 1.3:

```
<Location /private/>
  AuthType Basic
  AuthName "My Private Files"
  AuthDBUserFile /var/www/authdb
  require valid-user
</Location>
```

To migrate this setting to version 2.0 of Apache HTTP Server, use the following structure:

```
<Location /private/>
  AuthType Basic
  AuthName "My Private Files"
  AuthDBMUserFile /var/www/authdb
  AuthDBMType DB
  require valid-user
</Location>
```

Note that the **AuthDBMUserFile** directive can also be used in **.htaccess** files.

The **dbmmanage** Perl script, used to manipulate username and password databases, has been replaced by **htdbm** in Apache HTTP Server 2.0. The **htdbm** program offers equivalent functionality and, like **mod_auth_dbm**, can operate a variety of database formats; the **-T** option can be used on the command line to specify the format to use.

Table 25.1, "Migrating from **dbmmanage** to **htdbm**" shows how to migrate from a DBM-format database to **htdbm** format using **dbmmanage**.

Table 25.1. Migrating from dbmmanage to htdbm

Action	dbmmanage command (1.3)	Equivalent htdbm command (2.0)
Add user to database (using given password)	**dbmmanage authdb add username password**	**htdbm -b -TDB authdb username password**
Add user to database (prompts for password)	**dbmmanage authdb adduser username**	**htdbm -TDB authdb username**
Remove user from database	**dbmmanage authdb delete username**	**htdbm -x -TDB authdb username**
List users in database	**dbmmanage authdb view**	**htdbm -l -TDB authdb**
Verify a password	**dbmmanage authdb check username**	**htdbm -v -TDB authdb username**

The **-m** and **-s** options work with both **dbmmanage** and **htdbm**, enabling the use of the MD5 or SHA1 algorithms for hashing passwords, respectively.

When creating a new database with **htdbm**, the **-c** option must be used.

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

- http://httpd.apache.org/docs-2.0/mod/mod_auth_dbm.html

25.2.2.4.6. The mod_perl Module

The configuration for **mod_perl** has been moved from **httpd.conf** into the file **/etc/httpd/conf.d/perl.conf**. For this file to be loaded, and hence for **mod_perl** to work, the statement **Include conf.d/*.conf** must be included in **httpd.conf** as described in Section 25.2.2.1.3, “Dynamic Shared Object (DSO) Support”.

Occurrences of **Apache::** in **httpd.conf** must be replaced with **ModPerl::**. Additionally, the manner in which handlers are registered has been changed.

This is a sample Apache HTTP Server 1.3 **mod_perl** configuration:

```
<Directory /var/www/perl>
  SetHandler perl-script
  PerlHandler Apache::Registry
  Options +ExecCGI
</Directory>
```

This is the equivalent **mod_perl** for Apache HTTP Server 2.0:

```
<Directory /var/www/perl>
  SetHandler perl-script
  PerlResponseHandler ModPerl::Registry
  Options +ExecCGI
</Directory>
```

Most modules for **mod_perl** 1.x should work without modification with **mod_perl** 2.x. XS modules require recompilation and may require minor Makefile modifications.

25.2.2.4.7. The mod_python Module

Configuration for **mod_python** has moved from **httpd.conf** to the **/etc/httpd/conf.d/python.conf** file. For this file to be loaded, and hence for **mod_python** to work, the statement **Include conf.d/*.conf** must be in **httpd.conf** as described in Section 25.2.2.1.3, “Dynamic Shared Object (DSO) Support”.

25.2.2.4.8. PHP

The configuration for PHP has been moved from **httpd.conf** into the file **/etc/httpd/conf.d/php.conf**. For this file to be loaded, the statement **Include conf.d/*.conf** must be in **httpd.conf** as described in Section 25.2.2.1.3, “Dynamic Shared Object (DSO) Support”.

Note

Any PHP configuration directives used in Apache HTTP Server 1.3 are now fully compatible, when migrating to Apache HTTP Server 2.0 on Red Hat Enterprise Linux 5.

In PHP version 4.2.0 and later the default set of predefined variables which are available in the global scope has changed. Individual input and server variables are, by default, no longer placed directly into the global scope. This change may cause scripts to break. Revert to the old behavior by setting **register_globals** to **On** in the file **/etc/php.ini**.

For more on this topic, refer to the following URL for details concerning the global scope changes:

- http://www.php.net/release_4_1_0.php

25.2.2.4.9. The mod_authz_ldap Module

Red Hat Enterprise Linux ships with the **mod_authz_ldap** module for the Apache HTTP Server. This module uses the short form of the distinguished name for a subject and the issuer of the client SSL certificate to determine the distinguished name of the user within an LDAP directory. It is also capable of authorizing users based on attributes of that user's LDAP directory entry, determining access to assets based on the user and group privileges of the asset, and denying access for users with expired passwords. The **mod_ssl** module is required when using the **mod_authz_ldap** module.

Important

The **mod_authz_ldap** module does not authenticate a user to an LDAP directory using an encrypted password hash. This functionality is provided by the experimental **mod_auth_ldap** module. Refer to the **mod_auth_ldap** module documentation online at http://httpd.apache.org/docs-2.0/mod/mod_auth_ldap.html for details on the status of this module.

The **/etc/httpd/conf.d/authz_ldap.conf** file configures the **mod_authz_ldap** module.

Refer to **/usr/share/doc/mod_authz_ldap-<*version*>/index.html** (replacing <*version*> with the version number of the package) or http://authzldap.othello.ch/ for more information on configuring the **mod_authz_ldap** third party module.

25.3. Starting and Stopping httpd

After installing the **httpd** package, review the Apache HTTP Server's documentation available online at http://httpd.apache.org/docs/2.2/.

The **httpd** RPM installs the **/etc/init.d/httpd** script, which can be accessed using the **/sbin/service** command.

Starting **httpd** using the **apachectl** control script sets the environmental variables in **/etc/sysconfig/httpd** and starts **httpd**. You can also set the environment variables using the init script.

To start the server using the **apachectl** control script as root type:

```
apachectl start
```

You can also start **httpd** using **/sbin/service httpd start**. This starts **httpd** but does not set the environment variables. If you are using the default **Listen** directive in **httpd.conf**, which is port 80, you will need to have root privileges to start the apache server.

To stop the server, as root type:

```
apachectl stop
```

You can also stop **`httpd`** using **`/sbin/service httpd stop`**. The **`restart`** option is a shorthand way of stopping and then starting the Apache HTTP Server.

You can restart the server as root by typing:

```
apachectl restart
```

or:

```
service httpd restart
```

Apache will display a message on the console or in the **`ErrorLog`** if it encounters an error while starting.

By default, the **`httpd`** service does *not* start automatically at boot time. If you would wish to have Apache startup at boot time, you will need to add a call to **`apachectl`** in your startup files within the **`rc.N`** directory. A typical file used is **`rc.local`**. As this starts Apache as root, it is recommended to properly configure your security and authentication before adding this call.

You can also configure the **`httpd`** service to start up at boot time, using an initscript utility, such as **`/sbin/chkconfig`**, **/usr/sbin/ntsysv**, or the **Services Configuration Tool** program.

You can also display the status of your httpd server by typing:

```
apachectl status
```

The status module **`mod_status`** however needs to be enabled in your **`httpd.conf`** configuration file for this to work. For more details on **`mod_status`** can be found on http://httpd.apache.org/docs/2.2/mod/mod_status.html.

Note

If running the Apache HTTP Server as a secure server, the secure server's password is required after the machine boots when using an encrypted private SSL key.

You can find more information on http://httpd.apache.org/docs/2.2/ssl

25.4. Apache HTTP Server Configuration

The **HTTP Configuration Tool** allows you to configure the **`/etc/httpd/conf/httpd.conf`** configuration file for the Apache HTTP Server. It does not use the old **`srm.conf`** or **`access.conf`** configuration files; leave them empty. Through the graphical interface, you can configure directives such as virtual hosts, logging attributes, and maximum number of connections. To start the HTTD Configuration Tool, click on **`System > Administration > Server Settings > HTTP`**.

Only modules provided with Red Hat Enterprise Linux can be configured with the **HTTP Configuration Tool**. If additional modules are installed, they can not be configured using this tool.

Warning

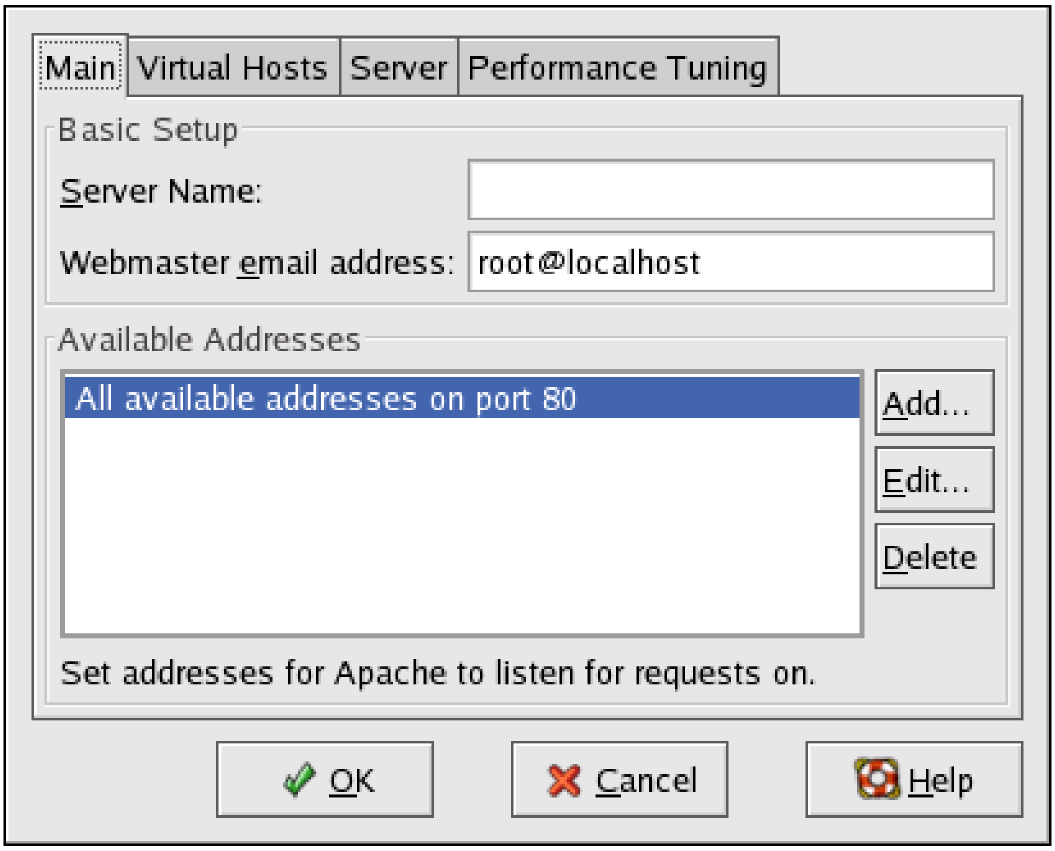
Do not edit the **`/etc/httpd/conf/httpd.conf`** configuration file by hand if you wish to use this tool. The **HTTP Configuration Tool** generates this file after you save your changes and exit the program. If you want to add additional modules or configuration options that are not available in **HTTP Configuration Tool**, you cannot use this tool.

The general steps for configuring the Apache HTTP Server using the **HTTP Configuration Tool** are as follows:

1. Configure the basic settings under the **`Main`** tab.
2. Click on the **`Virtual Hosts`** tab and configure the default settings.
3. Under the **`Virtual Hosts`** tab, configure the Default Virtual Host.
4. To serve more than one URL or virtual host, add any additional virtual hosts.
5. Configure the server settings under the **`Server`** tab.
6. Configure the connections settings under the **`Performance Tuning`** tab.
7. Copy all necessary files to the **`DocumentRoot`** and **`cgi-bin`** directories.
8. Exit the application and select to save your settings.

25.4.1. Basic Settings

Use the **`Main`** tab to configure the basic server settings.



Server Name:

Webmaster email address: root@localhost

All available addresses on port 80

Set addresses for Apache to listen for requests on.

Figure 25.1. Basic Settings

Enter a fully qualified domain name that you have the right to use in the **`Server Name`** text area. This option corresponds to the [`ServerName`](#) directive in **`httpd.conf`**. The **`ServerName`** directive sets the hostname of the Web server. It is used when creating redirection URLs. If you do not define a server name, the Web server attempts to resolve it from the IP address of the system. The server name does not have to be the domain name resolved from the IP address of the server. For example, you might set the server name to www.example.com while the server's real DNS name is foo.example.com.

Enter the email address of the person who maintains the Web server in the **`Webmaster email address`** text area. This option corresponds to the [`ServerAdmin`](#) directive in **`httpd.conf`**. If you configure the server's error pages to contain an email address, this email address is used so that users can report a problem to the server's administrator. The default value is root@localhost.

Use the **`Available Addresses`** area to define the ports on which the server accepts incoming requests. This option corresponds to the [`Listen`](#) directive in **`httpd.conf`**. By default, Red Hat configures the Apache HTTP Server to listen to port 80 for non-secure Web communications.

Click the **`Add`** button to define additional ports on which to accept requests. A window as shown in Figure 25.2, “Available Addresses” appears. Either choose the **`Listen to all addresses`** option to listen to all IP addresses on the defined port or specify a particular IP address over which the server accepts connections in the **`Address`** field. Only specify one IP address per port number. To specify more than one IP address with the same port number, create an entry for each IP address. If at all possible, use an IP address instead of a domain name to prevent a DNS lookup failure. Refer to http://httpd.apache.org/docs/2.2/dns-caveats.html for more information about *Issues Regarding DNS and Apache*.

Entering an asterisk (*) in the **`Address`** field is the same as choosing **`Listen to all addresses`**. Clicking the **`Edit`** button in the **`Available Addresses`** frame shows the same window as the **`Add`** button except with the fields populated for the selected entry. To delete an entry, select it and click the **`Delete`** button.

Note

If you set the server to listen to a port under 1024, you must be root to start it. For port 1024 and above, **`httpd`** can be started as a regular user.

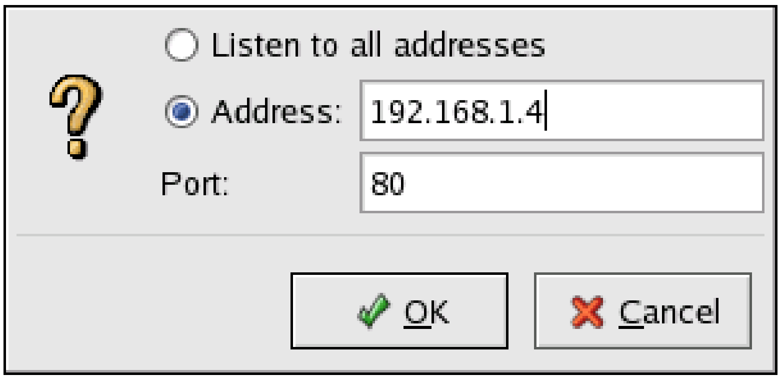
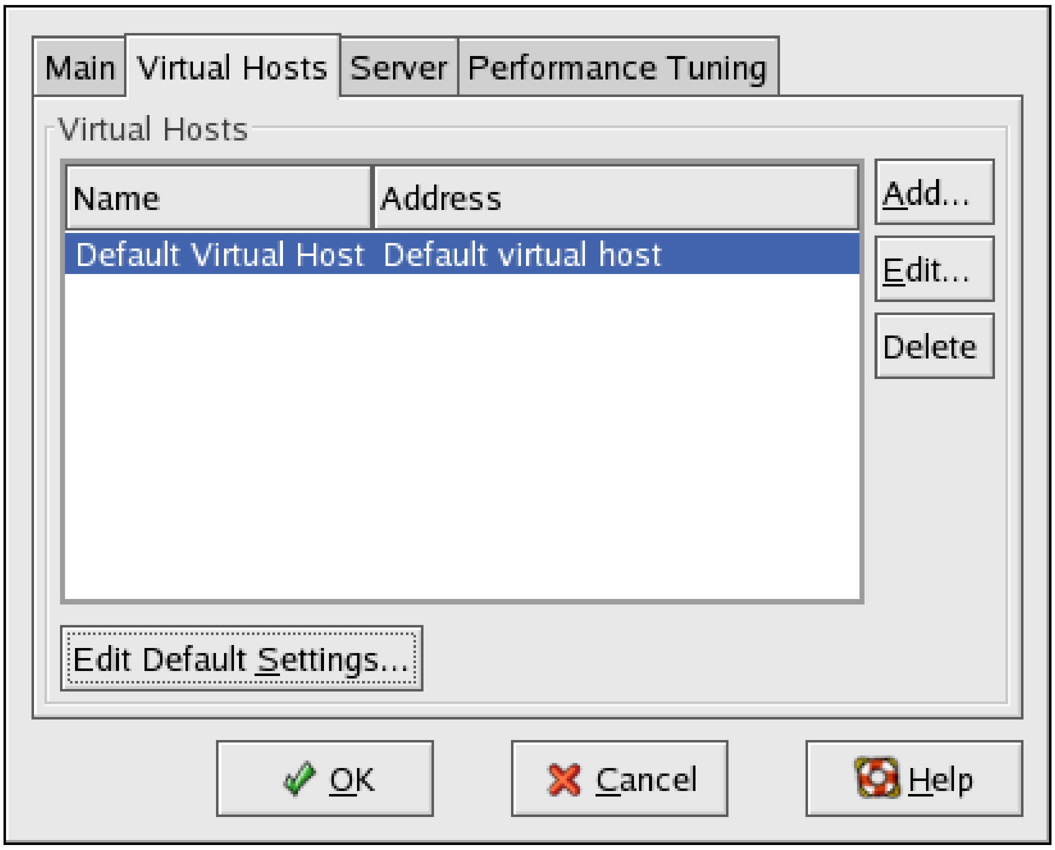

Listen to all addresses
[✓ Address:] 192.168.1.4
Port: 80

Figure 25.2. Available Addresses

25.4.2. Default Settings

After defining the **`Server Name`**, **`Webmaster email address`**, and **`Available Addresses`**, click the **`Virtual Hosts`** tab. The figure below illustrates the **`Virtual Hosts`** tab.



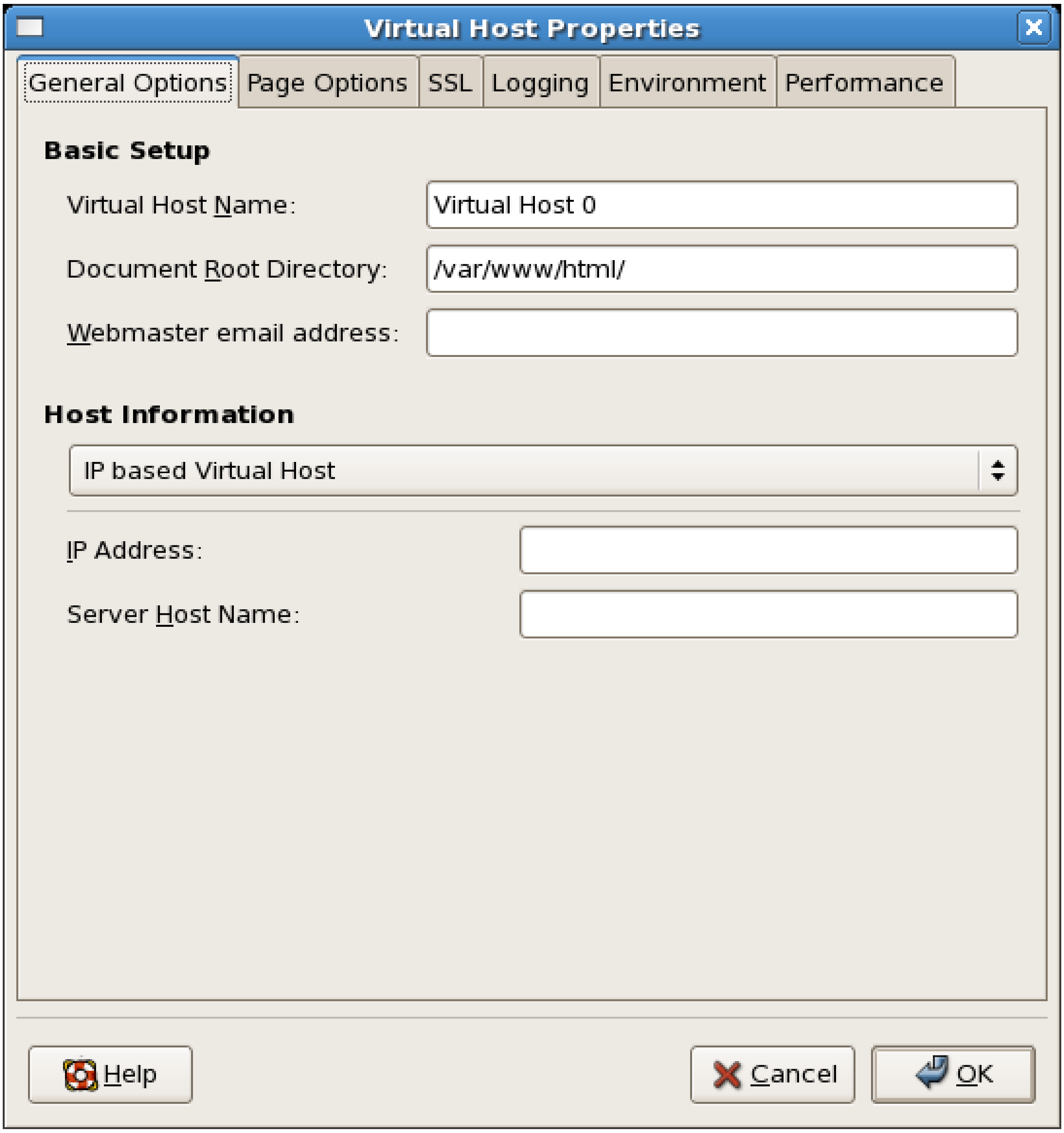
Virtual Hosts

Name	Address
Default Virtual Host	Default virtual host

Figure 25.3. Virtual Hosts Tab

Clicking on **`Edit`** will display the **`Virtual Host Properties`** window from which you can set your preferred settings. To add new settings, click on the **`Add`** button which will also display the **`Virtual Host Properties`** window. Clicking on the **`Edit Default Settings`** button, displays the **`Virtual Host Properties`** window without the **`General Options`** tab.

In the **`General Options`** tab, you can change the hostname, the document root directory and also set the webmaster's email address. In the Host information, you can set the Virtual Host's IP Address and Host Name. The figure below illustrates the **`General Options`** tab.



Basic Setup

Virtual Host Name: Virtual Host 0

Document Root Directory: /var/www/html/

Webmaster email address:

Host Information

IP based Virtual Host

IP Address:

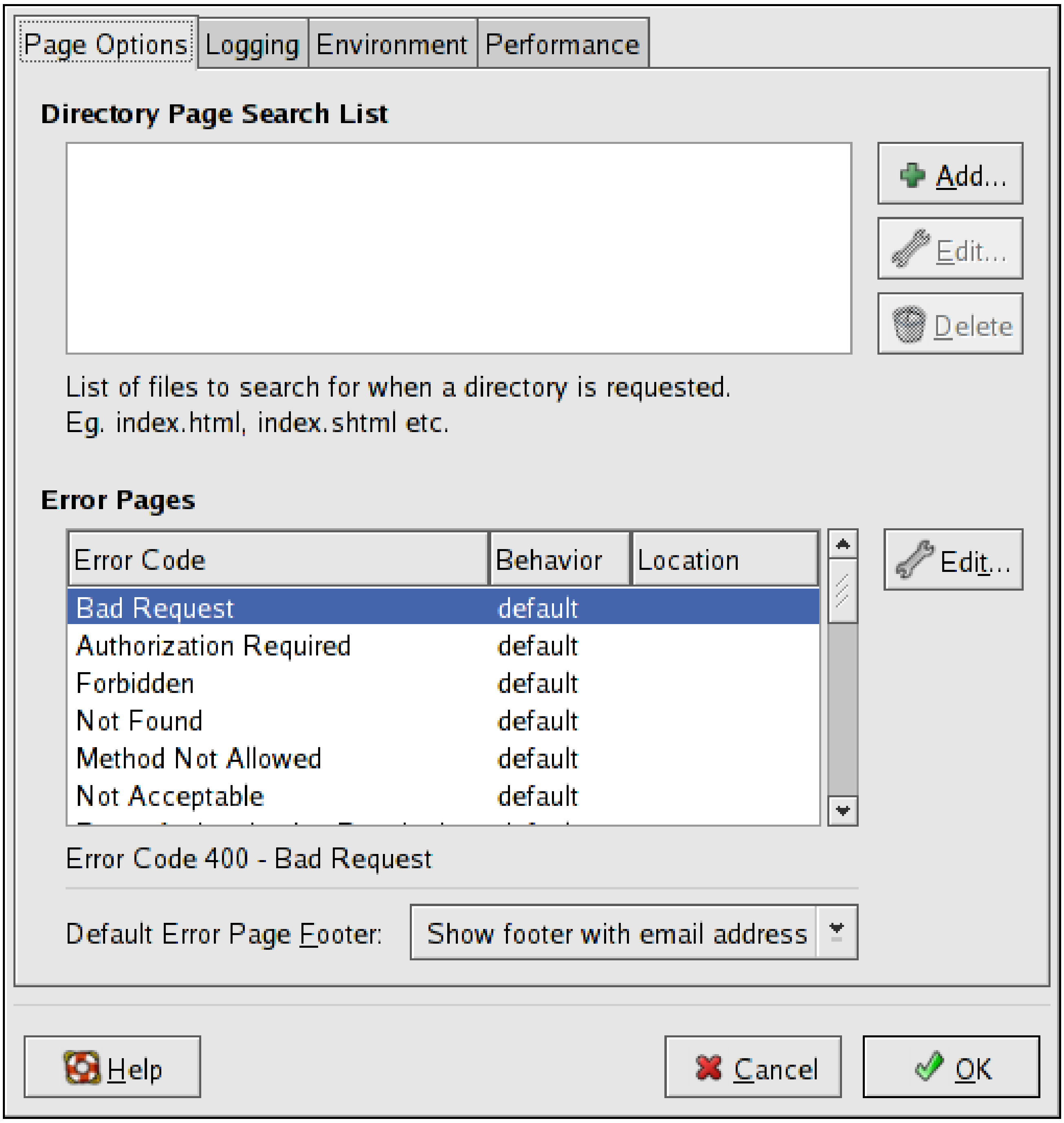
Server Host Name:

Figure 25.4. General Options

If you add a virtual host, the settings you configure for the virtual host take precedence for that virtual host. For a directive not defined within the virtual host settings, the default value is used.

25.4.2.1. Site Configuration

The figure below illustrates the **`Page Options`**tab from which you can configure the **`Directory Page Search List`** and **`Error Pages`**. If you are unsure of these settings, do not modify them.



Directory Page Search List

List of files to search for when a directory is requested.
Eg. index.html, index.shtml etc.

Error Pages

Error Code	Behavior	Location
Bad Request	default	
Authorization Required	default	
Forbidden	default	
Not Found	default	
Method Not Allowed	default	
Not Acceptable	default	

Error Code 400 - Bad Request

Default Error Page Footer: Show footer with email address

Figure 25.5. Site Configuration

The entries listed in the **`Directory Page Search List`** define the `DirectoryIndex` directive. The **`DirectoryIndex`** is the default page served by the server when a user requests an index of a directory by specifying a forward slash (/) at the end of the directory name.

For example, when a user requests the page **`http://www.example.com/this_directory/`**, they are going to get either the **`DirectoryIndex`** page, if it exists, or a server-generated directory list. The server tries to find one of the files listed in the **`DirectoryIndex`** directive and returns the first one it finds. If it does not find any of these files and if **`Options Indexes`** is set for that directory, the server generates and returns a list, in HTML format, of the subdirectories and files in the directory.

Use the **`Error Code`** section to configure Apache HTTP Server to redirect the client to a local or external URL in the event of a problem or error. This option corresponds to the `ErrorDocument` directive. If a problem or error occurs when a client tries to connect to the Apache HTTP Server, the default action is to display the short error message shown in the **`Error Code`** column. To override this default configuration, select the error code and click the **`Edit`** button. Choose **Default** to display the default short error message. Choose **URL** to redirect the client to an external URL and enter a complete URL, including the **`http://`**, in the **`Location`** field. Choose **File** to redirect the client to an internal URL and enter a file location under the document root for the Web server. The location must begin the a slash (/) and be relative to the Document Root.

For example, to redirect a 404 Not Found error code to a webpage that you created in a file called **`404.html`**, copy **`404.html`** to ***`DocumentRoot`*`/../error/404.html`**. In this case, *DocumentRoot* is the Document Root directory that you have defined (the default is **`/var/www/html/`**). If the Document Root is left as the default location, the file should be copied to **`/var/www/error/404.html`**. Then, choose **File** as the Behavior for **`404 - Not Found`** error code and enter **`/error/404.html`** as the **Location**.

From the **`Default Error Page Footer`** menu, you can choose one of the following options:

- **`Show footer with email address`** — Display the default footer at the bottom of all error pages along with the email address of the website maintainer specified by the **`ServerAdmin`** directive.
- **`Show footer`** — Display just the default footer at the bottom of error pages.
- **`No footer`** — Do not display a footer at the bottom of error pages.

25.4.2.2. SSL Support

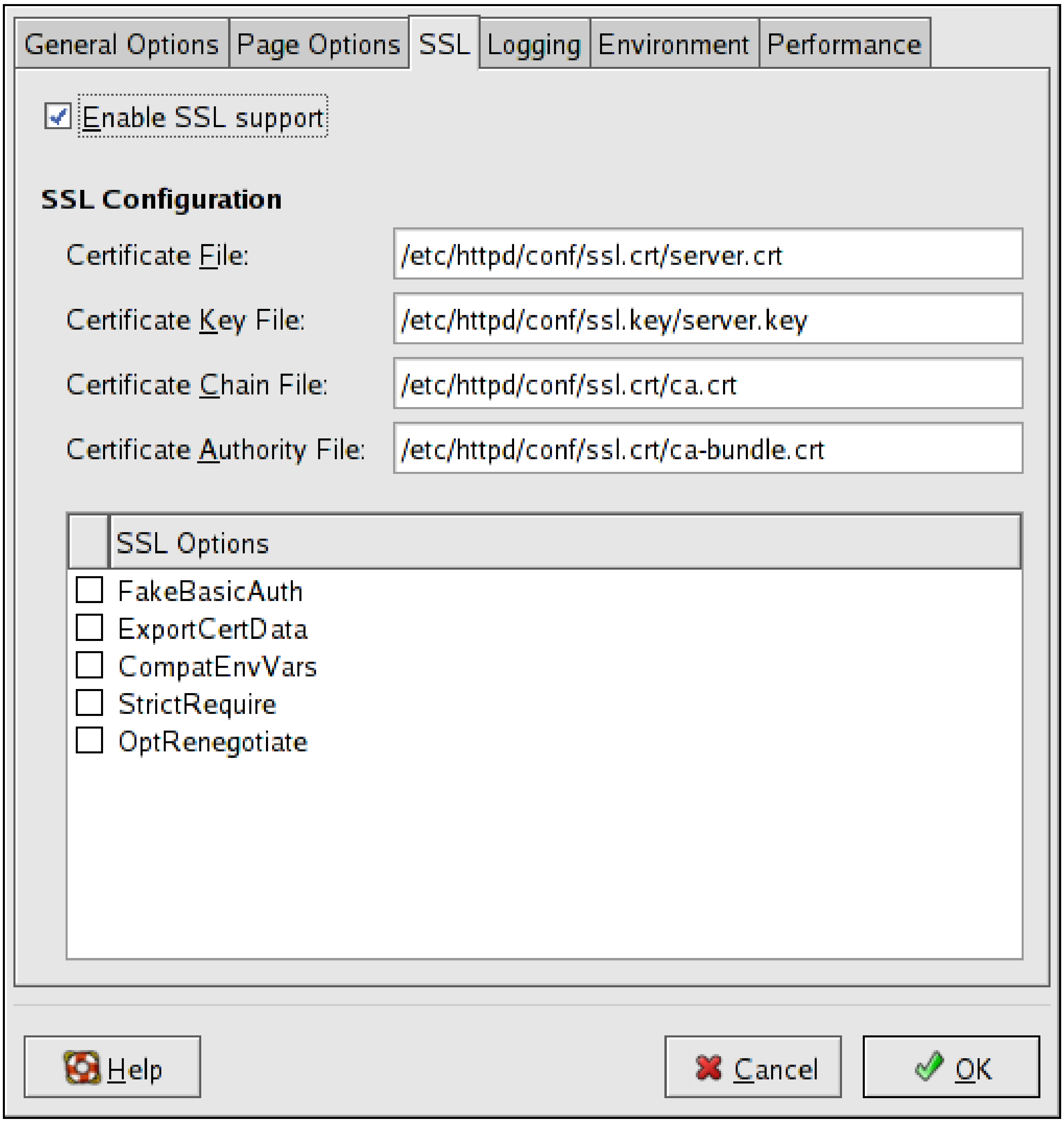
The **`mod_ssl`** enables encryption of the HTTP protocol over SSL. SSL (Secure Sockets Layer) protocol is used for communication and encryption over TCP/IP networks. The SSL tab enables you to configure SSL for your server. To configure SSL you need to provide the path to your:

- Certificate file - equivalent to using the **`SSLCertificateFile`** directive which points the path to the PEM (Privacy Enhanced Mail)-encoded server certificate file.
- Key file - equivalent to using the **`SSLCertificateKeyFile`** directive which points the path to the PEM-encoded server private key file.
- Certificate chain file - equivalent to using the **`SSLCertificateChainFile`** directive which points the path to the certificate file containing all the server's chain of certificates.
- Certificate authority file - is an encrypted file used to confirm the authenticity or identity of parties communicating with the server.

You can find out more about configuration directives for SSL on http://httpd.apache.org/docs/2.2/mod/directives.html#S. You also need to determine which SSL options to enable. These are equivalent to using the **`SSLOptions`** with the following options:

- FakeBasicAuth - enables standard authentication methods used by Apache. This means that the Client X509 certificate's Subject Distinguished Name (DN) is translated into a basic HTTP username.
- ExportCertData - creates CGI environment variables in **`SSL_SERVER_CERT`**, **`SSL_CLIENT_CERT`** and **`SSL_CLIENT_CERT_CHAIN_n`** where n is a number 0,1,2,3,4... These files are used for more certificate checks by CGI scripts.
- CompatEnvVars - enables backward compatibility for Apache SSL by adding CGI environment variables.
- StrictRequire - enables strict access which forces denial of access whenever the **`SSLRequireSSL`** and **`SSLRequire`** directives indicate access is forbidden.
- OptRenegotiate - enables avoidance of unnecessary handshakes by **`mod_ssl`** which also performs safe parameter checks. It is recommended to enable OptRenegotiate on a per directory basis.

More information on the above SSL options can be found on http://httpd.apache.org/docs/2.2/mod/mod_ssl.html#ssloptions. The figure below illustrates the SSL tab and the options discussed above.



☑ Enable SSL support

SSL Configuration

Certificate File:	/etc/httpd/conf/ssl.crt/server.crt
Certificate Key File:	/etc/httpd/conf/ssl.key/server.key
Certificate Chain File:	/etc/httpd/conf/ssl.crt/ca.crt
Certificate Authority File:	/etc/httpd/conf/ssl.crt/ca-bundle.crt

	SSL Options
☐	FakeBasicAuth
☐	ExportCertData
☐	CompatEnvVars
☐	StrictRequire
☐	OptRenegotiate

Figure 25.6. SSL

Important

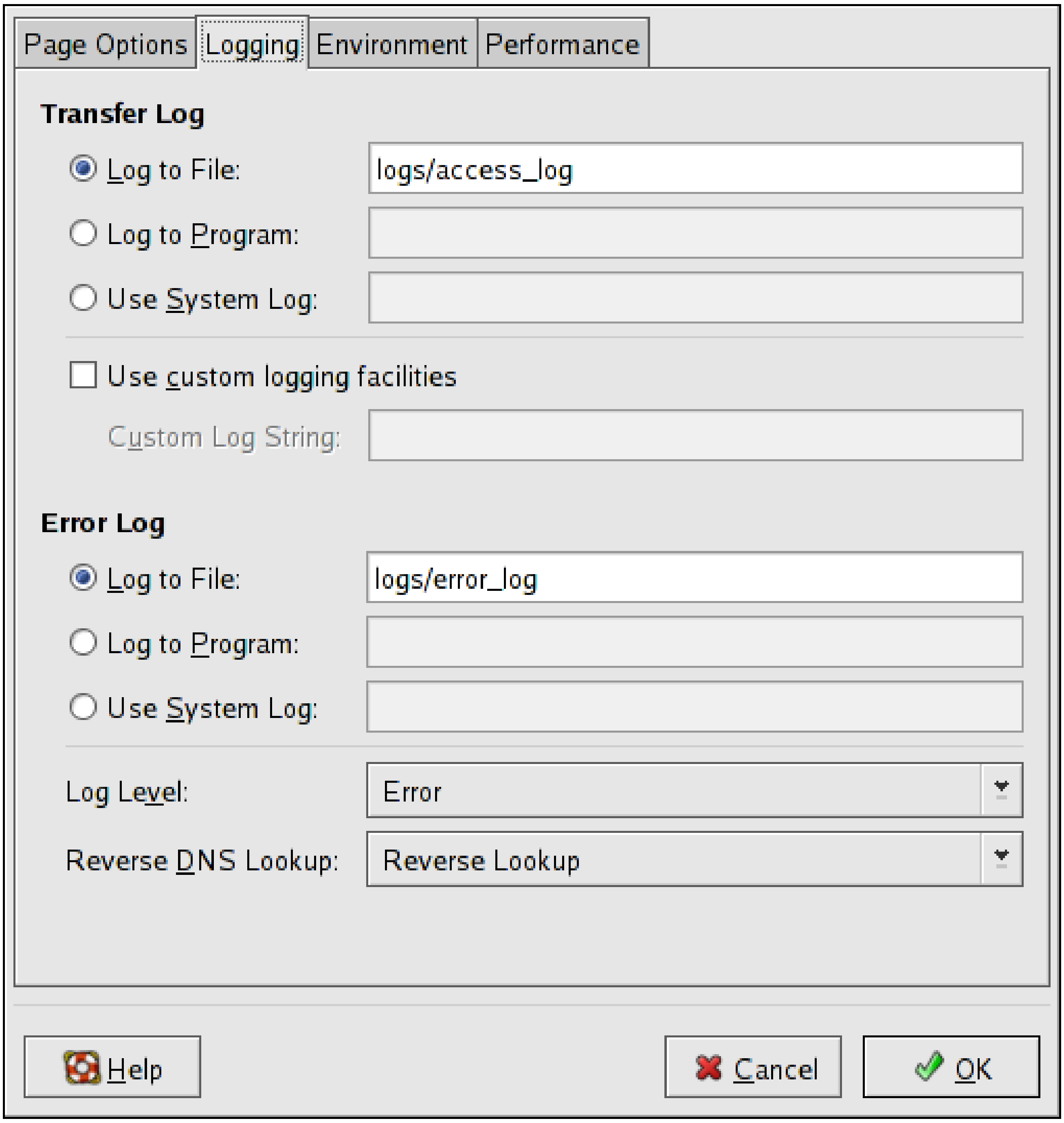
Due to the vulnerability described in *POODLE: SSLv3 vulnerability (CVE-2014-3566)*, Red Hat recommends disabling **SSL** and using only **`TLSv1.1`** or **`TLSv1.2`**. Backwards compatibility can be achieved using **`TLSv1.0`**. Many products Red Hat supports have the ability to use **SSLv2** or **SSLv3** protocols, or enable them by default. However, the use of **SSLv2** or **SSLv3** is now strongly recommended against.

25.4.2.3. Logging

Use the **`Logging`** tab to configure options for specific transfer and error logs.

By default, the server writes the transfer log to the **`/var/log/httpd/access_log`** file and the error log to the **`/var/log/httpd/error_log`** file.

The transfer log contains a list of all attempts to access the Web server. It records the IP address of the client that is attempting to connect, the date and time of the attempt, and the file on the Web server that it is trying to retrieve. Enter the name of the path and file in which to store this information. If the path and file name do not start with a slash (/), the path is relative to the server root directory as configured. This option corresponds to the `TransferLog` directive.



Transfer Log

Log to File: [✓] logs/access_log

Log to Program:

Use System Log:

Use custom logging facilities [ ]

Custom Log String:

Error Log

Log to File: [✓] logs/error_log

Log to Program:

Use System Log:

Log Level: Error

Reverse DNS Lookup: Reverse Lookup

Figure 25.7. Logging

You can configure a custom log format by checking **`Use custom logging facilities`** and entering a custom log string in the **`Custom Log String`** field. This configures the **`LogFormat`** directive. Refer to http://httpd.apache.org/docs/2.2/mod/mod_log_config.html#logformat for details on the format of this directive.

The error log contains a list of any server errors that occur. Enter the name of the path and file in which to store this information. If the path and file name do not start with a slash (/), the path is relative to the server root directory as configured. This option corresponds to the **`ErrorLog`** directive.

Use the **`Log Level`** menu to set the verbosity of the error messages in the error logs. It can be set (from least verbose to most verbose) to emerg, alert, crit, error, warn, notice, info or debug. This option corresponds to the **`LogLevel`** directive.

The value chosen with the **`Reverse DNS Lookup`** menu defines the **`HostnameLookups`** directive. Choosing **`No Reverse Lookup`** sets the value to off. Choosing **`Reverse Lookup`** sets the value to on. Choosing **`Double Reverse Lookup`** sets the value to double.

If you choose **`Reverse Lookup`**, your server automatically resolves the IP address for each connection which requests a document from your Web server. Resolving the IP address means that your server makes one or more connections to the DNS in order to find out the hostname that corresponds to a particular IP address.

If you choose **`Double Reverse Lookup`**, your server performs a double-reverse DNS. In other words, after a reverse lookup is performed, a forward lookup is performed on the result. At least one of the IP addresses in the forward lookup must match the address from the first reverse lookup.

Generally, you should leave this option set to **`No Reverse Lookup`**, because the DNS requests add a load to your server and may slow it down. If your server is busy, the effects of trying to perform these reverse lookups or double reverse lookups may be quite noticeable.

Reverse lookups and double reverse lookups are also an issue for the Internet as a whole. Each individual connection made to look up each hostname adds up. Therefore, for your own Web server's benefit, as well as for the Internet's benefit, you should leave this option set to **`No Reverse Lookup`**.

25.4.2.4. Environment Variables

Use the **`Environment`** tab to configure options for specific variables to set, pass, or unset for CGI scripts.

Sometimes it is necessary to modify environment variables for CGI scripts or server-side include (SSI) pages. The Apache HTTP Server can use the **`mod_env`** module to configure the environment variables which are passed to CGI scripts and SSI pages. Use the **`Environment Variables`** page to configure the directives for this module.

Use the **`Set for CGI Scripts`** section to set an environment variable that is passed to CGI scripts and SSI pages. For example, to set the environment variable **`MAXNUM`** to **`50`**, click the **`Add`** button inside the **`Set for CGI Script`** section, as shown in Figure 25.8, “Environment Variables”, and type **`MAXNUM`** in the **`Environment Variable`** text field and **`50`** in the **`Value to set`** text field. Click **`OK`** to add it to the list. The **`Set for CGI Scripts`** section configures the **`SetEnv`** directive.

Use the **`Pass to CGI Scripts`** section to pass the value of an environment variable when the server is first started to CGI scripts. To see this environment variable, type the command **`env`** at a shell prompt. Click the **`Add`** button inside the **`Pass to CGI Scripts`** section and enter the name of the environment variable in the resulting dialog box. Click **`OK`** to add it to the list. The **`Pass to CGI Scripts`** section configures the **`PassEnv`** directive.

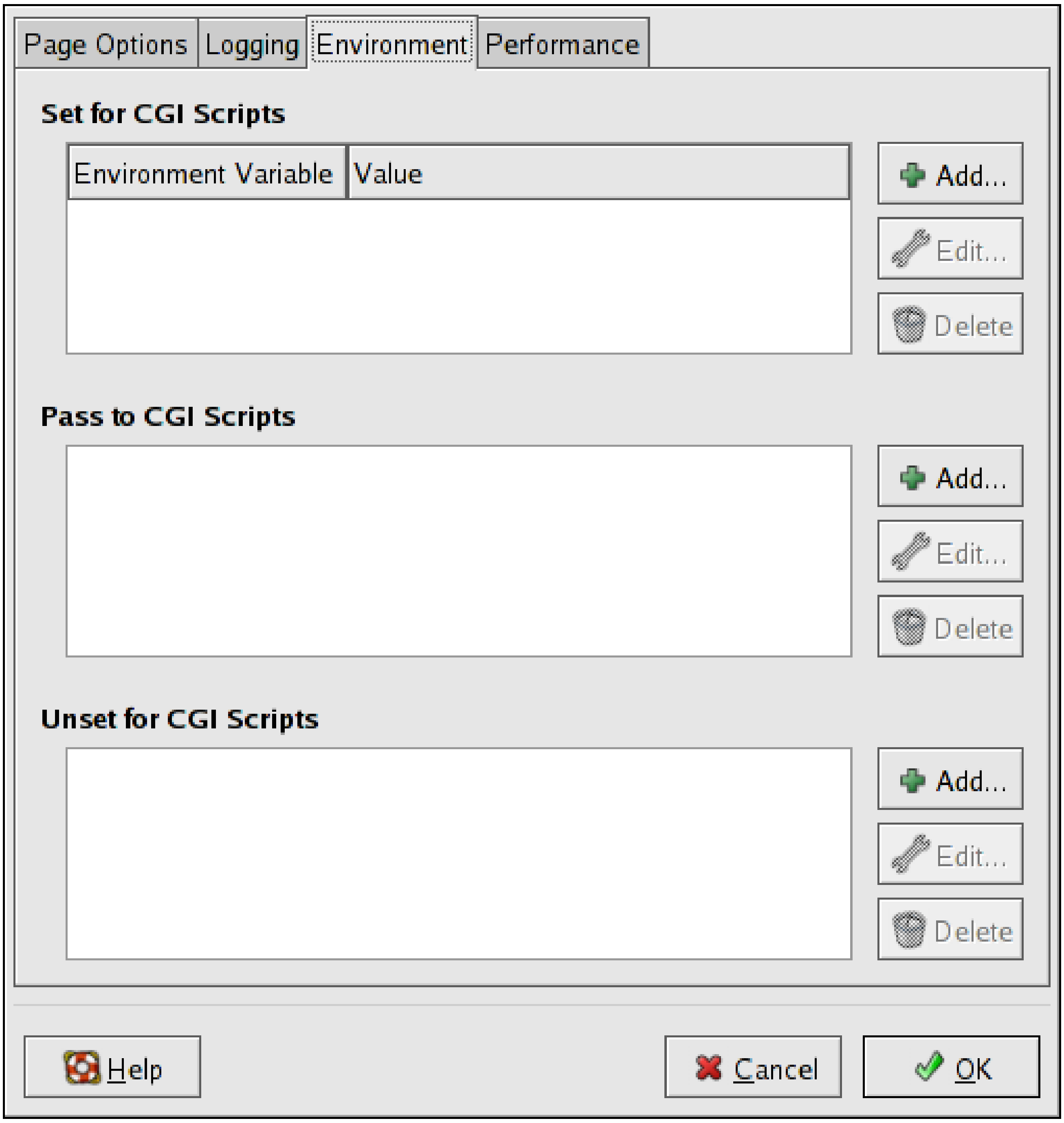


Figure 25.8. Environment Variables

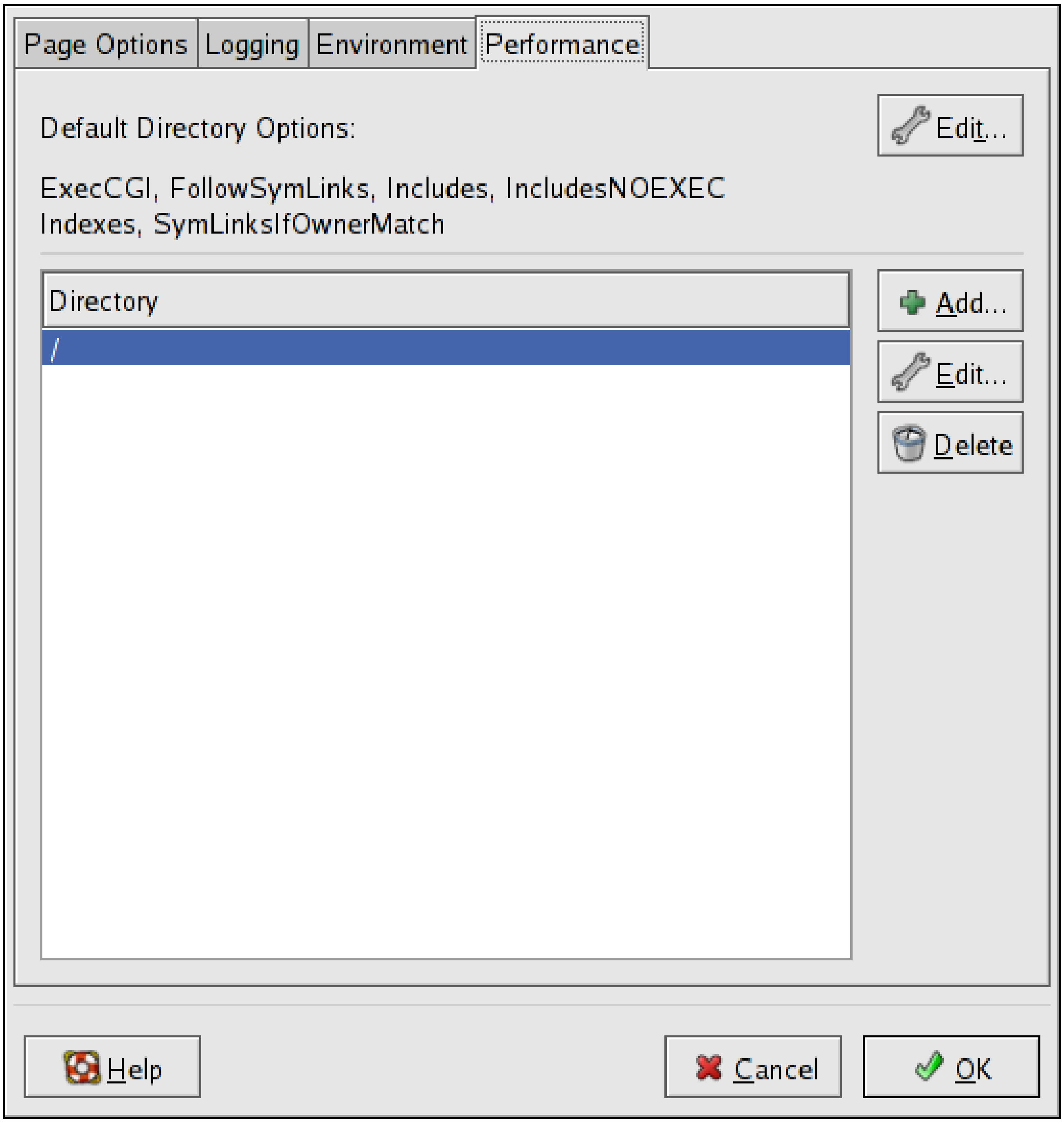
To remove an environment variable so that the value is not passed to CGI scripts and SSI pages, use the **`Unset for CGI Scripts`** section. Click **Add** in the **`Unset for CGI Scripts`** section, and enter the name of the environment variable to unset. Click **OK** to add it to the list. This corresponds to the[`UnsetEnv`] directive.

To edit any of these environment values, select it from the list and click the corresponding **`Edit`** button. To delete any entry from the list, select it and click the corresponding **`Delete`** button.

To learn more about environment variables in the Apache HTTP Server, refer to the following: http://httpd.apache.org/docs/2.2/env.html

25.4.2.5. Directories

Use the **Directories** page in the **Performance** tab to configure options for specific directories. This corresponds to the `<Directory>` directive.



Default Directory Options:

ExecCGI, FollowSymLinks, Includes, IncludesNOEXEC
Indexes, SymLinksIfOwnerMatch

Directory
/

Figure 25.9. Directories

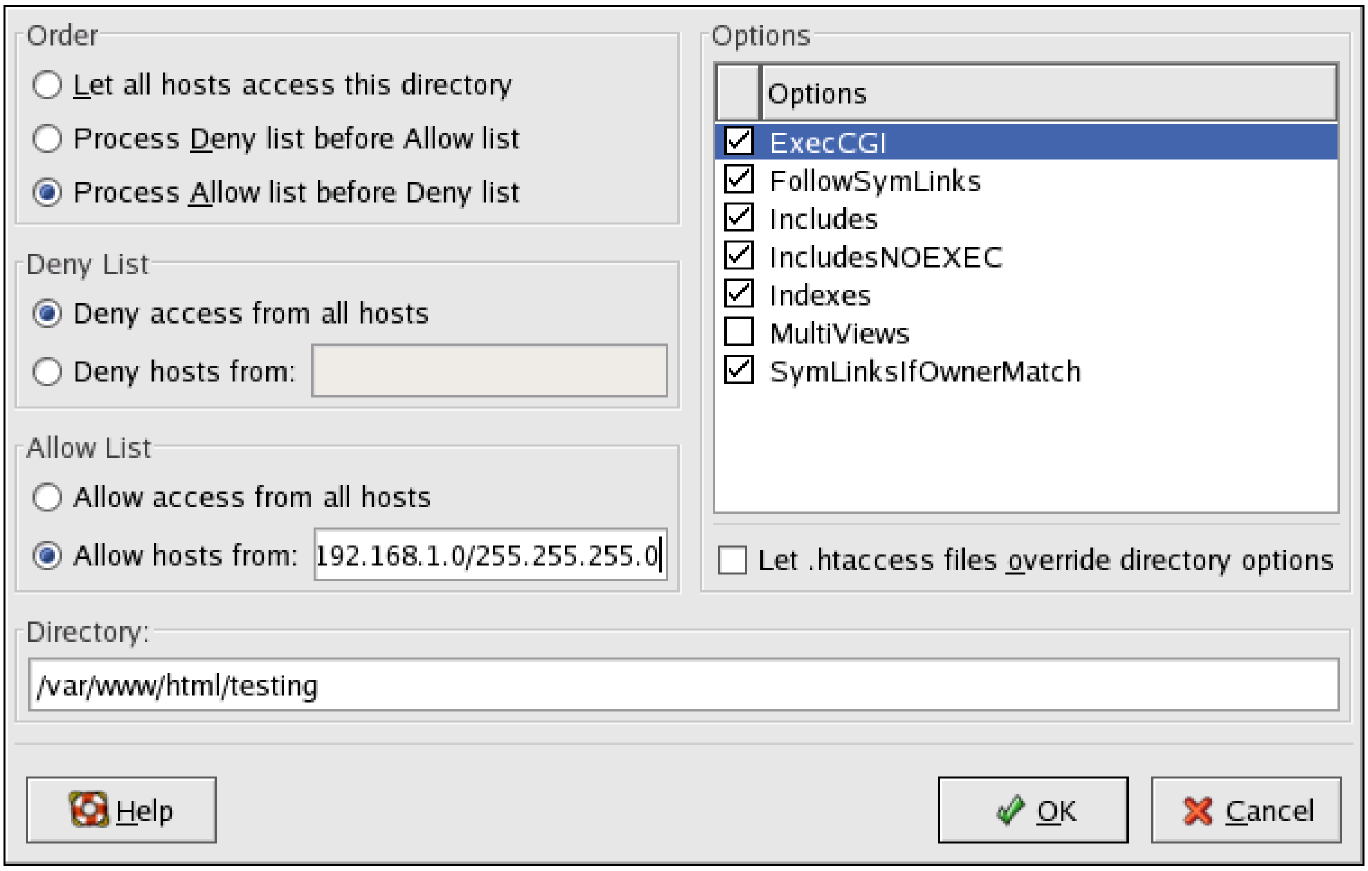
Click the **Edit** button in the top right-hand corner to configure the **Default Directory Options** for all directories that are not specified in the **Directory** list below it. The options that you choose are listed as the `Options` directive within the `<Directory>` directive. You can configure the following options:

- **ExecCGI** — Allow execution of CGI scripts. CGI scripts are not executed if this option is not chosen.
- **FollowSymLinks** — Allow symbolic links to be followed.
- **Includes** — Allow server-side includes.
- **IncludesNOEXEC** — Allow server-side includes, but disable the **#exec** and **#include** commands in CGI scripts.

- **`Indexes`** — Display a formatted list of the directory's contents, if no **`DirectoryIndex`** (such as **`index.html`**) exists in the requested directory.
- **`Multiview`** — Support content-negotiated multiviews; this option is disabled by default.
- **`SymLinksIfOwnerMatch`** — Only follow symbolic links if the target file or directory has the same owner as the link.

To specify options for specific directories, click the **Add** button beside the **`Directory`** list box. A window as shown in Figure 25.10, "Directory Settings" appears. Enter the directory to configure in the **`Directory`** text field at the bottom of the window. Select the options in the right-hand list and configure the **`Order`** directive with the left-hand side options. The **`Order`** directive controls the order in which allow and deny directives are evaluated. In the **`Allow hosts from`** and **`Deny hosts from`** text field, you can specify one of the following:

- Allow all hosts — Type **`all`** to allow access to all hosts.
- Partial domain name — Allow all hosts whose names match or end with the specified string.
- Full IP address — Allow access to a specific IP address.
- A subnet — Such as **`192.168.1.0/255.255.255.0`**
- A network CIDR specification — such as **`10.3.0.0/16`**


Order
- ○ Let all hosts access this directory
- ○ Process Deny list before Allow list
- [✓ Process Allow list before Deny list]

Deny List
- [✓ Deny access from all hosts]
- ○ Deny hosts from:

Allow List
- ○ Allow access from all hosts
- [✓ Allow hosts from:] 192.168.1.0/255.255.255.0

Options

	Options
☑	ExecCGI
☑	FollowSymLinks
☑	Includes
☑	IncludesNOEXEC
☑	Indexes
☐	MultiViews
☑	SymLinksIfOwnerMatch

☐ Let .htaccess files override directory options

Directory: /var/www/html/testing

Figure 25.10. Directory Settings

If you check the **`Let .htaccess files override directory options`**, the configuration directives in the **`.htaccess`** file take precedence.

25.5. Configuration Directives in `httpd.conf`

The Apache HTTP Server configuration file is **`/etc/httpd/conf/httpd.conf`**. The **`httpd.conf`** file is well-commented and mostly self-explanatory. The default configuration works for most situations; however, it is a good idea to become familiar some of the more important configuration options.

Warning

With the release of Apache HTTP Server 2.2, many configuration options have changed. If migrating from version 1.3 to 2.2, please firstly read Section 25.2.2, “Migrating Apache HTTP Server 1.3 Configuration Files to 2.0”.

25.5.1. General Configuration Tips

If configuring the Apache HTTP Server, edit **`/etc/httpd/conf/httpd.conf`** and then either reload, restart, or stop and start the **`httpd`** process as outlined in Section 25.3, “Starting and Stopping **`httpd`**”.

Before editing **`httpd.conf`**, make a copy the original file. Creating a backup makes it easier to recover from mistakes made while editing the configuration file.

If a mistake is made and the Web server does not work correctly, first review recently edited passages in **`httpd.conf`** to verify there are no typos.

Next look in the Web server's error log, **`/var/log/httpd/error_log`**. The error log may not be easy to interpret, depending on your level of expertise. However, the last entries in the error log should provide useful information.

The following subsections contain a list of short descriptions for many of the directives included in **`httpd.conf`**. These descriptions are not exhaustive. For more information, refer to the Apache documentation online at http://httpd.apache.org/docs/2.2/.

For more information about **`mod_ssl`** directives, refer to the documentation online at http://httpd.apache.org/docs/2.2/mod/mod_ssl.html.

AccessFileName

`AccessFileName` names the file which the server should use for access control information in each directory. The default is **`.htaccess`**.

Immediately after the **`AccessFileName`** directive, a set of **`Files`** tags apply access control to any file beginning with a **`.ht`**. These directives deny Web access to any **`.htaccess`** files (or other files which begin with **`.ht`**) for security reasons.

Action

`Action` specifies a MIME content type and CGI script pair, so that when a file of that media type is requested, a particular CGI script is executed.

AddDescription

When using **`FancyIndexing`** as an **`IndexOptions`** parameter, the **`AddDescription`** directive can be used to display user-specified descriptions for certain files or file types in a server generated directory listing. The **`AddDescription`** directive supports listing specific files, wildcard expressions, or file extensions.

AddEncoding

AddEncoding names file name extensions which should specify a particular encoding type. **AddEncoding** can also be used to instruct some browsers to uncompress certain files as they are downloaded.

AddHandler

AddHandler maps file extensions to specific handlers. For example, the **cgi-script** handler can be matched with the extension **.cgi** to automatically treat a file ending with **.cgi** as a CGI script. The following is a sample **AddHandler** directive for the **.cgi** extension.

```
AddHandler cgi-script .cgi
```

This directive enables CGIs outside of the **cgi-bin** to function in any directory on the server which has the **ExecCGI** option within the directories container. Refer to Directory for more information about setting the **ExecCGI** option for a directory.

In addition to CGI scripts, the **AddHandler** directive is used to process server-parsed HTML and image-map files.

AddIcon

AddIcon specifies which icon to show in server generated directory listings for files with certain extensions. For example, the Web server is set to show the icon **binary.gif** for files with **.bin** or **.exe** extensions.

AddIconByEncoding

This directive names icons which are displayed by files with MIME encoding in server generated directory listings. For example, by default, the Web server shows the **compressed.gif** icon next to MIME encoded x-compress and x-gzip files in server generated directory listings.

AddIconByType

This directive names icons which are displayed next to files with MIME types in server generated directory listings. For example, the server shows the icon **text.gif** next to files with a mime-type of **text**, in server generated directory listings.

AddLanguage

AddLanguage associates file name extensions with specific languages. This directive is useful for Apache HTTP Servers which serve content in multiple languages based on the client Web browser's language settings.

AddType

Use the **AddType** directive to define or override a default MIME type and file extension pairs. The following example directive tells the Apache HTTP Server to recognize the **.tgz** file extension:

```
AddType application/x-tar .tgz
```

Alias

The **Alias** setting allows directories outside the **DocumentRoot** directory to be accessible. Any URL ending in the alias automatically resolves to the alias' path. By default, one alias for an **icons/** directory is already set up. An **icons/** directory can be accessed by the Web server, but the directory is not in the **DocumentRoot**.

Allow

`Allow` specifies which client can access a given directory. The client can be **`all`**, a domain name, an IP address, a partial IP address, a network/netmask pair, and so on. The **`DocumentRoot`** directory is configured to **`Allow`** requests from **`all`**, meaning everyone has access.

AllowOverride

The **`AllowOverride`** directive sets whether any **`Options`** can be overridden by the declarations in an **`.htaccess`** file. By default, both the root directory and the **`DocumentRoot`** are set to allow no **`.htaccess`** overrides.

BrowserMatch

The **`BrowserMatch`** directive allows the server to define environment variables and take appropriate actions based on the User-Agent HTTP header field — which identifies the client's Web browser type. By default, the Web server uses **`BrowserMatch`** to deny connections to specific browsers with known problems and also to disable keepalives and HTTP header flushes for browsers that are known to have problems with those actions.

Cache Directives

A number of commented cache directives are supplied by the default Apache HTTP Server configuration file. In most cases, uncommenting these lines by removing the hash mark (**`#`**) from the beginning of the line is sufficient. The following, however, is a list of some of the more important cache-related directives.

- **`CacheEnable`** — Specifies whether the cache is a disk, memory, or file descriptor cache. By default **`CacheEnable`** configures a disk cache for URLs at or below **`/`**.
- **`CacheRoot`** — Specifies the name of the directory containing cached files. The default **`CacheRoot`** is the **`/var/httpd/proxy/`** directory.
- **`CacheSize`** — Specifies how much space the cache can use in kilobytes. The default **`CacheSize`** is **`5`** KB.

The following is a list of some of the other common cache-related directives.

- **`CacheMaxExpire`** — Specifies how long HTML documents are retained (without a reload from the originating Web server) in the cache. The default is **`24`** hours (**`86400`** seconds).
- **`CacheLastModifiedFactor`** — Specifies the creation of an expiry (expiration) date for a document which did not come from its originating server with its own expiry set. The default **`CacheLastModifiedFactor`** is set to **`0.1`**, meaning that the expiry date for such documents equals one-tenth of the amount of time since the document was last modified.
- **`CacheDefaultExpire`** — Specifies the expiry time in hours for a document that was received using a protocol that does not support expiry times. The default is set to **`1`** hour (**`3600`** seconds).
- **`NoProxy`** — Specifies a space-separated list of subnets, IP addresses, domains, or hosts whose content is not cached. This setting is most useful for Intranet sites.

CacheNegotiatedDocs

By default, the Web server asks proxy servers not to cache any documents which were negotiated on the basis of content (that is, they may change over time or because of the input from the requester). If **`CacheNegotiatedDocs`** is set to **`on`**, this function is disabled and proxy servers are allowed to cache such documents.

CustomLog

CustomLog identifies the log file and the log file format. By default, the access log is recorded to the **/var/log/httpd/access_log** file while errors are recorded in the **/var/log/httpd/error_log** file.

The default **CustomLog** format is the **combined** log file format, as illustrated here:

```
remotehost rfc931 user date "request" status bytes referrer user-agent
```

DefaultIcon

DefaultIcon specifies the icon displayed in server generated directory listings for files which have no other icon specified. The **unknown.gif** image file is the default.

DefaultType

DefaultType sets a default content type for the Web server to use for documents whose MIME types cannot be determined. The default is **text/plain**.

Deny

Deny works similar to **Allow**, except it specifies who is denied access. The **DocumentRoot** is not configured to **Deny** requests from anyone by default.

Directory

<Directory /path/to/directory> and **</Directory>** tags create a container used to enclose a group of configuration directives which apply only to a specific directory and its subdirectories. Any directive which is applicable to a directory may be used within **Directory** tags.

By default, very restrictive parameters are applied to the root directory (**/**), using the **Options** (refer to Options) and **AllowOverride** (refer to AllowOverride) directives. Under this configuration, any directory on the system which needs more permissive settings has to be explicitly given those settings.

In the default configuration, another **Directory** container is configured for the **DocumentRoot** which assigns less rigid parameters to the directory tree so that the Apache HTTP Server can access the files residing there.

The **Directory** container can be also be used to configure additional **cgi-bin** directories for server-side applications outside of the directory specified in the **ScriptAlias** directive (refer to ScriptAlias for more information).

To accomplish this, the **Directory** container must set the **ExecCGI** option for that directory.

For example, if CGI scripts are located in **/home/my_cgi_directory**, add the following **Directory** container to the **httpd.conf** file:

```
<Directory /home/my_cgi_directory>
  Options +ExecCGI
</Directory>
```

Next, the **AddHandler** directive must be uncommented to identify files with the **.cgi** extension as CGI scripts. Refer to AddHandler for instructions on setting **AddHandler**.

For this to work, permissions for CGI scripts, and the entire path to the scripts, must be set to 0755.

DirectoryIndex

The **`DirectoryIndex`** is the default page served by the server when a user requests an index of a directory by specifying a forward slash (/) at the end of the directory name.

When a user requests the page http://*example*/*this_directory*/, they get either the **`DirectoryIndex`** page, if it exists, or a server-generated directory list. The default for **`DirectoryIndex`** is **`index.html`** and the **`index.html.var`** type map. The server tries to find either of these files and returns the first one it finds. If it does not find one of these files and **`Options Indexes`** is set for that directory, the server generates and returns a listing, in HTML format, of the subdirectories and files within the directory, unless the directory listing feature is turned off.

DocumentRoot

`DocumentRoot` is the directory which contains most of the HTML files which are served in response to requests. The default **`DocumentRoot`**, for both the non-secure and secure Web servers, is the **`/var/www/html`** directory. For example, the server might receive a request for the following document:

```
http://example.com/foo.html
```

The server looks for the following file in the default directory:

```
/var/www/html/foo.html
```

To change the **`DocumentRoot`** so that it is not shared by the secure and the non-secure Web servers, refer to Section 25.7, “Virtual Hosts”.

ErrorDocument

The **`ErrorDocument`** directive associates an HTTP response code with a message or a URL to be sent back to the client. By default, the Web server outputs a simple and usually cryptic error message when an error occurs. The **`ErrorDocument`** directive forces the Web server to instead output a customized message or page.

Important

To be valid, the message *must* be enclosed in a pair of double quotes **`"`**.

ErrorLog

`ErrorLog` specifies the file where server errors are logged. By default, this directive is set to **`/var/log/httpd/error_log`**.

ExtendedStatus

The **`ExtendedStatus`** directive controls whether Apache generates basic (**`off`**) or detailed server status information (**`on`**), when the **`server-status`** handler is called. The **`server-status`** handler is called using **`Location`** tags. More information on calling **`server-status`** is included in Location.

Group

Specifies the group name of the Apache HTTP Server processes.

This directive has been deprecated for the configuration of virtual hosts.

By default, **`Group`** is set to **`apache`**.

HeaderName

`HeaderName` names the file which, if it exists in the directory, is prepended to the start of server generated directory listings. Like **`ReadmeName`**, the server tries to include it as an HTML document if possible or in plain text if not.

HostnameLookups

`HostnameLookups` can be set to **`on`**, **`off`**, or **`double`**. If **`HostnameLookups`** is set to **`on`**, the server automatically resolves the IP address for each connection. Resolving the IP address means that the server makes one or more connections to a DNS server, adding processing overhead. If **`HostnameLookups`** is set to **`double`**, the server performs a double-reverse DNS look up adding even more processing overhead.

To conserve resources on the server, **`HostnameLookups`** is set to **`off`** by default.

If hostnames are required in server log files, consider running one of the many log analyzer tools that perform the DNS lookups more efficiently and in bulk when rotating the Web server log files.

IfDefine

The **`IfDefine`** tags surround configuration directives that are applied if the "test" stated in the**`IfDefine`** tag is true. The directives are ignored if the test is false.

The test in the **`IfDefine`** tags is a parameter name (for example, **`HAVE_PERL`**). If the parameter is defined, meaning that it is provided as an argument to the server's start-up command, then the test is true. In this case, when the Web server is started, the test is true and the directives contained in the **`IfDefine`** tags are applied.

IfModule

`<IfModule>` and **`</IfModule>`** tags create a conditional container which are only activated if the specified module is loaded. Directives within the **`IfModule`** container are processed under one of two conditions. The directives are processed if the module contained within the starting **`<IfModule>`** tag is loaded. Or, if an exclamation point **`!`** appears before the module name, the directives are processed only if the module specified in the **`<IfModule>`** tag is *not* loaded.

For more information about Apache HTTP Server modules, refer to Section 25.6, “Adding Modules”.

Include

`Include` allows other configuration files to be included at runtime.

The path to these configuration files can be absolute or relative to the **`ServerRoot`**.

Important

For the server to use individually packaged modules, such as **mod_ssl**, **mod_perl**, and **php**, the following directive must be included in **Section 1: Global Environment** of **httpd.conf**:

```
Include conf.d/*.conf
```

IndexIgnore

IndexIgnore lists file extensions, partial file names, wildcard expressions, or full file names. The Web server does not include any files which match any of those parameters in server generated directory listings.

IndexOptions

IndexOptions controls the appearance of server generated directing listings, by adding icons, file descriptions, and so on. If **Options Indexes** is set (refer to Options), the Web server generates a directory listing when the Web server receives an HTTP request for a directory without an index.

First, the Web server looks in the requested directory for a file matching the names listed in the **DirectoryIndex** directive (usually, **index.html**). If an **index.html** file is not found, Apache HTTP Server creates an HTML directory listing of the requested directory. The appearance of this directory listing is controlled, in part, by the **IndexOptions** directive.

The default configuration turns on **FancyIndexing**. This means that a user can re-sort a directory listing by clicking on column headers. Another click on the same header switches from ascending to descending order. **FancyIndexing** also shows different icons for different files, based upon file extensions.

The **AddDescription** option, when used in conjunction with **FancyIndexing**, presents a short description for the file in server generated directory listings.

IndexOptions has a number of other parameters which can be set to control the appearance of server generated directories. The **IconHeight** and **IconWidth** parameters require the server to include HTML **HEIGHT** and **WIDTH** tags for the icons in server generated webpages. The **IconsAreLinks** parameter combines the graphical icon with the HTML link anchor, which contains the URL link target.

KeepAlive

KeepAlive sets whether the server allows more than one request per connection and can be used to prevent any one client from consuming too much of the server's resources.

By default **Keepalive** is set to **off**. If **Keepalive** is set to **on** and the server becomes very busy, the server can quickly spawn the maximum number of child processes. In this situation, the server slows down significantly. If **Keepalive** is enabled, it is a good idea to set the **KeepAliveTimeout** low (refer to KeepAliveTimeout for more information about the **KeepAliveTimeout** directive) and monitor the **/var/log/httpd/error_log** log file on the server. This log reports when the server is running out of child processes.

KeepAliveTimeout

KeepAliveTimeout sets the number of seconds the server waits after a request has been served before it closes the connection. Once the server receives a request, the **Timeout** directive applies instead. The **KeepAliveTimeout** directive is set to 15 seconds by default.

LanguagePriority

`LanguagePriority` sets precedence for different languages in case the client Web browser has no language preference set.

Listen

The **`Listen`** command identifies the ports on which the Web server accepts incoming requests. By default, the Apache HTTP Server is set to listen to port 80 for non-secure Web communications and (in the **`/etc/httpd/conf.d/ssl.conf`** file which defines any secure servers) to port 443 for secure Web communications.

If the Apache HTTP Server is configured to listen to a port under 1024, only the root user can start it. For port 1024 and above, **`httpd`** can be started as a regular user.

The **`Listen`** directive can also be used to specify particular IP addresses over which the server accepts connections.

LoadModule

`LoadModule` is used to load Dynamic Shared Object (DSO) modules. More information on the Apache HTTP Server's DSO support, including instructions for using the **`LoadModule`** directive, can be found in Section 25.6, “Adding Modules”. Note, the load order of the modules is *no longer important* with Apache HTTP Server 2.0. Refer to Section 25.2.2.1.3, “Dynamic Shared Object (DSO) Support” for more information about Apache HTTP Server 2.0 DSO support.

Location

The **`<Location>`** and **`</Location>`** tags create a container in which access control based on URL can be specified.

For instance, to allow people connecting from within the server's domain to see status reports, use the following directives:

```
<Location /server-status>
  SetHandler server-status
  Order deny,allow
  Deny from all
  Allow from <.example.com>
</Location>
```

Replace <*.example.com*> with the second-level domain name for the Web server.

To provide server configuration reports (including installed modules and configuration directives) to requests from inside the domain, use the following directives:

```
<Location /server-info>
  SetHandler server-info
  Order deny,allow
  Deny from all
  Allow from <.example.com>
</Location>
```

Again, replace <*.example.com*> with the second-level domain name for the Web server.

LogFormat

The **`LogFormat`** directive configures the format of the various Web server log files. The actual **`LogFormat`** used depends on the settings given in the **`CustomLog`** directive (refer to CustomLog).

The following are the format options if the **`CustomLog`** directive is set to **`combined`**:

`%h` (remote host's IP address or hostname)

Lists the remote IP address of the requesting client. If **`HostnameLookups`** is set to **`on`**, the client hostname is recorded unless it is not available from DNS.

`%l` (rfc931)

Not used. A hyphen **`-`** appears in the log file for this field.

`%u` (authenticated user)

Lists the username of the user recorded if authentication was required. Usually, this is not used, so a hyphen **`-`** appears in the log file for this field.

`%t` (date)

Lists the date and time of the request.

`%r` (request string)

Lists the request string exactly as it came from the browser or client.

`%s` (status)

Lists the HTTP status code which was returned to the client host.

`%b` (bytes)

Lists the size of the document.

`%\"%{Referer}i\"` (referrer)

Lists the URL of the webpage which referred the client host to Web server.

`%\"%{User-Agent}i\"` (user-agent)

Lists the type of Web browser making the request.

LogLevel

`LogLevel` sets how verbose the error messages in the error logs are. **`LogLevel`** can be set (from least verbose to most verbose) to **`emerg`**, **`alert`**, **`crit`**, **`error`**, **`warn`**, **`notice`**, **`info`**, or **`debug`**. The default **`LogLevel`** is **`warn`**.

MaxKeepAliveRequests

This directive sets the maximum number of requests allowed per persistent connection. The Apache Project recommends a high setting, which improves the server's performance. **`MaxKeepAliveRequests`** is set to **`100`** by default, which should be appropriate for most situations.

NameVirtualHost

The **NameVirtualHost** directive associates an IP address and port number, if necessary, for any name-based virtual hosts. Name-based virtual hosting allows one Apache HTTP Server to serve different domains without using multiple IP addresses.

Note

Name-based virtual hosts *only* work with non-secure HTTP connections. If using virtual hosts with a secure server, use IP address-based virtual hosts instead.

To enable name-based virtual hosting, uncomment the **NameVirtualHost** configuration directive and add the correct IP address. Then add additional **VirtualHost** containers for each virtual host as is necessary for your configuration.

Options

The **Options** directive controls which server features are available in a particular directory. For example, under the restrictive parameters specified for the root directory, **Options** is only set to the **FollowSymLinks** directive. No features are enabled, except that the server is allowed to follow symbolic links in the root directory.

By default, in the **DocumentRoot** directory, **Options** is set to include **Indexes** and **FollowSymLinks**. **Indexes** permits the server to generate a directory listing for a directory if no **DirectoryIndex** (for example, **index.html**) is specified. **FollowSymLinks** allows the server to follow symbolic links in that directory.

Note

Options statements from the main server configuration section need to be replicated to each **VirtualHost** container individually. Refer to VirtualHost for more information.

Order

The **Order** directive controls the order in which **allow** and **deny** directives are evaluated. The server is configured to evaluate the **Allow** directives before the **Deny** directives for the **DocumentRoot** directory.

PidFile

PidFile names the file where the server records its process ID (PID). By default the PID is listed in **/var/run/httpd.pid**.

Proxy

<Proxy *> and **</Proxy>** tags create a container which encloses a group of configuration directives meant to apply only to the proxy server. Many directives which are allowed within a **<Directory>** container may also be used within **<Proxy>** container.

ProxyRequests

To configure the Apache HTTP Server to function as a proxy server, remove the hash mark (#) from the beginning of the **`<IfModule mod_proxy.c>`** line, the ProxyRequests, and each line in the **`<Proxy>`** stanza. Set the **`ProxyRequests`** directive to **`On`**, and set which domains are allowed access to the server in the **`Allow from`** directive of the **`<Proxy>`** stanza.

ReadmeName

`ReadmeName` names the file which, if it exists in the directory, is appended to the end of server generated directory listings. The Web server first tries to include the file as an HTML document and then tries to include it as plain text. By default, **`ReadmeName`** is set to **`README.html`**.

Redirect

When a webpage is moved, **`Redirect`** can be used to map the file location to a new URL. The format is as follows:

```
Redirect /<old-path>/<file-name> http://<current-domain>/<current-
path>/<file-name>
```

In this example, replace *<old-path>* with the old path information for *<file-name>* and *<current-domain>* and *<current-path>* with the current domain and path information for *<file-name>*.

In this example, any requests for *<file-name>* at the old location is automatically redirected to the new location.

For more advanced redirection techniques, use the **`mod_rewrite`** module included with the Apache HTTP Server. For more information about configuring the **`mod_rewrite`** module, refer to the Apache Software Foundation documentation online at http://httpd.apache.org/docs/2.2/mod/mod_rewrite.html.

ScriptAlias

The **`ScriptAlias`** directive defines where CGI scripts are located. Generally, it is not good practice to leave CGI scripts within the **`DocumentRoot`**, where they can potentially be viewed as text documents. For this reason, a special directory outside of the **`DocumentRoot`** directory containing server-side executables and scripts is designated by the **`ScriptAlias`** directive. This directory is known as a **`cgi-bin`** and is set to **`/var/www/cgi-bin/`** by default.

It is possible to establish directories for storing executables outside of the **`cgi-bin/`** directory. For instructions on doing so, refer to AddHandler and Directory.

ServerAdmin

Sets the **`ServerAdmin`** directive to the email address of the Web server administrator. This email address shows up in error messages on server-generated Web pages, so users can report a problem by sending email to the server administrator.

By default, **`ServerAdmin`** is set to **`root@localhost`**.

A common way to set up **`ServerAdmin`** is to set it to **`webmaster@example.com`**. Once set, alias **`webmaster`** to the person responsible for the Web server in **`/etc/aliases`** and run **`/usr/bin/newaliases`**.

ServerName

`ServerName` specifies a hostname and port number (matching the **`Listen`** directive) for the server. The **`ServerName`** does not need to match the machine's actual hostname. For example, the Web server may be

www.example.com, but the server's hostname is actually **foo.example.com**. The value specified in **ServerName** must be a valid Domain Name Service (DNS) name that can be resolved by the system — do not make something up.

The following is a sample **ServerName** directive:

```
ServerName www.example.com:80
```

When specifying a **ServerName**, be sure the IP address and server name pair are included in the **/etc/hosts** file.

ServerRoot

The **ServerRoot** directive specifies the top-level directory containing website content. By default, **ServerRoot** is set to **"/etc/httpd"** for both secure and non-secure servers.

ServerSignature

The **ServerSignature** directive adds a line containing the Apache HTTP Server server version and the **ServerName** to any server-generated documents, such as error messages sent back to clients. **ServerSignature** is set to **on** by default.

ServerSignature can be set to **EMail** which adds a **mailto:ServerAdmin** HTML tag to the signature line of auto-generated responses. **ServerSignature** can also be set to **Off** to stop Apache from sending out its version number and module information. Please also check the **ServerTokens** settings.

ServerTokens

The **ServerTokens** directive determines if the Server response header field sent back to clients should include details of the Operating System type and information about compiled-in modules. By default, **ServerTokens** is set to **Full** which sends information about the Operating System type and compiled-in modules. Setting the **ServerTokens** to **Prod** sends the product name only and is recommended as many hackers check information in the Server header when scanning for vulnerabilities. You can also set the **ServerTokens** to **Min** (minimal) or to **OS** (operating system).

SuexecUserGroup

The **SuexecUserGroup** directive, which originates from the **mod_suexec** module, allows the specification of user and group execution privileges for CGI programs. Non-CGI requests are still processed with the user and group specified in the **User** and **Group** directives.

Note

From version 2.0, the **SuexecUserGroup** directive replaced the Apache HTTP Server 1.3 configuration of using the **User** and **Group** directives inside the configuration of **VirtualHosts** sections.

Timeout

Timeout defines, in seconds, the amount of time that the server waits for receipts and transmissions during communications. **Timeout** is set to **300** seconds by default, which is appropriate for most situations.

TypesConfig

`TypesConfig` names the file which sets the default list of MIME type mappings (file name extensions to content types). The default **`TypesConfig`** file is **`/etc/mime.types`**. Instead of editing **`/etc/mime.types`**, the recommended way to add MIME type mappings is to use the **`AddType`** directive.

For more information about **`AddType`**, refer to AddType.

UseCanonicalName

When set to **`on`**, this directive configures the Apache HTTP Server to reference itself using the value specified in the **`ServerName`** and **`Port`** directives. When **`UseCanonicalName`** is set to **`off`**, the server instead uses the value used by the requesting client when referring to itself.

`UseCanonicalName` is set to **`off`** by default.

User

The **`User`** directive sets the username of the server process and determines what files the server is allowed to access. Any files inaccessible to this user are also inaccessible to clients connecting to the Apache HTTP Server.

By default **`User`** is set to **`apache`**.

This directive has been deprecated for the configuration of virtual hosts.

Note

For security reasons, the Apache HTTP Server does not run as the root user.

UserDir

`UserDir` is the subdirectory within each user's home directory where they should place personal HTML files which are served by the Web server. This directive is set to **`disable`** by default.

The name for the subdirectory is set to **`public_html`** in the default configuration. For example, the server might receive the following request:

```
http://example.com/~username/foo.html
```

The server would look for the file:

```
/home/username/public_html/foo.html
```

In the above example, **`/home/username/`** is the user's home directory (note that the default path to users' home directories may vary).

Make sure that the permissions on the users' home directories are set correctly. Users' home directories must be set to 0711. The read (r) and execute (x) bits must be set on the users' **`public_html`** directories (0755 also works). Files that are served in a users' **`public_html`** directories must be set to at least 0644.

VirtualHost

`<VirtualHost>` and **`</VirtualHost>`** tags create a container outlining the characteristics of a virtual host. The **`VirtualHost`** container accepts most configuration directives.

A commented **`VirtualHost`** container is provided in **`httpd.conf`**, which illustrates the minimum set of configuration directives necessary for each virtual host. Refer to Section 25.7, “Virtual Hosts” for more information about virtual hosts.

Note

The default SSL virtual host container now resides in the file **`/etc/httpd/conf.d/ssl.conf`**.

25.5.2. Configuration Directives for SSL

The directives in **`/etc/httpd/conf.d/ssl.conf`** file can be configured to enable secure Web communications using TLS. See *Resolution for POODLE SSLv3.0 vulnerability (CVE-2014-3566) in httpd* for important information on disabling SSL while enabling TLS.

Important

Due to the vulnerability described in *POODLE: SSLv3 vulnerability (CVE-2014-3566)*, Red Hat recommends disabling **`SSL`** and using only **`TLSv1.1`** or **`TLSv1.2`**. Backwards compatibility can be achieved using **`TLSv1.0`**. Many products Red Hat supports have the ability to use **`SSLv2`** or **`SSLv3`** protocols, or enable them by default. However, the use of **`SSLv2`** or **`SSLv3`** is now strongly recommended against.

SetEnvIf

`SetEnvIf` sets environment variables based on the headers of incoming connections. It is *not* solely an SSL directive, though it is present in the supplied **`/etc/httpd/conf.d/ssl.conf`** file. It's purpose in this context is to disable HTTP keepalive and to allow SSL to close the connection without a closing notification from the client browser. This setting is necessary for certain browsers that do not reliably shut down the SSL connection.

For more information on other directives within the SSL configuration file, refer to the following URLs:

- http://localhost/manual/mod/mod_ssl.html
- http://httpd.apache.org/docs/2.2/mod/mod_ssl.html

Note

In most cases, SSL directives are configured appropriately during the installation of Red Hat Enterprise Linux. Be careful when altering Apache HTTP Secure Server directives, misconfiguration can lead to security vulnerabilities.

25.5.3. MPM Specific Server-Pool Directives

As explained in Section 25.2.2.1.2, “Server-Pool Size Regulation”, the responsibility for managing characteristics of the server-pool falls to a module group called MPMs under Apache HTTP Server 2.0. The characteristics of the server-pool differ depending upon which MPM is used. For this reason, an **`IfModule`** container is necessary to define the server-pool for the MPM in use.

By default, Apache HTTP Server 2.0 defines the server-pool for both the **`prefork`** and **`worker`** MPMs.

The following section list directives found within the MPM-specific server-pool containers.

MaxClients

`MaxClients` sets a limit on the total number of server processes, or simultaneously connected clients, that can run at one time. The main purpose of this directive is to keep a runaway Apache HTTP Server from crashing the operating system. For busy servers this value should be set to a high value. The server's default is set to 150 regardless of the MPM in use. However, it is not recommended that the value for **`MaxClients`** exceeds **`256`** when using the **`prefork`** MPM.

MaxRequestsPerChild

`MaxRequestsPerChild` sets the total number of requests each child server process serves before the child dies. The main reason for setting **`MaxRequestsPerChild`** is to avoid long-lived process induced memory leaks. The default **`MaxRequestsPerChild`** for the **`prefork`** MPM is **`4000`** and for the **`worker`** MPM is **`0`**.

MinSpareServers and MaxSpareServers

These values are only used with the **`prefork`** MPM. They adjust how the Apache HTTP Server dynamically adapts to the perceived load by maintaining an appropriate number of spare server processes based on the number of incoming requests. The server checks the number of servers waiting for a request and kills some if there are more than **`MaxSpareServers`** or creates some if the number of servers is less than **`MinSpareServers`**.

The default **`MinSpareServers`** value is **`5`**; the default **`MaxSpareServers`** value is **`20`**. These default settings should be appropriate for most situations. Be careful not to increase the **`MinSpareServers`** to a large number as doing so creates a heavy processing load on the server even when traffic is light.

MinSpareThreads and MaxSpareThreads

These values are only used with the **`worker`** MPM. They adjust how the Apache HTTP Server dynamically adapts to the perceived load by maintaining an appropriate number of spare server threads based on the number of incoming requests. The server checks the number of server threads waiting for a request and kills some if there are more than **`MaxSpareThreads`** or creates some if the number of servers is less than **`MinSpareThreads`**.

The default **`MinSpareThreads`** value is **`25`**; the default **`MaxSpareThreads`** value is **`75`**. These default settings should be appropriate for most situations. The value for **`MaxSpareThreads`** must be greater than or equal to the sum of **`MinSpareThreads`** and **`ThreadsPerChild`**, else the Apache HTTP Server automatically corrects it.

StartServers

The **`StartServers`** directive sets how many server processes are created upon startup. Since the Web server dynamically kills and creates server processes based on traffic load, it is not necessary to change this parameter. The Web server is set to start **`8`** server processes at startup for the **`prefork`** MPM and **`2`** for the **`worker`** MPM.

ThreadsPerChild

This value is only used with the **`worker`** MPM. It sets the number of threads within each child process. The default value for this directive is **`25`**.

25.6. Adding Modules

25.6. Adding Modules

The Apache HTTP Server is distributed with a number of modules. More information about Apache HTTP modules can be found on http://httpd.apache.org/docs/2.2/mod/.

The Apache HTTP Server supports *Dynamically Shared Objects* (*DSO*s), or modules, which can easily be loaded at runtime as necessary.

The Apache Project provides complete DSO documentation online at http://httpd.apache.org/docs/2.2/dso.html. Or, if the **http-manual** package is installed, documentation about DSOs can be found online at http://localhost/manual/mod/.

For the Apache HTTP Server to use a DSO, it must be specified in a **LoadModule** directive within **/etc/httpd/conf/httpd.conf**. If the module is provided by a separate package, the line must appear within the modules configuration file in the **/etc/httpd/conf.d/** directory. Refer to LoadModule for more information.

If adding or deleting modules from **http.conf**, Apache HTTP Server must be reloaded or restarted, as referred to in Section 25.3, “Starting and Stopping **httpd**”.

If creating a new module, first install the **httpd-devel** package which contains the include files, the header files, as well as the *APache eXtenSion* (**/usr/sbin/apxs**) application, which uses the include files and the header files to compile DSOs.

After writing a module, use **/usr/sbin/apxs** to compile the module sources outside the Apache source tree. For more information about using the **/usr/sbin/apxs** command, refer to the Apache documentation online at http://httpd.apache.org/docs/2.2/dso.html as well as the **apxs** man page.

Once compiled, put the module in the **/usr/lib/httpd/modules/** directory. For RHEL platforms using default-64-bit userspace (x86_64, ia64, ?) this path will be **/usr/lib64/httpd/modules/**. Then add a **LoadModule** line to the **httpd.conf**, using the following structure:

```
LoadModule <module-name> <path/to/module.so>
```

Where <*module-name*> is the name of the module and <*path/to/module.so*> is the path to the DSO.

25.7. Virtual Hosts

The Apache HTTP Server's built in virtual hosting allows the server to provide different information based on which IP address, hostname, or port is being requested. A complete guide to using virtual hosts is available online at http://httpd.apache.org/docs/2.2/vhosts/.

25.7.1. Setting Up Virtual Hosts

To create a name-based virtual host, it is best to use the virtual host container provided in **httpd.conf** as an example.

The virtual host example read as follows:

```
#NameVirtualHost *:80
#
#<VirtualHost *:80>
# ServerAdmin webmaster@dummy-host.example.com
# DocumentRoot /www/docs/dummy-host.example.com
# ServerName dummy-host.example.com
# ErrorLog logs/dummy-host.example.com-error_log
# CustomLog logs/dummy-host.example.com-access_log common #</VirtualHost>
```

To activate name-based virtual hosting, uncomment the **`NameVirtualHost`** line by removing the hash mark (`#`) and replace the asterisk (**`*`**) with the IP address assigned to the machine.

Next, configure a virtual host by uncommenting and customizing the **`<VirtualHost>`** container.

On the **`<VirtualHost>`** line, change the asterisk (**`*`**) to the server's IP address. Change the **`ServerName`** to a *valid* DNS name assigned to the machine, and configure the other directives as necessary.

The **`<VirtualHost>`** container is highly customizable and accepts almost every directive available within the main server configuration.

Note

If configuring a virtual host to listen on a non-default port, that port must be added to the **`Listen`** directive in the global settings section of **`/etc/httpd/conf/httpd.conf`** file.

To activate a newly created virtual host, the Apache HTTP Server must be reloaded or restarted. Refer to Section 25.3, "Starting and Stopping **`httpd`**" for further instructions.

Comprehensive information about creating and configuring both name-based and IP address-based virtual hosts is provided online at http://httpd.apache.org/docs/2.2/vhosts/.

25.8. Apache HTTP Secure Server Configuration

This section provides basic information on the Apache HTTP Server with the **`mod_ssl`** security module enabled to use the OpenSSL library and toolkit. The combination of these three components are referred to in this section as the secure Web server or just as the secure server.

The **`mod_ssl`** module is a security module for the Apache HTTP Server. The **`mod_ssl`** module uses the tools provided by the OpenSSL Project to add a very important feature to the Apache HTTP Server — the ability to encrypt communications. In contrast, regular HTTP communications between a browser and a Web server are sent in plain text, which could be intercepted and read by someone along the route between the browser and the server.

This section is not meant to be complete and exclusive documentation for any of these programs. When possible, this guide points to appropriate places where you can find more in-depth documentation on particular subjects.

This section shows you how to install these programs. You can also learn the steps necessary to generate a private key and a certificate request, how to generate your own self-signed certificate, and how to install a certificate to use with your secure server.

The **`mod_ssl`** configuration file is located at **`/etc/httpd/conf.d/ssl.conf`**. For this file to be loaded, and hence for **`mod_ssl`** to work, you must have the statement **`Include conf.d/*.conf`** in the **`/etc/httpd/conf/httpd.conf`** file. This statement is included by default in the default Apache HTTP Server configuration file.

25.8.1. An Overview of Security-Related Packages

To enable the secure server, you must have the following packages installed at a minimum:

`httpd`

> The **`httpd`** package contains the **`httpd`** daemon and related utilities, configuration files, icons, Apache HTTP Server modules, man pages, and other files used by the Apache HTTP Server.

`mod_ssl`

> The **`mod_ssl`** package includes the **`mod_ssl`** module, which provides strong cryptography for the Apache HTTP Server via the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols.

`openssl`

> The **`openssl`** package contains the OpenSSL toolkit. The OpenSSL toolkit implements the SSL and TLS protocols, and also includes a general purpose cryptography library.

Additionally, other software packages provide certain security functionalities (but are not required by the secure server to function):

25.8.2. An Overview of Certificates and Security

Your secure server provides security using a combination of the Secure Sockets Layer (SSL) protocol and (in most cases) a digital certificate from a Certificate Authority (CA). SSL handles the encrypted communications as well as the mutual authentication between browsers and your secure server. The CA-approved digital certificate provides authentication for your secure server (the CA puts its reputation behind its certification of your organization's identity). When your browser is communicating using SSL encryption, the **`https://`** prefix is used at the beginning of the Uniform Resource Locator (URL) in the navigation bar.

Encryption depends upon the use of keys (think of them as secret encoder/decoder rings in data format). In conventional or symmetric cryptography, both ends of the transaction have the same key, which they use to decode each other's transmissions. In public or asymmetric cryptography, two keys co-exist: a public key and a private key. A person or an organization keeps their private key a secret and publishes their public key. Data encoded with the public key can only be decoded with the private key; data encoded with the private key can only be decoded with the public key.

To set up your secure server, use public cryptography to create a public and private key pair. In most cases, you send your certificate request (including your public key), proof of your company's identity, and payment to a CA. The CA verifies the certificate request and your identity, and then sends back a certificate for your secure server.

A secure server uses a certificate to identify itself to Web browsers. You can generate your own certificate (called a "self-signed" certificate), or you can get a certificate from a CA. A certificate from a reputable CA guarantees that a website is associated with a particular company or organization.

Alternatively, you can create your own self-signed certificate. Note, however, that self-signed certificates should not be used in most production environments. Self-signed certificates are not automatically accepted by a user's browser — users are prompted by the browser to accept the certificate and create the secure connection. Refer to Section 25.8.4, “Types of Certificates” for more information on the differences between self-signed and CA-signed certificates.

Once you have a self-signed certificate or a signed certificate from the CA of your choice, you must install it on your secure server.

25.8.3. Using Pre-Existing Keys and Certificates

If you already have an existing key and certificate (for example, if you are installing the secure server to replace another company's secure server product), you can probably use your existing key and certificate with the secure server. The following two situations provide instances where you are not able to use your existing key and certificate:

- *If you are changing your IP address or domain name* — Certificates are issued for a particular IP address and domain name pair. You must get a new certificate if you are changing your IP address or domain name.
- *If you have a certificate from VeriSign and you are changing your server software* — VeriSign is a widely used CA. If you already have a VeriSign certificate for another purpose, you may have been considering using your existing VeriSign certificate with your new secure server. However, you are not be allowed to because VeriSign issues certificates for one specific server software and IP address/domain name combination.

  If you change either of those parameters (for example, if you previously used a different secure server product), the VeriSign certificate you obtained to use with the previous configuration will not work with the new configuration. You must obtain a new certificate.

If you have an existing key and certificate that you can use, you do not have to generate a new key and obtain a new certificate. However, you may need to move and rename the files which contain your key and certificate.

Move your existing key file to:

```
/etc/pki/tls/private/server.key
```

Move your existing certificate file to:

```
/etc/pki/tls/certs/server.crt
```

If you are upgrading from the Red Hat Secure Web Server, your old key (**`httpsd.key`**) and certificate (**`httpsd.crt`**) are located in **`/etc/httpd/conf/`**. Move and rename your key and certificate so that the secure server can use them. Use the following two commands to move and rename your key and certificate files:

```
mv /etc/httpd/conf/httpsd.key /etc/pki/tls/private/server.key
mv /etc/httpd/conf/httpsd.crt /etc/pki/tls/certs/server.crt
```

Then, start your secure server with the command:

```
service httpd start
```

25.8.4. Types of Certificates

If you installed your secure server from the RPM package provided by Red Hat, a randomly generated private key and a test certificate are generated and put into the appropriate directories. Before you begin using your secure server, however, you must generate your own key and obtain a certificate which correctly identifies your server.

You need a key and a certificate to operate your secure server — which means that you can either generate a self-signed certificate or purchase a CA-signed certificate from a CA. What are the differences between the two?

A CA-signed certificate provides two important capabilities for your server:

- Browsers (usually) automatically recognize the certificate and allow a secure connection to be made, without prompting the user.
- When a CA issues a signed certificate, they are guaranteeing the identity of the organization that is providing the webpages to the browser.

If your secure server is being accessed by the public at large, your secure server needs a certificate signed by a CA so that people who visit your website know that the website is owned by the organization who claims to own it. Before signing a certificate, a CA verifies that the organization requesting the certificate was actually who they claimed to be.

Most Web browsers that support SSL have a list of CAs whose certificates they automatically accept. If a browser encounters a certificate whose authorizing CA is not in the list, the browser asks the user to either accept or decline the connection.

You can generate a self-signed certificate for your secure server, but be aware that a self-signed certificate does not provide the same functionality as a CA-signed certificate. A self-signed certificate is not automatically recognized by most Web browsers and does not provide any guarantee concerning the identity of the organization that is providing the website. A CA-signed certificate provides both of these important capabilities for a secure server. If your secure server is to be used in a production environment, a CA-signed certificate is recommended.

The process of getting a certificate from a CA is fairly easy. A quick overview is as follows:

1. Create an encryption private and public key pair.
2. Create a certificate request based on the public key. The certificate request contains information about your server and the company hosting it.
3. Send the certificate request, along with documents proving your identity, to a CA. Red Hat does not make recommendations on which certificate authority to choose. Your decision may be based on your past experiences, on the experiences of your friends or colleagues, or purely on monetary factors.

   Once you have decided upon a CA, you need to follow the instructions they provide on how to obtain a certificate from them.
4. When the CA is satisfied that you are indeed who you claim to be, they provide you with a digital certificate.
5. Install this certificate on your secure server and begin handling secure transactions.

Whether you are getting a certificate from a CA or generating your own self-signed certificate, the first step is to generate a key. Refer to Section 25.8.5, “Generating a Key” for instructions.

25.8.5. Generating a Key

You must be root to generate a key.

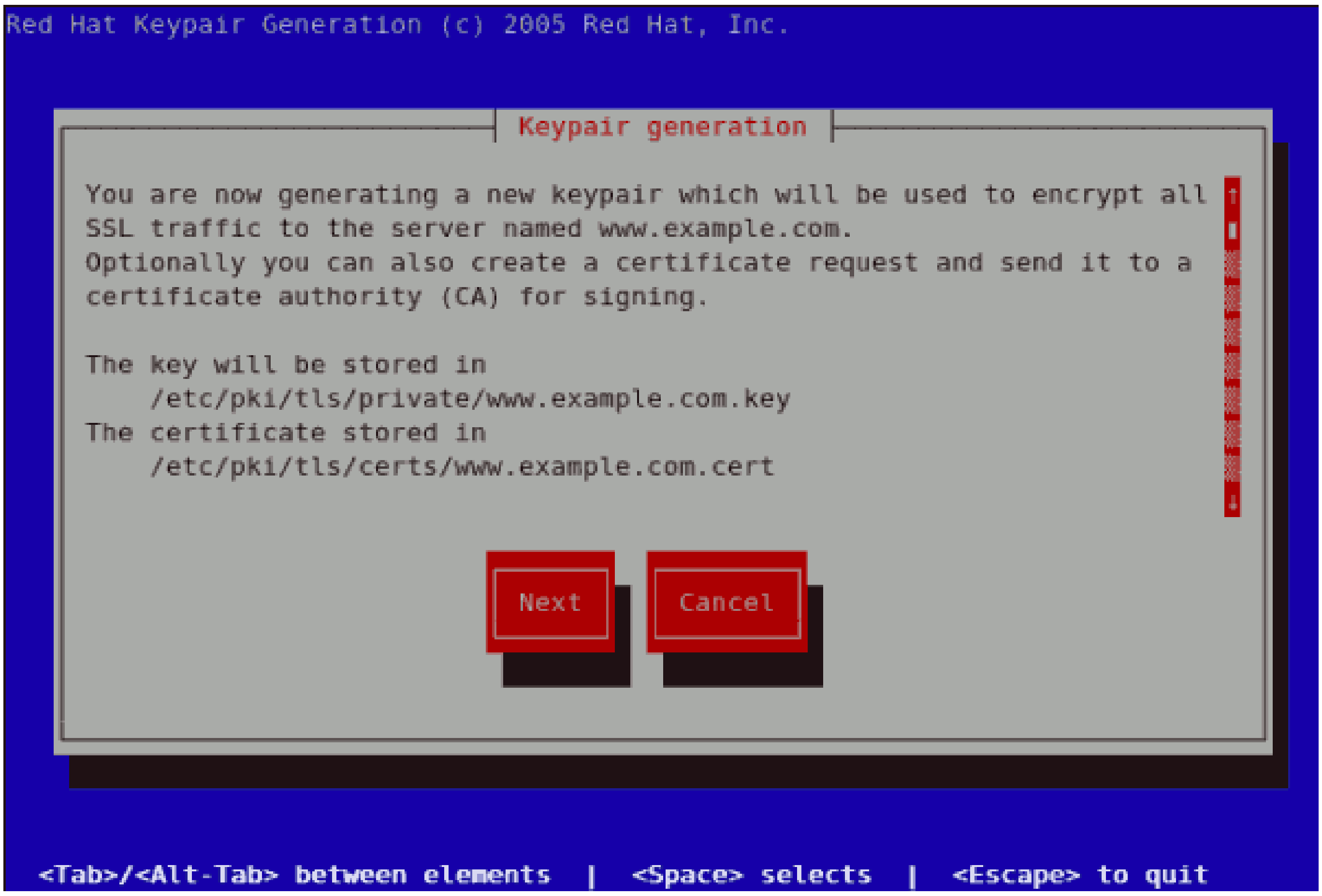
First, use the **cd** command to change to the **/etc/pki/tls/** directory. Remove the fake key and certificate that were generated during the installation with the following commands:

```
rm private/server.key
rm certs/server.crt
```

The **`crypto-utils`** package contains the **`genkey`** utility which you can use to generate keys as the name implies. To create your own private key, please ensure the **`crypto-utils`** package is installed. You can view more options by typing **`man genkey`** in your terminal. Assuming you wish to generate keys for www.example.com using the **`genkey`** utility, type in the following command in your terminal:

```
genkey www.example.com
```

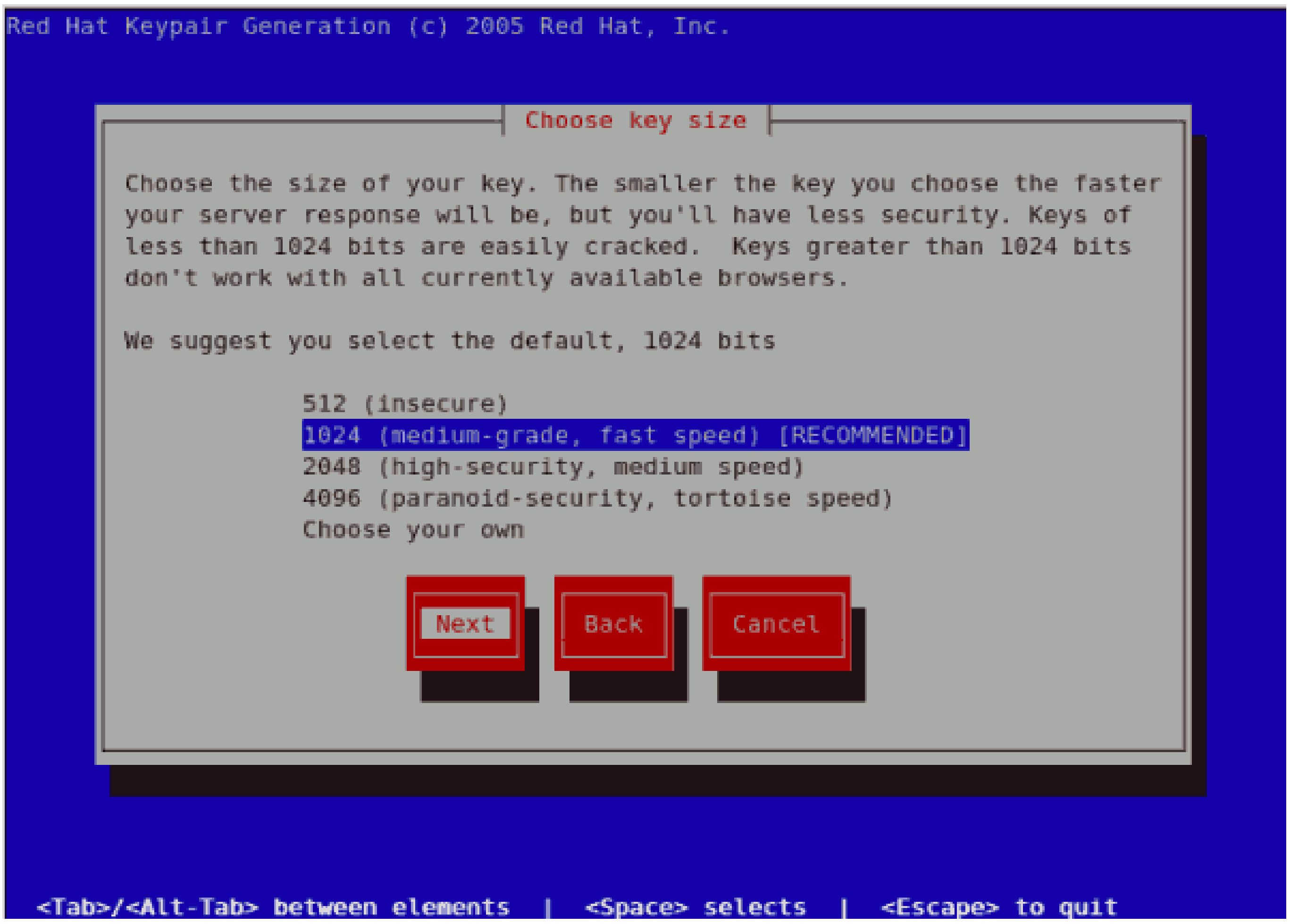
Please note that the **`make`** based process is no longer shipped with RHEL 5. This will start the **`genkey`** graphical user interface. The figure below illustrates the first screen. To navigate, use the keyboard arrow and tab keys. This windows indicates where your key will be stored and prompts you to proceed or cancel the operation. To proceed to the next step, select **`Next`** and press the Return (Enter) key.



```
Red Hat Keypair Generation (c) 2005 Red Hat, Inc.

Keypair generation

You are now generating a new keypair which will be used to encrypt all
SSL traffic to the server named www.example.com.
Optionally you can also create a certificate request and send it to a
certificate authority (CA) for signing.

The key will be stored in
    /etc/pki/tls/private/www.example.com.key
The certificate stored in
    /etc/pki/tls/certs/www.example.com.cert

Next    Cancel

<Tab>/<Alt-Tab> between elements   |  <Space> selects   |  <Escape> to quit
```

Figure 25.11. Keypair generation

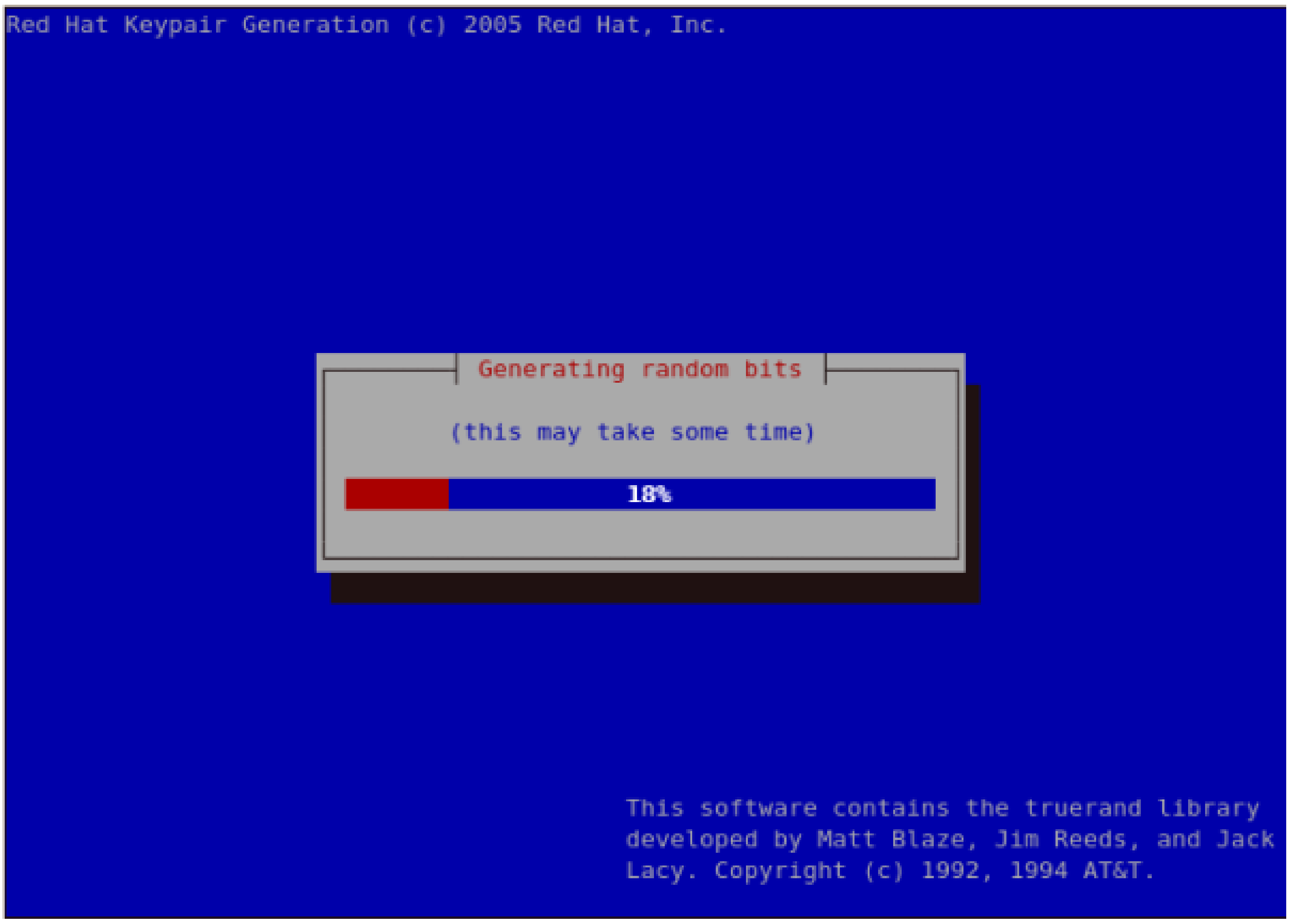
The next screen prompts you to choose the size of your key. As indicated, the smaller the size of your key, the faster will the response from your server be and the lesser your level of security. On selecting your preferred, key size using the arrow keys, select **`Next`** to proceed to the next step. The figure below illustrates the key size selection screen.



```
Red Hat Keypair Generation (c) 2005 Red Hat, Inc.

                    Choose key size

  Choose the size of your key. The smaller the key you choose the faster
  your server response will be, but you'll have less security. Keys of
  less than 1024 bits are easily cracked.  Keys greater than 1024 bits
  don't work with all currently available browsers.

  We suggest you select the default, 1024 bits

              512 (insecure)
              1024 (medium-grade, fast speed) [RECOMMENDED]
              2048 (high-security, medium speed)
              4096 (paranoid-security, tortoise speed)
              Choose your own

                  Next      Back      Cancel

 <Tab>/<Alt-Tab> between elements   |  <Space> selects   |  <Escape> to quit
```

Figure 25.12. Choose key size

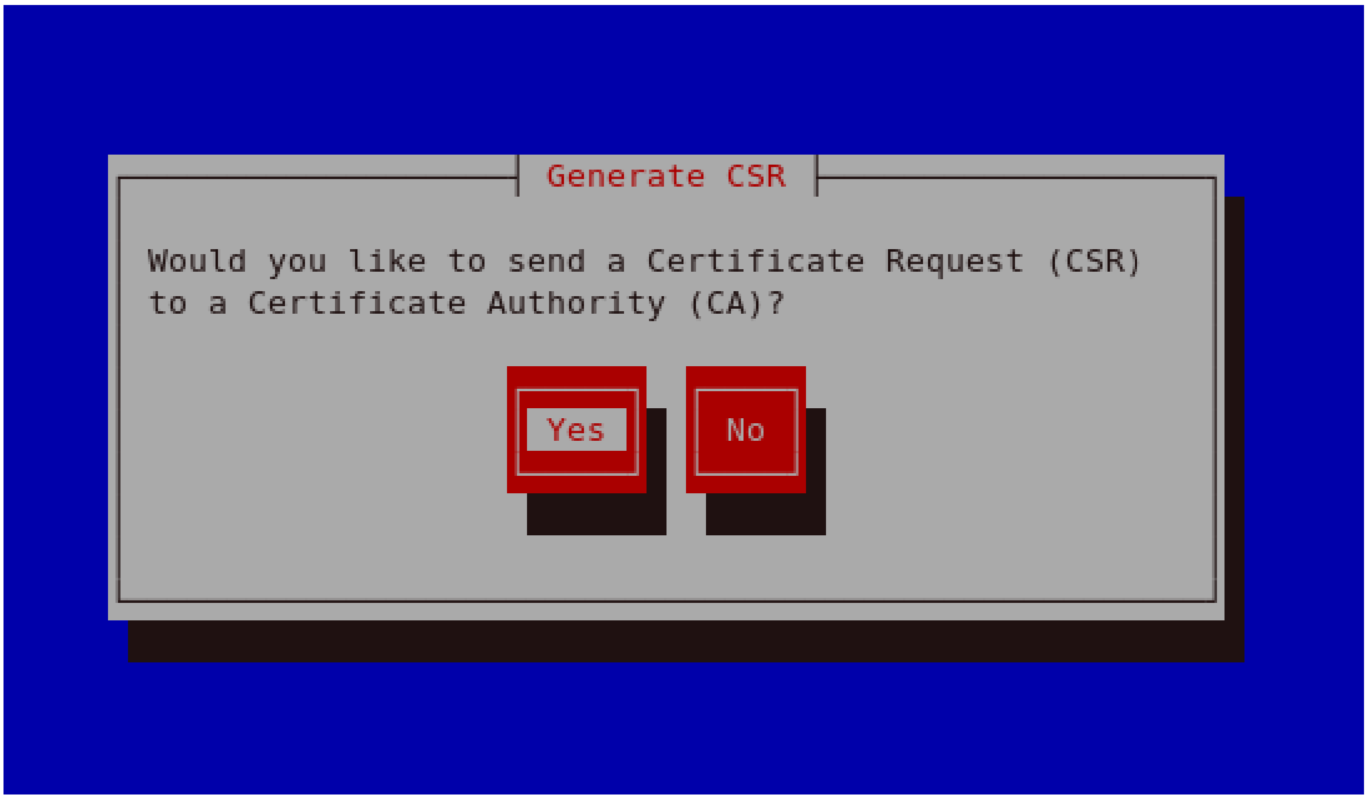
Selecting the next step will initiate the random bits generation process which may take some time depending on the size of your selected key. The larger the size of your key, the longer it will take to generate it.



```
Red Hat Keypair Generation (c) 2005 Red Hat, Inc.

                    Generating random bits

                  (this may take some time)

                            18%

                          This software contains the truerand library
                          developed by Matt Blaze, Jim Reeds, and Jack
                          Lacy. Copyright (c) 1992, 1994 AT&T.
```

Figure 25.13. Generating random bits

On generating your key, you will be prompted to send a Certificate Request (CSR) to a Certificate Authority (CA).



```
                   Generate CSR

 Would you like to send a Certificate Request (CSR)
 to a Certificate Authority (CA)?
```

Figure 25.14. Generate CSR

Selecting **`Yes`** will prompt you to select the Certificate Authority you wish to send your request to. Selecting **`No`** will allow you to generate a self-signed certificate. The next step for this is illustrated in Figure 25.17, "Generating a self signed certificate for your server".

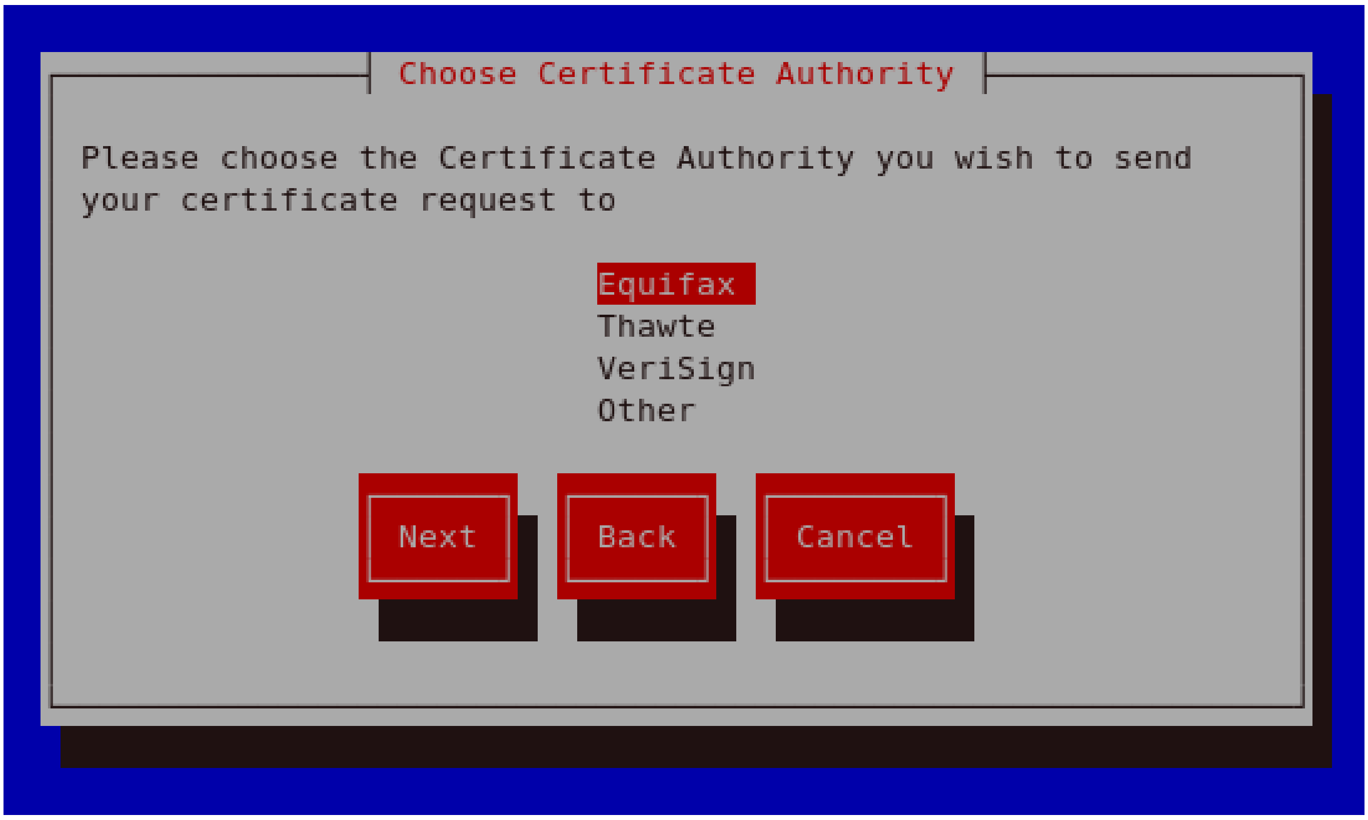
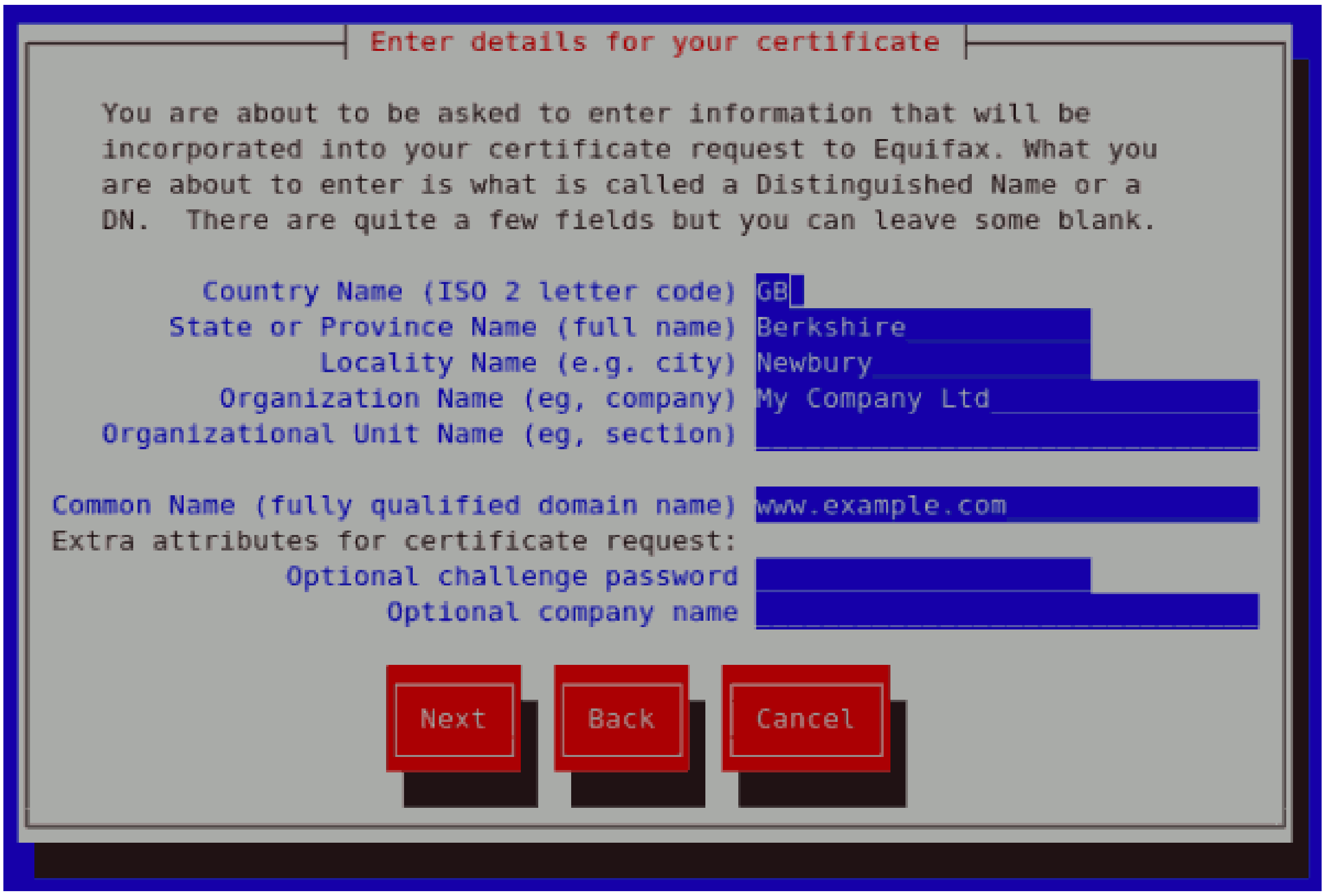

```
Choose Certificate Authority

Please choose the Certificate Authority you wish to send
your certificate request to

Equifax
Thawte
VeriSign
Other
```

Figure 25.15. Choose Certificate Authority (CA)

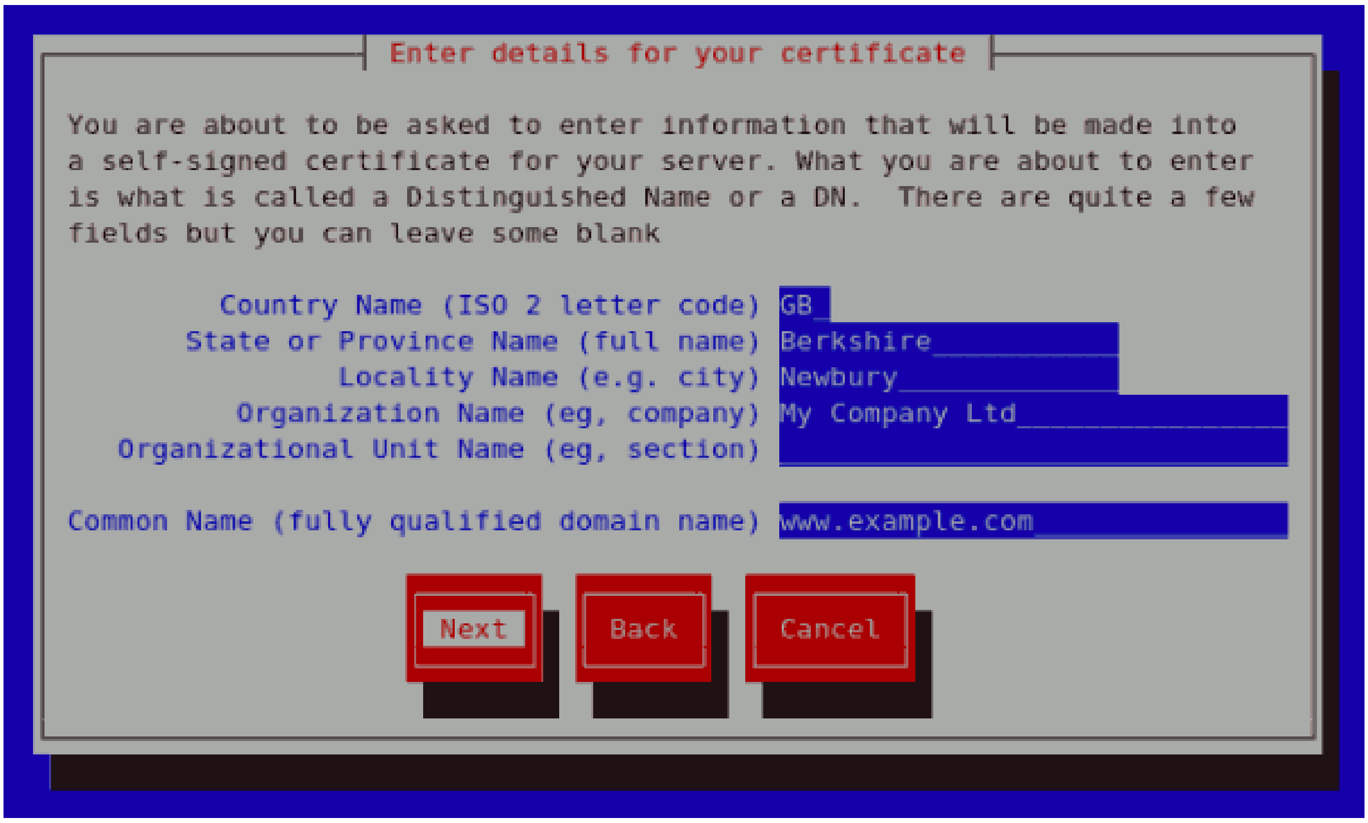
On Selecting your preferred option, select **`Next`** to proceed to the next step. The next screen allows you to enter the details of your certificate.



```
Enter details for your certificate

  You are about to be asked to enter information that will be
  incorporated into your certificate request to Equifax. What you
  are about to enter is what is called a Distinguished Name or a
  DN.  There are quite a few fields but you can leave some blank.

         Country Name (ISO 2 letter code) GB
       State or Province Name (full name) Berkshire
              Locality Name (e.g. city) Newbury
          Organization Name (eg, company) My Company Ltd
    Organizational Unit Name (eg, section)

Common Name (fully qualified domain name) www.example.com
Extra attributes for certificate request:
               Optional challenge password
                   Optional company name

              Next      Back      Cancel
```

Figure 25.16. Enter details for your certificate

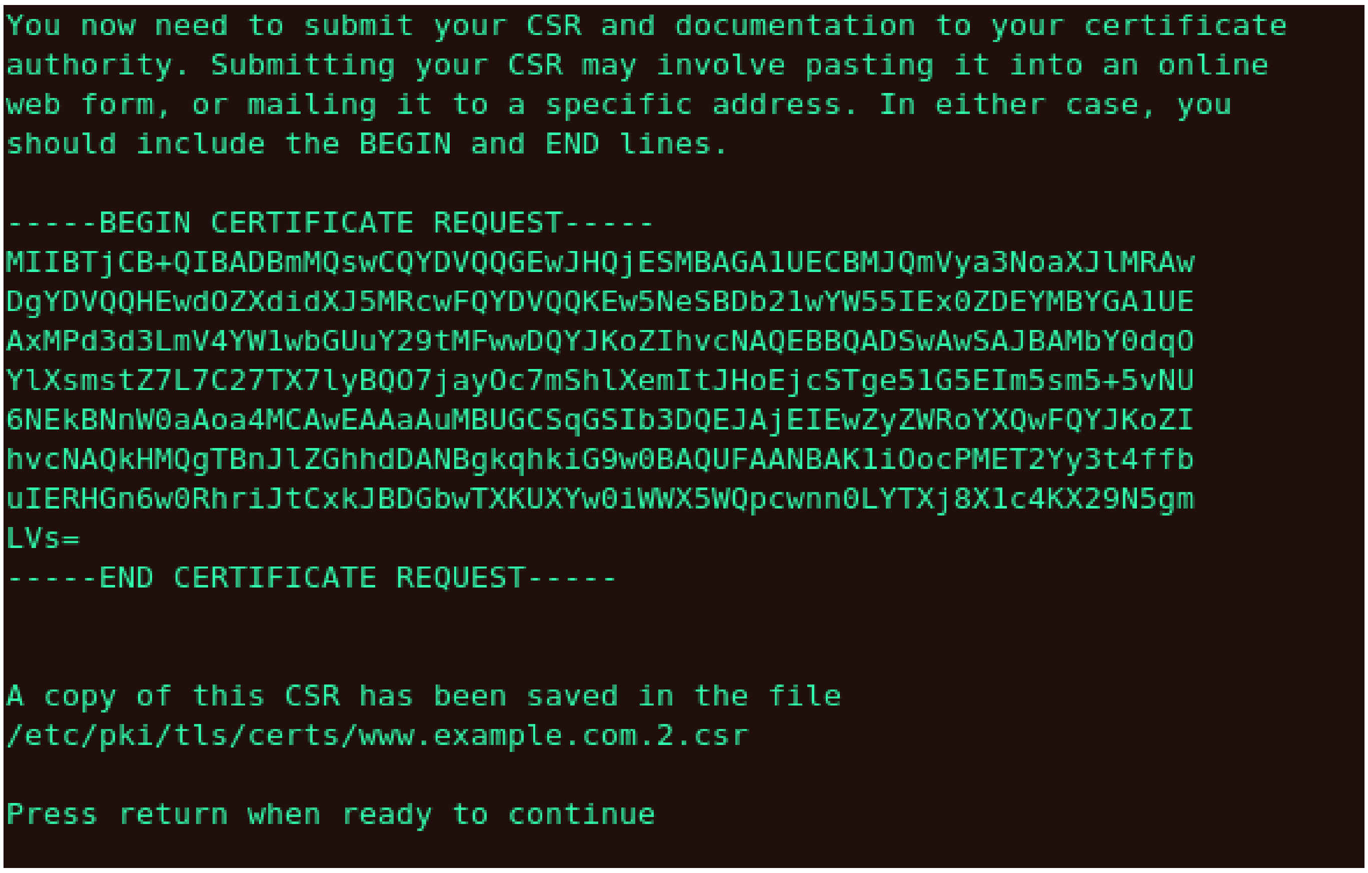
If you prefer to generate a self signed cert key pair, you should not generate a CSR. To do this, select **No** as your preferred option in the Generate CSR screen. This will display the figure below from which you can enter your certificate details. Entering your certificate details and pressing the return key will display the Figure 25.19, “Protecting your private key” from which you can choose to encrypt your private key or not.



```
Enter details for your certificate

You are about to be asked to enter information that will be made into
a self-signed certificate for your server. What you are about to enter
is what is called a Distinguished Name or a DN.  There are quite a few
fields but you can leave some blank

         Country Name (ISO 2 letter code) GB
       State or Province Name (full name) Berkshire
              Locality Name (e.g. city) Newbury
          Organization Name (eg, company) My Company Ltd
    Organizational Unit Name (eg, section)

Common Name (fully qualified domain name) www.example.com

              Next      Back      Cancel
```

Figure 25.17. Generating a self signed certificate for your server

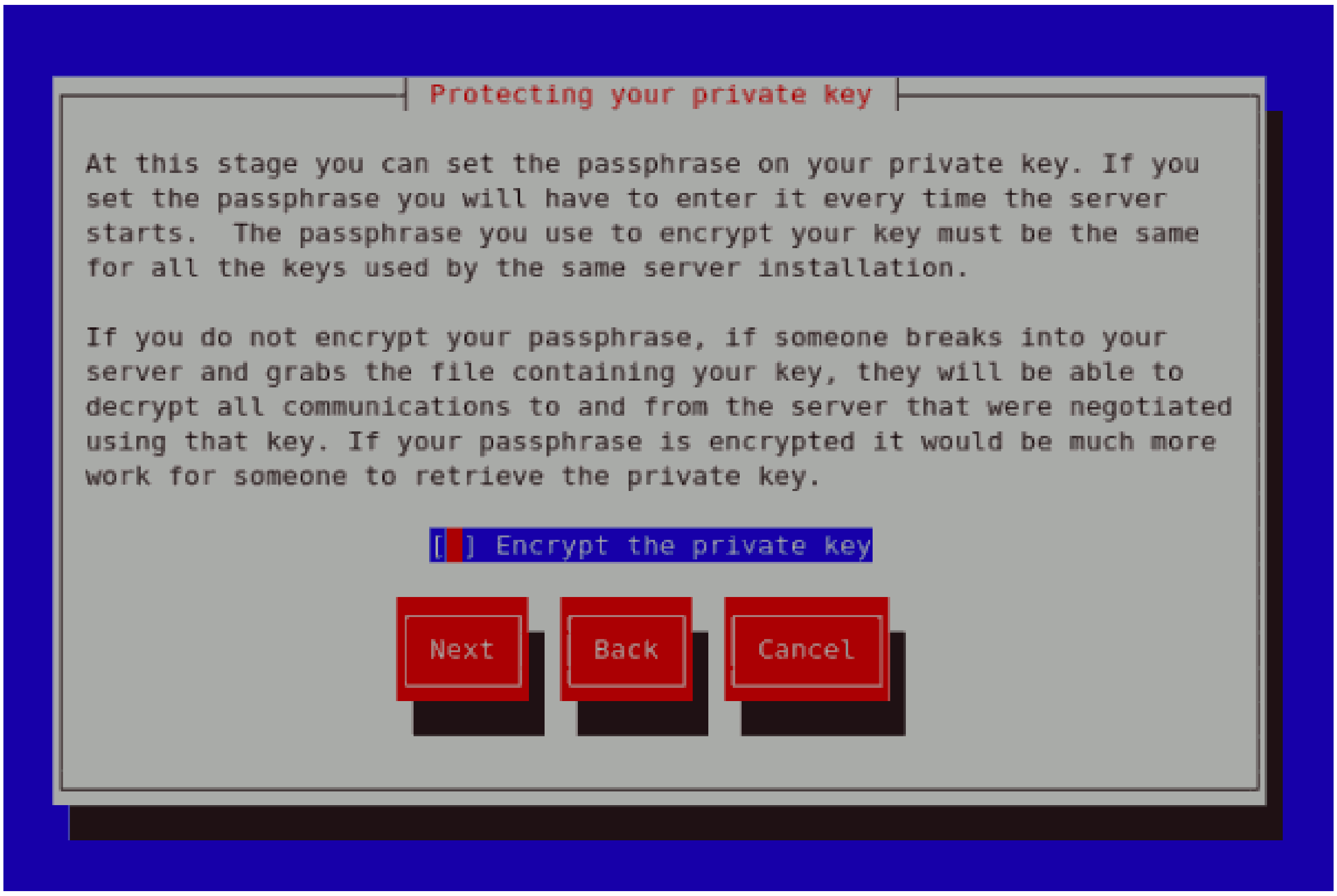
On entering the details of your certificate, select **`Next`** to proceed. The figure below illustrates an example of a the next screen displayed after completing the details for a certificate to be sent to Equifax. Please note that if you are generating a self signed key, for your server, this screen is not displayed.



```
You now need to submit your CSR and documentation to your certificate
authority. Submitting your CSR may involve pasting it into an online
web form, or mailing it to a specific address. In either case, you
should include the BEGIN and END lines.

-----BEGIN CERTIFICATE REQUEST-----
MIIBTjCB+QIBADBmMQswCQYDVQQGEwJHQjESMBAGA1UECBMJQmVya3NoaXJlMRAw
DgYDVQQHEwdOZXdidXJ5MRcwFQYDVQQKEw5NeSBDb21wYW55IEx0ZDEYMBYGA1UE
AxMPd3d3LmV4YW1wbGUuY29tMFwwDQYJKoZIhvcNAQEBBQADSwAwSAJBAMbY0dqO
YlXsmstZ7L7C27TX7lyBQO7jayOc7mShlXemItJHoEjcSTge51G5EIm5sm5+5vNU
6NEkBNnW0aAoa4MCAwEAAaAuMBUGCSqGSIb3DQEJAjEIEwZyZWRoYXQwFQYJKoZI
hvcNAQkHMQgTBnJlZGhhdDANBgkqhkiG9w0BAQUFAANBAK1iOocPMET2Yy3t4ffb
uIERHGn6w0RhriJtCxkJBDGbwTXKUXYw0iWWX5WQpcwnn0LYTXj8X1c4KX29N5gm
LVs=
-----END CERTIFICATE REQUEST-----


A copy of this CSR has been saved in the file
/etc/pki/tls/certs/www.example.com.2.csr

Press return when ready to continue
```

Figure 25.18. Begin certificate request

Pressing the return key, will display the next screen from which you can enable or disable the encryption of the private key. Use the spacebar to enable or disable this. When enabled, a [*] character will be displayed. On selecting your preferred option, select **`Next`** to proceed to the next step.



Protecting your private key

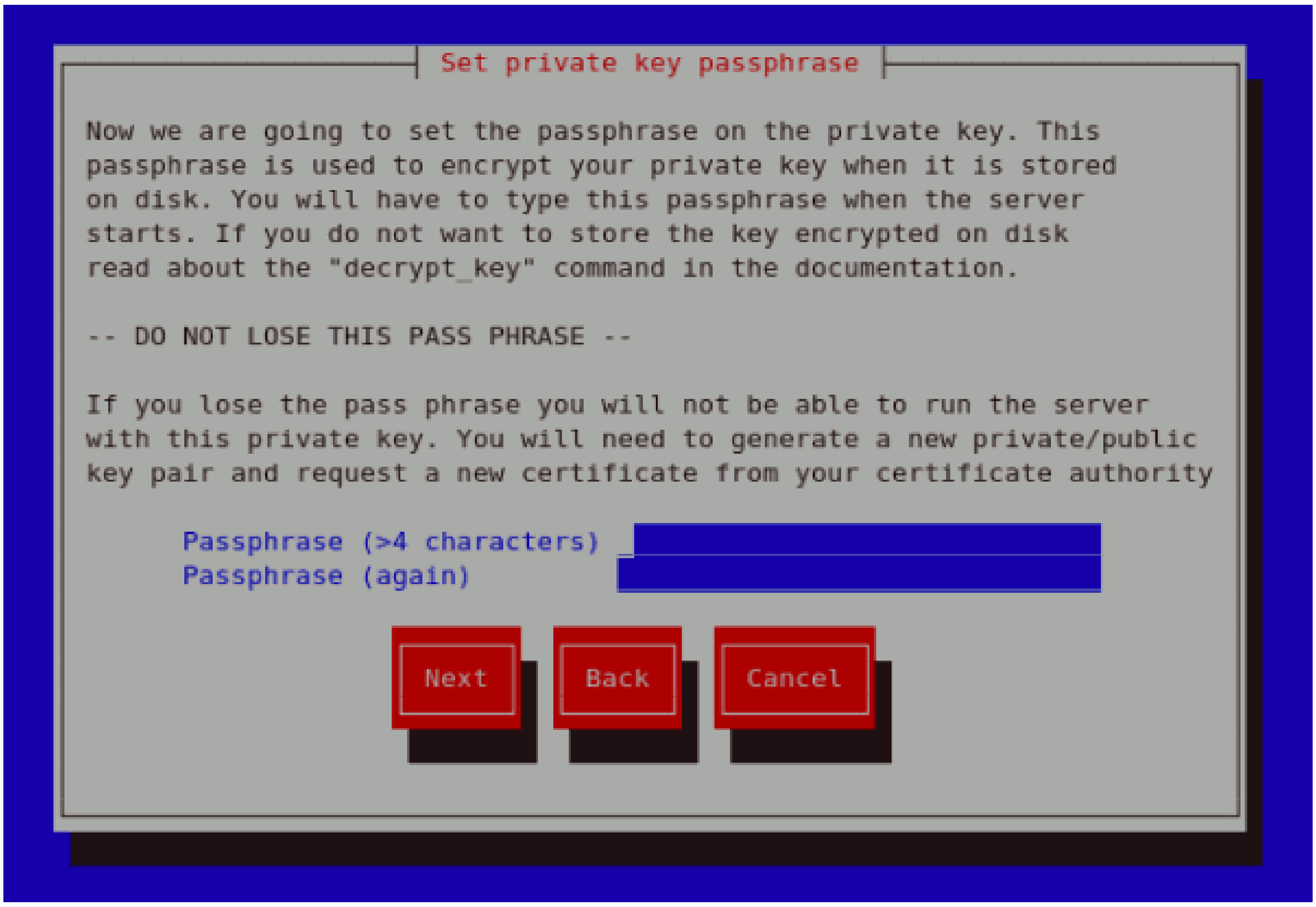
At this stage you can set the passphrase on your private key. If you set the passphrase you will have to enter it every time the server starts. The passphrase you use to encrypt your key must be the same for all the keys used by the same server installation.

If you do not encrypt your passphrase, if someone breaks into your server and grabs the file containing your key, they will be able to decrypt all communications to and from the server that were negotiated using that key. If your passphrase is encrypted it would be much more work for someone to retrieve the private key.

[ ] Encrypt the private key

Figure 25.19. Protecting your private key

The next screen allows you to set your key passphrase. Please do not lose this passphrase as you will not be able to run the server without it. You will need to regenerate a new private or public key pair and request a new certificate from your CA as indicated. For security, the passphrase is not displayed as you type. On typing your preferred passphrase, select **`Next`** to go back to your terminal.



```
Set private key passphrase

Now we are going to set the passphrase on the private key. This
passphrase is used to encrypt your private key when it is stored
on disk. You will have to type this passphrase when the server
starts. If you do not want to store the key encrypted on disk
read about the "decrypt_key" command in the documentation.

-- DO NOT LOSE THIS PASS PHRASE --

If you lose the pass phrase you will not be able to run the server
with this private key. You will need to generate a new private/public
key pair and request a new certificate from your certificate authority

      Passphrase (>4 characters) _
      Passphrase (again)
```

Figure 25.20. Set passphrase

If you attempt to run **`genkey www.example.com`** on a server that already has an existing key pair for the particular hostname, an error message will be displayed as illustrated below. You need to delete your existing key file as indicated to generate a new key pair.

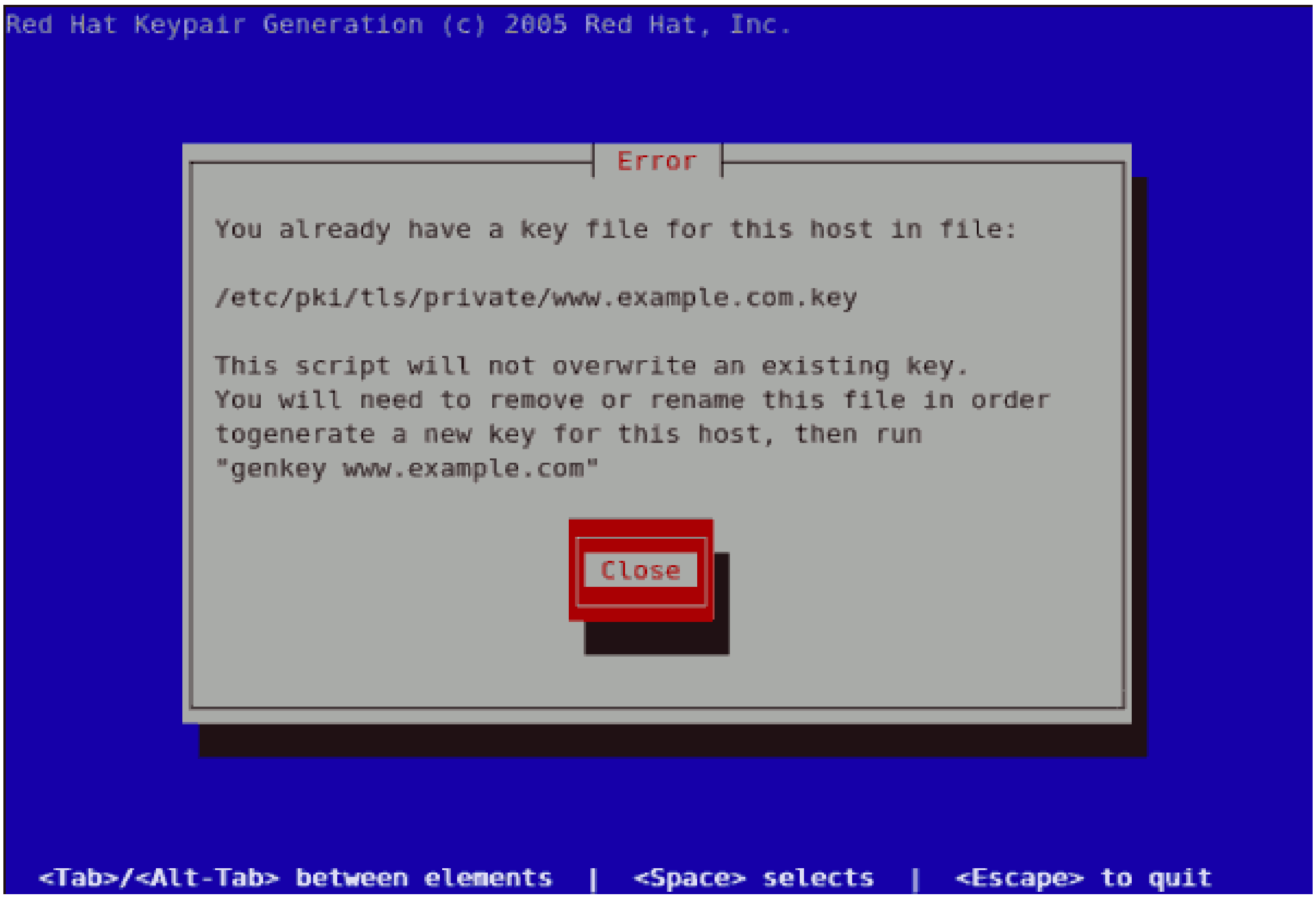

```
Red Hat Keypair Generation (c) 2005 Red Hat, Inc.

Error

You already have a key file for this host in file:

/etc/pki/tls/private/www.example.com.key

This script will not overwrite an existing key.
You will need to remove or rename this file in order
togenerate a new key for this host, then run
"genkey www.example.com"
```
Close
<Tab>/<Alt-Tab> between elements | <Space> selects | <Escape> to quit

Figure 25.21. genkey error

- http://httpd.apache.org/docs/2.2/ssl/
- http://httpd.apache.org/docs/2.2/vhosts/

25.8.6. How to configure the server to use the new key

The steps to configure the Apache HTTP Server to use the new key are:

- Obtain the signed certificate from the CA after submitting the CSR.
- Copy the certificate to the path, for example **`/etc/pki/tls/certs/www.example.com.crt`**
- Edit **`/etc/httpd/conf.d/ssl.conf`**. Change the SSLCertificateFile and SSLCertificateKey lines to be.

  ```
  SSLCertificateFile /etc/pki/tls/certs/www.example.com.crt
  SSLCertificateKeyFile /etc/pki/tls/private/www.example.com.key
  ```

  Note that the “www.example.com” part should match the argument passed on the **`genkey`** command.

25.9. Additional Resources

To learn more about the Apache HTTP Server, refer to the following resources.

25.9.1. Useful Websites

- http://httpd.apache.org/ — The official website for the Apache HTTP Server with documentation on all the directives and default modules.
- http://www.modssl.org/ — The official website for **`mod_ssl`**.
- http://www.apacheweek.com/ — A comprehensive online weekly newsletter about all things Apache.

Chapter 26. FTP

File Transfer Protocol (FTP) is one of the oldest and most commonly used protocols found on the Internet today. Its purpose is to reliably transfer files between computer hosts on a network without requiring the user to log directly into the remote host or have knowledge of how to use the remote system. It allows users to access files on remote systems using a standard set of simple commands.

This chapter outlines the basics of the FTP protocol, as well as configuration options for the primary FTP server shipped with Red Hat Enterprise Linux, **`vsftpd`**.

26.1. The File Transfer Protocol

FTP uses a client server architecture to transfer files using the TCP network protocol. Because FTP is an older protocol, it uses unencrypted username and password authentication. For this reason, it is considered an insecure protocol and should not be used unless absolutely necessary. A good substitute for FTP is **`sftp`** from the OpenSSH suite of tools. For information about configuring OpenSSH, refer to Chapter 20, *OpenSSH*.

However, because FTP is so prevalent on the Internet, it is often required to share files to the public. System administrators, therefore, should be aware of the FTP protocol's unique characteristics.

26.1.1. Multiple Ports, Multiple Modes

Unlike most protocols used on the Internet, FTP requires multiple network ports to work properly. When an FTP client application initiates a connection to an FTP server, it opens port 21 on the server — known as the *command port*. This port is used to issue all commands to the server. Any data requested from the server is returned to the client via a *data port*. The port number for data connections, and the way in which data connections are initialized, vary depending upon whether the client requests the data in *active* or *passive* mode.

The following defines these modes:

active mode

Active mode is the original method used by the FTP protocol for transferring data to the client application. When an active mode data transfer is initiated by the FTP client, the server opens a connection from port 20 on the server to the IP address and a random, unprivileged port (greater than 1024) specified by the client. This arrangement means that the client machine must be allowed to accept connections over any port above 1024. With the growth of insecure networks, such as the Internet, the use of firewalls to protect client machines is now prevalent. Because these client-side firewalls often deny incoming connections from active mode FTP servers, passive mode was devised.

passive mode

Passive mode, like active mode, is initiated by the FTP client application. When requesting data from the server, the FTP client indicates it wants to access the data in passive mode and the server provides the IP address and a random, unprivileged port (greater than 1024) on the server. The client then connects to that port on the server to download the requested information.

While passive mode resolves issues for client-side firewall interference with data connections, it can complicate administration of the server-side firewall. You can reduce the number of open ports on a server by limiting the range of unprivileged ports on the FTP server. This also simplifies the process of configuring firewall rules for the server. Refer to Section 26.2.5.8, “Network Options” for more about limiting passive ports.

26.2. FTP Servers

Red Hat Enterprise Linux ships with two different FTP servers:

- **Red Hat Content Accelerator** — A kernel-based Web server that delivers high performance Web server and FTP services. Since speed as its primary design goal, it has limited functionality and runs only as an anonymous FTP server. For more information about configuring and administering **Red Hat Content Accelerator**, consult the documentation available online at http://www.redhat.com/docs/manuals/tux/.

- **`vsftpd`** — A fast, secure FTP daemon which is the preferred FTP server for Red Hat Enterprise Linux. The remainder of this chapter focuses on **`vsftpd`**.

26.2.1. `vsftpd`

The Very Secure FTP Daemon (**`vsftpd`**) is designed from the ground up to be fast, stable, and, most importantly, secure. **`vsftpd`** is the only stand-alone FTP server distributed with Red Hat Enterprise Linux, due to its ability to handle large numbers of connections efficiently and securely.

The security model used by **`vsftpd`** has three primary aspects:

- *Strong separation of privileged and non-privileged processes* — Separate processes handle different tasks, and each of these processes run with the minimal privileges required for the task.

- *Tasks requiring elevated privileges are handled by processes with the minimal privilege necessary* — By leveraging compatibilities found in the **`libcap`** library, tasks that usually require full root privileges can be executed more safely from a less privileged process.

- *Most processes run in a **`chroot`** jail* — Whenever possible, processes are change-rooted to the directory being shared; this directory is then considered a **`chroot`** jail. For example, if the directory **`/var/ftp/`** is the primary shared directory, **`vsftpd`** reassigns **`/var/ftp/`** to the new root directory, known as **`/`**. This disallows any potential malicious hacker activities for any directories not contained below the new root directory.

Use of these security practices has the following effect on how **`vsftpd`** deals with requests:

- *The parent process runs with the least privileges required* — The parent process dynamically calculates the level of privileges it requires to minimize the level of risk. Child processes handle direct interaction with the FTP clients and run with as close to no privileges as possible.

- *All operations requiring elevated privileges are handled by a small parent process* — Much like the Apache HTTP Server, **`vsftpd`** launches unprivileged child processes to handle incoming connections. This allows the privileged, parent process to be as small as possible and handle relatively few tasks.

- *All requests from unprivileged child processes are distrusted by the parent process* — Communication with child processes are received over a socket, and the validity of any information from child processes is checked before being acted on.

- *Most interaction with FTP clients is handled by unprivileged child processes in a **`chroot`** jail* — Because these child processes are unprivileged and only have access to the directory being shared, any crashed processes only allows the attacker access to the shared files.

26.2.2. Files Installed with `vsftpd`

The **`vsftpd`** RPM installs the daemon (**`/usr/sbin/vsftpd`**), its configuration and related files, as well as FTP directories onto the system. The following lists the files and directories related to **`vsftpd`** configuration:

- **`/etc/rc.d/init.d/vsftpd`** — The *initialization script* (*initscript*) used by the **`/sbin/service`** command to start, stop, or reload **`vsftpd`**. Refer to Section 26.2.3, "Starting and Stopping **`vsftpd`**" for more information about using this script.
- **`/etc/pam.d/vsftpd`** — The Pluggable Authentication Modules (PAM) configuration file for **`vsftpd`**. This file specifies the requirements a user must meet to login to the FTP server. For more information, refer to Section 48.4, "Pluggable Authentication Modules (PAM)".
- **`/etc/vsftpd/vsftpd.conf`** — The configuration file for **`vsftpd`**. Refer to Section 26.2.5, "**`vsftpd`** Configuration Options" for a list of important options contained within this file.
- **`/etc/vsftpd.ftpusers`** — A list of users not allowed to log into **`vsftpd`**. By default, this list includes the **`root`**, **`bin`**, and **`daemon`** users, among others.
- **`/etc/vsftpd.user_list`** — This file can be configured to either deny or allow access to the users listed, depending on whether the **`userlist_deny`** directive is set to **`YES`** (default) or **`NO`** in **`/etc/vsftpd/vsftpd.conf`**. If **`/etc/vsftpd.user_list`** is used to grant access to users, the usernames listed must *not* appear in **`/etc/vsftpd.ftpusers`**.
- **`/var/ftp/`** — The directory containing files served by **`vsftpd`**. It also contains the **`/var/ftp/pub/`** directory for anonymous users. Both directories are world-readable, but writable only by the root user.

26.2.3. Starting and Stopping `vsftpd`

The **`vsftpd`** RPM installs the **`/etc/rc.d/init.d/vsftpd`** script, which can be accessed using the **`/sbin/service`** command.

To start the server, as root type:

```
service vsftpd start
```

To stop the server, as root type:

```
service vsftpd stop
```

The **`restart`** option is a shorthand way of stopping and then starting **`vsftpd`**. This is the most efficient way to make configuration changes take effect after editing the configuration file for **`vsftpd`**.

To restart the server, as root type:

```
service vsftpd restart
```

The **`condrestart`** (*conditional restart*) option only starts **`vsftpd`** if it is currently running. This option is useful for scripts, because it does not start the daemon if it is not running.

To conditionally restart the server, as root type:

```
service vsftpd condrestart
```

By default, the **`vsftpd`** service does *not* start automatically at boot time. To configure the **`vsftpd`** service to start at boot time, use an initscript utility, such as **`/sbin/chkconfig`**, **/usr/sbin/ntsysv**, or the **Services Configuration Tool** program. Refer to Chapter 18, *Controlling Access to Services* for more information regarding these tools.

26.2.3.1. Starting Multiple Copies of`vsftpd`

Sometimes one computer is used to serve multiple FTP domains. This is a technique called *multihoming*. One way to multihome using **`vsftpd`** is by running multiple copies of the daemon, each with its own configuration file.

To do this, first assign all relevant IP addresses to network devices or alias network devices on the system. Refer to Chapter 17, *Network Configuration* for more information about configuring network devices and device aliases. Additional information can be found about network configuration scripts in Chapter 16, *Network Interfaces*.

Next, the DNS server for the FTP domains must be configured to reference the correct machine. For information about BIND and its configuration files, refer to Chapter 19, *Berkeley Internet Name Domain (BIND)*.

For **`vsftpd`** to answer requests on different IP addresses, multiple copies of the daemon must be running. The first copy must be run using the **`vsftpd`** initscripts, as outlined in Section 26.2.3, "Starting and Stopping **`vsftpd`**". This copy uses the standard configuration file, **`/etc/vsftpd/vsftpd.conf`**.

Each additional FTP site must have a configuration file with a unique name in the **`/etc/vsftpd/`** directory, such as **`/etc/vsftpd/vsftpd-site-2.conf`**. Each configuration file must be readable and writable only by root. Within each configuration file for each FTP server listening on an IPv4 network, the following directive must be unique:

```
listen_address=N.N.N.N
```

Replace *N.N.N.N* with the *unique* IP address for the FTP site being served. If the site is using IPv6, use the **`listen_address6`** directive instead.

Once each additional server has a configuration file, the **`vsftpd`** daemon must be launched from a root shell prompt using the following command:

```
vsftpd /etc/vsftpd/<configuration-file> [amp   ]
```

In the above command, replace *<configuration-file>* with the unique name for the server's configuration file, such as **`/etc/vsftpd/vsftpd-site-2.conf`**.

Other directives to consider altering on a per-server basis are:

- **`anon_root`**
- **`local_root`**
- **`vsftpd_log_file`**
- **`xferlog_file`**

For a detailed list of directives available within **`vsftpd`**'s configuration file, refer to Section 26.2.5, "**`vsftpd`** Configuration Options".

To configure any additional servers to start automatically at boot time, add the above command to the end of the **`/etc/rc.local`** file.

26.2.4. Encrypting vsftpd Connections Using TLS

In order to counter the inherently insecure nature of **FTP**, which transmits user names, passwords, and data without encryption by default, the **vsftpd** daemon can be configured to utilize the **TLS** protocol to authenticate connections and encrypt all transfers. Note that an **FTP** client that supports **TLS** is needed to communicate with **vsftpd** with **TLS** enabled.

Note

SSL (Secure Sockets Layer) is the name of an older implementation of the security protocol. The new versions are called **TLS** (Transport Layer Security). Only the newer versions (**TLS**) should be used as **SSL** suffers from serious security vulnerabilities. The documentation included with the **vsftpd** server, as well as the configuration directives used in the **vsftpd.conf** file, use the **SSL** name when referring to security-related matters, but **TLS** is supported and used by default when the **ssl_enable** directive is set to **YES**.

Set the **ssl_enable** configuration directive in the **vsftpd.conf** file to **YES** to turn on **TLS** support. The default settings of other **TLS**-related directives that become automatically active when the **ssl_enable** option is enabled provide for a reasonably well-configured **TLS** set up. This includes, among other things, the requirement to only use the **TLS** v1 protocol for all connections (the use of the insecure **SSL** protocol versions is disabled by default) or forcing all non-anonymous logins to use **TLS** for sending passwords and data transfers.

Example 26.1. Configuring vsftpd to Use TLS

In this example, the configuration directives explicitly disable the older **SSL** versions of the security protocol in the **vsftpd.conf** file:

```
ssl_enable=YES
ssl_tlsv1=YES
ssl_sslv2=NO
ssl_sslv3=NO
```

Restart the **vsftpd** service after you modify its configuration:

```
~]# service vsftpd restart
```

See the vsftpd.conf(5) manual page for other **TLS**-related configuration directives for fine-tuning the use of **TLS** by **vsftpd**. Also, see Section 26.2.5, “**vsftpd** Configuration Options” for a description of other commonly used **vsftpd.conf** configuration directives.

26.2.5. vsftpd Configuration Options

Although **vsftpd** may not offer the level of customization other widely available FTP servers have, it offers enough options to fill most administrator's needs. The fact that it is not overly feature-laden limits configuration and programmatic errors.

All configuration of **vsftpd** is handled by its configuration file, **/etc/vsftpd/vsftpd.conf**. Each directive is on its own line within the file and follows the following format:

```
<directive>=<value>
```

For each directive, replace <*directive*> with a valid directive and <*value*> with a valid value.

Important

There must not be any spaces between the <*directive*>, equal symbol, and the <*value*> in a directive.

Comment lines must be preceded by a hash mark (**#**) and are ignored by the daemon.

For a complete list of all directives available, refer to the man page for **vsftpd.conf**.

Important

For an overview of ways to secure **vsftpd**, refer to Section 48.2, "Server Security".

The following is a list of some of the more important directives within **/etc/vsftpd/vsftpd.conf**. All directives not explicitly found within **vsftpd**'s configuration file are set to their default value.

26.2.5.1. Daemon Options

The following is a list of directives which control the overall behavior of the **vsftpd** daemon.

- **listen** — When enabled, **vsftpd** runs in stand-alone mode. Red Hat Enterprise Linux sets this value to **YES**. This directive cannot be used in conjunction with the **listen_ipv6** directive.

  The default value is **NO**.

- **listen_ipv6** — When enabled, **vsftpd** runs in stand-alone mode, but listens only to IPv6 sockets. This directive cannot be used in conjunction with the **listen** directive.

  The default value is **NO**.

- **session_support** — When enabled, **vsftpd** attempts to maintain login sessions for each user through Pluggable Authentication Modules (PAM). Refer to Section 48.4, "Pluggable Authentication Modules (PAM)" for more information. If session logging is not necessary, disabling this option allows**vsftpd** to run with less processes and lower privileges.

  The default value is **YES**.

26.2.5.2. Log In Options and Access Controls

The following is a list of directives which control the login behavior and access control mechanisms.

- **anonymous_enable** — When enabled, anonymous users are allowed to log in. The usernames **anonymous** and **ftp** are accepted.

  The default value is **YES**.

  Refer to Section 26.2.5.3, "Anonymous User Options" for a list of directives affecting anonymous users.

- **banned_email_file** — If the **deny_email_enable** directive is set to **YES**, this directive specifies the file containing a list of anonymous email passwords which are not permitted access to the server.

  The default value is **/etc/vsftpd.banned_emails**.

- **`banner_file`** — Specifies the file containing text displayed when a connection is established to the server. This option overrides any text specified in the **`ftpd_banner`** directive.

  There is no default value for this directive.

- **`cmds_allowed`** — Specifies a comma-delimited list of FTP commands allowed by the server. All other commands are rejected.

  There is no default value for this directive.

- **`deny_email_enable`** — When enabled, any anonymous user utilizing email passwords specified in the **`/etc/vsftpd.banned_emails`** are denied access to the server. The name of the file referenced by this directive can be specified using the **`banned_email_file`** directive.

  The default value is **`NO`**.

- **`ftpd_banner`** — When enabled, the string specified within this directive is displayed when a connection is established to the server. This option can be overridden by the **`banner_file`** directive.

  By default **`vsftpd`** displays its standard banner.

- **`local_enable`** — When enabled, local users are allowed to log into the system.

  The default value is **`YES`**.

  Refer to Section 26.2.5.4, “Local User Options” for a list of directives affecting local users.

- **`pam_service_name`** — Specifies the PAM service name for **`vsftpd`**.

  The default value is **`ftp`**. On Red Hat Enterprise Linux 5.10, this option is set to **`vsftpd`** in the configuration file.

- **`tcp_wrappers`** — When enabled, TCP wrappers are used to grant access to the server. If the FTP server is configured on multiple IP addresses, the **`VSFTPD_LOAD_CONF`** option can be used to load different configuration files based on the IP address being requested by the client.

  The default value is **`NO`**. On Red Hat Enterprise Linux 5.10, this option is set to **`YES`** in the configuration file.

  Refer to Section 48.5, “TCP Wrappers and xinetd” for more information about TCP wrappers.

- **`userlist_deny`** — When used in conjunction with the **`userlist_enable`** directive and set to **`NO`**, all local users are denied access unless the username is listed in the file specified by the **`userlist_file`** directive. Because access is denied before the client is asked for a password, setting this directive to **`NO`** prevents local users from submitting unencrypted passwords over the network.

  The default value is **`YES`**.

- **`userlist_enable`** — When enabled, the users listed in the file specified by the **`userlist_file`** directive are denied access. Because access is denied before the client is asked for a password, users are prevented from submitting unencrypted passwords over the network.

  The default value is **`NO`**. On Red Hat Enterprise Linux 5.10, this option is set to **`YES`** in the configuration file.

- **`userlist_file`** — Specifies the file referenced by **`vsftpd`** when the **`userlist_enable`** directive is enabled.

  The default value is **`/etc/vsftpd.user_list`** and is created during installation.

26.2.5.3. Anonymous User Options

The following lists directives which control anonymous user access to the server. To use these options, the **`anonymous_enable`** directive must be set to **`YES`**.

- **`anon_mkdir_write_enable`** — When enabled in conjunction with the **`write_enable`** directive, anonymous users are allowed to create new directories within a parent directory which has write permissions.

  The default value is **`NO`**.

- **`anon_root`** — Specifies the directory **`vsftpd`** changes to after an anonymous user logs in.

  There is no default value for this directive.

- **`anon_upload_enable`** — When enabled in conjunction with the **`write_enable`** directive, anonymous users are allowed to upload files within a parent directory which has write permissions.

  The default value is **`NO`**.

- **`anon_world_readable_only`** — When enabled, anonymous users are only allowed to download world-readable files.

  The default value is **`YES`**.

- **`ftp_username`** — Specifies the local user account (listed in **`/etc/passwd`**) used for the anonymous FTP user. The home directory specified in **`/etc/passwd`** for the user is the root directory of the anonymous FTP user.

  The default value is **`ftp`**.

- **`no_anon_password`** — When enabled, the anonymous user is not asked for a password.

  The default value is **`NO`**.

- **`secure_email_list_enable`** — When enabled, only a specified list of email passwords for anonymous logins are accepted. This is a convenient way to offer limited security to public content without the need for virtual users.

  Anonymous logins are prevented unless the password provided is listed in **`/etc/vsftpd.email_passwords`**. The file format is one password per line, with no trailing white spaces.

  The default value is **`NO`**.

26.2.5.4. Local User Options

The following lists directives which characterize the way local users access the server. To use these options, the **`local_enable`** directive must be set to **`YES`**.

- **`chmod_enable`** — When enabled, the FTP command **`SITE CHMOD`** is allowed for local users. This command allows the users to change the permissions on files.

  The default value is **`YES`**.

- **`chroot_list_enable`** — When enabled, the local users listed in the file specified in the **`chroot_list_file`** directive are placed in a **`chroot`** jail upon log in.

  If enabled in conjunction with the **`chroot_local_user`** directive, the local users listed in the file specified in the **chroot_list_file** directive are *not* placed in a **chroot** jail upon log in.

The default value is **NO**.

- **chroot_list_file** — Specifies the file containing a list of local users referenced when the **chroot_list_enable** directive is set to **YES**.

  The default value is **/etc/vsftpd.chroot_list**.

- **chroot_local_user** — When enabled, local users are change-rooted to their home directories after logging in.

  The default value is **NO**.

Warning

Enabling **chroot_local_user** opens up a number of security issues, especially for users with upload privileges. For this reason, it is *not* recommended.

- **guest_enable** — When enabled, all non-anonymous users are logged in as the user **guest**, which is the local user specified in the **guest_username** directive.

  The default value is **NO**.

- **guest_username** — Specifies the username the **guest** user is mapped to.

  The default value is **ftp**.

- **local_root** — Specifies the directory **vsftpd** changes to after a local user logs in.

  There is no default value for this directive.

- **local_umask** — Specifies the umask value for file creation. Note that the default value is in octal form (a numerical system with a base of eight), which includes a "0" prefix. Otherwise the value is treated as a base-10 integer.

  The default value is **022**.

- **passwd_chroot_enable** — When enabled in conjunction with the **chroot_local_user** directive, **vsftpd** change-roots local users based on the occurrence of the **/./** in the home directory field within **/etc/passwd**.

  The default value is **NO**.

- **user_config_dir** — Specifies the path to a directory containing configuration files bearing the name of local system users that contain specific setting for that user. Any directive in the user's configuration file overrides those found in **/etc/vsftpd/vsftpd.conf**.

  There is no default value for this directive.

26.2.5.5. Directory Options

The following lists directives which affect directories.

- **dirlist_enable** — When enabled, users are allowed to view directory lists.

The default value is **YES**.

- **dirmessage_enable** — When enabled, a message is displayed whenever a user enters a directory with a message file. This message resides within the current directory. The name of this file is specified in the **message_file** directive and is **.message** by default.

  The default value is **NO**. On Red Hat Enterprise Linux 5.10, this option is set to **YES** in the configuration file.

- **force_dot_files** — When enabled, files beginning with a dot (**.**) are listed in directory listings, with the exception of the **.** and **..** files.

  The default value is **NO**.

- **hide_ids** — When enabled, all directory listings show **ftp** as the user and group for each file.

  The default value is **NO**.

- **message_file** — Specifies the name of the message file when using the **dirmessage_enable** directive.

  The default value is **.message**.

- **text_userdb_names** — When enabled, text usernames and group names are used in place of UID and GID entries. Enabling this option may slow performance of the server.

  The default value is **NO**.

- **use_localtime** — When enabled, directory listings reveal the local time for the computer instead of GMT.

  The default value is **NO**.

26.2.5.6. File Transfer Options

The following lists directives which affect directories.

- **download_enable** — When enabled, file downloads are permitted.

  The default value is **YES**.

- **chown_uploads** — When enabled, all files uploaded by anonymous users are owned by the user specified in the **chown_username** directive.

  The default value is **NO**.

- **chown_username** — Specifies the ownership of anonymously uploaded files if the **chown_uploads** directive is enabled.

  The default value is **root**.

- **write_enable** — When enabled, FTP commands which can change the file system are allowed, such as **DELE**, **RNFR**, and **STOR**.

  The default value is **YES**.

26.2.5.7. Logging Options

The following lists directives which affect **vsftpd**'s logging behavior.

- **dual_log_enable** — When enabled in conjunction with **xferlog_enable**, **vsftpd** writes two files simultaneously: a **wu-ftpd**-compatible log to the file specified in the **xferlog_file** directive (**/var/log/xferlog** by default) and a standard **vsftpd** log file specified in the **vsftpd_log_file** directive (**/var/log/vsftpd.log** by default).

  The default value is **NO**.

- **log_ftp_protocol** — When enabled in conjunction with **xferlog_enable** and with **xferlog_std_format** set to **NO**, all FTP commands and responses are logged. This directive is useful for debugging.

  The default value is **NO**.

- **syslog_enable** — When enabled in conjunction with **xferlog_enable**, all logging normally written to the standard **vsftpd** log file specified in the **vsftpd_log_file** directive (**/var/log/vsftpd.log** by default) is sent to the system logger instead under the FTPD facility.

  The default value is **NO**.

- **vsftpd_log_file** — Specifies the **vsftpd** log file. For this file to be used, **xferlog_enable** must be enabled and **xferlog_std_format** must either be set to **NO** or, if **xferlog_std_format** is set to **YES**, **dual_log_enable** must be enabled. It is important to note that if **syslog_enable** is set to **YES**, the system log is used instead of the file specified in this directive.

  The default value is **/var/log/vsftpd.log**.

- **xferlog_enable** — When enabled, **vsftpd** logs connections (**vsftpd** format only) and file transfer information to the log file specified in the **vsftpd_log_file** directive (**/var/log/vsftpd.log** by default). If **xferlog_std_format** is set to **YES**, file transfer information is logged but connections are not, and the log file specified in **xferlog_file** (**/var/log/xferlog** by default) is used instead. It is important to note that both log files and log formats are used if **dual_log_enable** is set to **YES**.

  The default value is **NO**. On Red Hat Enterprise Linux 5.10, this option is set to **YES** in the configuration file.

- **xferlog_file** — Specifies the **wu-ftpd**-compatible log file. For this file to be used, **xferlog_enable** must be enabled and **xferlog_std_format** must be set to **YES**. It is also used if **dual_log_enable** is set to **YES**.

  The default value is **/var/log/xferlog**.

- **xferlog_std_format** — When enabled in conjunction with **xferlog_enable**, only a **wu-ftpd**-compatible file transfer log is written to the file specified in the **xferlog_file** directive (**/var/log/xferlog** by default). It is important to note that this file only logs file transfers and does not log connections to the server.

  The default value is **NO**. On Red Hat Enterprise Linux 5.10, this option is set to **YES** in the configuration file.

Important

To maintain compatibility with log files written by the older **`wu-ftpd`** FTP server, the **`xferlog_std_format`** directive is set to **`YES`** under Red Hat Enterprise Linux. However, this setting means that connections to the server are not logged.

To both log connections in **`vsftpd`** format and maintain a **`wu-ftpd`**-compatible file transfer log, set **`dual_log_enable`** to **`YES`**.

If maintaining a **`wu-ftpd`**-compatible file transfer log is not important, either set **`xferlog_std_format`** to **`NO`**, comment the line with a hash mark (**`#`**), or delete the line entirely.

26.2.5.8. Network Options

The following lists directives which affect how **`vsftpd`** interacts with the network.

- **`accept_timeout`** — Specifies the amount of time for a client using passive mode to establish a connection.

  The default value is **`60`**.

- **`anon_max_rate`** — Specifies the maximum data transfer rate for anonymous users in bytes per second.

  The default value is **`0`**, which does not limit the transfer rate.

- **`connect_from_port_20`** When enabled, **`vsftpd`** runs with enough privileges to open port 20 on the server during active mode data transfers. Disabling this option allows **`vsftpd`** to run with less privileges, but may be incompatible with some FTP clients.

  The default value is **`NO`**. On Red Hat Enterprise Linux 5.10, this option is set to **`YES`** in the configuration file.

- **`connect_timeout`** — Specifies the maximum amount of time a client using active mode has to respond to a data connection, in seconds.

  The default value is **`60`**.

- **`data_connection_timeout`** — Specifies maximum amount of time data transfers are allowed to stall, in seconds. Once triggered, the connection to the remote client is closed.

  The default value is **`300`**.

- **`ftp_data_port`** — Specifies the port used for active data connections when **`connect_from_port_20`** is set to **`YES`**.

  The default value is **`20`**.

- **`idle_session_timeout`** — Specifies the maximum amount of time between commands from a remote client. Once triggered, the connection to the remote client is closed.

  The default value is **`300`**.

- **`listen_address`** — Specifies the IP address on which **`vsftpd`** listens for network connections.

  There is no default value for this directive.

> **Note**
>
> If running multiple copies of **vsftpd** serving different IP addresses, the configuration file for each copy of the **vsftpd** daemon must have a different value for this directive. Refer to Section 26.2.3.1, "Starting Multiple Copies of **vsftpd**" for more information about multihomed FTP servers.

- **listen_address6** — Specifies the IPv6 address on which **vsftpd** listens for network connections when **listen_ipv6** is set to **YES**.

  There is no default value for this directive.

> **Note**
>
> If running multiple copies of **vsftpd** serving different IP addresses, the configuration file for each copy of the **vsftpd** daemon must have a different value for this directive. Refer to Section 26.2.3.1, "Starting Multiple Copies of **vsftpd**" for more information about multihomed FTP servers.

- **listen_port** — Specifies the port on which **vsftpd** listens for network connections.

  The default value is **21**.

- **local_max_rate** — Specifies the maximum rate data is transferred for local users logged into the server in bytes per second.

  The default value is **0**, which does not limit the transfer rate.

- **max_clients** — Specifies the maximum number of simultaneous clients allowed to connect to the server when it is running in standalone mode. Any additional client connections would result in an error message.

  The default value is **0**, which does not limit connections.

- **max_per_ip** — Specifies the maximum of clients allowed to connected from the same source IP address.

  The default value is **0**, which does not limit connections.

- **pasv_address** — Specifies the IP address for the public facing IP address of the server for servers behind Network Address Translation (NAT) firewalls. This enables **vsftpd** to hand out the correct return address for passive mode connections.

  There is no default value for this directive.

- **pasv_enable** — When enabled, passive mode connects are allowed.

  The default value is **YES**.

- **pasv_max_port** — Specifies the highest possible port sent to the FTP clients for passive mode connections. This setting is used to limit the port range so that firewall rules are easier to create.

  The default value is **0**, which does not limit the highest passive port range. The value must not exceed **65535**.

- **`pasv_min_port`** — Specifies the lowest possible port sent to the FTP clients for passive mode connections. This setting is used to limit the port range so that firewall rules are easier to create.

  The default value is **`0`**, which does not limit the lowest passive port range. The value must not be lower **`1024`**.

- **`pasv_promiscuous`** — When enabled, data connections are not checked to make sure they are originating from the same IP address. This setting is only useful for certain types of tunneling.

  The default value is **`NO`**.

  Warning

  Do not enable this option unless absolutely necessary as it disables an important security feature which verifies that passive mode connections originate from the same IP address as the control connection that initiates the data transfer.

- **`port_enable`** — When enabled, active mode connects are allowed.

  The default value is **`YES`**.

26.2.6. Additional Resources

For more information about **`vsftpd`**, refer to the following resources.

26.2.6.1. Installed Documentation

- The **`/usr/share/doc/vsftpd-<version-number>/`** directory — Replace <*version-number*> with the installed version of the **`vsftpd`** package. This directory contains a **`README`** with basic information about the software. The **`TUNING`** file contains basic performance tuning tips and the **`SECURITY/`** directory contains information about the security model employed by **`vsftpd`**.

- **`vsftpd`** related man pages — There are a number of man pages for the daemon and configuration files. The following lists some of the more important man pages.

  Server Applications

  - **`man vsftpd`** — Describes available command line options for **`vsftpd`**.

  Configuration Files

  - **`man vsftpd.conf`** — Contains a detailed list of options available within the configuration file for **`vsftpd`**.

  - **`man 5 hosts_access`** — Describes the format and options available within the TCP wrappers configuration files: **`hosts.allow`** and **`hosts.deny`**.

26.2.6.2. Useful Websites

- http://vsftpd.beasts.org/ — The **`vsftpd`** project page is a great place to locate the latest documentation and to contact the author of the software.

- http://slacksite.com/other/ftp.html — This website provides a concise explanation of the differences between active and passive mode FTP.

- http://www.ietf.org/rfc/rfc0959.txt — The original *Request for Comments* (*RFC*) of the FTP protocol from the IETF.

Chapter 27. Email

The birth of electronic mail (*email*) occurred in the early 1960s. The mailbox was a file in a user's home directory that was readable only by that user. Primitive mail applications appended new text messages to the bottom of the file, making the user wade through the constantly growing file to find any particular message. This system was only capable of sending messages to users on the same system.

The first network transfer of an electronic mail message file took place in 1971 when a computer engineer named Ray Tomlinson sent a test message between two machines via ARPANET — the precursor to the Internet. Communication via email soon became very popular, comprising 75 percent of ARPANET's traffic in less than two years.

Today, email systems based on standardized network protocols have evolved into some of the most widely used services on the Internet. Red Hat Enterprise Linux offers many advanced applications to serve and access email.

This chapter reviews modern email protocols in use today and some of the programs designed to send and receive email.

27.1. Email Protocols

Today, email is delivered using a client/server architecture. An email message is created using a mail client program. This program then sends the message to a server. The server then forwards the message to the recipient's email server, where the message is then supplied to the recipient's email client.

To enable this process, a variety of standard network protocols allow different machines, often running different operating systems and using different email programs, to send and receive email.

The following protocols discussed are the most commonly used in the transfer of email.

27.1.1. Mail Transport Protocols

Mail delivery from a client application to the server, and from an originating server to the destination server, is handled by the *Simple Mail Transfer Protocol* (*SMTP*).

27.1.1.1. SMTP

The primary purpose of SMTP is to transfer email between mail servers. However, it is critical for email clients as well. To send email, the client sends the message to an outgoing mail server, which in turn contacts the destination mail server for delivery. For this reason, it is necessary to specify an SMTP server when configuring an email client.

Under Red Hat Enterprise Linux, a user can configure an SMTP server on the local machine to handle mail delivery. However, it is also possible to configure remote SMTP servers for outgoing mail.

One important point to make about the SMTP protocol is that it does not require authentication. This allows anyone on the Internet to send email to anyone else or even to large groups of people. It is this characteristic of SMTP that makes junk email or *spam* possible. Imposing relay restrictions limits random users on the Internet from sending email through your SMTP server, to other servers on the internet. Servers that do not impose such restrictions are called *open relay* servers.

By default, Sendmail (**`/usr/sbin/sendmail`**) is the default SMTP program under Red Hat Enterprise Linux. However, a simpler mail server application called Postfix (**`/usr/sbin/postfix`**) is also available.

27.1.2. Mail Access Protocols

There are two primary protocols used by email client applications to retrieve email from mail servers: the *Post Office Protocol* (*POP*) and the *Internet Message Access Protocol* (*IMAP*).

27.1.2.1. POP

The default POP server under Red Hat Enterprise Linux is **`/usr/lib/cyrus-imapd/pop3d`** and is provided by the **`cyrus-imapd`** package. When using a POP server, email messages are downloaded by email client applications. By default, most POP email clients are automatically configured to delete the message on the email server after it has been successfully transferred, however this setting usually can be changed.

POP is fully compatible with important Internet messaging standards, such as *Multipurpose Internet Mail Extensions* (*MIME*), which allow for email attachments.

POP works best for users who have one system on which to read email. It also works well for users who do not have a persistent connection to the Internet or the network containing the mail server. Unfortunately for those with slow network connections, POP requires client programs upon authentication to download the entire content of each message. This can take a long time if any messages have large attachments.

The most current version of the standard POP protocol is POP3.

There are, however, a variety of lesser-used POP protocol variants:

- *APOP* — POP3 with MDS authentication. An encoded hash of the user's password is sent from the email client to the server rather then sending an unencrypted password.
- *KPOP* — POP3 with Kerberos authentication. Refer to Section 48.6, “Kerberos” for more information.
- *RPOP* — POP3 with RPOP authentication. This uses a per-user ID, similar to a password, to authenticate POP requests. However, this ID is not encrypted, so RPOP is no more secure than standard POP.

For added security, it is possible to use *Secure Socket Layer* (*SSL*) encryption for client authentication and data transfer sessions. This can be enabled by using the **`ipop3s`** service or by using the **`/usr/sbin/stunnel`** program. Refer to Section 27.6.1, “Securing Communication” for more information.

27.1.2.2. IMAP

The default IMAP server under Red Hat Enterprise Linux is **`/usr/lib/cyrus-imapd/imapd`** and is provided by the **`cyrus-imapd`** package. When using an IMAP mail server, email messages remain on the server where users can read or delete them. IMAP also allows client applications to create, rename, or delete mail directories on the server to organize and store email.

IMAP is particularly useful for those who access their email using multiple machines. The protocol is also convenient for users connecting to the mail server via a slow connection, because only the email header information is downloaded for messages until opened, saving bandwidth. The user also has the ability to delete messages without viewing or downloading them.

For convenience, IMAP client applications are capable of caching copies of messages locally, so the user can browse previously read messages when not directly connected to the IMAP server.

IMAP, like POP, is fully compatible with important Internet messaging standards, such as MIME, which allow for email attachments.

For added security, it is possible to use *SSL* encryption for client authentication and data transfer sessions. This can be enabled by using the **`imaps`** service, or by using the **`/usr/sbin/stunnel`** program. Refer to Section 27.6.1, “Securing Communication” for more information.

Other free, as well as commercial, IMAP clients and servers are available, many of which extend the IMAP protocol and provide additional functionality. A comprehensive list can be found online at http://www.imap.org/products/longlist.htm.

27.1.2.3. Dovecot

The **`imap-login`** and **`pop3-login`** daemons which implement the IMAP and POP3 protocols are included in the **`dovecot`** package. The use of IMAP and POP is configured through **`dovecot`**. By default **`dovecot`** listens on both IMAP and POP3 ports and also their secure variants. To start using **`dovecot`**:

1. Start the daemon:

   ```
   service dovecot start
   ```

2. Make the daemon start automatically after the next reboot:

   ```
   chkconfig dovecot on
   ```

   Note that **`dovecot`** only reports that it started the IMAP server, but also starts the POP3 server.

Unlike SMTP, both of these protocols require connecting clients to authenticate using a username and password. By default, passwords for both protocols are passed over the network unencrypted.

To configure SSL on dovecot:

- Edit the **`dovecot`** configuration file **`/etc/pki/dovecot/dovecot-openssl.cnf`** as you prefer. However in a typical installation, this file does not require modification.
- Rename, move or delete the files **`/etc/pki/dovecot/certs/dovecot.pem`** and **`/etc/pki/dovecot/private/dovecot.pem`**.
- Execute the **`/usr/share/doc/dovecot-1.0/examples/mkcert.sh`** script which creates the dovecot self signed certificates. The certificates are copied in the **`/etc/pki/dovecot/certs`** and **`/etc/pki/dovecot/private`** directories. To implement the changes, restart **`dovecot`** (**`/sbin/service dovecot restart`**).

More details on **`dovecot`** can be found online at http://www.dovecot.org.

Note

By default, **SSLv2** and **SSLv3** are disallowed by **Dovecot**. This is due to the POODLE SSL vulnerability (CVE-2014-3566). See Resolution for POODLE SSL 3.0 vulnerability (CVE-2014-3566) in Postfix and Dovecot for details.

27.2. Email Program Classifications

In general, all email applications fall into at least one of three classifications. Each classification plays a specific role in the process of moving and managing email messages. While most users are only aware of the specific email program they use to receive and send messages, each one is important for ensuring that email arrives at the correct destination.

27.2.1. Mail Transport Agent

A *Mail Transport Agent* (*MTA*) transports email messages between hosts using SMTP. A message may involve several MTAs as it moves to its intended destination.

While the delivery of messages between machines may seem rather straightforward, the entire process of deciding if a particular MTA can or should accept a message for delivery is quite complicated. In addition, due to problems from spam, use of a particular MTA is usually restricted by the MTA's configuration or the access configuration for the network on which the MTA resides.

Many modern email client programs can act as an MTA when sending email. However, this action should not be confused with the role of a true MTA. The sole reason email client programs are capable of sending email like an MTA is because the host running the application does not have its own MTA. This is particularly true for email client programs on non-UNIX-based operating systems. However, these client programs only send outbound messages to an MTA they are authorized to use and do not directly deliver the message to the intended recipient's email server.

Since Red Hat Enterprise Linux installs two MTAs, Sendmail and Postfix, email client programs are often not required to act as an MTA. Red Hat Enterprise Linux also includes a special purpose MTA called Fetchmail.

For more information on Sendmail, Postfix, and Fetchmail, refer to Section 27.3, "Mail Transport Agents".

27.2.2. Mail Delivery Agent

A *Mail Delivery Agent* (*MDA*) is invoked by the MTA to file incoming email in the proper user's mailbox. In many cases, the MDA is actually a *Local Delivery Agent* (*LDA*), such as **`mail`** or Procmail.

Any program that actually handles a message for delivery to the point where it can be read by an email client application can be considered an MDA. For this reason, some MTAs (such as Sendmail and Postfix) can fill the role of an MDA when they append new email messages to a local user's mail spool file. In general, MDAs do not transport messages between systems nor do they provide a user interface; MDAs distribute and sort messages on the local machine for an email client application to access.

27.2.3. Mail User Agent

A *Mail User Agent* (*MUA*) is synonymous with an email client application. An MUA is a program that, at the very least, allows a user to read and compose email messages. Many MUAs are capable of retrieving messages via the POP or IMAP protocols, setting up mailboxes to store messages, and sending outbound messages to an MTA.

MUAs may be graphical, such as Evolution, or have a very simple, text-based interface, such as **`mutt`**.

27.3. Mail Transport Agents

Red Hat Enterprise Linux includes two primary MTAs, Sendmail and Postfix. Sendmail is configured as the default MTA, although it is easy to switch the default MTA to Postfix.

27.3.1. Sendmail

Sendmail's core purpose, like other MTAs, is to safely transfer email among hosts, usually using the SMTP protocol. However, Sendmail is highly configurable, allowing control over almost every aspect of how email is handled, including the protocol used. Many system administrators elect to use Sendmail as their MTA due to its power and scalability.

27.3.1.1. Purpose and Limitations

It is important to be aware of what Sendmail is and what it can do, as opposed to what it is not. In these days of monolithic applications that fulfill multiple roles, Sendmail may seem like the only application needed to run an email server within an organization. Technically, this is true, as Sendmail can spool mail to each users' directory and deliver outbound mail for users. However, most users actually require much more than simple email delivery. Users usually want to interact with their email using an MUA, that uses POP or IMAP, to download their messages to their local machine. Or, they may prefer a Web interface to gain access to their mailbox. These other applications can work in conjunction with Sendmail, but they actually exist for different reasons and can operate separately from one another.

It is beyond the scope of this section to go into all that Sendmail should or could be configured to do. With literally hundreds of different options and rule sets, entire volumes have been dedicated to helping explain everything that can be done and how to fix things that go wrong. Refer to the Section 27.7, “Additional Resources” for a list of Sendmail resources.

This section reviews the files installed with Sendmail by default and reviews basic configuration changes, including how to stop unwanted email (spam) and how to extend Sendmail with the *Lightweight Directory Access Protocol (LDAP)*.

27.3.1.2. The Default Sendmail Installation

The Sendmail executable is **`/usr/sbin/sendmail`**.

Sendmail's lengthy and detailed configuration file is **`/etc/mail/sendmail.cf`**. Avoid editing the **`sendmail.cf`** file directly. To make configuration changes to Sendmail, edit the **`/etc/mail/sendmail.mc`** file, back up the original **`/etc/mail/sendmail.cf`**, and use the following alternatives to generate a new configuration file:

- Use the included makefile in **`/etc/mail`** (**`make all -C /etc/mail`**) to create a new **`/etc/mail/sendmail.cf`** configuration file. All other generated files in **`/etc/mail`** (db files) will be regenerated if needed. The old makemap commands are still usable. The make command will automatically be used by **`service sendmail start | restart | reload`** if the **`make`** package is installed.
- Alternatively you may use the included **`m4`** macro processor to create a new **`/etc/mail/sendmail.cf`**.

More information on configuring Sendmail can be found in Section 27.3.1.3, “Common Sendmail Configuration Changes”.

Various Sendmail configuration files are installed in the **`/etc/mail/`** directory including:

- **`access`** — Specifies which systems can use Sendmail for outbound email.
- **`domaintable`** — Specifies domain name mapping.
- **`local-host-names`** — Specifies aliases for the host.
- **`mailertable`** — Specifies instructions that override routing for particular domains.
- **`virtusertable`** — Specifies a domain-specific form of aliasing, allowing multiple virtual domains to be hosted on one machine.

Several of the configuration files in **`/etc/mail/`**, such as **`access`**, **`domaintable`**, **`mailertable`** and **`virtusertable`**, must actually store their information in database files before Sendmail can use any configuration changes. To include any changes made to these configurations in their database files, run the following command:

```
makemap hash /etc/mail/<name> < /etc/mail/<name>
```

where *<name>* is replaced with the name of the configuration file to convert.

For example, to have all emails addressed to the **`example.com`** domain delivered to `bob@other-example.com`, add the following line to the **`virtusertable`** file:

```
@example.com     bob@other-example.com
```

To finalize the change, the **`virtusertable.db`** file must be updated using the following command as root:

```
makemap hash /etc/mail/virtusertable < /etc/mail/virtusertable
```

This creates an updated **`virtusertable.db`** file containing the new configuration.

27.3.1.3. Common Sendmail Configuration Changes

When altering the Sendmail configuration file, it is best not to edit an existing file, but to generate an entirely new **`/etc/mail/sendmail.cf`** file.

Warning

Before changing the **`sendmail.cf`** file, it is a good idea to create a backup copy.

To add the desired functionality to Sendmail, edit the **`/etc/mail/sendmail.mc`** file as the root user. When finished, use the **`m4`** macro processor to generate a new **`sendmail.cf`** by executing the following command:

```
m4 /etc/mail/sendmail.mc > /etc/mail/sendmail.cf
```

By default, the **`m4`** macro processor is installed with Sendmail but is part of the **`m4`** package.

After creating a new **`/etc/mail/sendmail.cf`** file, restart Sendmail for the changes to take effect. The easiest way to do this is to type the following command:

```
service sendmail restart
```

Important

The default **`sendmail.cf`** file does not allow Sendmail to accept network connections from any host other than the local computer. To configure Sendmail as a server for other clients, edit the **`/etc/mail/sendmail.mc`** file, and either change the address specified in the **`Addr=`** option of the **`DAEMON_OPTIONS`** directive from **`127.0.0.1`** to the IP address of an active network device or comment out the **`DAEMON_OPTIONS`** directive all together by placing **`dnl`** at the beginning of the line. When finished, regenerate **`/etc/mail/sendmail.cf`** by executing the following command:

```
m4 /etc/mail/sendmail.mc > /etc/mail/sendmail.cf
```

The default configuration which ships with Red Hat Enterprise Linux works for most SMTP-only sites. However, it does not work for UUCP (UNIX to UNIX Copy) sites. If using UUCP mail transfers, the **`/etc/mail/sendmail.mc`** file must be reconfigured and a new **`/etc/mail/sendmail.cf`** must be generated.

Consult the **/usr/share/sendmail-cf/README** file before editing any files in the directories under the **/usr/share/sendmail-cf** directory, as they can affect the future configuration of **/etc/mail/sendmail.cf** files.

27.3.1.4. Masquerading

One common Sendmail configuration is to have a single machine act as a mail gateway for all machines on the network. For instance, a company may want to have a machine called **mail.example.com** that handles all of their email and assigns a consistent return address to all outgoing mail.

In this situation, the Sendmail server must masquerade the machine names on the company network so that their return address is **user@example.com** instead of **user@host.example.com**.

To do this, add the following lines to **/etc/mail/sendmail.mc**:

```
FEATURE(always_add_domain)dnl
FEATURE(`masquerade_entire_domain')dnl
FEATURE(`masquerade_envelope')dnl
FEATURE(`allmasquerade')dnl
MASQUERADE_AS(`bigcorp.com.')dnl
MASQUERADE_DOMAIN(`bigcorp.com.')dnl
MASQUERADE_AS(bigcorp.com)dnl
```

After generating a new **sendmail.cf** using **m4**, this configuration makes all mail from inside the network appear as if it were sent from **bigcorp.com**.

27.3.1.5. Stopping Spam

Email spam can be defined as unnecessary and unwanted email received by a user who never requested the communication. It is a disruptive, costly, and widespread abuse of Internet communication standards.

Sendmail makes it relatively easy to block new spamming techniques being employed to send junk email. It even blocks many of the more usual spamming methods by default. Main anti-spam features available in sendmail are *header checks, relaying denial (default from version 8.9), access database and sender information checks.*

For example, forwarding of SMTP messages, also called relaying, has been disabled by default since Sendmail version 8.9. Before this change occurred, Sendmail directed the mail host (**x.edu**) to accept messages from one party (**y.com**) and sent them to a different party (**z.net**). Now, however, Sendmail must be configured to permit any domain to relay mail through the server. To configure relay domains, edit the **/etc/mail/relay-domains** file and restart Sendmail.

However, many times users are bombarded with spam from other servers throughout the Internet. In these instances, Sendmail's access control features available through the **/etc/mail/access** file can be used to prevent connections from unwanted hosts. The following example illustrates how this file can be used to both block and specifically allow access to the Sendmail server:

```
badspammer.com          ERROR:550 "Go away and do not spam us anymore"
tux.badspammer.com      OK
10.0                    RELAY
```

This example shows that any email sent from **badspammer.com** is blocked with a 550 RFC-821 compliant error code, with a message sent back to the spammer. Email sent from the **tux.badspammer.com** sub-domain, is accepted. The last line shows that any email sent from the 10.0.*.* network can be relayed through the mail server.

Because **/etc/mail/access.db** is a database, use **makemap** to activate any changes. Do this using the following command as root:

```
makemap hash /etc/mail/access < /etc/mail/access
```

Message header analysis allows you to reject mail based on header contents. SMTP servers store information about an emails journey in the message header. As the message travels from one MTA to another, each puts in a "Received" header above all the other Received headers. It is however important to note that this information may be altered by spammers.

The above examples only represent a small part of what Sendmail can do in terms of allowing or blocking access. Refer to the **/usr/share/sendmail-cf/README** for more information and examples.

Since Sendmail calls the Procmail MDA when delivering mail, it is also possible to use a spam filtering program, such as SpamAssassin, to identify and file spam for users. Refer to Section 27.5.2.6, “Spam Filters” for more about using SpamAssassin.

27.3.1.6. Using Sendmail with LDAP

Using the *Lightweight Directory Access Protocol (LDAP)* is a very quick and powerful way to find specific information about a particular user from a much larger group. For example, an LDAP server can be used to look up a particular email address from a common corporate directory by the user's last name. In this kind of implementation, LDAP is largely separate from Sendmail, with LDAP storing the hierarchical user information and Sendmail only being given the result of LDAP queries in pre-addressed email messages.

However, Sendmail supports a much greater integration with LDAP, where it uses LDAP to replace separately maintained files, such as **aliases** and **virtusertables**, on different mail servers that work together to support a medium- to enterprise-level organization. In short, LDAP abstracts the mail routing level from Sendmail and its separate configuration files to a powerful LDAP cluster that can be leveraged by many different applications.

The current version of Sendmail contains support for LDAP. To extend the Sendmail server using LDAP, first get an LDAP server, such as **OpenLDAP**, running and properly configured. Then edit the **/etc/mail/sendmail.mc** to include the following:

```
LDAPROUTE_DOMAIN('yourdomain.com')dnl
FEATURE('ldap_routing')dnl
```

Note

This is only for a very basic configuration of Sendmail with LDAP. The configuration can differ greatly from this depending on the implementation of LDAP, especially when configuring several Sendmail machines to use a common LDAP server.

Consult **/usr/share/sendmail-cf/README** for detailed LDAP routing configuration instructions and examples.

Next, recreate the **/etc/mail/sendmail.cf** file by running **m4** and restarting Sendmail. Refer to Section 27.3.1.3, “Common Sendmail Configuration Changes” for instructions.

For more information on LDAP, refer to Chapter 28, *Lightweight Directory Access Protocol (LDAP)*.

27.3.2. Postfix

Originally developed at IBM by security expert and programmer Wietse Venema, Postfix is a Sendmail-compatible MTA that is designed to be secure, fast, and easy to configure.

To improve security, Postfix uses a modular design, where small processes with limited privileges are launched by a *master* daemon. The smaller, less privileged processes perform very specific tasks related to the various stages of mail delivery and run in a changed root environment to limit the effects of attacks.

Configuring Postfix to accept network connections from hosts other than the local computer takes only a few minor changes in its configuration file. Yet for those with more complex needs, Postfix provides a variety of configuration options, as well as third party add ons that make it a very versatile and full-featured MTA.

The configuration files for Postfix are human readable and support upward of 250 directives. Unlike Sendmail, no macro processing is required for changes to take effect and the majority of the most commonly used options are described in the heavily commented files.

Important

Before using Postfix, the default MTA must be switched from Sendmail to Postfix.

27.3.2.1. The Default Postfix Installation

The Postfix executable is **`/usr/sbin/postfix`**. This daemon launches all related processes needed to handle mail delivery.

Postfix stores its configuration files in the **`/etc/postfix/`** directory. The following is a list of the more commonly used files:

- **`access`** — Used for access control, this file specifies which hosts are allowed to connect to Postfix.
- **`aliases`** — A configurable list required by the mail protocol.
- **`main.cf`** — The global Postfix configuration file. The majority of configuration options are specified in this file.
- **`master.cf`** — Specifies how Postfix interacts with various processes to accomplish mail delivery.
- **`transport`** — Maps email addresses to relay hosts.

Important

The default **`/etc/postfix/main.cf`** file does not allow Postfix to accept network connections from a host other than the local computer. For instructions on configuring Postfix as a server for other clients, refer to Section 27.3.2.2, “Basic Postfix Configuration”.

When changing some options within files in the **`/etc/postfix/`** directory, it may be necessary to restart the **`postfix`** service for the changes to take effect. The easiest way to do this is to type the following command:

```
service postfix restart
```

27.3.2.2. Basic Postfix Configuration

By default, Postfix does not accept network connections from any host other than the local host. Perform the following steps as root to enable mail delivery for other hosts on the network:

- Edit the **`/etc/postfix/main.cf`** file with a text editor, such as **`vi`**.
- Uncomment the **`mydomain`** line by removing the hash mark (**`#`**), and replace *domain.tld* with the domain the mail server is servicing, such as **`example.com`**.
- Uncomment the **`myorigin = $mydomain`** line.
- Uncomment the **`myhostname`** line, and replace *host.domain.tld* with the hostname for the machine.
- Uncomment the **`mydestination = $myhostname, localhost.$mydomain`** line.
- Uncomment the **`mynetworks`** line, and replace *168.100.189.0/28* with a valid network setting for hosts that can connect to the server.
- Uncomment the **`inet_interfaces = all`** line.
- Comment the **`inet_interfaces = localhost`** line.
- Restart the **`postfix`** service.

Once these steps are complete, the host accepts outside emails for delivery.

Postfix has a large assortment of configuration options. One of the best ways to learn how to configure Postfix is to read the comments within the **`/etc/postfix/main.cf`** file. Also, with Red Hat Enterprise Linux version 5.9, Postfix provides MySQL maps support which allows Postfix to use a MySQL database and configure various lookup tables for various operations over MySQL databases. For example, the **`virtual`** table for handling global mail redirection, the **`access`** table for controlling access to an SMTP server and the **`aliases`** table for managing system-wide mail redirection. Configuration details and examples, as well as other additional resources including information about LDAP and SpamAssassin integration are available online at http://www.postfix.org/.

27.3.3. Fetchmail

Fetchmail is an MTA which retrieves email from remote servers and delivers it to the local MTA. Many users appreciate the ability to separate the process of downloading their messages located on a remote server from the process of reading and organizing their email in an MUA. Designed with the needs of dial-up users in mind, Fetchmail connects and quickly downloads all of the email messages to the mail spool file using any number of protocols, including POP3 and IMAP. It can even forward email messages to an SMTP server, if necessary.

Fetchmail is configured for each user through the use of a **`.fetchmailrc`** file in the user's home directory.

Using preferences in the **`.fetchmailrc`** file, Fetchmail checks for email on a remote server and downloads it. It then delivers it to port 25 on the local machine, using the local MTA to place the email in the correct user's spool file. If Procmail is available, it is launched to filter the email and place it in a mailbox so that it can be read by an MUA.

27.3.3.1. Fetchmail Configuration Options

Although it is possible to pass all necessary options on the command line to check for email on a remote server when executing Fetchmail, using a **`.fetchmailrc`** file is much easier. Place any desired configuration options in the **`.fetchmailrc`** file for those options to be used each time the **`fetchmail`** command is issued. It is possible to override these at the time Fetchmail is run by specifying that option on the command line.

A user's **.fetchmailrc** file contains three classes of configuration options:

- *global options* — Gives Fetchmail instructions that control the operation of the program or provide settings for every connection that checks for email.
- *server options* — Specifies necessary information about the server being polled, such as the hostname, as well as preferences for specific email servers, such as the port to check or number of seconds to wait before timing out. These options affect every user using that server.
- *user options* — Contains information, such as username and password, necessary to authenticate and check for email using a specified email server.

Global options appear at the top of the **.fetchmailrc** file, followed by one or more server options, each of which designate a different email server that Fetchmail should check. User options follow server options for each user account checking that email server. Like server options, multiple user options may be specified for use with a particular server as well as to check multiple email accounts on the same server.

Server options are called into service in the **.fetchmailrc** file by the use of a special option verb, **poll** or **skip**, that precedes any of the server information. The **poll** action tells Fetchmail to use this server option when it is run, which checks for email using the specified user options. Any server options after a **skip** action, however, are not checked unless this server's hostname is specified when Fetchmail is invoked. The **skip** option is useful when testing configurations in **.fetchmailrc** because it only checks skipped servers when specifically invoked, and does not affect any currently working configurations.

A sample **.fetchmailrc** file looks similar to the following example:

```
set postmaster "user1"
set bouncemail
 poll pop.domain.com proto pop3
  user 'user1' there with password 'secret' is user1 here
 poll mail.domain2.com
  user 'user5' there with password 'secret2' is user1 here
 user 'user7' there with password 'secret3' is user1 here
```

In this example, the global options specify that the user is sent email as a last resort (**postmaster** option) and all email errors are sent to the postmaster instead of the sender (**bouncemail** option). The **set** action tells Fetchmail that this line contains a global option. Then, two email servers are specified, one set to check using POP3, the other for trying various protocols to find one that works. Two users are checked using the second server option, but all email found for any user is sent to **user1**'s mail spool. This allows multiple mailboxes to be checked on multiple servers, while appearing in a single MUA inbox. Each user's specific information begins with the **user** action.

Note

Users are not required to place their password in the **.fetchmailrc** file. Omitting the **with password '<password>'** section causes Fetchmail to ask for a password when it is launched.

Fetchmail has numerous global, server, and local options. Many of these options are rarely used or only apply to very specific situations. The **fetchmail** man page explains each option in detail, but the most common ones are listed here.

27.3.3.2. Global Options

Each global option should be placed on a single line after a **set** action.

- **`daemon <seconds>`** — Specifies daemon-mode, where Fetchmail stays in the background. Replace *<seconds>* with the number of seconds Fetchmail is to wait before polling the server.
- **`postmaster`** — Specifies a local user to send mail to in case of delivery problems.
- **`syslog`** — Specifies the log file for errors and status messages. By default, this is **`/var/log/maillog`**.

27.3.3.3. Server Options

Server options must be placed on their own line in **`.fetchmailrc`** after a **`poll`** or **`skip`** action.

- **`auth <auth-type>`** — Replace *<auth-type>* with the type of authentication to be used. By default, **`password`** authentication is used, but some protocols support other types of authentication, including **`kerberos_v5`**, **`kerberos_v4`**, and **`ssh`**. If the **`any`** authentication type is used, Fetchmail first tries methods that do not require a password, then methods that mask the password, and finally attempts to send the password unencrypted to authenticate to the server.
- **`interval <number>`** — Polls the specified server every **`<number>`** of times that it checks for email on all configured servers. This option is generally used for email servers where the user rarely receives messages.
- **`port <port-number>`** — Replace *<port-number>* with the port number. This value overrides the default port number for the specified protocol.
- **`proto <protocol>`** — Replace *<protocol>* with the protocol, such as **`pop3`** or **`imap`**, to use when checking for messages on the server.
- **`timeout <seconds>`** — Replace *<seconds>* with the number of seconds of server inactivity after which Fetchmail gives up on a connection attempt. If this value is not set, a default of **`300`** seconds is assumed.

27.3.3.4. User Options

User options may be placed on their own lines beneath a server option or on the same line as the server option. In either case, the defined options must follow the **`user`** option (defined below).

- **`fetchall`** — Orders Fetchmail to download all messages in the queue, including messages that have already been viewed. By default, Fetchmail only pulls down new messages.
- **`fetchlimit <number>`** — Replace *<number>* with the number of messages to be retrieved before stopping.
- **`flush`** — Deletes all previously viewed messages in the queue before retrieving new messages.
- **`limit <max-number-bytes>`** — Replace *<max-number-bytes>* with the maximum size in bytes that messages are allowed to be when retrieved by Fetchmail. This option is useful with slow network links, when a large message takes too long to download.
- **`password '<password>'`** — Replace *<password>* with the user's password.
- **`preconnect "<command>"`** — Replace *<command>* with a command to be executed before retrieving messages for the user.
- **`postconnect "<command>"`** — Replace *<command>* with a command to be executed after retrieving messages for the user.
- **`ssl`** — Activates SSL encryption.

- **`sslproto`** — Defines allowed SSL or TLS protocols. Possible values are **`SSL2`**, **`SSL3`**, **`SSL23`**, and **`TLS1`**; however, due to the POODLE: SSLv3 vulnerability (CVE-2014-3566), be sure to set this option to **`TLS1`**.
- **`user "<username>"`** — Replace *<username>* with the username used by Fetchmail to retrieve messages. *This option must precede all other user options.*

27.3.3.5. Fetchmail Command Options

Most Fetchmail options used on the command line when executing the **`fetchmail`** command mirror the **`.fetchmailrc`** configuration options. In this way, Fetchmail may be used with or without a configuration file. These options are not used on the command line by most users because it is easier to leave them in the **`.fetchmailrc`** file.

There may be times when it is desirable to run the **`fetchmail`** command with other options for a particular purpose. It is possible to issue command options to temporarily override a **`.fetchmailrc`** setting that is causing an error, as any options specified at the command line override configuration file options.

27.3.3.6. Informational or Debugging Options

Certain options used after the **`fetchmail`** command can supply important information.

- **`--configdump`** — Displays every possible option based on information from **`.fetchmailrc`** and Fetchmail defaults. No email is retrieved for any users when using this option.
- **`-s`** — Executes Fetchmail in silent mode, preventing any messages, other than errors, from appearing after the **`fetchmail`** command.
- **`-v`** — Executes Fetchmail in verbose mode, displaying every communication between Fetchmail and remote email servers.
- **`-V`** — Displays detailed version information, lists its global options, and shows settings to be used with each user, including the email protocol and authentication method. No email is retrieved for any users when using this option.

27.3.3.7. Special Options

These options are occasionally useful for overriding defaults often found in the **`.fetchmailrc`** file.

- **`-a`** — Fetchmail downloads all messages from the remote email server, whether new or previously viewed. By default, Fetchmail only downloads new messages.
- **`-k`** — Fetchmail leaves the messages on the remote email server after downloading them. This option overrides the default behavior of deleting messages after downloading them.
- **`-l <max-number-bytes>`** — Fetchmail does not download any messages over a particular size and leaves them on the remote email server.
- **`--quit`** — Quits the Fetchmail daemon process.

More commands and **`.fetchmailrc`** options can be found in the **`fetchmail`** man page.

27.4. Mail Transport Agent (MTA) Configuration

A *Mail Transport Agent* (MTA) is essential for sending email. A *Mail User Agent* (MUA) such as **Evolution**, **Thunderbird**, and **Mutt**, is used to read and compose email. When a user sends an email from an MUA, the message is handed off to the MTA, which sends the message through a series of MTAs until it reaches its destination.

Even if a user does not plan to send email from the system, some automated tasks or system programs might use the **`/bin/mail`** command to send email containing log messages to the root user of the local system.

Red Hat Enterprise Linux 5 provides three MTAs: Sendmail, Postfix, and Exim. If all three are installed, **`sendmail`** is the default MTA. The **Mail Transport Agent Switcher** allows for the selection of either **`sendmail`**, **`postfix`**, or **`exim`** as the default MTA for the system.

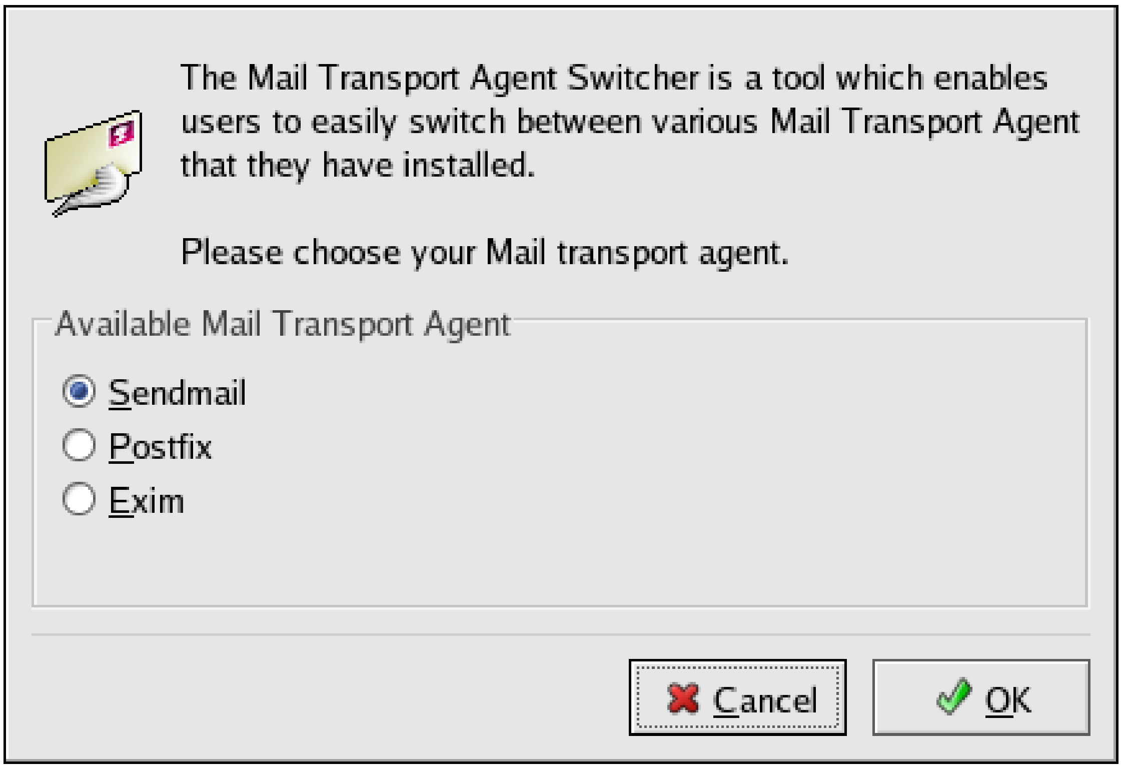
The **`system-switch-mail`** RPM package must be installed to use the text-based version of the **Mail Transport Agent Switcher** program. If you want to use the graphical version, the **`system-switch-mail-gnome`** package must also be installed.

Note

For more information on installing RPM packages, refer to Part II, "Package Management".

To start the **Mail Transport Agent Switcher**, select **System** (the main menu on the panel) > **Administration** > **Mail Transport Agent Switcher**, or type the command **`system-switch-mail`** at a shell prompt (for example, in an XTerm or GNOME terminal).

The program automatically detects if the X Window System is running. If it is running, the program starts in graphical mode as shown in Figure 27.1, " Mail Transport Agent Switcher ". If X is not detected, it starts in text-mode. To force **Mail Transport Agent Switcher** to run in text-mode, use the command **`system-switch-mail-nox`**.


The Mail Transport Agent Switcher is a tool which enables users to easily switch between various Mail Transport Agent that they have installed.

Please choose your Mail transport agent.

Available Mail Transport Agent

(●) Sendmail
( ) Postfix
( ) Exim

Figure 27.1. Mail Transport Agent Switcher

If you select **`OK`** to change the MTA, the selected mail daemon is enabled to start at boot time, and the unselected mail daemons are disabled so that they do not start at boot time. The selected mail daemon is started, and any other mail daemon is stopped; thus making the changes take place immediately.

27.5. Mail Delivery Agents

Red Hat Enterprise Linux includes two primary MDAs, Procmail and **`mail`**. Both of the applications are considered LDAs and both move email from the MTA's spool file into the user's mailbox. However, Procmail provides a robust filtering system.

This section details only Procmail. For information on the **`mail`** command, consult its man page.

Procmail delivers and filters email as it is placed in the mail spool file of the localhost. It is powerful, gentle on system resources, and widely used. Procmail can play a critical role in delivering email to be read by email client applications.

Procmail can be invoked in several different ways. Whenever an MTA places an email into the mail spool file, Procmail is launched. Procmail then filters and files the email for the MUA and quits. Alternatively, the MUA can be configured to execute Procmail any time a message is received so that messages are moved into their correct mailboxes. By default, the presence of **`/etc/procmailrc`** or of a **`.procmailrc`** file (also called an *rc* file) in the user's home directory invokes Procmail whenever an MTA receives a new message.

Whether Procmail acts upon an email message depends upon whether the message matches a specified set of conditions or *recipes* in the **`rc`** file. If a message matches a recipe, then the email is placed in a specified file, is deleted, or is otherwise processed.

When Procmail starts, it reads the email message and separates the body from the header information. Next, Procmail looks for **`/etc/procmailrc`** and **`rc`** files in the **`/etc/procmailrcs`** directory for default, system-wide, Procmail environmental variables and recipes. Procmail then searches for a **`.procmailrc`** file in the user's home directory. Many users also create additional **`rc`** files for Procmail that are referred to within the **`.procmailrc`** file in their home directory.

By default, no system-wide **`rc`** files exist in the **`/etc/`** directory and no **`.procmailrc`** files exist in any user's home directory. Therefore, to use Procmail, each user must construct a **`.procmailrc`** file with specific environment variables and rules.

27.5.1. Procmail Configuration

The Procmail configuration file contains important environmental variables. These variables specify things such as which messages to sort and what to do with the messages that do not match any recipes.

These environmental variables usually appear at the beginning of **`.procmailrc`** in the following format:

```
<env-variable>="<value>"
```

In this example, ***`<env-variable>`*** is the name of the variable and ***`<value>`*** defines the variable.

There are many environment variables not used by most Procmail users and many of the more important environment variables are already defined by a default value. Most of the time, the following variables are used:

- **`DEFAULT`** — Sets the default mailbox where messages that do not match any recipes are placed.

  The default **`DEFAULT`** value is the same as **`$ORGMAIL`**.

- **`INCLUDERC`** — Specifies additional **`rc`** files containing more recipes for messages to be checked against. This breaks up the Procmail recipe lists into individual files that fulfill different roles, such as blocking spam and managing email lists, that can then be turned off or on by using comment characters in the user's **`.procmailrc`** file.

For example, lines in a user's **`.procmailrc`** file may look like this:

```
MAILDIR=$HOME/Msgs
INCLUDERC=$MAILDIR/lists.rc
INCLUDERC=$MAILDIR/spam.rc
```

If the user wants to turn off Procmail filtering of their email lists but leave spam control in place, they would comment out the first **`INCLUDERC`** line with a hash mark character (**`#`**).

- **`LOCKSLEEP`** — Sets the amount of time, in seconds, between attempts by Procmail to use a particular lockfile. The default is eight seconds.
- **`LOCKTIMEOUT`** — Sets the amount of time, in seconds, that must pass after a lockfile was last modified before Procmail assumes that the lockfile is old and can be deleted. The default is 1024 seconds.
- **`LOGFILE`** — The file to which any Procmail information or error messages are written.
- **`MAILDIR`** — Sets the current working directory for Procmail. If set, all other Procmail paths are relative to this directory.
- **`ORGMAIL`** — Specifies the original mailbox, or another place to put the messages if they cannot be placed in the default or recipe-required location.

  By default, a value of **`/var/spool/mail/$LOGNAME`** is used.
- **`SUSPEND`** — Sets the amount of time, in seconds, that Procmail pauses if a necessary resource, such as swap space, is not available.
- **`SWITCHRC`** — Allows a user to specify an external file containing additional Procmail recipes, much like the **`INCLUDERC`** option, except that recipe checking is actually stopped on the referring configuration file and only the recipes on the **`SWITCHRC`**-specified file are used.
- **`VERBOSE`** — Causes Procmail to log more information. This option is useful for debugging.

Other important environmental variables are pulled from the shell, such as **`LOGNAME`**, which is the login name; **`HOME`**, which is the location of the home directory; and **`SHELL`**, which is the default shell.

A comprehensive explanation of all environments variables, as well as their default values, is available in the **`procmailrc`** man page.

27.5.2. Procmail Recipes

New users often find the construction of recipes the most difficult part of learning to use Procmail. To some extent, this is understandable, as recipes do their message matching using *regular expressions*, which is a particular format used to specify qualifications for a matching string. However, regular expressions are not very difficult to construct and even less difficult to understand when read. Additionally, the consistency of the way Procmail recipes are written, regardless of regular expressions, makes it easy to learn by example. To see example Procmail recipes, refer to Section 27.5.2.5, “Recipe Examples”.

Procmail recipes take the following form:

```
:0<flags>: <lockfile-name>
* <special-condition-character>
    <condition-1>
* <special-condition-character>
    <condition-2>
```

```
* <special-condition-character>
    <condition-N>
    <special-action-character>
    <action-to-perform>
```

The first two characters in a Procmail recipe are a colon and a zero. Various flags can be placed after the zero to control how Procmail processes the recipe. A colon after the ***<flags>*** section specifies that a lockfile is created for this message. If a lockfile is created, the name can be specified by replacing ***<lockfile-name>*** .

A recipe can contain several conditions to match against the message. If it has no conditions, every message matches the recipe. Regular expressions are placed in some conditions to facilitate message matching. If multiple conditions are used, they must all match for the action to be performed. Conditions are checked based on the flags set in the recipe's first line. Optional special characters placed after the **`*`** character can further control the condition.

The ***<action-to-perform>*** specifies the action taken when the message matches one of the conditions. There can only be one action per recipe. In many cases, the name of a mailbox is used here to direct matching messages into that file, effectively sorting the email. Special action characters may also be used before the action is specified. Refer to Section 27.5.2.4, “Special Conditions and Actions” for more information.

27.5.2.1. Delivering vs. Non-Delivering Recipes

The action used if the recipe matches a particular message determines whether it is considered a *delivering* or *non-delivering* recipe. A delivering recipe contains an action that writes the message to a file, sends the message to another program, or forwards the message to another email address. A non-delivering recipe covers any other actions, such as a *nesting block*. A nesting block is a set of actions, contained in braces **`{ }`**, that are performed on messages which match the recipe's conditions. Nesting blocks can be nested inside one another, providing greater control for identifying and performing actions on messages.

When messages match a delivering recipe, Procmail performs the specified action and stops comparing the message against any other recipes. Messages that match non-delivering recipes continue to be compared against other recipes.

27.5.2.2. Flags

Flags are essential to determine how or if a recipe's conditions are compared to a message. The following flags are commonly used:

- **`A`** — Specifies that this recipe is only used if the previous recipe without an **`A`** or **`a`** flag also matched this message.
- **`a`** — Specifies that this recipe is only used if the previous recipe with an **`A`** or **`a`** flag also matched this message *and* was successfully completed.
- **`B`** — Parses the body of the message and looks for matching conditions.
- **`b`** — Uses the body in any resulting action, such as writing the message to a file or forwarding it. This is the default behavior.
- **`c`** — Generates a carbon copy of the email. This is useful with delivering recipes, since the required action can be performed on the message and a copy of the message can continue being processed in the **`rc`** files.
- **`D`** — Makes the **`egrep`** comparison case-sensitive. By default, the comparison process is not case-sensitive.

- **`E`** — While similar to the **`A`** flag, the conditions in the recipe are only compared to the message if the immediately preceding the recipe without an **`E`** flag did not match. This is comparable to an *else* action.
- **`e`** — The recipe is compared to the message only if the action specified in the immediately preceding recipe fails.
- **`f`** — Uses the pipe as a filter.
- **`H`** — Parses the header of the message and looks for matching conditions. This occurs by default.
- **`h`** — Uses the header in a resulting action. This is the default behavior.
- **`w`** — Tells Procmail to wait for the specified filter or program to finish, and reports whether or not it was successful before considering the message filtered.
- **`W`** — Is identical to **`w`** except that "Program failure" messages are suppressed.

For a detailed list of additional flags, refer to the **`procmailrc`** man page.

27.5.2.3. Specifying a Local Lockfile

Lockfiles are very useful with Procmail to ensure that more than one process does not try to alter a message simultaneously. Specify a local lockfile by placing a colon (**`:`**) after any flags on a recipe's first line. This creates a local lockfile based on the destination file name plus whatever has been set in the **`LOCKEXT`** global environment variable.

Alternatively, specify the name of the local lockfile to be used with this recipe after the colon.

27.5.2.4. Special Conditions and Actions

Special characters used before Procmail recipe conditions and actions change the way they are interpreted.

The following characters may be used after the **`*`** character at the beginning of a recipe's condition line:

- **`!`** — In the condition line, this character inverts the condition, causing a match to occur only if the condition does not match the message.
- **`<`** — Checks if the message is under a specified number of bytes.
- **`>`** — Checks if the message is over a specified number of bytes.

The following characters are used to perform special actions:

- **`!`** — In the action line, this character tells Procmail to forward the message to the specified email addresses.
- **`$`** — Refers to a variable set earlier in the **`rc`** file. This is often used to set a common mailbox that is referred to by various recipes.
- **`|`** — Starts a specified program to process the message.
- **`{`** and **`}`** — Constructs a nesting block, used to contain additional recipes to apply to matching messages.

If no special character is used at the beginning of the action line, Procmail assumes that the action line is specifying the mailbox in which to write the message.

27.5.2.5. Recipe Examples

Procmail is an extremely flexible program, but as a result of this flexibility, composing Procmail recipes from scratch can be difficult for new users.

The best way to develop the skills to build Procmail recipe conditions stems from a strong understanding of regular expressions combined with looking at many examples built by others. A thorough explanation of regular expressions is beyond the scope of this section. The structure of Procmail recipes and useful sample Procmail recipes can be found at various places on the Internet (such as http://www.iki.fi/era/procmail/links.html). The proper use and adaptation of regular expressions can be derived by viewing these recipe examples. In addition, introductory information about basic regular expression rules can be found in the **`grep`** man page.

The following simple examples demonstrate the basic structure of Procmail recipes and can provide the foundation for more intricate constructions.

A basic recipe may not even contain conditions, as is illustrated in the following example:

```
:0:
new-mail.spool
```

The first line specifies that a local lockfile is to be created but does not specify a name, so Procmail uses the destination file name and appends the value specified in the **`LOCKEXT`** environment variable. No condition is specified, so every message matches this recipe and is placed in the single spool file called **`new-mail.spool`**, located within the directory specified by the **`MAILDIR`** environment variable. An MUA can then view messages in this file.

A basic recipe, such as this, can be placed at the end of all **`rc`** files to direct messages to a default location.

The following example matched messages from a specific email address and throws them away.

```
:0
* ^From: spammer@domain.com
/dev/null
```

With this example, any messages sent by **`spammer@domain.com`** are sent to the **`/dev/null`** device, deleting them.

Warning

Be certain that rules are working as intended before sending messages to **`/dev/null`** for permanent deletion. If a recipe inadvertently catches unintended messages, and those messages disappear, it becomes difficult to troubleshoot the rule.

A better solution is to point the recipe's action to a special mailbox, which can be checked from time to time to look for false positives. Once satisfied that no messages are accidentally being matched, delete the mailbox and direct the action to send the messages to **`/dev/null`**.

The following recipe grabs email sent from a particular mailing list and places it in a specified folder.

```
:0:
* ^(From|CC|To).*tux-lug
tuxlug
```

Any messages sent from the **`tux-lug@domain.com`** mailing list are placed in the **`tuxlug`** mailbox automatically for the MUA. Note that the condition in this example matches the message if it has the mailing list's email address on the **`From`**, **`CC`**, or **`To`** lines.

Consult the many Procmail online resources available in Section 27.7, “Additional Resources” for more detailed and powerful recipes.

27.5.2.6. Spam Filters

Because it is called by Sendmail, Postfix, and Fetchmail upon receiving new emails, Procmail can be used as a powerful tool for combating spam.

This is particularly true when Procmail is used in conjunction with SpamAssassin. When used together, these two applications can quickly identify spam emails, and sort or destroy them.

SpamAssassin uses header analysis, text analysis, blacklists, a spam-tracking database, and self-learning Bayesian spam analysis to quickly and accurately identify and tag spam.

The easiest way for a local user to use SpamAssassin is to place the following line near the top of the **`~/.procmailrc`** file:

```
INCLUDERC=/etc/mail/spamassassin/spamassassin-default.rc
```

The **`/etc/mail/spamassassin/spamassassin-default.rc`** contains a simple Procmail rule that activates SpamAssassin for all incoming email. If an email is determined to be spam, it is tagged in the header as such and the title is prepended with the following pattern:

```
*****SPAM*****
```

The message body of the email is also prepended with a running tally of what elements caused it to be diagnosed as spam.

To file email tagged as spam, a rule similar to the following can be used:

```
:0 Hw
* ^X-Spam-Status: Yes
spam
```

This rule files all email tagged in the header as spam into a mailbox called **`spam`**.

Since SpamAssassin is a Perl script, it may be necessary on busy servers to use the binary SpamAssassin daemon (**`spamd`**) and client application (**`spamc`**). Configuring SpamAssassin this way, however, requires root access to the host.

To start the **`spamd`** daemon, type the following command as root:

```
service spamassassin start
```

To start the SpamAssassin daemon when the system is booted, use an initscript utility, such as the **Services Configuration Tool** (**`system-config-services`**), to turn on the **`spamassassin`** service. Refer to Chapter 18, *Controlling Access to Services* for more information about initscript utilities.

To configure Procmail to use the SpamAssassin client application instead of the Perl script, place the following line near the top of the **`~/.procmailrc`** file. For a system-wide configuration, place it in **`/etc/procmailrc`**:

```
INCLUDERC=/etc/mail/spamassassin/spamassassin-spamc.rc
```

27.6. Mail User Agents

There are scores of mail programs available under Red Hat Enterprise Linux. There are full-featured, graphical email client programs, such as **Ximian Evolution**, as well as text-based email programs such as **`mutt`**.

The remainder of this section focuses on securing communication between the client and server.

27.6.1. Securing Communication

Popular MUAs included with Red Hat Enterprise Linux, such as **Ximian Evolution** and **`mutt`** offer SSL-encrypted email sessions.

Like any other service that flows over a network unencrypted, important email information, such as usernames, passwords, and entire messages, may be intercepted and viewed by users on the network. Additionally, since the standard POP and IMAP protocols pass authentication information unencrypted, it is possible for an attacker to gain access to user accounts by collecting usernames and passwords as they are passed over the network.

27.6.1.1. Secure Email Clients

Most Linux MUAs designed to check email on remote servers support SSL encryption. To use SSL when retrieving email, it must be enabled on both the email client and server.

SSL is easy to enable on the client-side, often done with the click of a button in the MUA's configuration window or via an option in the MUA's configuration file. Secure IMAP and POP have known port numbers (993 and 995, respectively) that the MUA uses to authenticate and download messages.

27.6.1.2. Securing Email Client Communications

Offering SSL encryption to IMAP and POP users on the email server is a simple matter.

First, create an SSL certificate. This can be done two ways: by applying to a *Certificate Authority* (*CA*) for an SSL certificate or by creating a self-signed certificate.

Warning

Self-signed certificates should be used for testing purposes only. Any server used in a production environment should use an SSL certificate granted by a CA.

To create a self-signed SSL certificate for IMAP, change to the **`/etc/pki/tls/certs/`** directory and type the following commands as root:

```
rm -f cyrus-imapd.pem make cyrus-imapd.pem
```

Answer all of the questions to complete the process.

To create a self-signed SSL certificate for POP, change to the **`/etc/pki/tls/certs/`** directory, and type the following commands as root:

```
rm -f ipop3d.pem make ipop3d.pem
```

Again, answer all of the questions to complete the process.

Important

Please be sure to remove the default **`imapd.pem`** and **`ipop3d.pem`** files before issuing each **`make`** command.

Disable insecure SSL protocols by adding the following line to the **`/etc/imapd.conf`** file:

```
tls_cipher_list: TLSv1+HIGH:!aNull:@STRENGTH
```

This is due to the POODLE SSL vulnerability (CVE-2014-3566). See POODLE: SSLv3 vulnerability (CVE-2014-3566) for details.

Once finished, execute the **`/sbin/service cyrus-imapd start`** command to start the **Cyrus** IMAP and POP daemons.

Alternatively, the **`stunnel`** command can be used as an SSL encryption wrapper around the standard, non-secure IMAP and POP protocols. In that case, however, you must disable IMAPS and POP3 in the **Cyrus** configuration file, **`/etc/cyrus.conf`**. To do so, comment out the lines containing **`imaps`** and **`pop3s`**, and restart the **`cyrus-imapd`** service.

The **`stunnel`** program uses external OpenSSL libraries included with Red Hat Enterprise Linux to provide strong cryptography and protect the connections. It is best to apply to a CA to obtain an SSL certificate, but it is also possible to create a self-signed certificate.

To create a self-signed SSL certificate, change to the **`/etc/pki/tls/certs/`** directory, and type the following command:

```
make stunnel.pem
```

Again, answer all of the questions to complete the process.

When you have a certificate, create a configuration file for **stunnel**. It is a text file in which every line specifies an option or the beginning of a service definition. You can also keep comments and empty lines in the file to improve its legibility, where comments start with a semicolon.

The *stunnel* RPM package contains the **`/etc/stunnel/`** directory, in which you can store the configuration file. Although **stunnel** does not require any special format of the file name or its extension, use **`/etc/stunnel/stunnel.conf`**. The following content configures **stunnel** as a TLS wrapper for secure IMAP and POP:

```
cert = /etc/pki/tls/certs/stunnel.pem
; Allow only TLS, thus avoiding SSL
options = NO_SSLv2
options = NO_SSLv3
chroot = /var/run/stunnel
setuid = nobody
setgid = nobody
pid = /stunnel.pid
socket = l:TCP_NODELAY=1
socket = r:TCP_NODELAY=1
```

```
[pop3s]
accept  = 995
connect = 110

[imaps]
accept  = 993
connect = 143
```

Finally, start **stunnel**:

```
stunnel /etc/stunnel/stunnel.conf
```

For more information about how to use **`stunnel`**, read the **`stunnel`** man page or refer to the documents in the **`/usr/share/doc/stunnel-<*version-number*>`** / directory, where <*version-number*> is the version number for **`stunnel`**.

27.7. Additional Resources

The following is a list of additional documentation about email applications.

27.7.1. Installed Documentation

- Information on configuring Sendmail is included with the **`sendmail`** and **`sendmail-cf`** packages.
  - **`/usr/share/sendmail-cf/README`** — Contains information on **`m4`**, file locations for Sendmail, supported mailers, how to access enhanced features, and more.

  In addition, the **`sendmail`** and **`aliases`** man pages contain helpful information covering various Sendmail options and the proper configuration of the Sendmail **`/etc/mail/aliases`** file.

- **`/usr/share/doc/postfix-<*version-number*>`** — Contains a large amount of information about ways to configure Postfix. Replace <*version-number*> with the version number of Postfix.

- **`/usr/share/doc/fetchmail-<*version-number*>`** — Contains a full list of Fetchmail features in the **`FEATURES`** file and an introductory **`FAQ`** document. Replace <*version-number*> with the version number of Fetchmail.

- **`/usr/share/doc/procmail-<*version-number*>`** — Contains a **`README`** file that provides an overview of Procmail, a **`FEATURES`** file that explores every program feature, and an **`FAQ`** file with answers to many common configuration questions. Replace <*version-number*> with the version number of Procmail.

  When learning how Procmail works and creating new recipes, the following Procmail man pages are invaluable:

  - **`procmail`** — Provides an overview of how Procmail works and the steps involved with filtering email.
  - **`procmailrc`** — Explains the **`rc`** file format used to construct recipes.
  - **`procmailex`** — Gives a number of useful, real-world examples of Procmail recipes.
  - **`procmailsc`** — Explains the weighted scoring technique used by Procmail to match a particular recipe to a message.

- **`/usr/share/doc/spamassassin-<version-number>/`** — Contains a large amount of information pertaining to SpamAssassin. Replace *<version-number>* with the version number of the **`spamassassin`** package.

27.7.2. Useful Websites

- http://www.sendmail.org/ — Offers a thorough technical breakdown of Sendmail features, documentation and configuration examples.
- http://www.sendmail.com/ — Contains news, interviews and articles concerning Sendmail, including an expanded view of the many options available.
- http://www.postfix.org/ — The Postfix project home page contains a wealth of information about Postfix. The mailing list is a particularly good place to look for information.
- http://fetchmail.berlios.de/ — The home page for Fetchmail, featuring an online manual, and a thorough FAQ.
- http://www.procmail.org/ — The home page for Procmail with links to assorted mailing lists dedicated to Procmail as well as various FAQ documents.
- http://partmaps.org/era/procmail/mini-faq.html — An excellent Procmail FAQ, offers troubleshooting tips, details about file locking, and the use of wildcard characters.
- http://www.uwasa.fi/~ts/info/proctips.html — Contains dozens of tips that make using Procmail much easier. Includes instructions on how to test **`.procmailrc`** files and use Procmail scoring to decide if a particular action should be taken.
- http://www.spamassassin.org/ — The official site of the SpamAssassin project.

27.7.3. Related Books

- *Sendmail Milters: A Guide for Fighting Spam* by Bryan Costales and Marcia Flynt; Addison-Wesley — A good Sendmail guide that can help you customise your mail filters.
- *Sendmail* by Bryan Costales with Eric Allman et al; O'Reilly & Associates — A good Sendmail reference written with the assistance of the original creator of Delivermail and Sendmail.
- *Removing the Spam: Email Processing and Filtering* by Geoff Mulligan; Addison-Wesley Publishing Company — A volume that looks at various methods used by email administrators using established tools, such as Sendmail and Procmail, to manage spam problems.
- *Internet Email Protocols: A Developer's Guide* by Kevin Johnson; Addison-Wesley Publishing Company — Provides a very thorough review of major email protocols and the security they provide.
- *Managing IMAP* by Dianna Mullet and Kevin Mullet; O'Reilly & Associates — Details the steps required to configure an IMAP server.

Chapter 28. Lightweight Directory Access Protocol (LDAP)

The *Lightweight Directory Access Protocol* (*LDAP*) is a set of open protocols used to access centrally stored information over a network. It is based on the *X.500* standard for directory sharing, but is less complex and resource-intensive. For this reason, LDAP is sometimes referred to as "*X.500 Lite*." The X.500 standard is a directory that contains hierarchical and categorized information, which could include information such as names, addresses, and phone numbers.

Like X.500, LDAP organizes information in a hierarchal manner using directories. These directories can store a variety of information and can even be used in a manner similar to the Network Information Service (NIS), enabling anyone to access their account from any machine on the LDAP enabled network.

In many cases, LDAP is used as a virtual phone directory, allowing users to easily access contact information for other users. But LDAP is more flexible than a traditional phone directory, as it is capable of referring a querent to other LDAP servers throughout the world, providing an ad-hoc global repository of information. Currently, however, LDAP is more commonly used within individual organizations, like universities, government departments, and private companies.

LDAP is a client/server system. The server can use a variety of databases to store a directory, each optimized for quick and copious read operations. When an LDAP client application connects to an LDAP server, it can either query a directory or attempt to modify it. In the event of a query, the server either answers the query locally, or it can refer the querent to an LDAP server which does have the answer. If the client application is attempting to modify information within an LDAP directory, the server verifies that the user has permission to make the change and then adds or updates the information.

This chapter refers to the configuration and use of OpenLDAP 2.0, an open source implementation of the LDAPv2 and LDAPv3 protocols.

28.1. Why Use LDAP?

The main benefit of using LDAP is that information for an entire organization can be consolidated into a central repository. For example, rather than managing user lists for each group within an organization, LDAP can be used as a central directory accessible from anywhere on the network. And because LDAP supports Transport Layer Security (TLS), sensitive data can be protected from prying eyes.

Important

Due to the vulnerability described in Resolution for POODLE SSLv3.0 vulnerability (CVE-2014-3566) for components that do not allow SSLv3 to be disabled via configuration settings, Red Hat recommends that you do not rely on the **`SSLv3`** protocol for security. OpenLDAP is one of the system components that do not provide configuration parameters that allow **`SSLv3`** to be effectively disabled. To mitigate the risk, it is recommended that you use the **`stunnel`** command to provide a secure tunnel, and disable **stunnel** from using **`SSLv3`**.

LDAP also supports a number of back-end databases in which to store directories. This allows administrators the flexibility to deploy the database best suited for the type of information the server is to disseminate. Because LDAP also has a well-defined client Application Programming Interface (API), the number of LDAP-enabled applications are numerous and increasing in quantity and quality.

28.1.1. OpenLDAP Features

OpenLDAP includes a number of important features.

- *LDAPv3 Support* — OpenLDAP supports Simple Authentication and Security Layer (SASL), and Transport Layer Security (TLS) among other improvements. Many of the changes in the protocol since LDAPv2 are designed to make LDAP more secure.
- *IPv6 Support* — OpenLDAP supports the next generation Internet Protocol version 6.
- *LDAP Over IPC* — OpenLDAP can communicate within a system using interprocess communication (IPC). This enhances security by eliminating the need to communicate over a network.
- *Updated C API* — Improves the way programmers can connect to and use LDAP directory servers.
- *LDIFv1 Support* — Provides full compliance with the LDAP Data Interchange Format (LDIF) version 1.
- *Enhanced Stand-Alone LDAP Server* — Includes an updated access control system, thread pooling, better tools, and much more.

28.2. LDAP Terminology

Any discussion of LDAP requires a basic understanding of a set of LDAP-specific terms:

- *entry* — A single unit within an LDAP directory. Each entry is identified by its unique *Distinguished Name (DN)*.
- *attributes* — Information directly associated with an entry. For example, an organization could be represented as an LDAP entry. Attributes associated with the organization might include a fax number, an address, and so on. People can also be represented as entries in an LDAP directory, with common attributes such as the person's telephone number and email address.

  Some attributes are required, while other attributes are optional. An *objectclass* definition sets which attributes are required for each entry. Objectclass definitions are found in various schema files, located in the **`/etc/openldap/schema/`** directory. For more information, refer to Section 28.5, "The `/etc/openldap/schema/` Directory".

  The assertion of an attribute and its corresponding value is also referred to as a *Relative Distinguished Name* (RDN). An RDN is only unique per entry, whereas a DN is globally unique.

- *LDIF* — The *LDAP Data Interchange Format* (LDIF) is an ASCII text representation of LDAP entries. Files used for importing data to LDAP servers must be in LDIF format. An LDIF entry looks similar to the following example:

```
[<id>] dn: <distinguished name>
<attrtype>: <attrvalue>
<attrtype>: <attrvalue>
<attrtype>: <attrvalue>
```

  Each entry can contain as many **`<attrtype>: <attrvalue>`** pairs as needed. A blank line indicates the end of an entry.

  Warning

  All **`<attrtype>`** and **`<attrvalue>`** pairs *must* be defined in a corresponding schema file to use this information.

  Any value enclosed within a **<** and a **>** is a variable and can be set whenever a new LDAP entry is created. This rule does not apply, however, to **`<id>`**. The **`<id>`** is a number determined by the application used to edit the entry.

28.3. OpenLDAP Daemons and Utilities

The suite of OpenLDAP libraries and tools are included within the following packages:

- **`openldap`** — Contains the libraries necessary to run the OpenLDAP server and client applications.
- **`openldap-clients`** — Contains command line tools for viewing and modifying directories on an LDAP server.
- **`openldap-servers`** — Contains the servers and other utilities necessary to configure and run an LDAP server.

There are two servers contained in the **`openldap-servers`** package: the *Standalone LDAP Daemon* (**`/usr/sbin/slapd`**) and the *Standalone LDAP Update Replication Daemon* (**`/usr/sbin/slurpd`**).

The **`slapd`** daemon is the standalone LDAP server while the **`slurpd`** daemon is used to synchronize changes from one LDAP server to other LDAP servers on the network. The **`slurpd`** daemon is only used when dealing with multiple LDAP servers.

To perform administrative tasks, the **`openldap-servers`** package installs the following utilities into the **`/usr/sbin/`** directory:

- **`slapadd`** — Adds entries from an LDIF file to an LDAP directory. For example, the command **`/usr/sbin/slapadd -l`** ***`ldif-input`*** reads in the LDIF file, ***`ldif-input`***, containing the new entries.

  Important

  Only the root user may use **`/usr/sbin/slapadd`**. However, the directory server runs as the **`ldap`** user. Therefore the directory server is unable to modify any files created by **`slapadd`**. To correct this issue, after using **`slapadd`**, type the following command:

  ```
  chown -R ldap /var/lib/ldap
  ```

- **`slapcat`** — Pulls entries from an LDAP directory in the default format, *Sleepycat Software's Berkeley DB* system, and saves them in an LDIF file. For example, the command **`/usr/sbin/slapcat -l`** ***`ldif-output`*** outputs an LDIF file called ***`ldif-output`*** containing the entries from the LDAP directory.
- **`slapindex`** — Re-indexes the **`slapd`** directory based on the current content. This tool should be run whenever indexing options within **`/etc/openldap/slapd.conf`** are changed.
- **`slappasswd`** — Generates an encrypted user password value for use with **`ldapmodify`** or the **`rootpw`** value in the **`slapd`** configuration file, **`/etc/openldap/slapd.conf`**. Execute the **`/usr/sbin/slappasswd`** command to create the password.

Warning

You must stop **slapd** by issuing the **/sbin/service ldap stop** command before using **slapadd**, **slapcat** or **slapindex**. Otherwise, the integrity of the LDAP directory is at risk.

For more information on using these utilities, refer to their respective man pages.

The **openldap-clients** package installs tools into **/usr/bin/** which are used to add, modify, and delete entries in an LDAP directory. These tools include the following:

- **ldapadd** — Adds entries to an LDAP directory by accepting input via a file or standard input; **ldapadd** is actually a hard link to **ldapmodify -a**.
- **ldapdelete** — Deletes entries from an LDAP directory by accepting user input at a shell prompt or via a file.
- **ldapmodify** — Modifies entries in an LDAP directory, accepting input via a file or standard input.
- **ldappasswd** — Sets the password for an LDAP user.
- **ldapsearch** — Searches for entries in an LDAP directory using a shell prompt.
- **ldapcompare** — Opens a connection to an LDAP server, binds, and performs a comparison using specified parameters.
- **ldapwhoami** — Opens a connection to an LDAP server, binds, and performs a**whoami** operation.
- **ldapmodrdn** — Opens a connection to an LDAP server, binds, and modifies the RDNs of entries.

With the exception of **ldapsearch**, each of these utilities is more easily used by referencing a file containing the changes to be made rather than typing a command for each entry to be changed within an LDAP directory. The format of such a file is outlined in the man page for each utility.

28.3.1. NSS, PAM, and LDAP

In addition to the OpenLDAP packages, Red Hat Enterprise Linux includes a package called **nss_ldap**, which enhances LDAP's ability to integrate into both Linux and other UNIX environments.

The **nss_ldap** package provides the following modules (where *<version>* refers to the version of **libnss_ldap** in use):

- **/lib/libnss_ldap-<*version*>.so**
- **/lib/security/pam_ldap.so**

The **nss_ldap** package provides the following modules for Itanium or AMD64 architectures:

- **/lib64/libnss_ldap-<*version*>.so**
- **/lib64/security/pam_ldap.so**

The **libnss_ldap-<*version*>.so** module allows applications to look up users, groups, hosts, and other information using an LDAP directory via the *Nameservice Switch* (NSS) interface of **glibc**. NSS allows applications to authenticate using LDAP in conjunction with the NIS name service and flat authentication files.

The **pam_ldap** module allows PAM-aware applications to authenticate users using information stored in an LDAP directory. PAM-aware applications include console login, POP and IMAP mail servers, and Samba. By deploying an LDAP server on a network, all of these applications can authenticate using the same user ID and password combination, greatly simplifying administration.

For more about configuring PAM, refer to Section 48.4, “Pluggable Authentication Modules (PAM)” and the PAM man pages.

28.3.2. PHP4, LDAP, and the Apache HTTP Server

Red Hat Enterprise Linux includes a package containing an LDAP module for the PHP server-side scripting language.

The **php-ldap** package adds LDAP support to the PHP4 HTML-embedded scripting language via the **/usr/lib/php4/ldap.so** module. This module allows PHP4 scripts to access information stored in an LDAP directory.

Red Hat Enterprise Linux ships with the **mod_authz_ldap** module for the Apache HTTP Server. This module uses the short form of the distinguished name for a subject and the issuer of the client SSL certificate to determine the distinguished name of the user within an LDAP directory. It is also capable of authorizing users based on attributes of that user's LDAP directory entry, determining access to assets based on the user and group privileges of the asset, and denying access for users with expired passwords. The **mod_ssl** module is required when using the **mod_authz_ldap** module.

Important

The **mod_authz_ldap** module does not authenticate a user to an LDAP directory using an encrypted password hash. This functionality is provided by the experimental **mod_auth_ldap** module, which is not included with Red Hat Enterprise Linux. Refer to the Apache Software Foundation website online at http://www.apache.org/ for details on the status of this module.

28.3.3. LDAP Client Applications

There are graphical LDAP clients available which support creating and modifying directories, but they are *not* included with Red Hat Enterprise Linux. One such application is **LDAP Browser/Editor** — A Java-based tool available online at http://www.iit.edu/~gawojar/ldap/.

Other LDAP clients access directories as read-only, using them to reference, but not alter, organization-wide information. Some examples of such applications are Sendmail, **Mozilla**, **Gnome Meeting**, and **Evolution**.

28.4. OpenLDAP Configuration Files

OpenLDAP configuration files are installed into the **/etc/openldap/** directory. The following is a brief list highlighting the most important directories and files:

- **/etc/openldap/ldap.conf** — This is the configuration file for all *client* applications which use the OpenLDAP libraries such as **ldapsearch**, **ldapadd**, Sendmail, **Evolution**, and **Gnome Meeting**.
- **/etc/openldap/slapd.conf** — This is the configuration file for the **slapd** daemon. Refer to Section 28.6.1, “Editing **/etc/openldap/slapd.conf**” for more information.
- **/etc/openldap/schema/** directory — This subdirectory contains the schema used by the **slapd** daemon. Refer to Section 28.5, “The **/etc/openldap/schema/** Directory” for more information.

Note

If the **`nss_ldap`** package is installed, it creates a file named **`/etc/ldap.conf`**. This file is used by the PAM and NSS modules supplied by the **`nss_ldap`** package. Refer to Section 28.7, "Configuring a System to Authenticate Using OpenLDAP" for more information.

28.5. The `/etc/openldap/schema/` Directory

The **`/etc/openldap/schema/`** directory holds LDAP definitions, previously located in the **`slapd.at.conf`** and **`slapd.oc.conf`** files. The **`/etc/openldap/schema/redhat/`** directory holds customized schemas distributed by Red Hat for Red Hat Enterprise Linux.

All *attribute syntax definitions* and *objectclass definitions* are now located in the different schema files. The various schema files are referenced in **`/etc/openldap/slapd.conf`** using **`include`** lines, as shown in this example:

```
include  /etc/openldap/schema/core.schema
include  /etc/openldap/schema/cosine.schema
include  /etc/openldap/schema/inetorgperson.schema
include  /etc/openldap/schema/nis.schema
include  /etc/openldap/schema/rfc822-MailMember.schema
include  /etc/openldap/schema/redhat/autofs.schema
```

Warning

Do not modify schema items defined in the schema files installed by OpenLDAP.

It is possible to extend the schema used by OpenLDAP to support additional attribute types and object classes using the default schema files as a guide. To do this, create a **`local.schema`** file in the **`/etc/openldap/schema/`** directory. Reference this new schema within **`slapd.conf`** by adding the following line below the default **`include`** schema lines:

```
include  /etc/openldap/schema/local.schema
```

Next, define new attribute types and object classes within the **`local.schema`** file. Many organizations use existing attribute types from the schema files installed by default and add new object classes to the **`local.schema`** file.

Extending the schema to match certain specialized requirements is quite involved and beyond the scope of this chapter. Refer to http://www.openldap.org/doc/admin/schema.html for information.

28.6. OpenLDAP Setup Overview

This section provides a quick overview for installing and configuring an OpenLDAP directory. For more details, refer to the following URLs:

- http://www.openldap.org/doc/admin/quickstart.html — The *Quick-Start Guide* on the OpenLDAP website.

- http://www.tldp.org/HOWTO/LDAP-HOWTO/index.html — The *LDAP Linux HOWTO* from the Linux Documentation Project.

The basic steps for creating an LDAP server are as follows:

1. Install the **openldap**, **openldap-servers**, and **openldap-clients** RPMs.
2. Edit the **/etc/openldap/slapd.conf** file to specify the LDAP domain and server. Refer to Section 28.6.1, “Editing **/etc/openldap/slapd.conf**” for more information.
3. Start **slapd** with the command:

   ```
   service ldap start
   ```

   After configuring LDAP, use **chkconfig**, **/usr/sbin/ntsysv**, or the **Services Configuration Tool** to configure LDAP to start at boot time. For more information about configuring services, refer to Chapter 18, *Controlling Access to Services*.
4. Add entries to an LDAP directory with **ldapadd**.
5. Use **ldapsearch** to determine if **slapd** is accessing the information correctly.
6. At this point, the LDAP directory should be functioning properly and can be configured with LDAP-enabled applications.

28.6.1. Editing /etc/openldap/slapd.conf

To use the **slapd** LDAP server, modify its configuration file, **/etc/openldap/slapd.conf**, to specify the correct domain and server.

The **suffix** line names the domain for which the LDAP server provides information and should be changed from:

```
suffix          "dc=your-domain,dc=com"
```

Edit it accordingly so that it reflects a fully qualified domain name. For example:

```
suffix          "dc=example,dc=com"
```

The **rootdn** entry is the Distinguished Name (DN) for a user who is unrestricted by access controls or administrative limit parameters set for operations on the LDAP directory. The **rootdn** user can be thought of as the root user for the LDAP directory. In the configuration file, change the **rootdn** line from its default value as in the following example:

```
rootdn          "cn=root,dc=example,dc=com"
```

When populating an LDAP directory over a network, change the **rootpw** line — replacing the default value with an encrypted password string. To create an encrypted password string, type the following command:

```
slappasswd
```

When prompted, type and then re-type a password. The program prints the resulting encrypted password to the shell prompt.

Next, copy the newly created encrypted password into the **/etc/openldap/slapd.conf** on one of the

rootpw lines and remove the hash mark (**#**).

When finished, the line should look similar to the following example:

```
rootpw {SSHA}vv2y+i6V6esazrIv70xSSnNAJE18bb2u
```

Warning

LDAP passwords, including the **rootpw** directive specified in **/etc/openldap/slapd.conf**, are sent over the network *unencrypted*, unless TLS encryption is enabled.

To enable TLS encryption, review the comments in **/etc/openldap/slapd.conf** and refer to the man page for **slapd.conf**.

For added security, the **rootpw** directive should be commented out after populating the LDAP directory by preceding it with a hash mark (**#**).

When using the **/usr/sbin/slapadd** command line tool locally to populate the LDAP directory, use of the **rootpw** directive is not necessary.

Important

Only the root user can use **/usr/sbin/slapadd**. However, the directory server runs as the **ldap** user. Therefore, the directory server is unable to modify any files created by **slapadd**. To correct this issue, after using **slapadd**, type the following command:

```
chown -R ldap /var/lib/ldap
```

28.7. Configuring a System to Authenticate Using OpenLDAP

This section provides a brief overview of how to configure OpenLDAP user authentication. Unless you are an OpenLDAP expert, more documentation than is provided here is necessary. Refer to the references provided in Section 28.9, “Additional Resources” for more information.

Install the Necessary LDAP Packages.

First, make sure that the appropriate packages are installed on both the LDAP server and the LDAP client machines. The LDAP server needs the **openldap-servers** package.

The **openldap**, **openldap-clients**, and **nss_ldap** packages need to be installed on all LDAP client machines.

Edit the Configuration Files.

- On the server, edit the **/etc/openldap/slapd.conf** file on the LDAP server to make sure it matches the specifics of the organization. Refer to Section 28.6.1, “Editing **/etc/openldap/slapd.conf**” for instructions about editing **slapd.conf**.

- On the client machines, both **/etc/ldap.conf** and **/etc/openldap/ldap.conf** need to contain the proper server and search base information for the organization.

  To do this, run the graphical **Authentication Configuration Tool** (**system-config-authentication**) and select **Enable LDAP Support** under the **User Information** tab.

  It is also possible to edit these files by hand.

- On the client machines, the **/etc/nsswitch.conf** must be edited to use LDAP.

  To do this, run the **Authentication Configuration Tool** (**system-config-authentication**) and select **Enable LDAP Support** under the **User Information** tab.

  If editing **/etc/nsswitch.conf** by hand, add **ldap** to the appropriate lines.

  For example:

  ```
  passwd: files ldap
  shadow: files ldap
  group: files ldap
  ```

28.7.1. PAM and LDAP

To have standard PAM-enabled applications use LDAP for authentication, run the **Authentication Configuration Tool** (**system-config-authentication**) and select **Enable LDAP Support** under the **Authentication** tab. For more about configuring PAM, refer to Section 48.4, “Pluggable Authentication Modules (PAM)” and the PAM man pages.

28.7.2. Migrating Old Authentication Information to LDAP Format

The **/usr/share/openldap/migration/** directory contains a set of shell and Perl scripts for migrating authentication information into an LDAP format.

Note

Perl must be installed on the system to use these scripts.

First, modify the **migrate_common.ph** file so that it reflects the correct domain. The default DNS domain should be changed from its default value to something like:

```
$DEFAULT_MAIL_DOMAIN = "example";
```

The default base should also be changed to something like:

```
$DEFAULT_BASE = "dc=example,dc=com";
```

The job of migrating a user database into a format that is LDAP readable falls to a group of migration scripts installed in the same directory. Using Table 28.1, “LDAP Migration Scripts”, decide which script to run to migrate the user database.

Run the appropriate script based on the existing name service.

The **`README`** and the **`migration-tools.txt`** files in the **`/usr/share/openldap/migration/`** directory provide more details on how to migrate the information.

Table 28.1. LDAP Migration Scripts

Existing name service	Is LDAP running?	Script to Use
`/etc` flat files	yes	**`migrate_all_online.sh`**
`/etc` flat files	no	**`migrate_all_offline.sh`**
NetInfo	yes	**`migrate_all_netinfo_online.sh`**
NetInfo	no	**`migrate_all_netinfo_offline.sh`**
NIS (YP)	yes	**`migrate_all_nis_online.sh`**
NIS (YP)	no	**`migrate_all_nis_offline.sh`**

28.8. Migrating Directories from Earlier Releases

With Red Hat Enterprise Linux, OpenLDAP uses Sleepycat Software's Berkeley DB system as its on-disk storage format for directories. Earlier versions of OpenLDAP used *GNU Database Manager* (*gdbm*). For this reason, before upgrading an LDAP implementation to Red Hat Enterprise Linux 5.10, original LDAP data should first be exported before the upgrade, and then reimported afterwards. This can be achieved by performing the following steps:

1. Before upgrading the operating system, run the command **`/usr/sbin/slapcat -l`** ***`ldif-output`***. This outputs an LDIF file called ***`ldif-output`*** containing the entries from the LDAP directory.

2. Upgrade the operating system, being careful not to reformat the partition containing the LDIF file.

3. Re-import the LDAP directory to the upgraded Berkeley DB format by executing the command **`/usr/sbin/slapadd -l`** ***`ldif-output`***.

28.9. Additional Resources

The following resources offer additional information on LDAP. It is highly recommended that you review these, especially the OpenLDAP website and the LDAP HOWTO, before configuring LDAP on your system(s).

28.9.1. Installed Documentation

- **`/usr/share/docs/openldap-`*****`<versionnumber>`*****`/`** directory — Contains a general **`README`** document and miscellaneous information.

- LDAP related man pages — There are a number of man pages for the various applications and configuration files involved with LDAP. The following is a list of some of the more important man pages.

  Client Applications

  - **`man ldapadd`** — Describes how to add entries to an LDAP directory.
  - **`man ldapdelete`** — Describes how to delete entries within an LDAP directory.
  - **`man ldapmodify`** — Describes how to modify entries within an LDAP directory.
  - **`man ldapsearch`** — Describes how to search for entries within an LDAP directory.

- **`man ldappasswd`** — Describes how to set or change the password of an LDAP user.
- **`man ldapcompare`** — Describes how to use the **`ldapcompare`** tool.
- **`man ldapwhoami`** — Describes how to use the **`ldapwhoami`** tool.
- **`man ldapmodrdn`** — Describes how to modify the RDNs of entries.

Server Applications

- **`man slapd`** — Describes command line options for the LDAP server.
- **`man slurpd`** — Describes command line options for the LDAP replication server.

Administrative Applications

- **`man slapadd`** — Describes command line options used to add entries to a **`slapd`** database.
- **`man slapcat`** — Describes command line options used to generate an LDIF file from a **`slapd`** database.
- **`man slapindex`** — Describes command line options used to regenerate an index based upon the contents of a **`slapd`** database.
- **`man slappasswd`** — Describes command line options used to generate user passwords for LDAP directories.

Configuration Files

- **`man ldap.conf`** — Describes the format and options available within the configuration file for LDAP clients.
- **`man slapd.conf`** — Describes the format and options available within the configuration file referenced by both the LDAP server applications (**`slapd`** and **`slurpd`**) and the LDAP administrative tools (**`slapadd`**, **`slapcat`**, and **`slapindex`**).

28.9.2. Useful Websites

- http://www.openldap.org/ — Home of the OpenLDAP Project. This website contains a wealth of information about configuring OpenLDAP as well as a future roadmap and version changes.
- http://www.padl.com/ — Developers of **`nss_ldap`** and **`pam_ldap`**, among other useful LDAP tools.
- http://www.kingsmountain.com/ldapRoadmap.shtml — Jeff Hodges' LDAP Road Map contains links to several useful FAQs and emerging news concerning the LDAP protocol.
- http://www.ldapman.org/articles/ — Articles that offer a good introduction to LDAP, including methods to design a directory tree and customizing directory structures.

28.9.3. Related Books

- *OpenLDAP by Example* by John Terpstra and Benjamin Coles; Prentice Hall.
- *Implementing LDAP* by Mark Wilcox; Wrox Press, Inc.
- *Understanding and Deploying LDAP Directory Services* by Tim Howes et al.; Macmillan Technical Publishing.

Chapter 29. Authentication Configuration

When a user logs in to a Red Hat Enterprise Linux system, the username and password combination must be verified, or *authenticated*, as a valid and active user. Sometimes the information to verify the user is located on the local system, and other times the system defers the authentication to a user database on a remote system.

The **Authentication Configuration Tool** provides a graphical interface for configuring user information retrieval from NIS, LDAP, and Hesiod servers. This tool also allows you to configure LDAP, Kerberos, and SMB as authentication protocols.

Note

If you configured a medium or high security level during installation (or with the **Security Level Configuration Tool**), then the firewall will prevent NIS (Network Information Service) authentication.

This chapter does not explain each of the different authentication types in detail. Instead, it explains how to use the **Authentication Configuration Tool** to configure them.

To start the graphical version of the **Authentication Configuration Tool** from the desktop, select the System (on the panel) > **Administration** > **Authentication** or type the command **`system-config-authentication`** at a shell prompt (for example, in an **XTerm** or a **GNOME** terminal).

Important

After exiting the authentication program, the changes made take effect immediately.

29.1. User Information

The **`User Information`** tab allows you to configure how users should be authenticated, and has several options. To enable an option, click the empty checkbox beside it. To disable an option, click the checkbox beside it to clear the checkbox. Click **`OK`** to exit the program and apply the changes.

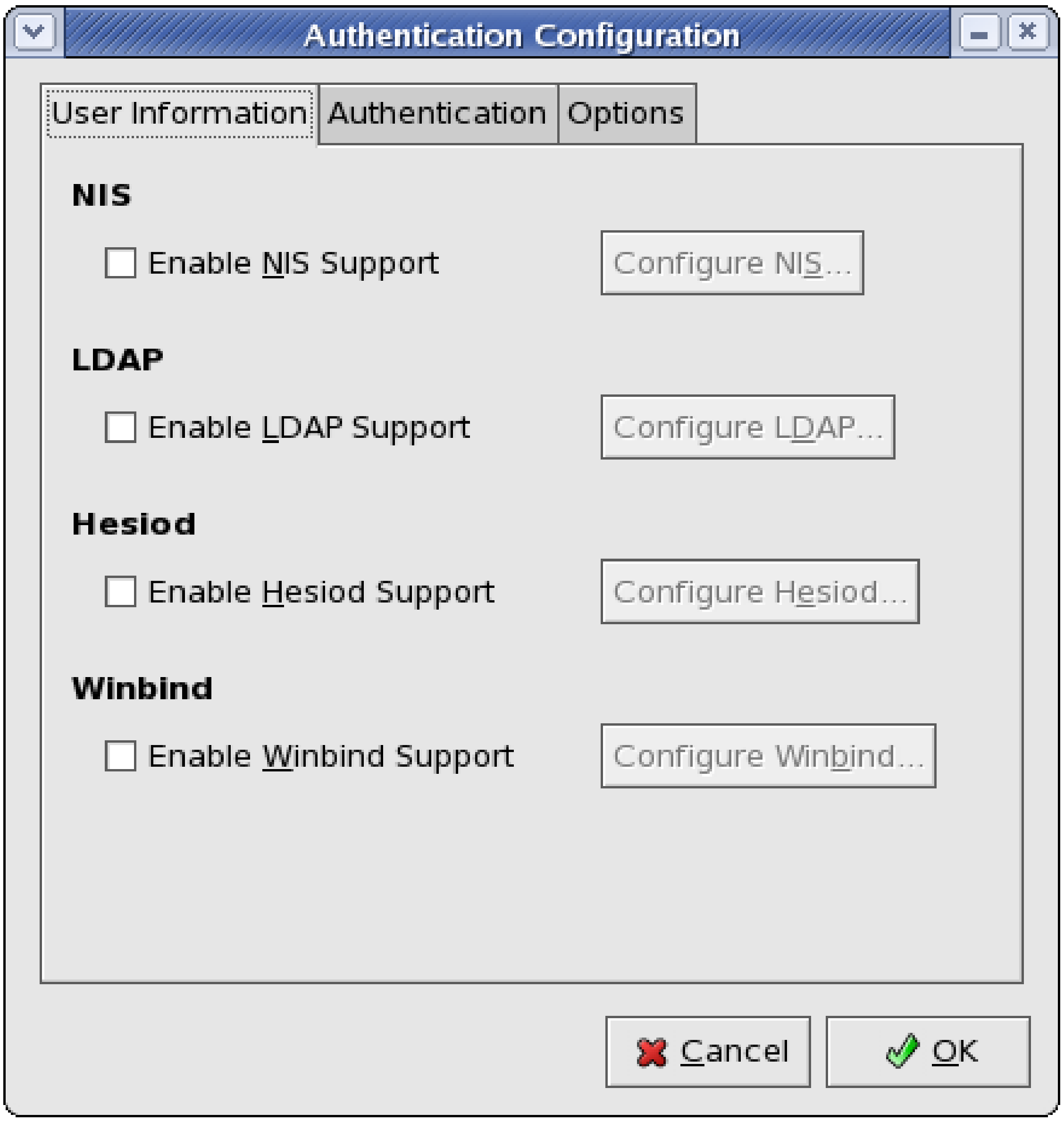


Figure 29.1. `User Information`

The following list explains what each option configures:

NIS

The **`Enable NIS Support`** option configures the system to connect to an NIS server (as an NIS client) for user and password authentication. Click the **`Configure NIS...`** button to specify the NIS domain and NIS server. If the NIS server is not specified, the daemon attempts to find it via broadcast.

The **`ypbind`** package must be installed for this option to work. If NIS support is enabled, the **`portmap`** and **`ypbind`** services are started and are also enabled to start at boot time.

For more information about NIS, refer to Section 48.2.3, “Securing NIS”.

LDAP

The **`Enable LDAP Support`** option instructs the system to retrieve user information via LDAP. Click the **`Configure LDAP...`** button to specify the following:

- **`LDAP Search Base DN`** — Specifies that user information should be retrieved using the listed Distinguished Name (DN).

- **`LDAP Server`** — Specifies the IP address of the LDAP server.
- **`Use TLS to encrypt connections`** — When enabled, Transport Layer Security will be used to encrypt passwords sent to the LDAP server. The **`Download CA Certificate`** option allows you to specify a URL from which to download a valid *CA (Certificate Authority) Certificate*. A valid CA Certificate must be in PEM (Privacy Enhanced Mail) format.

  For more information about CA Certificates, refer to Section 25.8.2, “An Overview of Certificates and Security”.

The **`openldap-clients`** package must be installed for this option to work.

For more information about LDAP, refer to Chapter 28, *Lightweight Directory Access Protocol (LDAP)*.

Hesiod

The **`Enable Hesiod Support`** option configures the system to retrieve information (including user information) from a remote Hesiod database. Click the **`Configure Hesiod...`** button to specify the following:

- **`Hesiod LHS`** — Specifies the domain prefix used for Hesiod queries.
- **`Hesiod RHS`** — Specifies the default Hesiod domain.

The **`hesiod`** package must be installed for this option to work.

For more information about Hesiod, refer to its man page using the command **`man hesiod`**. You can also refer to the **`hesiod.conf`** man page (**`man hesiod.conf`**) for more information on LHS and RHS.

Winbind

The **`Enable Winbind Support`** option configures the system to connect to a Windows Active Directory or a Windows domain controller. User information from the specified directory or domain controller can then be accessed, and server authentication options can be configured. Click the **`Configure Winbind...`** button to specify the following:

- **`Winbind Domain`** — Specifies the Windows Active Directory or domain controller to connect to.
- **`Security Model`** — Allows you to select a security model, which configures how clients should respond to Samba. The drop-down list allows you select any of the following:
  - **`user`** — This is the default mode. With this level of security, a client must first log in with a valid username and password. Encrypted passwords can also be used in this security mode.
  - **`server`** — In this mode, Samba will attempt to validate the username/password by authenticating it through another SMB server (for example, a Windows NT Server). If the attempt fails, the **`user`** mode will take effect instead.
  - **`domain`** — In this mode, Samba will attempt to validate the username/password by authenticating it through a Windows NT Primary or Backup Domain Controller, similar to how a Windows NT Server would.
  - **`ads`** — This mode instructs Samba to act as a domain member in an Active Directory Server (ADS) realm. To operate in this mode, the **`krb5-server`** package must be installed, and Kerberos must be configured properly.
- **`Winbind ADS Realm`** — When the **`ads`** Security Model is selected, this allows you to specify the ADS Realm the Samba server should act as a domain member of.

- **`Winbind Domain Controllers`** — Use this option to specify which domain controller **`winbind`** should use. For more information about domain controllers, please refer to Section 22.6.3, "Domain Controller".
- **`Template Shell`** — When filling out the user information for a Windows NT user, the **`winbindd`** daemon uses the value chosen here to to specify the login shell for that user.

For more information about the **`winbind`** service, refer to **`winbindd`** under Section 22.2, "Samba Daemons and Related Services".

29.2. Authentication

The **`Authentication`** tab allows for the configuration of network authentication methods. To enable an option, click the empty checkbox beside it. To disable an option, click the checkbox beside it to clear the checkbox.

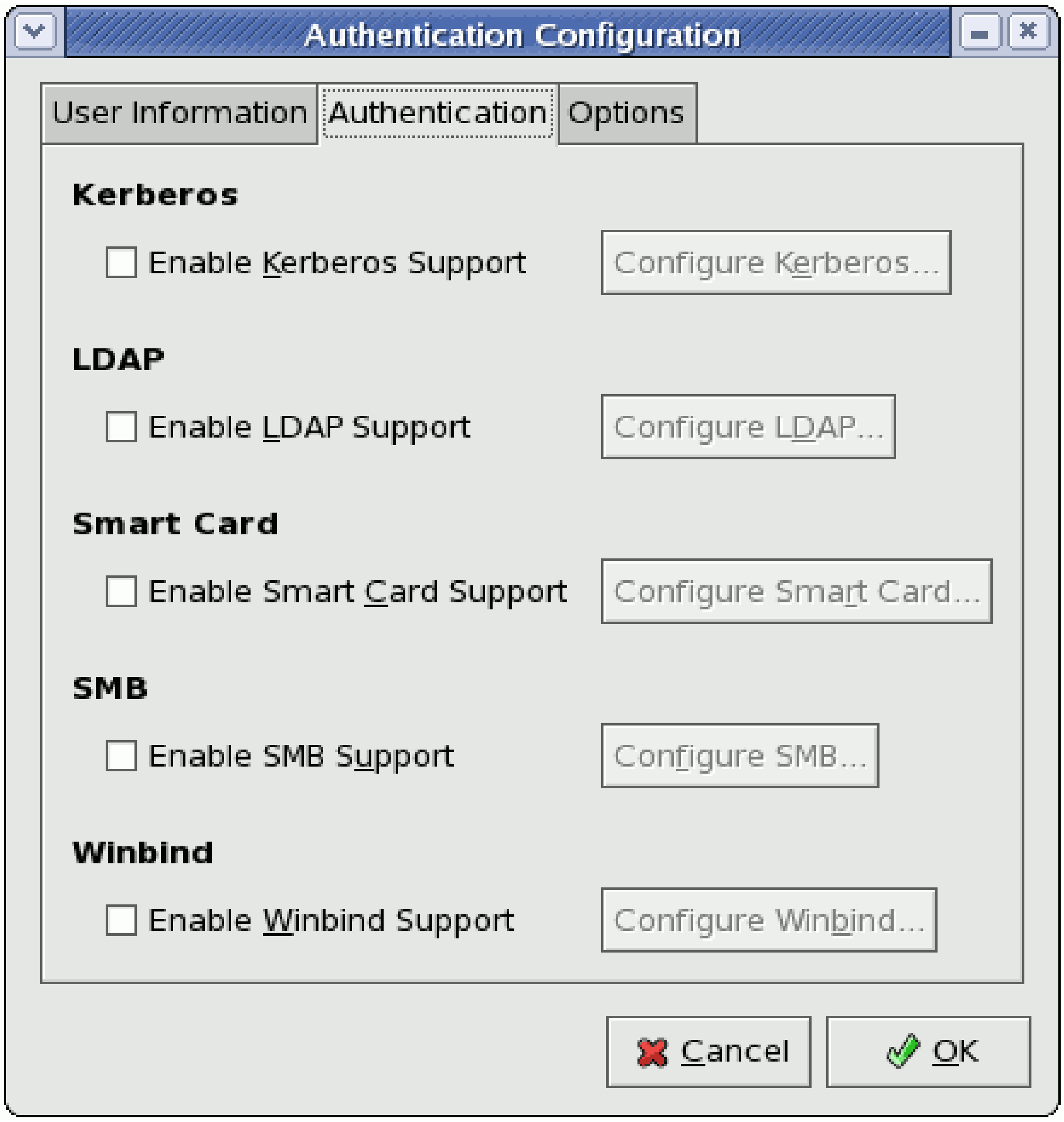


Figure 29.2. `Authentication`

The following explains what each option configures:

Kerberos

The **`Enable Kerberos Support`** option enables Kerberos authentication. Click the **`Configure Kerberos...`** button to open the **Kerberos Settings** dialogue and configure the following:

- **`Realm`** — Configures the realm for the Kerberos server. The realm is the network that uses Kerberos, composed of one or more KDCs and a potentially large number of clients.
- **`KDC`** — Defines the Key Distribution Center (KDC), which is the server that issues Kerberos tickets.
- **`Admin Servers`** — Specifies the administration server(s) running **`kadmind`**.

The **Kerberos Settings** dialogue also allows you to use DNS to resolve hosts to realms and locate KDCs for realms.

The **`krb5-libs`** and **`krb5-workstation`** packages must be installed for this option to work. For more information about Kerberos, refer to Section 48.6, “Kerberos”.

LDAP

The **`Enable LDAP Support`** option instructs standard PAM-enabled applications to use LDAP for authentication. The **`Configure LDAP...`** button allows you to configure LDAP support with options identical to those present in **`Configure LDAP...`** under the **User Information** tab. For more information about these options, refer to Section 29.1, “User Information”.

The **`openldap-clients`** package must be installed for this option to work.

Smart Card

The **`Enable Smart Card Support`** option enables Smart Card authentication. This allows users to log in using a certificate and key associated stored on a smart card. Click the **`Configure Smart Card...`** button for more options.

The **`pam_pkcs11`** and **`coolkey`** packages must be installed for this option to work. For more information about smart cards, refer to Section 48.3.1.3, “Supported Smart Cards” under Section 48.3, “Single Sign-on (SSO)”.

SMB

The **`Enable SMB Support`** option configures PAM to use a Server Message Block (SMB) server to authenticate users. SMB refers to a client-server protocol used for cross-system communication; it is also the protocol used by Samba to appear as a Windows server to Windows clients. Click the **`Configure SMB...`** button to specify the following:

- **`Workgroup`** — Specifies the SMB workgroup to use.
- **`Domain Controllers`** — Specifies the SMB domain controllers to use.

Winbind

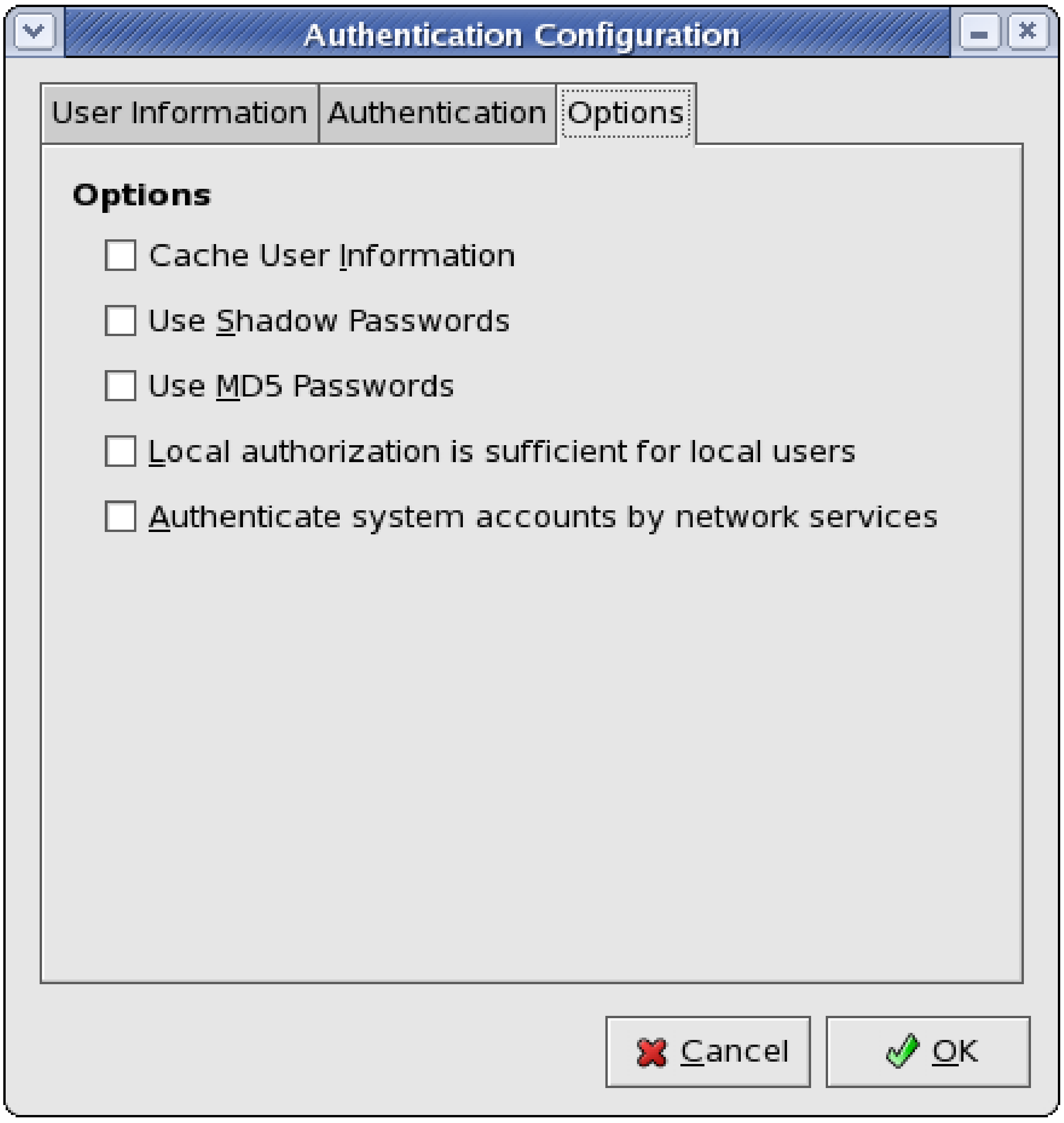
The **`Enable Winbind Support`** option configures the system to connect to a Windows Active Directory or a Windows domain controller. User information from the specified directory or domain controller can then be accessed, and server authentication options can be configured.

The **`Configure Winbind...`** options are identical to those in the **`Configure Winbind...`** button on the **`User Information`** tab. Please refer to Winbind (under Section 29.1, “User Information”) for more information.

29.3. Options

.

This tab allows other configuration options, as listed below.



Options

- [ ] Cache User Information
- [ ] Use Shadow Passwords
- [ ] Use MD5 Passwords
- [ ] Local authorization is sufficient for local users
- [ ] Authenticate system accounts by network services

Figure 29.3. `Options`

`Cache User Information`

Select this option to enable the name service cache daemon (**`nscd`**) and configure it to start at boot time.

The **`nscd`** package must be installed for this option to work. For more information about**`nscd`**, refer to its man page using the command **`man nscd`**.

`Use Shadow Passwords`

Select this option to store passwords in shadow password format in the **`/etc/shadow`** file instead of **`/etc/passwd`**. Shadow passwords are enabled by default during installation and are highly recommended to increase the security of the system.

The **`shadow-utils`** package must be installed for this option to work. For more information about shadow passwords, refer to Section 37.6, “Shadow Passwords”.

`Use MD5 Passwords`

Select this option to enable MD5 passwords, which allows passwords to be up to 256 characters instead of eight characters or less. It is selected by default during installation and is highly recommended for increased security.

`Local authorization is sufficient for local users`

When this option is enabled, the system will not check authorization from network services (such as LDAP or Kerberos) for user accounts maintained in its **`/etc/passwd`** file.

`Authenticate system accounts by network services`

Enabling this option configures the system to allow network services (such as LDAP or Kerberos) to authenticate system accounts (including root) in the machine.

29.4. Command Line Version

The **Authentication Configuration Tool** can also be run as a command line tool with no interface. The command line version can be used in a configuration script or a kickstart script. The authentication options are summarized in Table 29.1, “Command Line Options”.

Note

These options can also be found in the **`authconfig`** man page or by typing **`authconfig --help`** at a shell prompt.

Table 29.1. Command Line Options

Option	Description
`--enableshadow`	Enable shadow passwords
`--disableshadow`	Disable shadow passwords
`--enablemd5`	Enable MD5 passwords
`--disablemd5`	Disable MD5 passwords
`--enablenis`	Enable NIS
`--disablenis`	Disable NIS
`--nisdomain=<`*`domain`*`>`	Specify NIS domain
`--nisserver=<`*`server`*`>`	Specify NIS server
`--enableldap`	Enable LDAP for user information
`--disableldap`	Disable LDAP for user information
`--enableldaptls`	Enable use of TLS with LDAP
`--disableldaptls`	Disable use of TLS with LDAP
`--enableldapauth`	Enable LDAP for authentication
`--disableldapauth`	Disable LDAP for authentication
`--ldapserver=<`*`server`*`>`	Specify LDAP server
`--ldapbasedn=<`*`dn`*`>`	Specify LDAP base DN
`--enablekrb5`	Enable Kerberos
`--disablekrb5`	Disable Kerberos

Option	Description
`--krb5kdc=<`*`kdc`*`>`	Specify Kerberos KDC
`--krb5adminserver=<`*`server`*`>`	Specify Kerberos administration server
`--krb5realm=<`*`realm`*`>`	Specify Kerberos realm
`--enablekrb5kdcdns`	Enable use of DNS to find Kerberos KDCs
`--disablekrb5kdcdns`	Disable use of DNS to find Kerberos KDCs
`--enablekrb5realmdns`	Enable use of DNS to find Kerberos realms
`--disablekrb5realmdns`	Disable use of DNS to find Kerberos realms
`--enablesmbauth`	Enable SMB
`--disablesmbauth`	Disable SMB
`--smbworkgroup=<`*`workgroup`*`>`	Specify SMB workgroup
`--smbservers=<`*`server`*`>`	Specify SMB servers
`--enablewinbind`	Enable winbind for user information by default
`--disablewinbind`	Disable winbind for user information by default
`--enablewinbindauth`	Enable winbindauth for authentication by default
`--disablewinbindauth`	Disable winbindauth for authentication by default
`--smbsecurity=<`*`user|server|domain|ads`*`>`	Security mode to use for Samba and winbind
`--smbrealm=<`*`STRING`*`>`	Default realm for Samba and winbind when **`security=ads`**
`--smbidmapuid=<`*`lowest-highest`*`>`	UID range winbind assigns to domain or ADS users
`--smbidmapgid=<`*`lowest-highest`*`>`	GID range winbind assigns to domain or ADS users
`--winbindseparator=<`*`\`*`>`	Character used to separate the domain and user part of winbind usernames if **`winbindusedefaultdomain`** is not enabled
`--winbindtemplatehomedir=<`*`/home/%D/%U`*`>`	Directory that winbind users have as their home
`--winbindtemplateprimarygroup=<`*`nobody`*`>`	Group that winbind users have as their primary group
`--winbindtemplateshell=<`*`/bin/false`*`>`	Shell that winbind users have as their default login shell
`--enablewinbindusedefaultdomain`	Configures winbind to assume that users with no domain in their usernames are domain users
`--disablewinbindusedefaultdomain`	Configures winbind to assume that users with no domain in their usernames are not domain users
`--winbindjoin=<`*`Administrator`*`>`	Joins the winbind domain or ADS realm now as this administrator
`--enablewins`	Enable WINS for hostname resolution

Option	Description
`--disablewins`	Disable WINS for hostname resolution
`--enablehesiod`	Enable Hesiod
`--disablehesiod`	Disable Hesiod
`--hesiodlhs=<lhs>`	Specify Hesiod LHS
`--hesiodrhs=<rhs>`	Specify Hesiod RHS
`--enablecache`	Enable **`nscd`**
`--disablecache`	Disable **`nscd`**
`--nostart`	Do not start or stop the **`portmap`**, **`ypbind`**, or **`nscd`** services even if they are configured
`--kickstart`	Do not display the user interface
`--probe`	Probe and display network defaults

Chapter 30. Using and Caching Credentials with SSSD

The System Security Services Daemon (SSSD) provides access to different identity and authentication providers. SSSD is an intermediary between local clients and any configured data store. The local clients connect to SSSD and then SSSD contacts the external providers. This brings a number of benefits for administrators:

- *Reducing the load on identification/authentication servers.* Rather than having every client service attempt to contact the identification server directly, all of the local clients can contact SSSD which can connect to the identification server or check its cache.
- *Permitting offline authentication.* SSSD can optionally keep a cache of user identities and credentials that it retrieves from remote services. This allows users to authenticate to resources successfully, even if the remote identification server is offline or the local machine is offline.
- *Using a single user account.* Remote users frequently have two (or even more) user accounts, such as one for their local system and one for the organizational system. This is necessary to connect to a virtual private network (VPN). Because SSSD supports caching and offline authentication, remote users can connect to network resources simply by authenticating to their local machine and then SSSD maintains their network credentials.

The System Security Services Daemon does not require any additional configuration or tuning to work with the Authentication Configuration Tool. However, SSSD can work with other applications, and the daemon may require configuration changes to improve the performance of those applications.

30.1. About the sssd.conf File

SSSD services and domains are configured in a **.conf** file. The default file is **/etc/sssd/sssd.conf**, although alternative files can be passed to SSSD by using the **-c** option with the **sssd** command:

```
# sssd -c /etc/sssd/customfile.conf
```

Both services and domains are configured individually, in separate sections on the configuration identified by *[type/name]* divisions, such as **[domain/LDAP]**. The configuration file uses simple *key* = *value* lines to set the configuration. Comment lines are set by either a hash sign (#) or a semicolon (;)

For example:

```
[section]
# Comment line
key1 = val1
key10 = val1,val2
```

30.2. Starting and Stopping SSSD

Note

Configure at least one domain before starting SSSD for the first time. See Section 30.4, “Creating Domains”.

Either the **service** command or the **/etc/init.d/sssd** script can start SSSD. For example:

```
# service sssd start
```

By default, SSSD is configured not to start automatically. To change this behavior, use the **chkconfig** command:

```
[root@server ~]# chkconfig sssd on
```

30.3. Configuring SSSD to Work with System Services

SSSD worked with specialized services that run in tandem with the SSSD process itself. SSSD and its associated services are configured in the **sssd.conf** file. on sections. The **[sssd]** section also lists the services that are active and should be started when **sssd** starts within the **services** directive.

SSSD currently provides several services:

- A Name Service Switch (NSS) provider service that answers name service requests from the **sssd_nss** module. This is configured in the **[nss]** section of the SSSD configuration.
- A PAM provider service that manages a PAM conversation through the **sssd_pam** module. This is configured in the **[pam]** section of the configuration.
- **monitor**, a special service that monitors and starts or restarts all other SSSD services. Its options are specified in the **[sssd]** section of the **/etc/sssd/sssd.conf** configuration file.

Note

If a DNS lookup fails to return an IPv4 address for a hostname, SSSD attempts to look up an IPv6 address before returning a failure. This only ensures that the asynchronous resolver identifies the correct address.

The hostname resolution behavior is configured in the ***lookup family order*** option in the **sssd.conf** configuration file.

30.3.1. Configuring NSS Services

SSSD provides an NSS module, **sssd_nss**, which instructs the system to use SSSD to retrieve user information. The NSS configuration must include a reference to the SSSD module, and then the SSSD configuration sets how SSSD interacts with NSS.

30.3.1.1. About NSS Service Maps and SSSD

The Name Service Switch (NSS) provides a central configuration for services to look up a number of configuration and name resolution services. NSS provides one method of mapping system identities and services with configuration sources.

SSSD works with NSS as a provider services for several types of NSS maps:

- Passwords (**passwd**)
- User groups (**shadow**)

- Groups (**groups**)
- Netgroups (**netgroups**)

30.3.1.2. Configuring NSS Services to Use SSSD

NSS can use multiple identity and configuration providers for any and all of its service maps. The default is to use system files for services; for SSSD to be included, the **nss_sss** module has to be included for the desired service type.

Use the Authentication Configuration tool to enable SSSD. This automatically configured the **nsswitch.conf** file to use SSSD as a provider.

```
[root@server ~]# authconfig --enablesssd --update
```

This automatically configures the password, shadow, group, and netgroups services maps to use the SSSD module:

```
passwd:     files sss
shadow:     files sss
group:      files sss

netgroup:   files sss
```

30.3.1.3. Configuring SSSD to Work with NSS

The options and configuration that SSSD uses to service NSS requests are configured in the SSSD configuration file, in the **[nss]** services section.

1. Open the **sssd.conf** file.

   ```
   [root@server ~]# vim /etc/sssd/sssd.conf
   ```

2. Make sure that NSS is listed as one of the services that works with SSSD.

   ```
   [sssd]
   config_file_version = 2
   reconnection_retries = 3
   sbus_timeout = 30
   services = nss, pam
   ```

3. In the **[nss]** section, change any of the NSS parameters. These are listed in Table 30.1, “SSSD [nss] Configuration Parameters”.

   ```
   [nss]
   filter_groups = root
   filter_users = root
   reconnection_retries = 3
   entry_cache_timeout = 300
   entry_cache_nowait_percentage = 75
   ```

4. Restart SSSD.

```
[root@server ~]# service sssd restart
```

Table 30.1. SSSD [nss] Configuration Parameters

Parameter	Value Format	[root@server ~] Description
enum_cache_timeout	integer	Specifies how long, in seconds, **`sssd_nss`** should cache requests for information about all users (enumerations).
entry_cache_nowait_percentage	integer	Specifies how long **`sssd_nss`** should return cached entries before refreshing the cache. Setting this to zero (**`0`**) disables the entry cache refresh. This configures the entry cache to update entries in the background automatically if they are requested if the time before the next update is a certain percentage of the next interval. For example, if the interval is 300 seconds and the cache percentage is 75, then the entry cache will begin refreshing when a request comes in at 225 seconds — 75% of the interval. The allowed values for this option are 0 to 99, which sets the percentage based on the **`entry_cache_timeout`** value. The default value is 50%.
entry_negative_timeout	integer	Specifies how long, in seconds, **`sssd_nss`** should cache *negative* cache hits. A negative cache hit is a query for an invalid database entries, including non-existent entries.
filter_users, filter_groups	string	Tells SSSD to exclude certain users from being fetched from the NSS database. This is particularly useful for system accounts such as **`root`**.
filter_users_in_groups	Boolean	Sets whether users listed in the **`filter_users`** list appear in group memberships when performing group lookups. If set to **`false`**, group lookups return all users that are members of that group. If not specified, this value defaults to **`true`**, which filters the group member lists.

30.3.2. Configuring the PAM Service

Warning

A mistake in the PAM configuration file can lock users out of the system completely. Always back up the configuration files before performing any changes, and keep a session open so that any changes can be reverted.

SSSD provides a PAM module, **sssd_pam**, which instructs the system to use SSSD to retrieve user information. The PAM configuration must include a reference to the SSSD module, and then the SSSD configuration sets how SSSD interacts with PAM.

To configure the PAM service:

1. The Authentication Configuration tool automatically writes to the **/etc/pam.d/system-auth-ac** file, which is symlinked to **/etc/pam.d/system-auth**. Any changes made to **/etc/pam.d/system-auth** are overwritten the next time that **authconfig** is run.

   So, remove the **/etc/pam.d/system-auth** symlink.

   ```
   [root@server ~]# rm /etc/pam.d/system-auth
   rm: remove symbolic link `/etc/pam.d/system-auth'? y
   ```

2. Create a new **/etc/pam.d/system-auth-local** file. One easy way to do this is simply to copy the **/etc/pam.d/system-auth-ac** file.

   ```
   [root@server ~]# cp /etc/pam.d/system-auth-ac /etc/pam.d/system-auth-
   local
   ```

3. Create a new symlink between the **/etc/pam.d/system-auth-local** file and **/etc/pam.d/system-auth**.

   ```
   [root@server ~]# ln -s /etc/pam.d/system-auth-local /etc/pam.d/system-
   auth
   ```

4. Edit the **/etc/pam.d/system-auth-local** file, and add all of the SSSD modules to the PAM configuration:

   ```
   #%PAM-1.0
   ...
   auth         sufficient    pam_sss.so use_first_pass
   auth         required      pam_deny.so

   ...
   account [default=bad success=ok user_unknown=ignore] pam_sss.so
   account      required      pam_permit.so

   ...
   password     sufficient    pam_sss.so use_authtok
   password     required      pam_deny.so

   ...
   session      sufficient    pam_sss.so
   ```

```
session     required      pam_unix.so
```

These modules can be set to **include** statements, as necessary.

5. Open the **sssd.conf** file.

```
# vim /etc/sssd/sssd.conf
```

6. Make sure that PAM is listed as one of the services that works with SSSD.

```
[sssd]
config_file_version = 2
reconnection_retries = 3
sbus_timeout = 30
services = nss, pam
```

7. In the **[pam]** section, change any of the PAM parameters. These are listed in Table 30.2, "SSSD [pam] Configuration Parameters".

```
[pam]
reconnection_retries = 3
offline_credentials_expiration = 2
offline_failed_login_attempts = 3
offline_failed_login_delay = 5
```

8. Restart SSSD.

```
[root@server ~]# service sssd restart
```

Table 30.2. SSSD [pam] Configuration Parameters

Parameter	Value Format	Description
offline_credentials_expiration	integer	Sets how long, in days, to allow cached logins if the authentication provider is offline. This value is measured from the last successful online login. If not specified, this defaults to zero (**0**), which is unlimited.
offline_failed_login_attempts	integer	Sets how many failed login attempts are allowed if the authentication provider is offline. If not specified, this defaults to zero (**0**), which is unlimited.

Parameter	Value Format	Description
offline_failed_login_delay	integer	Sets how long to prevent login attempts if a user hits the failed login attempt limit. If set to zero (**`0`**), the user cannot authenticate while the provider is offline once he hits the failed attempt limit. Only a successful online authentication can re-enable offline authentication. If not specified, this defaults to five (5).

30.4. Creating Domains

SSSD recognizes *domains*, which are associated with the different identity servers. Domains are a combination of an identity provider and an authentication method. SSSD works with LDAP identity providers (including OpenLDAP, Red Hat Directory Server, and Microsoft Active Directory) and can use native LDAP authentication or Kerberos authentication.

As long as they belong to different domains, SSSD can recognize different users with the same username. For example, SSSD can successfully authenticate both **`jsmith`** in the **`ldap.example.com`** domain and **`jsmith`** in the **`ldap.otherexample.com`** domain. SSSD allows requests using fully-qualified domain names, so requesting information for **`jsmith@ldap.example.com`** returns the proper user account. Specifying only the username returns the user for whichever domain comes first in the lookup order.

Note

SSSD has a **`filter_users`** option, which excludes the specified users from being returned in a search.

Configuring a domain defines both *where* user information is stored and *how* those users are allowed to authenticate to the system. The possible combinations are listed in Table 30.3, "Identity Store and Authentication Type Combinations".

- Section 30.4.1, "General Rules and Options for Configuring a Domain"
- Section 30.4.2, "Configuring an LDAP Domain"
- Section 30.4.3, "Configuring Kerberos Authentication with a Domain"
- Section 30.4.4, "Configuring a Proxy Domain"

Table 30.3. Identity Store and Authentication Type Combinations

Identification Provider	Authentication Provider
LDAP	LDAP
LDAP	Kerberos
proxy	LDAP
proxy	Kerberos
proxy	proxy

30.4.1. General Rules and Options for Configuring a Domain

A domain configuration defines the *identity provider*, the *authentication provider*, and any specific configuration to access the information in those providers. There are two types of identity providers — LDAP and proxy —three types of authentication providers — LDAP, Kerberos, and proxy. The identity and authentication providers can be configured in any combination in a domain entry.

Along with the domain entry itself, the domain name must be added to the list of domains that SSSD will query. For example:

```
domains = LOCAL,Name

[domain/Name]
id_provider = type
auth_provider = type
provider_specific = value
global = value
```

global attributes are available to any type of domain, such as cache and time out settings. Each identity and authentication provider has its own set of required and optional configuration parameters.

Table 30.4. General [domain] Configuration Parameters

Parameter	Value Format	Description
id_provider	string	Specifies the data provider identity backend to use for this domain. The supported identity backends are: • ldap • ipa, compatible with FreeIPA version 2.x • proxy for a legacy NSS provider, such as **`nss_nis`**. Using a proxy ID provider also requires specifying the legacy NSS library to load to start successfully, set in the **`proxy_lib_name`** option. • local, the SSSD internal local provider
auth_provider	string	Sets the authentication provider used for the domain. The default value for this option is the value of **`id_provider`**. The supported authentication providers are ldap, ipa, krb5 (Kerberos), proxy, and none.

Parameter	Value Format	Description
min_id,max_id	integer	*Optional.* Specifies the UID and GID range for the domain. If a domain contains entries that are outside that range, they are ignored. The default value for **`min_id`** is **`1`**; the default value for **`max_id`** is **`0`**, which is unlimited. **Important** The default **`min_id`** value is the same for all types of identity provider. If LDAP directories are using UID numbers that start at one, it could cause conflicts with users in the local **`/etc/passwd`** file. To avoid these conflicts, set **`min_id`** to **`1000`** or higher as possible.

Parameter	Value Format	Description
enumerate	Boolean	*Optional.* Specifies whether to list the users and groups of a domain. Enumeration means that the entire set of available users and groups on the remote source is cached on the local machine. When enumeration is disabled, users and groups are only cached as they are requested. **Warning** When enumeration is enabled, reinitializing a client results in a complete refresh of the entire set of available users and groups from the remote source. Similarly, when SSSD is connected to a new server, the entire set of available users and groups from the remote source is pulled and cached on the local machine. In a domain with a large number of clients connected to a remote source, this refresh process can harm the network performance because of frequent queries from the clients. If the set of available users and groups is large enough, it degrades client performance as well. The default value for this parameter is **`false`**, which disables enumeration.
cache_credentials	Boolean	*Optional.* Specifies whether to store user credentials in the local SSSD domain database cache. The default value for this parameter is **`false`**. Set this value to **`true`** for domains other than the LOCAL domain to enable offline authentication.
entry_cache_timeout	integer	*Optional.* Specifies how long, in seconds, SSSD should cache *positive* cache hits. A positive cache hit is a successful query.

Parameter	Value Format	Description
use_fully_qualified_names	Boolean	*Optional.* Specifies whether requests to this domain require fully-qualified domain names. If set to **`true`**, all requests to this domain must use fully-qualified domain names. It also means that the output from the request displays the fully-qualified name. Restricting requests to fully-qualified user names allows SSSD to differentiate between domains with users with conflicting usernames. If ***`use_fully_qualified_names`*** is set to **`false`**, it is possible to use the fully-qualified name in the requests, but only the simplified version is displayed in the output. SSSD can only parse names based on the domain name, not the realm name. The same name can be used for both domains and realms, however.

30.4.2. Configuring an LDAP Domain

An LDAP domain simply means that SSSD uses an LDAP directory as the identity provider (and, optionally, also as an authentication provider). SSSD supports several major directory services:

- Red Hat Directory Server
- OpenLDAP
- Microsoft Active Directory 2008, with Subsystem for UNIX-based Applications

Note

DNS service discovery allows the LDAP backend to find the appropriate DNS servers to connect to automatically using a special DNS query.

- Section 30.4.2.1, “Parameters for Configuring an LDAP Domain”
- Section 30.4.2.2, “LDAP Domain Example”
- Section 30.4.2.3, “Active Directory Domain Example”
- Section 30.4.2.4, “Using IP Addresses in Certificate Subject Names”

30.4.2.1. Parameters for Configuring an LDAP Domain

An LDAP directory can function as both an identity provider and an authentication provider. The configuration requires enough information to identify and connect to the user directory in the LDAP server, but the way that those connection parameters are defined is flexible.

Other options are available to provide more fine-grained control, like specifying a user account to use to connect to the LDAP server or using different LDAP servers for password operations. The most common options are listed in Table 30.5, "LDAP Domain Configuration Parameters". All of the options listed in Section 30.4.1, "General Rules and Options for Configuring a Domain" are also available for LDAP domains.

Note

Many other options are listed in the man page for LDAP domain configuration, `sssd-ldap(5)`.

Table 30.5. LDAP Domain Configuration Parameters

Parameter	Description
ldap_uri	Gives a comma-separated list of the URIs of the LDAP servers to which SSSD will connect. The list is given in order of preference, so the first server in the list is tried first. Listing additional servers provides failover protection. This can be detected from the DNS SRV records if it is not given.
ldap_search_base	Gives the base DN to use for performing LDAP user operations.
ldap_tls_reqcert	Specifies how to check for SSL server certificates in a TLS session. There are four options: • *never* disables requests for certificates. • *allow* requests a certificate, but proceeds normally even if no certificate is given or a bad certificate is given. • *try* requests a certificate and proceeds normally if no certificate is given, If a bad certificate is given, the session terminates. • *demand* and *hard* are the same option. This requires a valid certificate or the session is terminated. The default is *hard*.
ldap_tls_cacert	Gives the full path and file name to the file that contains the CA certificates for all of the CAs that SSSD recognizes. SSSD will accept any certificate issued by these CAs. This uses the OpenLDAP system defaults if it is not given explicitly.
ldap_referrals	Sets whether SSSD will use LDAP referrals, meaning forwarding queries from one LDAP database to another. SSSD supports database-level and subtree referrals. For referrals within the same LDAP server, SSSD will adjust the DN of the entry being queried. For referrals that go to different LDAP servers, SSSD does an exact match on the DN. Setting this value to `true` enables referrals. This is the default.

Parameter	Description

ldap_schema	Sets what version of schema to use when searching for user entries. This can be either **`rfc2307`** or **`rfc2307bis`**. The default is **`rfc2307`**.
	
	In RFC 2307, group objects use a multi-valued attribute, ***`memberuid`***, which lists the names of the users that belong to that group. In RFC 2307bis, group objects use the ***`member`*** attribute, which contains the full distinguished name (DN) of a user or group entry. RFC 2307bis allows nested groups usning the ***`member`*** attribute. Because these different schema use different definitions for group membership, using the wrong LDAP schema with SSSD can affect both viewing and managing network resources, even if the appropriate permissions are in place.
	
	For example, with RFC 2307bis, all groups are returned when using nested groups or primary/secondary groups.
	
	`$ id`
	`uid=500(myserver) gid=500(myserver)`
	`groups=500(myserver),510(myothergro`
	`up)`
	
	If SSSD is using RFC 2307 schema, only the primary group is returned.
	
	This setting only affects how SSSD determines the group members. It does not change the actual user data.

ldap_search_timeout	Sets the time, in seconds, that LDAP searches are allowed to run before they are canceled and cached results are returned. This defaults to five when the **`enumerate`** value is false and defaults to 30 when **`enumerate`** is true. When an LDAP search times out, SSSD automatically switches to offline mode.
ldap_network_timeout	Sets the time, in seconds, SSSD attempts to poll an LDAP server after a connection attempt fails. The default is six seconds.
ldap_opt_timeout	Sets the time, in seconds, to wait before aborting synchronous LDAP operations if no response is received from the server. This option also controls the timeout when communicating with the KDC in case of a SASL bind. The default is five seconds.

30.4.2.2. LDAP Domain Example

The LDAP configuration is very flexible, depending on your specific environment and the SSSD behavior. These are some common examples of an LDAP domain, but the SSSD configuration is not limited to these examples.

Note

Along with creating the domain entry, add the new domain to the list of domains for SSSD to query in the **sssd.conf** file. For example:

```
domains = LOCAL,LDAP1,AD,PROXYNIS
```

Example 30.1. A Basic LDAP Domain Configuration

An LDAP domain requires three things:

- An LDAP server
- The search base
- A way to establish a secure connection

The last item depends on the LDAP environment. SSSD requires a secure connection since it handles sensitive information. This connection can be a dedicated TLS/SSL connection or it can use Start TLS.

Using a dedicated TLS/SSL connection simply uses an LDAPS connection to connect to the server and is therefore set as part of the **ldap_uri** option:

```
# An LDAP domain
[domain/LDAP]
enumerate = false
cache_credentials = true

id_provider = ldap
auth_provider = ldap

ldap_uri = ldaps://ldap.example.com:636
ldap_search_base = dc=example,dc=com
```

Using Start TLS requires a way to input the certificate information to establish a secure connection dynamically over an insecure port. This is done using the **ldap_id_use_start_tls** option to use Start TLS and then **ldap_tls_cacert** to identify the CA certificate which issued the SSL server certificates.

```
# An LDAP domain
[domain/LDAP]
enumerate = false
cache_credentials = true

id_provider = ldap
auth_provider = ldap

ldap_uri = ldap://ldap.example.com
```

```
ldap_search_base = dc=example,dc=com
ldap_id_use_start_tls = true
ldap_tls_reqcert = demand
ldap_tls_cacert = /etc/pki/tls/certs/ca-bundle.crt
```

30.4.2.3. Active Directory Domain Example

For SSSD to work with an Active Directory domain, both the Active Directory domain and the local system have to be configured specially to communicate with one another.

Note

The Microsoft Active Directory documentation has complete procedures for configuring the Active Directory domain.

1. Using **authconfig**, set the Linux client to use Active Directory as its LDAP identity provider. For example:

   ```
   authconfig --enableldap --enableldapauth --
   ldapserver=ldap://ad.example.com:389 --enablekrb5 --krb5realm AD-
   REALM.EXAMPLE.COM --krb5kdc ad-kdc.example.com:88 --krb5adminserver
   ad-kdc.example.com:749 --update
   ```

   The **authconfig** command is described in Section 29.4, “Command Line Version”.

2. Create the Active Directory Domain Services role.

3. Add the Identity Management for UNIX service to the Active Directory Domain Services role. Use the Unix NIS domain as the domain name in the configuration.

4. On the Active Directory server, create a new **Computer** object with the name of the Linux client.

   a. In the **Administrative Tools** menu, select the **Active Directory Users and Computers** application.

   b. Expand the Active Directory root object, such as **ad.example.com**.

   c. Right-click **Computers**, and select the **New** and the **Computer** item.

   d. Enter the name for the Linux client, such as **rhel-server**, and click **OK**.

   e. Expand the **Computers** object.

   f. Right-click the **rhel-server** object, and select **Properties**.

   g. In the **UNIX Attributes**, enter the name of the Linux NIS domain and the IP address of the Linux server.

      Click **OK**.

5. From the command prompt on the Active Directory server, create a machine account, password, and UPN for the Linux host principal.

   ```
   C:\> setspn -A host/rhel-server.example.com@AD-REALM.EXAMPLE.COM rhel-
   ```

```
server
Registering ServicePrincipalNames for CN=rhel
server,CN=Computers,DC=ad,DC=example,DC=com
        host/rhel server.example.com@AD-REALM.EXAMPLE.COM
Updated object

C:\> setspn -L rhel-server
Registered ServicePrincipalNames for CN=rhel
server,CN=Computers,DC=ad,DC=example,DC=com:
        host/rhel server.example.com@AD-REALM.EXAMPLE.COM

C:\> ktpass /princ host/rhel-server.example.com@AD-REALM.EXAMPLE.COM
/out rhel-server.keytab /crypto all /ptype KRB5_NT_PRINCIPAL -desonly
/mapuser AD\rhel-server$ +rndPass

Targeting domain controller:
    ad.example.com
Using legacy password setting method
Successfully mapped host/rhel server.redhat.com
... 8< ...
```

6. Copy the keytab from the Active Directory server to the Linux client, and save it as **`/etc/krb5.keytab`**.

7. On the Linux system, reset the permissions and owner for the keytab file.

```
[root@rhel-server ~]# chown root:root /etc/krb5.keytab

[root@rhel-server ~]# chmod 0600 /etc/krb5.keytab
```

8. Restore the SELinux file permissions for the keytab.

```
[root@rhel-server ~]# restorecon /etc/krb5.keytab
```

9. Verify that the host can connect to the Active Directory domain.

```
[root@rhel-server ~]# kinit -k -t /etc/krb5.keytab host/rhel-
server.example.com@AD-REALM.EXAMPLE.COM
```

10. On the Active Directory server, create a a group for the Linux users.

    a. Create a new group named *unixusers*.

    b. Open the **`unixusers`** group and open the **`Unix Attributes`** tab.

    c. Configure the Unix settings:

       - The NIS domain
       - The UID
       - The login shell, to **`/bin/bash`**
       - The home directory, to **`/home/aduser`**
       - The primary group name, to **`unixusers`**

11. Then, configure the SSSD domain on the Linux machine.

Example 30.2. An Active Directory 2008 Domain

```
[root@rhel-server ~]# vim /etc/sssd/sssd.conf

[sssd]
config_file_version = 2
domains = ad.example.com
services = nss, pam

[nss]

[pam]

[domain/ad.example.com]
cache_credentials = true
enumerate = false

id_provider = ldap
auth_provider = krb5
chpass_provider = krb5
access_provider = ldap

ldap_sasl_mech = GSSAPI
ldap_sasl_authid = host/rhel-server.example.com@AD-REALM.EXAMPLE.COM


ldap_schema = rfc2307bis

ldap_user_search_base = ou=user accounts,dc=ad,dc=example,dc=com
ldap_user_object_class = user
ldap_user_home_directory = unixHomeDirectory
ldap_user_principal = userPrincipalName
ldap_user_name = sAMAccountName

ldap_group_search_base = ou=groups,dc=ad,dc=example,dc=com
ldap_group_object_class = group

ldap_access_order = expire
ldap_account_expire_policy = ad
ldap_force_upper_case_realm = true
ldap_disable_referrals = true

#krb5_server = server.ad.example.com
krb5_realm = AD-REALM.EXAMPLE.COM
```

These options are described in the man page for LDAP domain configuration, **`sssd-ldap(5)`**.

12. Restart SSSD.

```
[root@rhel-server ~]# service sssd restart
```

30.4.2.4. Using IP Addresses in Certificate Subject Names

30.4.2.1. Using IP Addresses in Certificate Subject Names

Using an IP address in the **`ldap_uri`** option instead of the server name may cause the TLS/SSL connection to fail. TLS/SSL certificates contain the server name, not the IP address. However, the *subject alternative name* field in the certificate can be used to include the IP address of the server, which allows a successful secure connection using an IP address.

1. Convert an existing certificate into a certificate request. The signing key (**`-signkey`**) is the key of the issuer of whatever CA originally issued the certificate. If this is done by an external CA, it requires a separate PEM file; if the certificate is self-signed, then this is the certificate itself. For example:

   ```
   openssl x509 -x509toreq -in old_cert.pem -out req.pem -signkey key.pem
   ```

   With a self-signed certificate:

   ```
   openssl x509 -x509toreq -in old_cert.pem -out req.pem -signkey
   old_cert.pem
   ```

2. Edit the **`/etc/pki/tls/openssl.cnf`** configuration file to include the server's IP address under the ***`[ v3_ca ]`*** section:

   ```
   subjectAltName = IP:10.0.0.10
   ```

3. Use the generated certificate request to generate a new self-signed certificate with the specified IP address:

   ```
   openssl x509 -req -in req.pem -out new_cert.pem -extfile ./openssl.cnf
   -extensions v3_ca -signkey old_cert.pem
   ```

   The **`-extensions`** option sets which extensions to use with the certificate. For this, it should be v3_ca to load the appropriate section.

4. Copy the private key block from the **`old_cert.pem`** file into the **`new_cert.pem`** file to keep all relevant information in one file.

When creating a certificate through the **certutil** utility provided by the *nss-tools* package, note that **certutil** supports DNS subject alternative names for certificate creation only.

30.4.3. Configuring Kerberos Authentication with a Domain

Both LDAP and proxy identity providers can use a separate Kerberos domain to supply authentication. Configuring a Kerberos authentication provider requires the *key distribution center* (KDC) and the Kerberos domain. All of the principal names must be available in the specified identity provider; if they are not, SSSD constructs the principals using the format *username@REALM*.

Note

Kerberos can only provide authentication; it cannot provide an identity database.

SSSD assumes that the Kerberos KDC is also a Kerberos kadmin server. However, production environments commonly have multiple, read-only replicas of the KDC and only a single kadmin server. Use the **`krb5_kpasswd`** option to specify where the password changing service is running or if it is running on a non-default port. If the **`krb5_kpasswd`** option is not defined, SSSD tries to use the Kerberos KDC to change the password.

The basic Kerberos configuration options are listed in Table 30.6, “Kerberos Authentication Configuration Parameters”. The **sssd-krb5(5)** man page has more information about Kerberos configuration options.

Example 30.3. Basic Kerberos Authentication

```
# A domain with identities provided by LDAP and authentication by Kerberos
[domain/KRBDOMAIN]
enumerate = false
id_provider = ldap
chpass_provider = krb5
ldap_uri = ldap://ldap.example.com
ldap_search_base = dc=example,dc=com
ldap-tls_reqcert = demand
ldap_tls_cacert = /etc/pki/tls/certs/ca-bundle.crt

auth_provider = krb5
krb5_server = 192.168.1.1, kerberos.example.com
krb5_realm = EXAMPLE.COM
krb5_kpasswd = kerberos.admin.example.com
krb5_auth_timeout = 15
```

Table 30.6. Kerberos Authentication Configuration Parameters

Parameter	Description
chpass_provider	Specifies which service to use for password change operations. This is assumed to be the same as the authentication provider. To use Kerberos, set this to *krb5*.
krb5_server	Gives a comma-separated list of IP addresses or hostnames of Kerberos servers to which SSSD will connect. The list is given in order of preference, so the first server in the list is tried first. Listing additional servers provides failover protection. When using service discovery for KDC or kpasswd servers, SSSD first searches for DNS entries that specify UDP as the connection protocol, and then falls back to TCP.
krb5_realm	Identifies the Kerberos realm served by the KDC.
krb5_lifetime	Requests a Kerberos ticket with the specified lifetime in seconds (s), minutes (m), hours (h) or days (d).
krb5_renewable_lifetime	Requests a renewable Kerberos ticket with a total lifetime that is specified in seconds (s), minutes (m), hours (h) or days (d).
krb5_renew_interval	Sets the time, in seconds, for SSSD to check if tickets should be renewed. Tickets are renewed automatically once they exceed half their lifetime. If this option is missing or set to zero, then automatic ticket renewal is disabled.

Parameter	Description
krb5_store_password_if_offline	Sets whether to store user passwords if the Kerberos authentication provider is offline, and then to use that cache to request tickets when the provider is back online. The default is **`false`**, which does not store passwords.
krb5_kpasswd	Lists alternate Kerberos kadmin servers to use if the change password service is not running on the KDC.
krb5_ccname_template	Gives the directory to use to store the user's credential cache. This can be templatized, and the following tokens are supported: • *%u*, the user's login name • *%U*, the user's login UID • *%p*, the user's principal name • *%r*, the realm name • *%h*, the user's home directory • *%d*, the value of the **`krb5ccache_dir`** parameter • *%P*, the process ID of the SSSD client. • *%%*, a literal percent sign (%) • *XXXXXX*, a string at the end of the template which instructs SSSD to create a unique filename safely For example: `krb5_ccname_template = FILE:%d/krb5cc_%U_XXXXXX`
krb5_ccachedir	Specifies the directory to store credential caches. This can be templatized, using the same tokens as **`krb5_ccname_template`**, except for **`%d`** and **`%P`**. If **`%u`**, **`%U`**, **`%p`**, or **`%h`** are used, then SSSD creates a private directory for each user; otherwise, it creates a public directory.
krb5_auth_timeout	Gives the time, in seconds, before an online authentication or change password request is aborted. If possible, the authentication request is continued offline. The default is 15 seconds.

30.4.4. Configuring a Proxy Domain

A proxy with SSSD is just a relay, an intermediary configuration. SSSD connects to its proxy service, and then that proxy loads the specified libraries. This allows SSSD to use some resources that it otherwise would not be able to use. For example, SSSD only supports LDAP and Kerberos as authentication providers, but using a proxy allows SSSD to use alternative authentication methods like a fingerprint scanner or smart card.

Table 30.7. Proxy Domain Configuration Parameters

Parameter	Description

Parameter	Description
proxy_pam_target	Specifies the target to which PAM must proxy as an authentication provider. The PAM target is a file containing PAM stack information in the default PAM directory, **`/etc/pam.d/`**. This is used to proxy an authentication provider. **Important** Ensure that the proxy PAM stack does *not* recursively include **`pam_sss.so`**.
proxy_lib_name	Specifies which existing NSS library to proxy identity requests through. This is used to proxy an identity provider.

Example 30.4. Proxy Identity and Kerberos Authentication

The proxy library is loaded using the **`proxy_lib_name`** parameter. This library can be anything as long as it is compatible with the given authentication service. For a Kerberos authentication provider, it must be a Kerberos-compatible library, like NIS.

```
[domain/PROXY_KRB5]
auth_provider = krb5
krb5_server = 192.168.1.1
krb5_realm = EXAMPLE.COM

id_provider = proxy
proxy_lib_name = nis
enumerate = true
cache_credentials = true
```

Example 30.5. LDAP Identity and Proxy Authentication

The proxy library is loaded using the **`proxy_pam_target`** parameter. This library must be a PAM module that is compatible with the given identity provider. For example, this uses a PAM fingerprint module with LDAP:

```
[domain/LDAP_PROXY]
id_provider = ldap
ldap_uri = ldap://example.com
ldap_search_base = dc=example,dc=com

auth_provider = proxy
proxy_pam_target = sssdpamproxy
enumerate = true
cache_credentials = true
```

After the SSSD domain is configured, make sure that the specified PAM files are configured. In this example, the target is **sssdpamproxy**, so create a **/etc/pam.d/sssdpamproxy** file and load the PAM/LDAP modules:

```
auth            required        pam_frprint.so
account         required        pam_frprint.so
password        required        pam_frprint.so
session         required        pam_frprint.so
```

Example 30.6. Proxy Identity and Authentication

SSSD can have a domain with both identity and authentication proxies. The only configuration given then are the proxy settings, **proxy_pam_target** for the authentication PAM module and **proxy_lib_name** for the service, like NIS or LDAP.

This example illustrates a possible configuration, but this is not a realistic configuration. If LDAP is used for identity and authentication, then both the identity and authentication providers should be set to the LDAP configuration, not a proxy.

```
[domain/PROXY_PROXY]
auth_provider = proxy
id_provider = proxy
proxy_lib_name = ldap
proxy_pam_target = sssdproxyldap
enumerate = true
cache_credentials = true
```

Once the SSSD domain is added, then update the system settings to configure the proxy service:

1. Create an **/etc/pam.d/sssdproxyldap** file which requires the **pam_ldap.so** module:

   ```
   auth            required        pam_ldap.so
   account         required        pam_ldap.so
   password        required        pam_ldap.so
   session         required        pam_ldap.so
   ```

2. Make sure the **nss-pam-ldap** package is installed.

   ```
   [root@server ~]# yum install nss-pam-ldap
   ```

3. Edit the **/etc/nslcd.conf** file, the configuration file for the LDAP name service daemon, to contain the information for the LDAP directory:

   ```
   uid nslcd
   gid ldap
   uri ldaps://ldap.example.com:636
   base dc=example,dc=com
   ssl on
   tls_cacertdir /etc/openldap/cacerts
   ```

30.5. Configuring Access Control for SSSD Domains

SSSD provides a rudimentary access control for domain configuration, allowing either simple user allow/deny lists or using the LDAP backend itself.

30.5.1. Using the Simple Access Provider

The *Simple Access Provider* allows or denies access based on a list of usernames or groups.

The Simple Access Provider is a way to restrict access to certain, specific machines. For example, if a company uses laptops, the Simple Access Provider can be used to restrict access to only a specific user or a specific group, even if a different user authenticated successfully against the same authentication provider.

The most common options are **`simple_allow_users`** and **`simple_allow_groups`**, which grant access explicitly to specific users (either the given users or group members) and deny access to everyone else. It is also possible to create deny lists (which deny access only to explicit people and implicitly allow everyone else access).

The Simple Access Provider adheres to the following four rules to determine which users should or should not be granted access:

- If both the allow and deny lists are empty, access is granted.
- If any list is provided, allow rules are evaluated first, and then deny rules. Practically, this means that deny rules supersede allow rules.
- If an allowed list is provided, then all users are denied access unless they are in the list.
- If only deny lists are provided, then all users are allowed access unless they are in the list.

This example grants access to two users and anyone who belongs to the IT group; implicitly, all other users are denied.

```
[domain/example.com]
access_provider = simple
simple_allow_users = jsmith,bjensen
simple_allow_groups = itgroup
```

Note

The LOCAL domain in SSSD does not support **`simple`** as an access provider.

Other options are listed in the **`sssd-simple`** man page, but these are rarely used.

30.5.2. Using the LDAP Access Filter

The LDAP server itself can provide the access control rules. The associated filter option (**`ldap_access_filter`**) specifies which users are granted access to the specified host. The user filter must be used or all users are denied access.

For example:

```
[domain/example.com]
access_provider = ldap
ldap_access_filter = memberOf=cn=allowedusers,ou=Groups,dc=example,dc=com
```

Note

Offline caching for LDAP access providers is limited to determining whether the user's last online login attempt was successful. Users that were granted access during their last login will continue to be granted access while offline.

SSSD can also check results by the account expiration policy and the ***authorizedService*** attribute.

30.6. Configuring Domain Failover

SSSD attempts to connect to machines and to services separately.

When SSSD tries to connect to one of its domain backends, it first tries to resolve the hostname of a given machine. If this resolution attempt fails, the machine is considered offline, and SSSD no longer attempts to connect to this machine for any other service.

If the resolution attempt succeeds, the backend tries to connect to a service on this machine. If the service connection attempt fails, then only this particular service is considered offline and the backend automatically switches over to the next service. The machine is still considered online and might still be tried for another service.

SSSD only tries the first IP address given in the DNS A record. To find multiple servers with a single request, SSSD relies on SRV records.

Connections are retried to offline machines or services every 30 seconds, until SSSD can successfully connect to the backend.

30.6.1. Configuring Failover

Configuring failover allows SSSD to switch automatically to a different server if the primary server fails. These servers are entered as a case-insensitive, comma-separated list in the *[domain/Name]* sections of the **/etc/sssd/sssd.conf** file. The servers are listed in order of preference. This list can contain any number of servers.

For example, for a native LDAP domain:

```
ldap_uri = ldap://ldap0.example.com, ldap://ldap1.example.com,
ldap://ldap2.example.com
```

The first entry, **ldap://ldap0.example.com**, is the primary server. If this server fails, SSSD first attempts to connect to **ldap1.example.com** and then **ldap2.example.com**.

If the server parameter is not specified, then SSSD uses service discovery to try to find another server on the network.

Important

The failover servers must be entered as a comma-separated list of values for a single key. If there are multiple keys, SSSD only recognizes the last entry.

30.6.2. Using SRV Records with Failover

SSSD supports SRV records in its failover configuration. The SSSD configuration can specify a server that is later resolved into a list of specific servers using SRV requests.

For every service with which to use service discovery, add a special DNS record to the DNS server:

```
_service._protocol._domain TTL priority weight port hostname
```

The *priority* and *weight* attributes of SRV records provide fine-grained control over which servers to contact first if the primary server fails.

A typical configuration contains multiple such records, each with a different priority for failover and different weights for load balancing.

For more information on SRV records, see RFC 2782.

30.7. Deleting Domain Cache Files

SSSD can define multiple domains of the same type and different types of domain. SSSD maintains a separate database file for each domain, meaning each domain has its own cache. These cache files are stored in the **`/var/lib/sss/db/`** directory.

If there is ever a problem with a domain, it is easy to purge the cache by stopping SSSD and deleting the cache file for that domain.

All cache files are named for the domain. For example, for a domain named **`exampleldap`**, the cache file is named **`cache_exampleldap.ldb`**.

Be careful when you delete a cache file. This operation has significant effects:

- Deleting the cache file deletes all user data, both identification and cached credentials. Consequently, do not delete a cache file unless the system is online and can authenticate with a username against the domain's servers. Without a credentials cache, offline authentication will fail.
- If the configuration is changed to reference a different identity provider, SSSD will recognize users from both providers until the cached entries from the original provider time out.

  It is possible to avoid this by purging the cache, but the better option is to use a different domain name for the new provider. When SSSD is restarted, it creates a new cache file with the new name and the old file is ignored.

30.8. Using NSCD with SSSD

SSSD is not designed to be used with the NSCD daemon. Even though SSSD does not directly conflict with NSCD, using both services can result in unexpected behavior, especially with how long entries are cached.

The most common evidence of a problem is conflicts with NFS. When using Network Manager to manage network connections, it may take several minutes for the network interface to come up. During this time, various services attempt to start. If these services start before the network is up and the DNS servers are available, these services fail to identify the forward or reverse DNS entries they need. These services will read an incorrect or possibly empty **`resolv.conf`** file. This file is typically only read once, and so any changes made to this file are not automatically applied. This can cause NFS locking to fail on the machine where the NSCD service is running, unless that service is manually restarted.

To avoid this problem, enable caching for hosts and services in the **`/etc/nscd.conf`** file and rely on the SSSD cache for the **`passwd`**, **`group`**, and **`netgroup`** entries.

Change the **`/etc/nscd.conf`** file:

```
enable-cache hosts yes
enable-cache passwd no
enable-cache group no
enable-cache netgroup no
```

With NSCD answering hosts requests, these entries will be cached by NSCD and returned by NSCD during the boot process. All other entries are handled by SSSD.

30.9. Troubleshooting SSSD

30.9.1. Checking SSSD Log Files

SSSD uses a number of log files to report information about its operation, located in the **`/var/log/sssd/`** directory. SSSD produces a log file for each domain, as well as an **`sssd_pam.log`** and an **`sssd_nss.log`** file.

Additionally, the **`/var/log/secure`** file logs authentication failures and the reason for the failure.

Increasing the log level can provide more information about problems with SSSD. To change the log level, set the ***`debug_level`*** parameter for each section in the **`sssd.conf`** file for which to product extra logs. For example:

```
[sssd]
config_file_version = 2
services = nss, pam
domains = LDAP
debug_level = 9
```

Table 30.8. Debug Log Levels

Level	Description
0	Fatal failures. Anything that would prevent SSSD from starting up or causes it to cease running.
1	Critical failures. An error that doesn't kill the SSSD, but one that indicates that at least one major feature is not going to work properly.
2	Serious failures. An error announcing that a particular request or operation has failed.
3	Minor failures. These are the errors that would percolate down to cause the operation failure of 2.
4	Configuration settings.
5	Function data.
6	Trace messages for operation functions.
7	Trace messages for internal control functions.
8	Contents of function-internal variables that may be interesting.
9	Extremely low-level tracing information.

30.9.2. Problems with SSSD Configuration

SSSD fails to start

SSSD requires that the configuration file be properly set up, with all the required entries, before the daemon will start.

- SSSD requires at least one properly configured domain before the service will start. Without a domain, attempting to start SSSD returns an error that no domains are configured:

  ```
  # sssd -d4

  [sssd] [ldb] (3): server_sort:Unable to register control with rootdse!
  [sssd] [confdb_get_domains] (0): No domains configured, fatal error!
  [sssd] [get_monitor_config] (0): No domains configured.
  ```

  Edit the **`/etc/sssd/sssd.conf`** file and create at least one domain.

- SSSD also requires at least one available service provider before it will start. If the problem is with the service provider configuration, the error message indicates that there are no services configured:

  ```
  [sssd] [get_monitor_config] (0): No services configured!
  ```

  Edit the **`/etc/sssd/sssd.conf`** file and configure at least one service provider.

Important

SSSD requires that service providers be configured as a comma-separated list in a single ***`services`*** entry in the **`/etc/sssd/sssd.conf`** file. If services are listed in multiple entries, only the last entry is recognized by SSSD.

I don't see any groups with 'id' or group members with 'getent group'.

This may be due to an incorrect **`ldap_schema`** setting in the **`[domain/DOMAINNAME]`** section of **`sssd.conf`**.

SSSD supports RFC 2307 and RFC 2307bis schema types. By default, SSSD uses the more common RFC 2307 schema.

The difference between RFC 2307 and RFC 2307bis is the way which group membership is stored in the LDAP server. In an RFC 2307 server, group members are stored as the multi-valued ***`memberuid`*** attribute, which contains the name of the users that are members. In an RFC2307bis server, group members are stored as the multi-valued ***`member`*** or ***`uniqueMember`*** attribute which contains the DN of the user or group that is a member of this group. RFC2307bis allows nested groups to be maintained as well.

If group lookups are not returning any information:

1. Set **`ldap_schema`** to **`rfc2307bis`**.
2. Delete **`/var/lib/sss/db/cache_DOMAINNAME.ldb`**.
3. Restarting SSSD.

If that doesn't work, add this line to **`sssd.conf`**:

```
ldap_group_name = uniqueMember
```

Then delete the cache and restart SSSD again.

Authentication fails against LDAP.

To perform authentication, SSSD requires that the communication channel be encrypted. This means that if **`sssd.conf`** is configured to connect over a standard protocol (**`ldap://`**), it attempts to encrypt the communication channel with Start TLS. If **`sssd.conf`** is configured to connect over a secure protocol (**`ldaps://`**), then SSSD uses SSL.

This means that the LDAP server must be configured to run in SSL or TLS. TLS must be enabled for the standard LDAP port (389) or SSL enabled on the secure LDAPS port (636). With either SSL or TLS, the LDAP server must also be configured with a valid certificate trust.

An invalid certificate trust is one of the most common issues with authenticating against LDAP. If the client does not have proper trust of the LDAP server certificate, it is unable to validate the connection, and SSSD refuses to send the password. The LDAP protocol requires that the password be sent in plaintext to the LDAP server. Sending the password in plaintext over an unencrypted connection is a security problem.

If the certificate is not trusted, a **`syslog`** message is written, indicating that TLS encryption could not be started. The certificate configuration can be tested by checking if the LDAP server is accessible apart from SSSD. For example, this tests an anonymous bind over a TLS connection to **`test.example.com`**:

```
$ ldapsearch -x -ZZ -h test.example.com -b dc=example,dc=com
```

If the certificate trust is not properly configured, the test fails with this error:

```
ldap_start_tls: Connect error (-11) additional info: TLS error -8179:Unknown
code ___f 13
```

To trust the certificate:

1. Obtain a copy of the public CA certificate for the certificate authority used to sign the LDAP server certificate and save it to the local system.

2. Add a line to the **`sssd.conf`** file that points to the CA certificate on the filesystem.

   ```
   ldap_tls_cacert = /path/to/cacert
   ```

3. If the LDAP server uses a self-signed certificate, remove the **`ldap_tls_reqcert`** line from the **`sssd.conf`** file.

   This parameter directs SSSD to trust any certificate issued by the CA certificate, which is a security risk with a self-signed CA certificate.

Connecting to LDAP servers on non-standard ports fail.

When running SELinux in enforcing mode, the client's SELinux policy has to be modified to connect to the LDAP server over the non-standard port. For example:

```
# semanage port -a -t ldap_port_t -p tcp 1389
```

NSS fails to return user information

This usually means that SSSD cannot connect to the NSS service.

- Ensure that NSS is running:

```
# service sssd status
```

- If NSS is running, make sure that the provider is properly configured in the **`[nss]`** section of the **`/etc/sssd/sssd.conf`** file. Especially check the ***`filter_users`*** and ***`filter_groups`*** attributes.
- Make sure that NSS is included in the list of services that SSSD uses.
- Check the configuration in the **`/etc/nsswitch.conf`** file.

NSS returns incorrect user information

If searches are returning the incorrect user information, check that there are not conflicting usernames in separate domains. When there are multiple domains, set the ***`use_fully_qualified_domains`*** attribute to **`true`** in the **`/etc/sssd/sssd.conf`** file. This differentiates between different users in different domains with the same name.

Setting the password for the local SSSD user prompts twice for the password

When attempting to change a local SSSD user's password, it may prompt for the password twice:

```
[root@clientF11 tmp]# passwd user1000
Changing password for user user1000.
New password:
Retype new password:
New Password:
Reenter new Password:
passwd: all authentication tokens updated successfully.
```

This is the result of an incorrect PAM configuration. Ensure that the ***`use_authtok`*** option is correctly configured in your **`/etc/pam.d/system-auth`** file.

Part IV. System Configuration

Part of a system administrator's job is configuring the system for various tasks, types of users, and hardware configurations. This section explains how to configure a Red Hat Enterprise Linux system.

Chapter 31. Console Access

When normal (non-root) users log into a computer locally, they are given two types of special permissions:

1. They can run certain programs that they would otherwise be unable to run.
2. They can access certain files (normally special device files used to access diskettes, CD-ROMs, and so on) that they would otherwise be unable to access.

Since there are multiple consoles on a single computer and multiple users can be logged into the computer locally at the same time, one of the users has to essentially win the race to access the files. The first user to log in at the console owns those files. Once the first user logs out, the next user who logs in owns the files.

In contrast, *every* user who logs in at the console is allowed to run programs that accomplish tasks normally restricted to the root user. If X is running, these actions can be included as menu items in a graphical user interface. As shipped, these console-accessible programs include **`halt`**, **`poweroff`**, and **`reboot`**.

31.1. Disabling Shutdown Via `Ctrl`+`Alt`+`Del`

By default, **`/etc/inittab`** specifies that your system is set to shutdown and reboot in response to a **`Ctrl`**+**`Alt`**+**`Del`** key combination used at the console. To completely disable this ability, comment out the following line in **`/etc/inittab`** by putting a hash mark (**`#`**) in front of it:

```
ca::ctrlaltdel:/sbin/shutdown -t3 -r now
```

Alternatively, you may want to allow certain non-root users the right to shutdown or reboot the system from the console using **`Ctrl`**+**`Alt`**+**`Del`** . You can restrict this privilege to certain users, by taking the following steps:

1. Add the **`-a`** option to the **`/etc/inittab`** line shown above, so that it reads:

   ```
   ca::ctrlaltdel:/sbin/shutdown -a -t3 -r now
   ```

   The **`-a`** flag tells **`shutdown`** to look for the **`/etc/shutdown.allow`** file.

2. Create a file named **`shutdown.allow`** in **`/etc`**. The **`shutdown.allow`** file should list the usernames of any users who are allowed to shutdown the system using **`Ctrl`**+**`Alt`**+**`Del`** . The format of the **`shutdown.allow`** file is a list of usernames, one per line, like the following:

   ```
   stephen
   jack
   sophie
   ```

According to this example **`shutdown.allow`** file, the users **`stephen`**, **`jack`**, and **`sophie`** are allowed to shutdown the system from the console using **`Ctrl`**+**`Alt`**+**`Del`** . When that key combination is used, the **`shutdown -a`** command in **`/etc/inittab`** checks to see if any of the users in **`/etc/shutdown.allow`** (or root) are logged in on a virtual console. If one of them is, the shutdown of the system continues; if not, an error message is written to the system console instead.

For more information on **`shutdown.allow`**, refer to the **`shutdown`** man page.

31.2. Disabling Console Program Access

To disable access by users to console programs, run the following command as root:

```
rm -f /etc/security/console.apps/*
```

In environments where the console is otherwise secured (BIOS and boot loader passwords are set, **Ctrl**+**Alt**+**Delete** is disabled, the power and reset switches are disabled, and so forth), you may not want to allow any user at the console to run **poweroff**, **halt**, and **reboot**, which are accessible from the console by default.

To disable these abilities, run the following commands as root:

```
rm -f /etc/security/console.apps/poweroff
rm -f /etc/security/console.apps/halt
rm -f /etc/security/console.apps/reboot
```

31.3. Defining the Console

The **pam_console.so** module uses the **/etc/security/console.perms** file to determine the permissions for users at the system console. The syntax of the file is very flexible; you can edit the file so that these instructions no longer apply. However, the default file has a line that looks like this:

```
<console>=tty[0-9][0-9]* vc/[0-9][0-9]* :[0-9]\.[0-9] :[0-9]
```

When users log in, they are attached to some sort of named terminal, which can be either an X server with a name like **:0** or **mymachine.example.com:1.0**, or a device like **/dev/ttyS0** or **/dev/pts/2**. The default is to define that local virtual consoles and local X servers are considered local, but if you want to consider the serial terminal next to you on port **/dev/ttyS1** to also be local, you can change that line to read:

```
<console>=tty[0-9][0-9]* vc/[0-9][0-9]* :[0-9]\.[0-9] :[0-9] /dev/ttyS1
```

31.4. Making Files Accessible From the Console

The default settings for individual device classes and permission definitions are defined in **/etc/security/console.perms.d/50-default.perms**. To edit file and device permissions, it is advisable to create a new default file in **/etc/security/console.perms.d/** containing your preferred settings for a specified set of files or devices. The name of the new default file must begin with a number higher than 50 (for example, **51-default.perms**) in order to override **50-default.perms**.

To do this, create a new file named **51-default.perms** in **/etc/security/console.perms.d/**:

```
touch /etc/security/console.perms.d/51-default.perms
```

Open the original default **perms** file, **50-default.perms**. The first section defines *device classes*, with lines similar to the following:

```
<floppy>=/dev/fd[0-1]* \
          /dev/floppy/* /mnt/floppy*
<sound>=/dev/dsp* /dev/audio* /dev/midi* \
   /dev/mixer* /dev/sequencer \
   /dev/sound/* /dev/beep \
   /dev/snd/*
<cdrom>=/dev/cdrom* /dev/cdroms/* /dev/cdwriter* /mnt/cdrom*
```

Items enclosed in brackets name the device; in the above example, **`<cdrom>`** refers to the CD-ROM drive. To add a new device, do not define it in the default **`50-default.perms`** file; instead, define it in **`51-default.perms`**. For example, to define a scanner, add the following line to **`51-default.perms`**:

```
<scanner>=/dev/scanner /dev/usb/scanner*
```

Of course, you must use the appropriate name for the device. Ensure that **`/dev/scanner`** is really your scanner and not some other device, such as your hard drive.

Once you have properly defined a device or file, the second step is to specify its *permission definitions*. The second section of **`/etc/security/console.perms.d/50-default.perms`** defines this, with lines similar to the following:

```
<console> 0660 <floppy> 0660 root.floppy
<console> 0600 <sound>  0640 root
<console> 0600 <cdrom>  0600 root.disk
```

To define permissions for a scanner, add a line similar to the following in **`51-default.perms`**:

```
<console> 0600 <scanner> 0600 root
```

Then, when you log in at the console, you are given ownership of the **`/dev/scanner`** device with the permissions of 0600 (readable and writable by you only). When you log out, the device is owned by root, and still has the permissions 0600 (now readable and writable by root only).

Warning

You must *never* edit the default **`50-default.perms`** file. To edit permissions for a device already defined in **`50-default.perms`**, add the desired permission definition for that device in **`51-default.perms`**. This will override whatever permissions are defined in **`50-default.perms`**.

31.5. Enabling Console Access for Other Applications

To make other applications accessible to console users, a bit more work is required.

First of all, console access *only* works for applications which reside in **`/sbin/`** or **`/usr/sbin/`**, so the application that you wish to run must be there. After verifying that, perform the following steps:

1. Create a link from the name of your application, such as our sample **`foo`** program, to the **`/usr/bin/consolehelper`** application:

   ```
   cd /usr/bin
   ln -s consolehelper foo
   ```

2. Create the file **`/etc/security/console.apps/foo`**:

   ```
   touch /etc/security/console.apps/foo
   ```

3. Create a PAM configuration file for the **foo** service in **/etc/pam.d/**. An easy way to do this is to copy the PAM configuration file of the **halt** service, and then modify the copy if you want to change the behavior:

   ```
   cp /etc/pam.d/halt /etc/pam.d/foo
   ```

Now, when **/usr/bin/*foo*** is executed, **consolehelper** is called, which authenticates the user with the help of **/usr/sbin/userhelper**. To authenticate the user, **consolehelper** asks for the user's password if **/etc/pam.d/*foo*** is a copy of **/etc/pam.d/halt** (otherwise, it does precisely what is specified in **/etc/pam.d/*foo***) and then runs **/usr/sbin/*foo*** with root permissions.

In the PAM configuration file, an application can be configured to use the *pam_timestamp* module to remember (or cache) a successful authentication attempt. When an application is started and proper authentication is provided (the root password), a timestamp file is created. By default, a successful authentication is cached for five minutes. During this time, any other application that is configured to use **pam_timestamp** and run from the same session is automatically authenticated for the user — the user does not have to enter the root password again.

This module is included in the **pam** package. To enable this feature, add the following lines to your PAM configuration file in **etc/pam.d/**:

```
auth            include         config-util
account         include         config-util
session         include         config-util
```

These lines can be copied from any of the **/etc/pam.d/system-config-*** configuration files. Note that these lines must be added *below* any other **auth sufficient session optional** lines in your PAM configuration file.

If an application configured to use **pam_timestamp** is successfully authenticated from the Applications (the main menu on the panel), the icon is displayed in the notification area of the panel if you are running the **GNOME** or **KDE** desktop environment. After the authentication expires (the default is five minutes), the icon disappears.

The user can select to forget the cached authentication by clicking on the icon and selecting the option to forget authentication.

31.6. The `floppy` Group

If, for whatever reason, console access is not appropriate for you and your non-root users require access to your system's diskette drive, this can be done using the **floppy** group. Add the user(s) to the **floppy** group using the tool of your choice. For example, the **gpasswd** command can be used to add user **fred** to the **floppy** group:

```
gpasswd -a fred floppy
```

Now, user **fred** is able to access the system's diskette drive from the console.

Chapter 32. The `sysconfig` Directory

The **`/etc/sysconfig/`** directory contains a variety of system configuration files for Red Hat Enterprise Linux.

This chapter outlines some of the files found in the **`/etc/sysconfig/`** directory, their function, and their contents. The information in this chapter is not intended to be complete, as many of these files have a variety of options that are only used in very specific or rare circumstances.

32.1. Files in the `/etc/sysconfig/` Directory

The following sections offer descriptions of files normally found in the **`/etc/sysconfig/`** directory. Files not listed here, as well as extra file options, are found in the **`/usr/share/doc/initscripts-<version-number>/sysconfig.txt`** file (replace *<version-number>* with the version of the **`initscripts`** package). Alternatively, looking through the initscripts in the **`/etc/rc.d/`** directory can prove helpful.

Note

If some of the files listed here are not present in the **`/etc/sysconfig/`** directory, then the corresponding program may not be installed.

32.1.1. `/etc/sysconfig/amd`

The **`/etc/sysconfig/amd`** file contains various parameters used by **`amd`**; these parameters allow for the automatic mounting and unmounting of file systems.

32.1.2. `/etc/sysconfig/apmd`

The **`/etc/sysconfig/apmd`** file is used by **`apmd`** to configure what power settings to start/stop/change on suspend or resume. This file configures how **`apmd`** functions at boot time, depending on whether the hardware supports *Advanced Power Management* (*APM*) or whether the user has configured the system to use it. The **`apm`** daemon is a monitoring program that works with power management code within the Linux kernel. It is capable of alerting users to low battery power on laptops and other power-related settings.

32.1.3. `/etc/sysconfig/arpwatch`

The **`/etc/sysconfig/arpwatch`** file is used to pass arguments to the **`arpwatch`** daemon at boot time. The **`arpwatch`** daemon maintains a table of Ethernet MAC addresses and their IP address pairings. By default, this file sets the owner of the **`arpwatch`** process to the user **`pcap`** and sends any messages to the **`root`** mail queue. For more information regarding available parameters for this file, refer to the **`arpwatch`** man page.

32.1.4. `/etc/sysconfig/authconfig`

The **`/etc/sysconfig/authconfig`** file sets the authorization to be used on the host. It contains one or more of the following lines:

- **`USEMD5=<value>`**, where **`<value>`** is one of the following:

  - **`yes`** — MD5 is used for authentication.
  - **`no`** — MD5 is not used for authentication.
- **`USEKERBEROS=<value>`**, where **`<value>`** is one of the following:
  - **`yes`** — Kerberos is used for authentication.
  - **`no`** — Kerberos is not used for authentication.
- **`USELDAPAUTH=<value>`**, where **`<value>`** is one of the following:
  - **`yes`** — LDAP is used for authentication.
  - **`no`** — LDAP is not used for authentication.

32.1.5. `/etc/sysconfig/autofs`

The **`/etc/sysconfig/autofs`** file defines custom options for the automatic mounting of devices. This file controls the operation of the automount daemons, which automatically mount file systems when you use them and unmount them after a period of inactivity. File systems can include network file systems, CD-ROMs, diskettes, and other media.

The **`/etc/sysconfig/autofs`** file may contain the following:

- **`LOCALOPTIONS="<value>"`**, where <*value*> is a string for defining machine-specific automount rules. The default value is an empty string (**`""`**).
- **`DAEMONOPTIONS="<value>"`**, where <*value*> is the timeout length in seconds before unmounting the device. The default value is 60 seconds (**`"--timeout=60"`**).
- **`UNDERSCORETODOT=<value>`**, where <*value*> is a binary value that controls whether to convert underscores in file names into dots. For example, **`auto_home`** to **`auto.home`** and **`auto_mnt`** to **`auto.mnt`**. The default value is 1 (true).
- **`DISABLE_DIRECT=<value>`**, where <*value*> is a binary value that controls whether to disable direct mount support, as the Linux implementation does not conform to the Sun Microsystems' automounter behavior. The default value is 1 (true), and allows for compatibility with the Sun automounter options specification syntax.

32.1.6. `/etc/sysconfig/clock`

The **`/etc/sysconfig/clock`** file controls the interpretation of values read from the system hardware clock.

The correct values are:

- **`UTC=<value>`**, where **`<value>`** is one of the following boolean values:
  - **`true`** or **`yes`** — The hardware clock is set to Universal Time.
  - **`false`** or **`no`** — The hardware clock is set to local time.
- **`ARC=<value>`**, where **`<value>`** is the following:
  - **`false`** or **`no`** — This value indicates that the normal UNIX epoch is in use. Other values are used by systems not supported by Red Hat Enterprise Linux.
- **`SRM=<value>`**, where **`<value>`** is the following:

  - **`false`** or **`no`** — This value indicates that the normal UNIX epoch is in use. Other values are used by systems not supported by Red Hat Enterprise Linux.

- **`ZONE=<filename>`** — The time zone file under **`/usr/share/zoneinfo`** that **`/etc/localtime`** is a copy of. The file contains information such as:

  ```
  ZONE="America/New York"
  ```

  Note that the **`ZONE`** parameter is read by the **Time and Date Properties Tool** (**`system-config-date`**), and manually editing it does not change the system timezone.

Earlier releases of Red Hat Enterprise Linux used the following values (which are deprecated):

- **`CLOCKMODE=<value>`**, where **`<value>`** is one of the following:

  - **`GMT`** — The clock is set to Universal Time (Greenwich Mean Time).

  - **`ARC`** — The ARC console's 42-year time offset is in effect (for Alpha-based systems only).

32.1.7. `/etc/sysconfig/desktop`

The **`/etc/sysconfig/desktop`** file specifies the desktop for new users and the display manager to run when entering runlevel 5.

Correct values are:

- **`DESKTOP="<value>"`**, where **`"<value>"`** is one of the following:

  - **`GNOME`** — Selects the **GNOME** desktop environment.

  - **`KDE`** — Selects the **KDE** desktop environment.

- **`DISPLAYMANAGER="<value>"`**, where **`"<value>"`** is one of the following:

  - **`GNOME`** — Selects the **GNOME Display Manager**.

  - **`KDE`** — Selects the **KDE Display Manager**.

  - **`XDM`** — Selects the **X Display Manager**.

For more information, refer to Chapter 35, *The X Window System*.

32.1.8. `/etc/sysconfig/dhcpd`

The **`/etc/sysconfig/dhcpd`** file is used to pass arguments to the **dhcpd** daemon at boot time. The **dhcpd** daemon implements the Dynamic Host Configuration Protocol (DHCP) and the Internet Bootstrap Protocol (BOOTP). DHCP and BOOTP assign hostnames to machines on the network. For more information about what parameters are available in this file, refer to the **dhcpd** man page.

32.1.9. `/etc/sysconfig/exim`

The **`/etc/sysconfig/exim`** file allows messages to be sent to one or more clients, routing the messages over whatever networks are necessary. The file sets the default values for exim to run. Its default values are set to run as a background daemon and to check its queue each hour in case something has backed up.

The values include:

- **DAEMON=<*value*>**, where **<*value*>** is one of the following:
  - **yes** — **exim** should be configured to listen to port 25 for incoming mail. **yes** implies the use of the Exim's **-bd** options.
  - **no** — **exim** should not be configured to listen to port 25 for incoming mail.
- **QUEUE=1h** which is given to **exim** as **-q$QUEUE**. The **-q** option is not given to **exim** if **/etc/sysconfig/exim** exists and **QUEUE** is empty or undefined.

32.1.10. /etc/sysconfig/firstboot

The first time the system boots, the **/sbin/init** program calls the **etc/rc.d/init.d/firstboot** script, which in turn launches the **Setup Agent**. This application allows the user to install the latest updates as well as additional applications and documentation.

The **/etc/sysconfig/firstboot** file tells the **Setup Agent** application not to run on subsequent reboots. To run it the next time the system boots, remove **/etc/sysconfig/firstboot** and execute **chkconfig --level 5 firstboot on**.

32.1.11. /etc/sysconfig/gpm

The **/etc/sysconfig/gpm** file is used to pass arguments to the **gpm** daemon at boot time. The **gpm** daemon is the mouse server which allows mouse acceleration and middle-click pasting. For more information about what parameters are available for this file, refer to the **gpm** man page. By default, the **DEVICE** directive is set to **/dev/input/mice**.

32.1.12. /etc/sysconfig/hwconf

The **/etc/sysconfig/hwconf** file lists all the hardware that **kudzu** detected on the system, as well as the drivers used, vendor ID, and device ID information. The **kudzu** program detects and configures new and/or changed hardware on a system. The **/etc/sysconfig/hwconf** file is not meant to be manually edited. If edited, devices could suddenly show up as being added or removed.

32.1.13. /etc/sysconfig/i18n

The **/etc/sysconfig/i18n** file sets the default language, any supported languages, and the default system font. For example:

```
LANG="en_US.UTF-8"
SUPPORTED="en_US.UTF-8:en_US:en"
SYSFONT="latarcyrheb-sun16"
```

32.1.14. /etc/sysconfig/init

The **/etc/sysconfig/init** file controls how the system appears and functions during the boot process.

The following values may be used:

- **BOOTUP=<*value*>**, where **<*value*>** is one of the following:
  - **color** — The standard color boot display, where the success or failure of devices and services starting up is shown in different colors.

    - **verbose** — An old style display which provides more information than purely a message of success or failure.
    - Anything else means a new display, but without ANSI-formatting.

- **RES_COL=*<value>***, where ***<value>*** is the number of the column of the screen to start status labels. The default is set to 60.
- **MOVE_TO_COL=*<value>***, where ***<value>*** moves the cursor to the value in the **RES_COL** line via the **echo -en** command.
- **SETCOLOR_SUCCESS=*<value>***, where ***<value>*** sets the success color via the **echo -en** command. The default color is set to green.
- **SETCOLOR_FAILURE=*<value>***, where ***<value>*** sets the failure color via the **echo -en** command. The default color is set to red.
- **SETCOLOR_WARNING=*<value>***, where ***<value>*** sets the warning color via the **echo -en** command. The default color is set to yellow.
- **SETCOLOR_NORMAL=*<value>***, where ***<value>*** resets the color to "normal" via the **echo -en**.
- **LOGLEVEL=*<value>***, where ***<value>*** sets the initial console logging level for the kernel. The default is 3; 8 means everything (including debugging), while 1 means only kernel panics. The **syslogd** daemon overrides this setting once started.
- **PROMPT=*<value>***, where ***<value>*** is one of the following boolean values:
    - **yes** — Enables the key check for interactive mode.
    - **no** — Disables the key check for interactive mode.

32.1.15. /etc/sysconfig/ip6tables-config

The **/etc/sysconfig/ip6tables-config** file stores information used by the kernel to set up IPv6 packet filtering at boot time or whenever the **ip6tables** service is started.

Do not modify this file by hand unless familiar with how to construct **ip6tables** rules. Rules also can be created manually using the **/sbin/ip6tables** command. Once created, add the rules to the **/etc/sysconfig/ip6tables** file by typing the following command:

```
service ip6tables save
```

Once this file exists, any firewall rules saved in it persists through a system reboot or a service restart.

For more information on **ip6tables**, refer to Section 48.9, “IPTables”.

32.1.16. /etc/sysconfig/iptables-config

The **/etc/sysconfig/iptables-config** file stores information used by the kernel to set up packet filtering services at boot time or whenever the service is started.

Do not modify this file by hand unless you are familiar with constructing **iptables** rules. The easiest way to add rules is to use the **Security Level Configuration Tool** (**system-config-securitylevel**) application to create a firewall. These applications automatically edit this file at the end of the process.

Rules can also be created manually using the **`/sbin/iptables`** command. Once created, add the rule(s) to the **`/etc/sysconfig/iptables`** file by typing the following command:

```
service iptables save
```

Once this file exists, any firewall rules saved in it persists through a system reboot or a service restart.

For more information on **`iptables`**, refer to Section 48.9, “IPTables”.

32.1.17. `/etc/sysconfig/irda`

The **`/etc/sysconfig/irda`** file controls how infrared devices on the system are configured at startup.

The following values may be used:

- **`IRDA=<value>`**, where **`<value>`** is one of the following boolean values:
  - **`yes`** — **`irattach`** runs and periodically checks to see if anything is trying to connect to the infrared port, such as another notebook computer trying to make a network connection. For infrared devices to work on the system, this line must be set to **`yes`**.
  - **`no`** — **`irattach`** does not run, preventing infrared device communication.
- **`DEVICE=<value>`**, where **`<value>`** is the device (usually a serial port) that handles infrared connections. A sample serial device entry could be **`/dev/ttyS2`**.
- **`DONGLE=<value>`**, where **`<value>`** specifies the type of dongle being used for infrared communication. This setting exists for people who use serial dongles rather than real infrared ports. A dongle is a device that is attached to a traditional serial port to communicate via infrared. This line is commented out by default because notebooks with real infrared ports are far more common than computers with add-on dongles. A sample dongle entry could be **`actisys+`**.
- **`DISCOVERY=<value>`**, where **`<value>`** is one of the following boolean values:
  - **`yes`** — Starts **`irattach`** in discovery mode, meaning it actively checks for other infrared devices. This must be turned on for the machine to actively look for an infrared connection (meaning the peer that does not initiate the connection).
  - **`no`** — Does not start **`irattach`** in discovery mode.

32.1.18. `/etc/sysconfig/keyboard`

The **`/etc/sysconfig/keyboard`** file controls the behavior of the keyboard. The following values may be used:

- **`KEYBOARDTYPE="sun|pc"`** where **`sun`** means a Sun keyboard is attached on **`/dev/kbd`**, or **`pc`** means a PS/2 keyboard connected to a PS/2 port.
- **`KEYTABLE="<file>"`**, where **`<file>`** is the name of a keytable file.

  For example: **`KEYTABLE="us"`**. The files that can be used as keytables start in **`/lib/kbd/keymaps/i386`** and branch into different keyboard layouts from there, all labeled **`<file>.kmap.gz`**. The first file found beneath **`/lib/kbd/keymaps/i386`** that matches the **`KEYTABLE`** setting is used.

32.1.19. `/etc/sysconfig/kudzu`

The **`/etc/sysconfig/kuzdu`** file triggers a safe probe of the system hardware by **`kudzu`** at boot time. A safe probe is one that disables serial port probing.

- **`SAFE=<`*`value`*`>`**, where **`<`*`value`*`>`** is one of the following:
  - **`yes`** — **`kuzdu`** does a safe probe.
  - **`no`** — **`kuzdu`** does a normal probe.

32.1.20. `/etc/sysconfig/named`

The **`/etc/sysconfig/named`** file is used to pass arguments to the **`named`** daemon at boot time. The **`named`** daemon is a *Domain Name System* (*DNS*) server which implements the *Berkeley Internet Name Domain* (*BIND*) version 9 distribution. This server maintains a table of which hostnames are associated with IP addresses on the network.

Currently, only the following values may be used:

- **`ROOTDIR="</`*`some/where`*`>"`**, where **`</`*`some/where`*`>`** refers to the full directory path of a configured chroot environment under which **`named`** runs. This chroot environment must first be configured. Type **`info chroot`** for more information.
- **`OPTIONS="<`*`value`*`>"`**, where **`<`*`value`*`>`** is any option listed in the man page for **`named`** except **`-t`**. In place of **`-t`**, use the **`ROOTDIR`** line above.

For more information about available parameters for this file, refer to the **`named`** man page. For detailed information on how to configure a BIND DNS server, refer to Chapter 19, *Berkeley Internet Name Domain (BIND)*. By default, the file contains no parameters.

32.1.21. `/etc/sysconfig/network`

The **`/etc/sysconfig/network`** file is used to specify information about the desired network configuration. The following values may be used:

- **`NETWORKING=<`*`value`*`>`**, where **`<`*`value`*`>`** is one of the following boolean values:
  - **`yes`** — Networking should be configured.
  - **`no`** — Networking should not be configured.
- **`HOSTNAME=<`*`value`*`>`**, where **`<`*`value`*`>`** should be the *Fully Qualified Domain Name* (*FQDN*), such as **`hostname.expample.com`**, but can be whatever hostname is necessary.
- **`GATEWAY=<`*`value`*`>`**, where **`<`*`value`*`>`** is the IP address of the network's gateway.
- **`GATEWAYDEV=<`*`value`*`>`**, where **`<`*`value`*`>`** is the gateway device, such as **`eth0`**. Configure this option if you have multiple interfaces on the same subnet, and require one of those interfaces to be the preferred route to the default gateway.
- **`NISDOMAIN=<`*`value`*`>`**, where **`<`*`value`*`>`** is the NIS domain name.
- **`NOZEROCONF=<`*`value`*`>`**, where setting **`<`*`value`*`>`** to **`true`** disables the zeroconf route.

  By default, the zeroconf route (169.254.0.0) is enabled when the system boots. For more information about zeroconf, refer to http://www.zeroconf.org/.

Warning

Do not use custom initscripts to configure network settings. When performing a post-boot network service restart, custom initscripts configuring network settings that are run outside of the network init script lead to unpredictable results.

32.1.22. `/etc/sysconfig/nfs`

NFS requires portmap, which dynamically assigns ports for RPC services. This causes problems for configuring firewall rules. To overcome this problem, use the **`/etc/sysconfig/nfs`** file to control which ports the required RPC services run on.

The **`/etc/sysconfig/nfs`** may not exist by default on all systems. If it does not exist, create it and add the following variables (alternatively, if the file exists, un-comment and change the default entries as required):

`MOUNTD_PORT=`*`x`*

control which TCP and UDP port mountd (rpc.mountd) uses. Replace *x* with an unused port number.

`STATD_PORT=`*`x`*

control which TCP and UDP port status (rpc.statd) uses. Replace *x* with an unused port number.

`LOCKD_TCPPORT=`*`x`*

control which TCP port nlockmgr (rpc.lockd) uses. Replace *x* with an unused port number.

`LOCKD_UDPPORT=`*`x`*

control which UDP port nlockmgr (rpc.lockd) uses. Replace *x* with an unused port number.

If NFS fails to start, check **`/var/log/messages`**. Normally, NFS will fail to start if you specify a port number that is already in use. After editing **`/etc/sysconfig/nfs`** restart the NFS service by running the **`service nfs restart`** command. Run the **`rpcinfo -p`** command to confirm the changes.

To configure a firewall to allow NFS:

1. Allow TCP and UDP port 2049 for NFS.
2. Allow TCP and UDP port 111 (portmap/sunrpc).
3. Allow the TCP and UDP port specified with **`MOUNTD_PORT="`*`x`*`"`**
4. Allow the TCP and UDP port specified with **`STATD_PORT="`*`x`*`"`**
5. Allow the TCP port specified with **`LOCKD_TCPPORT="`*`x`*`"`**
6. Allow the UDP port specified with **`LOCKD_UDPPORT="`*`x`*`"`**

32.1.23. `/etc/sysconfig/ntpd`

The **`/etc/sysconfig/ntpd`** file is used to pass arguments to the **`ntpd`** daemon at boot time. The **`ntpd`** daemon sets and maintains the system clock to synchronize with an Internet standard time server. It implements version 4 of the Network Time Protocol (NTP). For more information about what parameters are available for this file, use a Web browser to view the following file:

`/usr/share/doc/ntp-<version>/ntpd.htm` (where *<version>* is the version number of **`ntpd`**). By default, this file sets the owner of the **`ntpd`** process to the user **`ntp`**.

32.1.24. `/etc/sysconfig/radvd`

The **`/etc/sysconfig/radvd`** file is used to pass arguments to the **`radvd`** daemon at boot time. The **`radvd`** daemon listens for router requests and sends router advertisements for the IP version 6 protocol. This service allows hosts on a network to dynamically change their default routers based on these router advertisements. For more information about available parameters for this file, refer to the **`radvd`** man page. By default, this file sets the owner of the **`radvd`** process to the user **`radvd`**.

32.1.25. `/etc/sysconfig/samba`

The **`/etc/sysconfig/samba`** file is used to pass arguments to the **`smbd`** and the **`nmbd`** daemons at boot time. The **`smbd`** daemon offers file sharing connectivity for Windows clients on the network. The **`nmbd`** daemon offers NetBIOS over IP naming services. For more information about what parameters are available for this file, refer to the **`smbd`** man page. By default, this file sets **`smbd`** and **`nmbd`** to run in daemon mode.

32.1.26. `/etc/sysconfig/selinux`

The **`/etc/sysconfig/selinux`** file contains the basic configuration options for SELinux. This file is a symbolic link to **`/etc/selinux/config`**.

32.1.27. `/etc/sysconfig/sendmail`

The **`/etc/sysconfig/sendmail`** file allows messages to be sent to one or more clients, routing the messages over whatever networks are necessary. The file sets the default values for the **Sendmail** application to run. Its default values are set to run as a background daemon and to check its queue each hour in case something has backed up.

Values include:

- **`DAEMON=<`*`value`*`>`**, where **`<`*`value`*`>`** is one of the following:
  - **`yes`** — **Sendmail** should be configured to listen to port 25 for incoming mail. **`yes`** implies the use of **Sendmail**'s **`-bd`** options.
  - **`no`** — **Sendmail** should not be configured to listen to port 25 for incoming mail.
- **`QUEUE=1h`** which is given to **Sendmail** as **`-q$QUEUE`**. The **`-q`** option is not given to **Sendmail** if **`/etc/sysconfig/sendmail`** exists and **`QUEUE`** is empty or undefined.

32.1.28. `/etc/sysconfig/spamassassin`

The **`/etc/sysconfig/spamassassin`** file is used to pass arguments to the **`spamd`** daemon (a daemonized version of **Spamassassin**) at boot time. **Spamassassin** is an email spam filter application. For a list of available options, refer to the **`spamd`** man page. By default, it configures **`spamd`** to run in daemon mode, create user preferences, and auto-create whitelists (allowed bulk senders).

For more information about **Spamassassin**, refer to Section 27.5.2.6, “Spam Filters”.

32.1.29. `/etc/sysconfig/squid`

The **`/etc/sysconfig/squid`** file is used to pass arguments to the **`squid`** daemon at boot time. The **`squid`** daemon is a proxy caching server for Web client applications. For more information on configuring a **`squid`** proxy server, use a Web browser to open the **`/usr/share/doc/squid-<version>/`** directory (replace <*version*> with the **`squid`** version number installed on the system). By default, this file sets **`squid`** to start in daemon mode and sets the amount of time before it shuts itself down.

32.1.30. `/etc/sysconfig/system-config-securitylevel`

The **`/etc/sysconfig/system-config-securitylevel`** file contains all options chosen by the user the last time the **Security Level Configuration Tool** (**`system-config-securitylevel`**) was run. Users should not modify this file by hand. For more information about the **Security Level Configuration Tool**, refer to Section 48.8.2, “Basic Firewall Configuration”.

32.1.31. `/etc/sysconfig/system-config-selinux`

The **`/etc/sysconfig/system-config-selinux`** file contains all options chosen by the user the last time the **SELinux Administration Tool** (**`system-config-selinux`**) was run. Users should not modify this file by hand. For more information about the **SELinux Administration Tool** and SELinux in general, refer to Section 49.2, “Introduction to SELinux”.

32.1.32. `/etc/sysconfig/system-config-users`

The **`/etc/sysconfig/system-config-users`** file is the configuration file for the graphical application, **User Manager**. This file is used to filter out system users such as **`root`**, **`daemon`**, or **`lp`**. This file is edited by the **Preferences** > **Filter system users and groups** pull-down menu in the **User Manager** application and should never be edited by hand. For more information on using this application, refer to Section 37.1, “User and Group Configuration”.

32.1.33. `/etc/sysconfig/system-logviewer`

The **`/etc/sysconfig/system-logviewer`** file is the configuration file for the graphical, interactive log viewing application, **Log Viewer**. This file is edited by the **Edit** > **Preferences** pull-down menu in the **Log Viewer** application and should not be edited by hand. For more information on using this application, refer to Chapter 40, *Log Files*.

32.1.34. `/etc/sysconfig/tux`

The **`/etc/sysconfig/tux`** file is the configuration file for the Red Hat Content Accelerator (formerly known as **TUX**), the kernel-based Web server. For more information on configuring the Red Hat Content Accelerator, use a Web browser to open the **`/usr/share/doc/tux-<version>/tux/index.html`** file (replace <*version*> with the version number of **TUX** installed on the system). The parameters available for this file are listed in **`/usr/share/doc/tux-<version>/tux/parameters.html`**.

32.1.35. `/etc/sysconfig/vncservers`

The **`/etc/sysconfig/vncservers`** file configures the way the *Virtual Network Computing* (*VNC*) server starts up.

VNC is a remote display system which allows users to view the desktop environment not only on the machine where it is running but across different networks on a variety of architectures.

It may contain the following:

- **VNCSERVERS=*<value>***, where ***<value>*** is set to something like **"1:fred"**, to indicate that a VNC server should be started for user fred on display :1. User fred must have set a VNC password using the **vncpasswd** command before attempting to connect to the remote VNC server.

32.1.36. /etc/sysconfig/xinetd

The **/etc/sysconfig/xinetd** file is used to pass arguments to the **xinetd** daemon at boot time. The **xinetd** daemon starts programs that provide Internet services when a request to the port for that service is received. For more information about available parameters for this file, refer to the **xinetd** man page. For more information on the **xinetd** service, refer to Section 48.5.3, "xinetd".

32.2. Directories in the /etc/sysconfig/ Directory

The following directories are normally found in **/etc/sysconfig/**.

apm-scripts/

This directory contains the APM suspend/resume script. Do not edit the files directly. If customization is necessary, create a file called **/etc/sysconfig/apm-scripts/apmcontinue** which is called at the end of the script. It is also possible to control the script by editing **/etc/sysconfig/apmd**.

cbq/

This directory contains the configuration files needed to do *Class Based Queuing* for bandwidth management on network interfaces. CBQ divides user traffic into a hierarchy of classes based on any combination of IP addresses, protocols, and application types.

networking/

This directory is used by the **Network Administration Tool** (**system-config-network**), and its contents should not be edited manually. For more information about configuring network interfaces using the **Network Administration Tool**, refer to Chapter 17, *Network Configuration*.

network-scripts/

This directory contains the following network-related configuration files:

- Network configuration files for each configured network interface, such as **ifcfg-eth0** for the **eth0** Ethernet interface.
- Scripts used to bring network interfaces up and down, such as **ifup** and **ifdown**.
- Scripts used to bring ISDN interfaces up and down, such as **ifup-isdn** and **ifdown-isdn**.
- Various shared network function scripts which should not be edited directly.

For more information on the **network-scripts** directory, refer to Chapter 16, *Network Interfaces*.

rhn/

Deprecated. This directory contains the configuration files and GPG keys used by the RHN Classic content service. No files in this directory should be edited by hand.

This directory is available for legacy systems which are still managed by RHN Classic. Systems which are registered against the Certificate-Based Red Hat Network do not use this directory.

32.3. Additional Resources

This chapter is only intended as an introduction to the files in the **`/etc/sysconfig/`** directory. The following source contains more comprehensive information.

32.3.1. Installed Documentation

- **`/usr/share/doc/initscripts-<`*`version-number`*`>/sysconfig.txt`** — This file contains a more authoritative listing of the files found in the **`/etc/sysconfig/`** directory and the configuration options available for them. The <*version-number*> in the path to this file corresponds to the version of the **`initscripts`** package installed.

Chapter 33. Date and Time Configuration

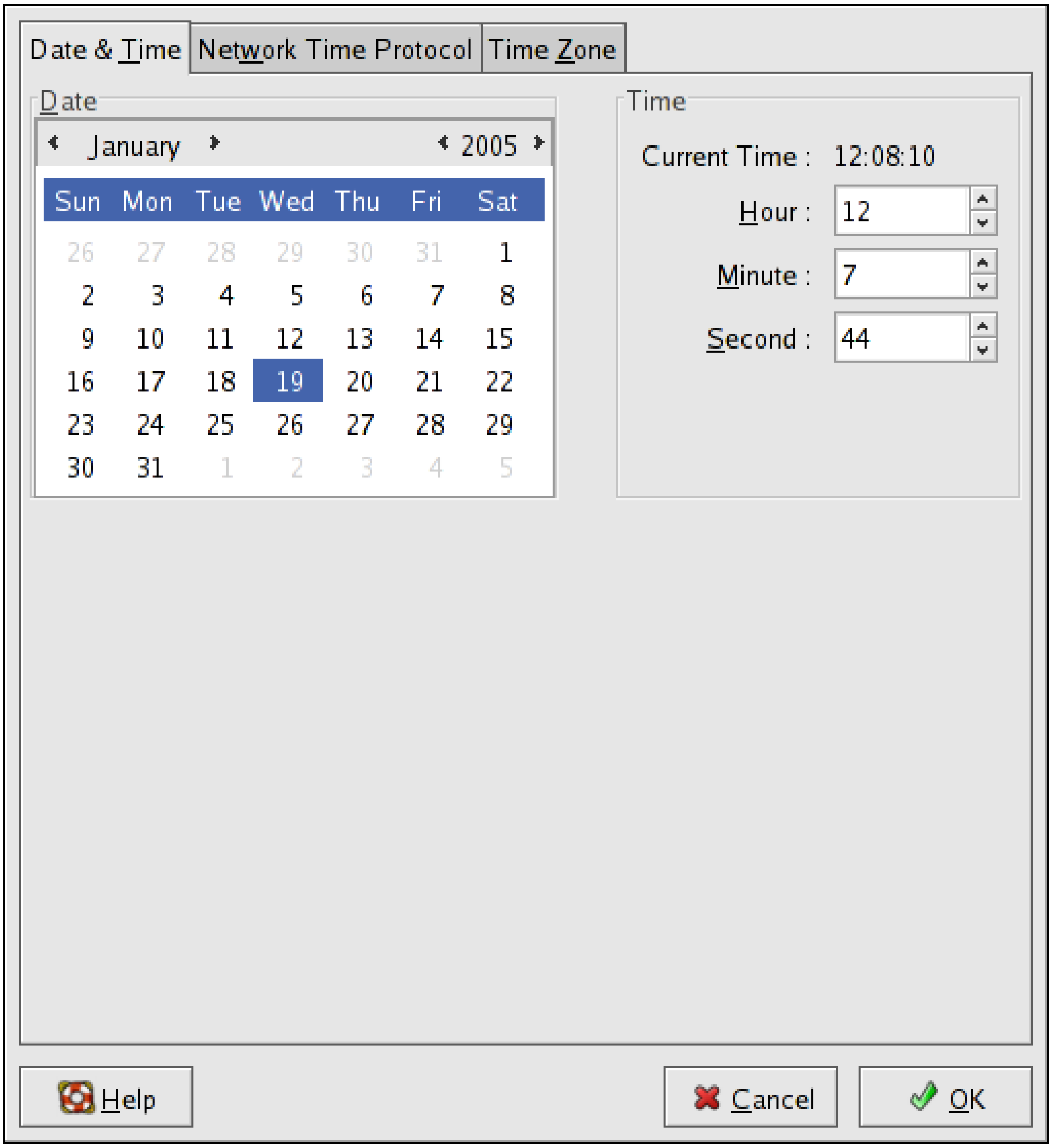
The **Time and Date Properties Tool** allows the user to change the system date and time, to configure the time zone used by the system, and to setup the Network Time Protocol (NTP) daemon to synchronize the system clock with a time server.

You must be running the **X** Window System and have root privileges to use the tool. There are three ways to start the application:

- From the desktop, go to Applications (the main menu on the panel) > **System Settings** > **Date & Time**
- From the desktop, right-click on the time in the toolbar and select **Adjust Date and Time**.
- Type the command **`system-config-date`**, **`system-config-time`**, or **`dateconfig`** at a shell prompt (for example, in an **XTerm** or a **GNOME** terminal).

33.1. Time and Date Properties

As shown in Figure 33.1, “Time and Date Properties”, the first tabbed window that appears is for configuring the system date and time.



Date

January 2005

Sun	Mon	Tue	Wed	Thu	Fri	Sat
26	27	28	29	30	31	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	1	2	3	4	5

Time

Current Time : 12:08:10

Hour : 12

Minute : 7

Second : 44

Figure 33.1. Time and Date Properties

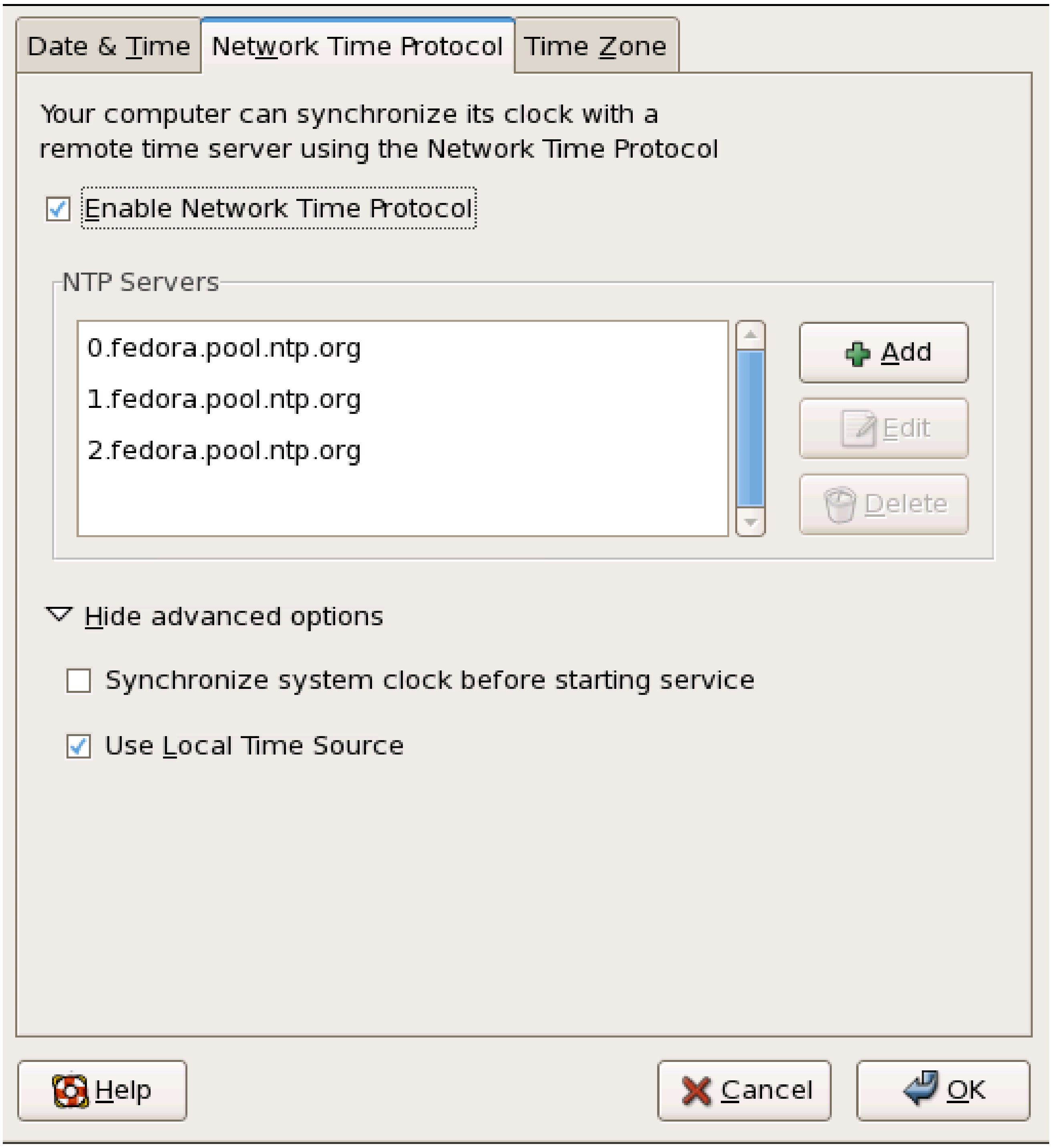
To change the date, use the arrows to the left and right of the month to change the month, use the arrows to the left and right of the year to change the year, and click on the day of the week to change the day of the week.

To change the time, use the up and down arrow buttons beside the **`Hour`**, **`Minute`**, and **`Second`** in the **`Time`** section.

Clicking the **`OK`** button applies any changes made to the date and time, the NTP daemon settings, and the time zone settings. It also exits the program.

33.2. Network Time Protocol (NTP) Properties

As shown in Figure 33.2, “NTP Properties”, the second tabbed window that appears is for configuring NTP.


Date & Time | Network Time Protocol | Time Zone

Your computer can synchronize its clock with a remote time server using the Network Time Protocol

[✓] Enable Network Time Protocol

NTP Servers

0.fedora.pool.ntp.org
1.fedora.pool.ntp.org
2.fedora.pool.ntp.org

[ ] Synchronize system clock before starting service

[✓] Use Local Time Source

Figure 33.2. NTP Properties

The Network Time Protocol (NTP) daemon synchronizes the system clock with a remote time server or time source. The application allows you to configure an NTP daemon to synchronize your system clock with a remote server. To enable this feature, select **`Enable Network Time Protocol`**. This enables the **`NTP Servers`** list and other options. You can choose one of the predefined servers, edit a predefined server by clicking the **`Edit`** or add a new server name by clicking **`Add`**. Your system does not start synchronizing with the NTP server until you click **`OK`**. After clicking **`OK`**, the configuration is saved and the NTP daemon is started (or restarted if it is already running).

Clicking the **`OK`** button applies any changes made to the date and time, the NTP daemon settings, and the time zone settings. It also exits the program.

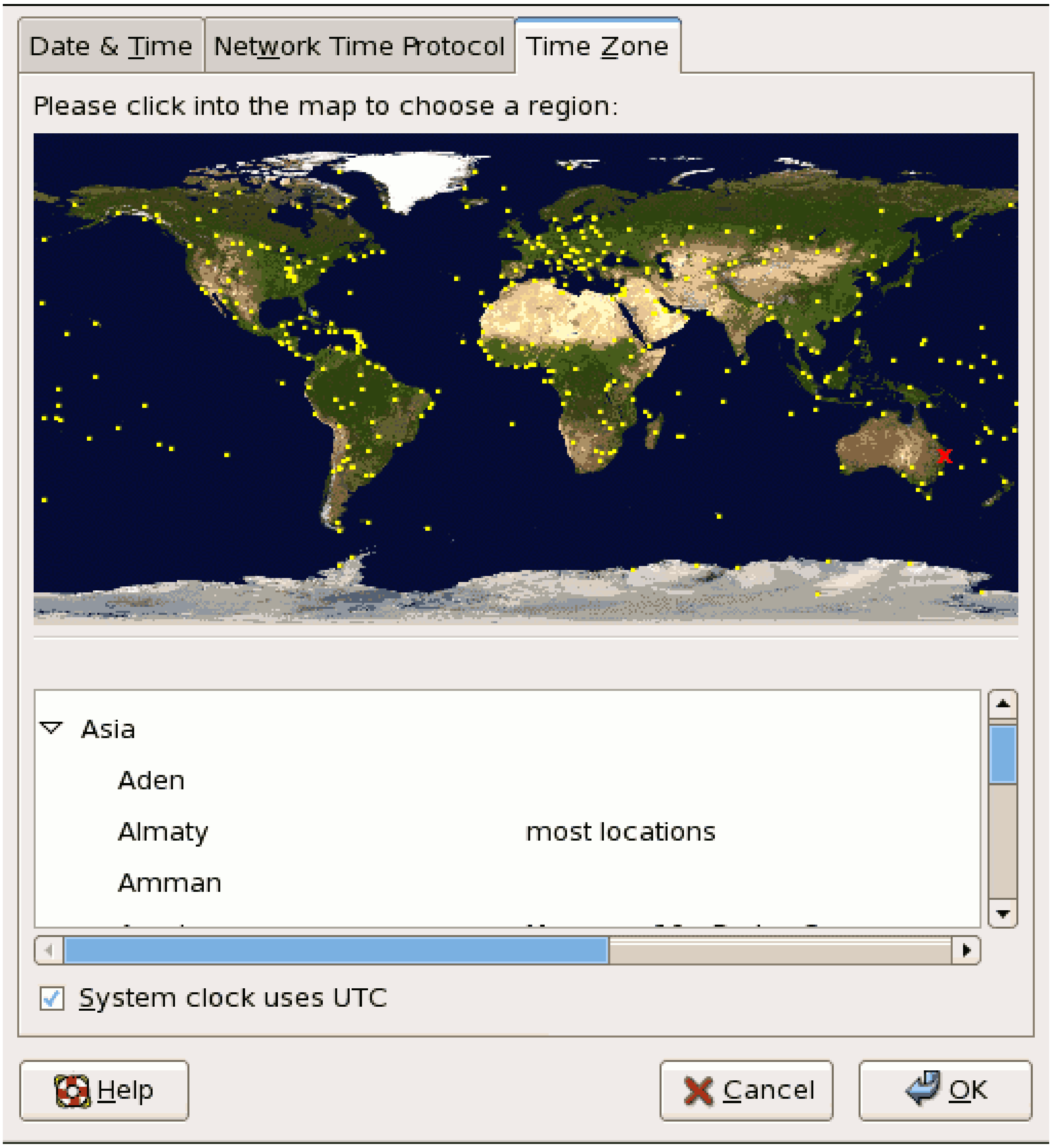
33.3. Time Zone Configuration

As shown in Figure 33.3, “Timezone Properties”, the third tabbed window that appears is for configuring the system time zone.

To configure the system time zone, click the **`Time Zone`** tab. The time zone can be changed by either using the interactive map or by choosing the desired time zone from the list below the map. To use the map, click on the desired region. The map zooms into the region selected, after which you may choose the city specific to your time zone. A red **`X`** appears and the time zone selection changes in the list below the map.

Alternatively, you can also use the list below the map. In the same way that the map lets you choose a region before choosing a city, the list of time zones is now a treelist, with cities and countries grouped within their specific continents. Non-geographic time zones have also been added to address needs in the scientific community.

Click **`OK`** to apply the changes and exit the program.



Asia
Aden
Almaty most locations
Amman

☑ System clock uses UTC

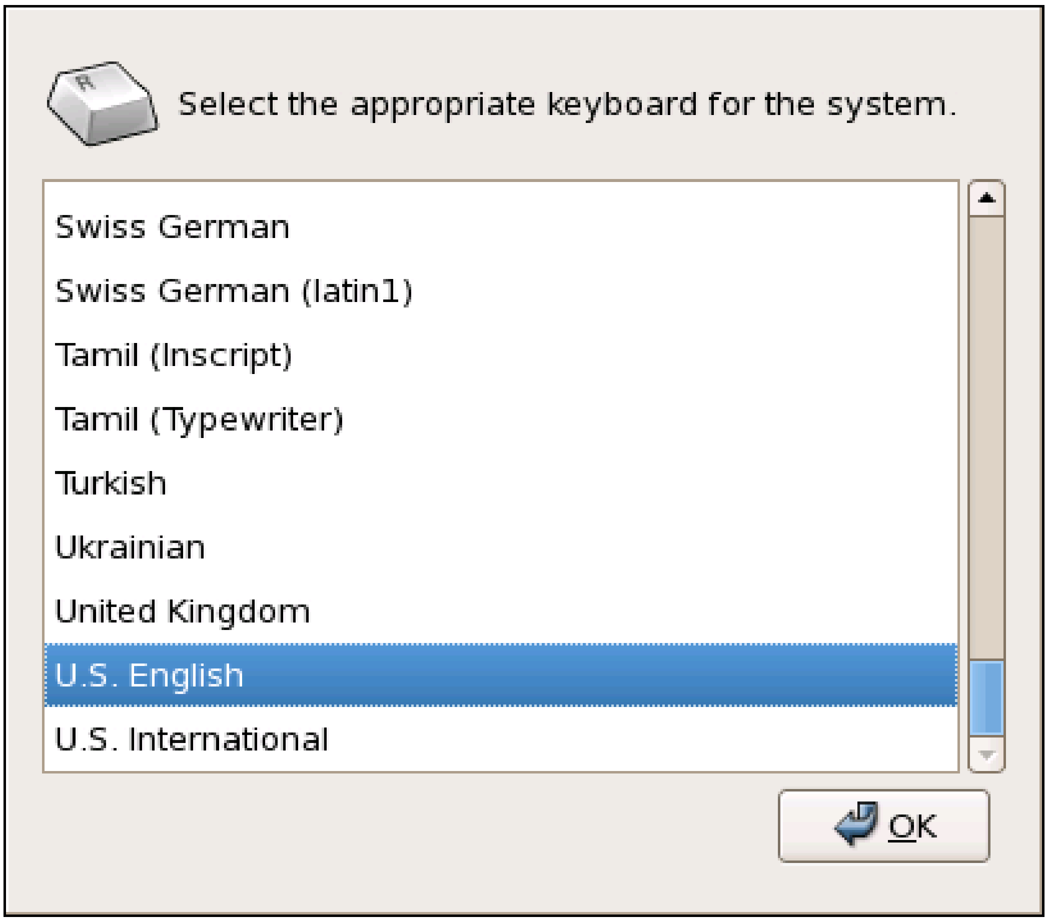
Figure 33.3. Timezone Properties

If your system clock is set to use UTC, select the **`System clock uses UTC`** option. UTC stands for the *Universal Time, Coordinated*, also known as Greenwich Mean Time (GMT). Other time zones are determined by adding or subtracting from the UTC time.

Chapter 34. Keyboard Configuration

The installation program allows you to configure a keyboard layout for your system. To configure a different keyboard layout after installation, use the **Keyboard Configuration Tool**.

To start the **Keyboard Configuration Tool**, select System (on the panel) > **Administration** > **Keyboard**, or type the command **`system-config-keyboard`** at a shell prompt.


Select the appropriate keyboard for the system.

Swiss German
Swiss German (latin1)
Tamil (Inscript)
Tamil (Typewriter)
Turkish
Ukrainian
United Kingdom
U.S. English
U.S. International

Figure 34.1. Keyboard Configuration Tool

Select a keyboard layout from the list (for example, **`U.S. English`**) and click **`OK`**.

Changes take effect immediately.

Chapter 35. The X Window System

While the heart of Red Hat Enterprise Linux is the kernel, for many users, the face of the operating system is the graphical environment provided by the *X Window System*, also called *X*.

Other windowing environments have existed in the UNIX world, including some that predate the release of the X Window System in June 1984. Nonetheless, X has been the default graphical environment for most UNIX-like operating systems, including Red Hat Enterprise Linux, for many years.

The graphical environment for Red Hat Enterprise Linux is supplied by the *X.Org Foundation*, an open source organization created to manage development and strategy for the X Window System and related technologies. X.Org is a large-scale, rapidly developing project with hundreds of developers around the world. It features a wide degree of support for a variety of hardware devices and architectures, and can run on a variety of different operating systems and platforms. This release for Red Hat Enterprise Linux specifically includes the X11R7.1 release of the X Window System.

The X Window System uses a client-server architecture. The *X server* (the **`Xorg`** binary) listens for connections from *X client* applications via a network or local loopback interface. The server communicates with the hardware, such as the video card, monitor, keyboard, and mouse. X client applications exist in the user-space, creating a *graphical user interface* (*GUI*) for the user and passing user requests to the X server.

35.1. The X11R7.1 Release

Red Hat Enterprise Linux 5.10 now uses the X11R7.1 release as the base X Window System, which includes several video driver, EXA, and platform support enhancements over the previous release, among others. In addition, this release also includes several automatic configuration features for the X server.

X11R7.1 is the first release to take specific advantage of the modularization of the X Window System. This modularization, which splits X into logically distinct modules, makes it easier for open source developers to contribute code to the system.

Important

Red Hat Enterprise Linux no longer provides the XFree86™ server packages. Before upgrading a system to the latest version of Red Hat Enterprise Linux, be sure that the system's video card is compatible with the X11R7.1 release by checking the Red Hat Hardware Compatibility List located online at http://hardware.redhat.com/.

In the X11R7.1 release, all libraries, headers, and binaries now live under **`/usr/`** instead of **`/usr/X11R6`**. The **`/etc/X11/`** directory contains configuration files for X client and server applications. This includes configuration files for the X server itself, the **`xfs`** font server, the X display managers, and many other base components.

The configuration file for the newer Fontconfig-based font architecture is still **`/etc/fonts/fonts.conf`**. For more on configuring and adding fonts, refer to Section 35.4, “Fonts”.

Because the X server performs advanced tasks on a wide array of hardware, it requires detailed information about the hardware it works on. The X server automatically detects some of this information; other details must be configured.

The installation program installs and configures X automatically, unless the X11R7.1 release packages are not selected for installation. However, if there are any changes to the monitor, video card or other devices managed by the X server, X must be reconfigured. The best way to do this is to use the **X Configuration Tool** (**`system-config-display`**), particularly for devices that are not detected manually.

In Red Hat Enterprise Linux's default graphical environment, the **X Configuration Tool** is available at System (on the panel) > **Administration** > **Display**.

Changes made with the **X Configuration Tool** take effect after logging out and logging back in.

For more information about **X Configuration Tool**, refer to Chapter 36, *X Window System Configuration*.

In some situations, reconfiguring the X server may require manually editing its configuration file, **`/etc/X11/xorg.conf`**. For information about the structure of this file, refer to Section 35.3, "X Server Configuration Files".

35.2. Desktop Environments and Window Managers

Once an X server is running, X client applications can connect to it and create a GUI for the user. A range of GUIs are possible with Red Hat Enterprise Linux, from the rudimentary *Tab Window Manager* to the highly developed and interactive *GNOME* desktop environment that most Red Hat Enterprise Linux users are familiar with.

To create the latter, more comprehensive GUI, two main classes of X client application must connect to the X server: a *desktop environment* and a *window manager*.

35.2.1. Desktop Environments

A desktop environment integrates various X clients to create a common graphical user environment and development platform.

Desktop environments have advanced features allowing X clients and other running processes to communicate with one another, while also allowing all applications written to work in that environment to perform advanced tasks, such as drag and drop operations.

Red Hat Enterprise Linux provides two desktop environments:

- *GNOME* — The default desktop environment for Red Hat Enterprise Linux based on the GTK+ 2 graphical toolkit.
- *KDE* — An alternative desktop environment based on the Qt 3 graphical toolkit.

Both GNOME and KDE have advanced productivity applications, such as word processors, spreadsheets, and Web browsers; both also provide tools to customize the look and feel of the GUI. Additionally, if both the GTK+ 2 and the Qt libraries are present, KDE applications can run in GNOME and vice-versa.

35.2.2. Window Managers

Window managers are X client programs which are either part of a desktop environment or, in some cases, stand-alone. Their primary purpose is to control the way graphical windows are positioned, resized, or moved. Window managers also control title bars, window focus behavior, and user-specified key and mouse button bindings.

Four window managers are included with Red Hat Enterprise Linux:

`kwin`

The *KWin* window manager is the default window manager for KDE. It is an efficient window manager which supports custom themes.

`metacity`

The *Metacity* window manager is the default window manager for GNOME. It is a simple and efficient window manager which also supports custom themes. To run this window manager, you need to install the **`metacity`** package.

`mwm`

The *Motif Window Manager* (**`mwm`**) is a basic, stand-alone window manager. Since it is designed to be a stand-alone window manager, it should not be used in conjunction with GNOME or KDE. To run this window manager, you need to install the **`openmotif`** package.

`twm`

The minimalist *Tab Window Manager* (**`twm`**, which provides the most basic tool set of any of the window managers, can be used either as a stand-alone or with a desktop environment. It is installed as part of the X11R7.1 release.

To run any of the aforementioned window managers, you will first need to boot into Runlevel 3. For instructions on how to do this, refer to Section 18.1, "Runlevels".

Once you are logged in to Runlevel 3, you will be presented with a terminal prompt, not a graphical environment. To start a window manager, type **`xinit -e`** ***`<path-to-window-manager>`*** at the prompt.

`<path-to-window-manager>` is the location of the window manager binary file. The binary file can be located by typing **`which`** ***`window-manager-name`***, where ***`window-manager-name`*** is the name of the window manager you want to run.

For example:

```
~]# which twm
/usr/bin/twm
~]# xinit -e /usr/bin/twm
```

The first command above returns the absolute path to the **`twm`** window manager, the second command starts **`twm`**.

To exit a window manager, close the last window or press **`Ctrl`**+**`Alt`**+**`Backspace`**. Once you have exited the window manager, you can log back into Runlevel 5 by typing **`startx`** at the prompt.

35.3. X Server Configuration Files

The X server is a single binary executable (**`/usr/bin/Xorg`**). Associated configuration files are stored in the **`/etc/X11/`** directory (as is a symbolic link — X — which points to **`/usr/bin/Xorg`**). The configuration file for the X server is **`/etc/X11/xorg.conf`**.

The directory **`/usr/lib/xorg/modules/`** contains X server modules that can be loaded dynamically at runtime. By default, only some modules in **`/usr/lib/xorg/modules/`** are automatically loaded by the X server.

To load optional modules, they must be specified in the X server configuration file, **`/etc/X11/xorg.conf`**. For more information about loading modules, refer to Section 35.3.1.5, "**`Module`**".

When Red Hat Enterprise Linux 5.10 is installed, the configuration files for X are created using information gathered about the system hardware during the installation process.

35.3.1. `xorg.conf`

While there is rarely a need to manually edit the **`/etc/X11/xorg.conf`** file, it is useful to understand the various sections and optional parameters available, especially when troubleshooting.

35.3.1.1. The Structure

The **`/etc/X11/xorg.conf`** file is comprised of many different sections which address specific aspects of the system hardware.

Each section begins with a **`Section "<section-name>"`** line (where *<section-name>* is the title for the section) and ends with an **`EndSection`** line. Each section contains lines that include option names and one or more option values. These are sometimes enclosed in double quotes (**`"`**).

Lines beginning with a hash mark (**`#`**) are not read by the X server and are used for human-readable comments.

Some options within the **`/etc/X11/xorg.conf`** file accept a boolean switch which turns the feature on or off. Acceptable boolean values are:

- **`1`**, **`on`**, **`true`**, or **`yes`** — Turns the option on.
- **`0`**, **`off`**, **`false`**, or **`no`** — Turns the option off.

The following are some of the more important sections in the order in which they appear in a typical **`/etc/X11/xorg.conf`** file. More detailed information about the X server configuration file can be found in the **`xorg.conf`** man page.

35.3.1.2. `ServerFlags`

The optional **`ServerFlags`** section contains miscellaneous global X server settings. Any settings in this section may be overridden by options placed in the **`ServerLayout`** section (refer to Section 35.3.1.3, "`ServerLayout`" for details).

Each entry within the **`ServerFlags`** section is on its own line and begins with the term **`Option`** followed by an option enclosed in double quotation marks (**`"`**).

The following is a sample **`ServerFlags`** section:

```
Section "ServerFlags"
 Option "DontZap" "true"
EndSection
```

The following lists some of the most useful options:

- **`"DontZap" "<boolean>"`** — When the value of *<boolean>* is set to true, this setting prevents the use of the **`Ctrl`**+**`Alt`**+**`Backspace`** key combination to immediately terminate the X server.
- **`"DontZoom" "<boolean>"`** — When the value of *<boolean>* is set to true, this setting prevents cycling through configured video resolutions using the **`Ctrl`**+**`Alt`**+**`Keypad-Plus`** and **`Ctrl`**+**`Alt`**+**`Keypad-Minus`** key combinations.

35.3.1.3. `ServerLayout`

The **`ServerLayout`** section binds together the input and output devices controlled by the X server. At a minimum, this section must specify one output device and one input device. By default, a monitor (output device) and keyboard (input device) are specified.

The following example illustrates a typical **`ServerLayout`** section:

```
Section  "ServerLayout"
 Identifier     "Default Layout"
 Screen      0  "Screen0" 0 0
 InputDevice    "Mouse0" "CorePointer"
 InputDevice    "Keyboard0" "CoreKeyboard"
EndSection
```

The following entries are commonly used in the **`ServerLayout`** section:

- **`Identifier`** — Specifies a unique name for this **`ServerLayout`** section.
- **`Screen`** — Specifies the name of a **`Screen`** section to be used with the X server. More than one **`Screen`** option may be present.

  The following is an example of a typical **`Screen`** entry:

  ```
  Screen      0  "Screen0" 0 0
  ```

  The first number in this example **`Screen`** entry (**`0`**) indicates that the first monitor connector or *head* on the video card uses the configuration specified in the **`Screen`** section with the identifier **`"Screen0"`**.

  An example of a **`Screen`** section with the identifier **`"Screen0"`** can be found in Section 35.3.1.9, “**`Screen`**”.

  If the video card has more than one head, another **`Screen`** entry with a different number and a different **`Screen`** section identifier is necessary .

  The numbers to the right of **`"Screen0"`** give the absolute X and Y coordinates for the upper-left corner of the screen (**`0 0`** by default).

- **`InputDevice`** — Specifies the name of an **`InputDevice`** section to be used with the X server.

  It is advisable that there be at least two **`InputDevice`** entries: one for the default mouse and one for the default keyboard. The options **`CorePointer`** and **`CoreKeyboard`** indicate that these are the primary mouse and keyboard.

- **`Option "<option-name>"`** — An optional entry which specifies extra parameters for the section. Any options listed here override those listed in the **`ServerFlags`** section.

  Replace *<option-name>* with a valid option listed for this section in the **`xorg.conf`** man page.

It is possible to put more than one **`ServerLayout`** section in the **`/etc/X11/xorg.conf`** file. By default, the server only reads the first one it encounters, however.

If there is an alternative **`ServerLayout`** section, it can be specified as a command line argument when starting an X session.

35.3.1.4. `Files`

The **`Files`** section sets paths for services vital to the X server, such as the font path. This is an optional section, these paths are normally detected automatically. This section may be used to override any automatically detected defaults.

The following example illustrates a typical **`Files`** section:

```
Section "Files"
 RgbPath      "/usr/share/X11/rgb.txt"
 FontPath     "unix/:7100"
EndSection
```

The following entries are commonly used in the **`Files`** section:

- **`RgbPath`** — Specifies the location of the RGB color database. This database defines all valid color names in X and ties them to specific RGB values.
- **`FontPath`** — Specifies where the X server must connect to obtain fonts from the **`xfs`** font server.

  By default, the **`FontPath`** is **`unix/:7100`**. This tells the X server to obtain font information using UNIX-domain sockets for inter-process communication (IPC) on port 7100.

  Refer to Section 35.4, “Fonts” for more information concerning X and fonts.
- **`ModulePath`** — An optional parameter which specifies alternate directories which store X server modules.

35.3.1.5. `Module`

By default, the X server automatically loads the following modules from the **`/usr/lib/xorg/modules/`** directory:

- **`extmod`**
- **`dbe`**
- **`glx`**
- **`freetype`**
- **`type1`**
- **`record`**
- **`dri`**

The default directory for loading these modules can be changed by specifying a different directory with the optional **`ModulePath`** parameter in the **`Files`** section. Refer to Section 35.3.1.4, “**`Files`**” for more information on this section.

Adding a **`Module`** section to **`/etc/X11/xorg.conf`** instructs the X server to load the modules listed in this section *instead* of the default modules.

For example, the following typical **`Module`** section:

```
Section "Module"
 Load  "fbdevhw"
EndSection
```

instructs the X server to load the **`fbdevhw`** instead of the default modules.

As such, if you add a **`Module`** section to **`/etc/X11/xorg.conf`**, you will need to specify any default modules you want to load as well as any extra modules.

35.3.1.6. `InputDevice`

Each **`InputDevice`** section configures one input device for the X server. Systems typically have at least one **`InputDevice`** section for the keyboard. It is perfectly normal to have no entry for a mouse, as most mouse settings are automatically detected.

The following example illustrates a typical **`InputDevice`** section for a keyboard:

```
Section "InputDevice"
        Identifier  "Keyboard0"
        Driver      "kbd"
        Option      "XkbModel" "pc105"
        Option      "XkbLayout" "us"
EndSection
```

The following entries are commonly used in the **`InputDevice`** section:

- **`Identifier`** — Specifies a unique name for this **`InputDevice`** section. This is a required entry.
- **`Driver`** — Specifies the name of the device driver X must load for the device.
- **`Option`** — Specifies necessary options pertaining to the device.

  A mouse may also be specified to override any autodetected defaults for the device. The following options are typically included when adding a mouse in the **`xorg.conf`**:

  - **`Protocol`** — Specifies the protocol used by the mouse, such as **`IMPS/2`**.
  - **`Device`** — Specifies the location of the physical device.
  - **`Emulate3Buttons`** — Specifies whether to allow a two-button mouse to act like a three-button mouse when both mouse buttons are pressed simultaneously.

  Consult the **`xorg.conf`** man page for a list of valid options for this section.

35.3.1.7. `Monitor`

Each **`Monitor`** section configures one type of monitor used by the system. This is an optional entry as well, as most monitors are now automatically detected.

The easiest way to configure a monitor is to configure X during the installation process or by using the **X Configuration Tool**. For more information about using the**X Configuration Tool**, refer to Chapter 36, *X Window System Configuration*.

This example illustrates a typical **`Monitor`** section for a monitor:

```
Section "Monitor"
 Identifier   "Monitor0"
 VendorName   "Monitor Vendor"
 ModelName    "DDC Probed Monitor - ViewSonic G773-2"
 DisplaySize  320 240
 HorizSync    30.0 - 70.0
 VertRefresh  50.0 - 180.0
EndSection
```

Warning

Be careful when manually editing values in the **`Monitor`** section of **`/etc/X11/xorg.conf`**. Inappropriate values can damage or destroy a monitor. Consult the monitor's documentation for a listing of safe operating parameters.

The following are commonly entries used in the **`Monitor`** section:

- **`Identifier`** — Specifies a unique name for this **`Monitor`** section. This is a required entry.
- **`VendorName`** — An optional parameter which specifies the vendor of the monitor.
- **`ModelName`** — An optional parameter which specifies the monitor's model name.
- **`DisplaySize`** — An optional parameter which specifies, in millimeters, the physical size of the monitor's picture area.
- **`HorizSync`** — Specifies the range of horizontal sync frequencies compatible with the monitor in kHz. These values help the X server determine the validity of built-in or specified **`Modeline`** entries for the monitor.
- **`VertRefresh`** — Specifies the range of vertical refresh frequencies supported by the monitor, in kHz. These values help the X server determine the validity of built in or specified **`Modeline`** entries for the monitor.
- **`Modeline`** — An optional parameter which specifies additional video modes for the monitor at particular resolutions, with certain horizontal sync and vertical refresh resolutions. Refer to the **`xorg.conf`** man page for a more detailed explanation of **`Modeline`** entries.
- **`Option "<option-name>"`** — An optional entry which specifies extra parameters for the section. Replace *<option-name>* with a valid option listed for this section in the **`xorg.conf`** man page.

35.3.1.8. `Device`

Each **`Device`** section configures one video card on the system. While one **`Device`** section is the minimum, additional instances may occur for each video card installed on the machine.

The best way to configure a video card is to configure X during the installation process or by using the **X Configuration Tool**. For more about using the **X Configuration Tool**, refer to *Chapter 36, X Window System Configuration*.

The following example illustrates a typical **`Device`** section for a video card:

```
Section "Device"
 Identifier  "Videocard0"
 Driver      "mga"
 VendorName  "Videocard vendor"
 BoardName   "Matrox Millennium G200"
 VideoRam    8192
 Option      "dpms"
EndSection
```

The following entries are commonly used in the **`Device`** section:

- **`Identifier`** — Specifies a unique name for this **`Device`** section. This is a required entry.

- **`Driver`** — Specifies which driver the X server must load to utilize the video card. A list of drivers can be found in **`/usr/share/hwdata/videodrivers`**, which is installed with the **`hwdata`** package.

- **`VendorName`** — An optional parameter which specifies the vendor of the video card.

- **`BoardName`** — An optional parameter which specifies the name of the video card.

- **`VideoRam`** — An optional parameter which specifies the amount of RAM available on the video card in kilobytes. This setting is only necessary for video cards the X server cannot probe to detect the amount of video RAM.

- **`BusID`** — An entry which specifies the bus location of the video card. On systems with only one video carc a **`BusID`** entry is optional and may not even be present in the default **`/etc/X11/xorg.conf`** file. On systems with more than one video card, however, a **`BusID`** entry must be present.

- **`Screen`** — An optional entry which specifies which monitor connector or head on the video card the **`Device`** section configures. This option is only useful for video cards with multiple heads.

  If multiple monitors are connected to different heads on the same video card, separate **`Device`** sections must exist and each of these sections must have a different **`Screen`** value.

  Values for the **`Screen`** entry must be an integer. The first head on the video card has a value of**`0`**. The value for each additional head increments this value by one.

- **`Option "<option-name>"`** — An optional entry which specifies extra parameters for the section. Replace *<option-name>* with a valid option listed for this section in the **`xorg.conf`** man page.

  One of the more common options is **`"dpms"`** (for Display Power Management Signaling, a VESA standard), which activates the Service Star energy compliance setting for the monitor.

35.3.1.9. `Screen`

Each **`Screen`** section binds one video card (or video card head) to one monitor by referencing the**`Device`** section and the **`Monitor`** section for each. While one **`Screen`** section is the minimum, additional instances may occur for each video card and monitor combination present on the machine.

The following example illustrates a typical **`Screen`** section:

```
Section "Screen"
 Identifier "Screen0"
 Device     "Videocard0"
 Monitor    "Monitor0"
 DefaultDepth     16
 SubSection "Display"
  Depth     24
  Modes    "1280x1024" "1280x960" "1152x864" "1024x768" "800x600" "640x480"
 EndSubSection
 SubSection "Display"
  Depth     16
  Modes    "1152x864" "1024x768" "800x600" "640x480"
 EndSubSection
EndSection
```

The following entries are commonly used in the **`Screen`** section:

- **`Identifier`** — Specifies a unique name for this **`Screen`** section. This is a required entry.

- **`Device`** — Specifies the unique name of a **`Device`** section. This is a required entry.
- **`Monitor`** — Specifies the unique name of a **`Monitor`** section. This is only required if a specific **`Monitor`** section is defined in the **`xorg.conf`** file. Normally, monitors are automatically detected.
- **`DefaultDepth`** — Specifies the default color depth in bits. In the previous example, **`16`** (which provides thousands of colors) is the default. Only one **`DefaultDepth`** is permitted, although this can be overridden with the Xorg command line option **`-depth <n>`**,where **`<n>`** is any additional depth specified.
- **`SubSection "Display"`** — Specifies the screen modes available at a particular color depth. The **`Screen`** section can have multiple **`Display`** subsections, which are entirely optional since screen modes are automatically detected.

  This subsection is normally used to override autodetected modes.
- **`Option "<option-name>"`** — An optional entry which specifies extra parameters for the section. Replace *<option-name>* with a valid option listed for this section in the **`xorg.conf`** man page.

35.3.1.10. `DRI`

The optional **`DRI`** section specifies parameters for the *Direct Rendering Infrastructure* (*DRI*). DRI is an interface which allows 3D software applications to take advantage of 3D hardware acceleration capabilities built into most modern video hardware. In addition, DRI can improve 2D performance via hardware acceleration, if supported by the video card driver.

This section rarely appears, as the DRI Group and Mode are automatically initialized to default values. If a different Group or Mode is desired, then adding this section to the **`xorg.conf`** file will override those defaults.

The following example illustrates a typical **`DRI`** section:

```
Section "DRI"
 Group        0
 Mode         0666
EndSection
```

Since different video cards use DRI in different ways, do not add to this section without first referring to http://dri.sourceforge.net/.

35.4. Fonts

Red Hat Enterprise Linux uses two subsystems to manage and display fonts under X: *Fontconfig* and **`xfs`**.

The newer Fontconfig font subsystem simplifies font management and provides advanced display features, such as anti-aliasing. This system is used automatically for applications programmed using the Qt 3 or GTK+ 2 graphical toolkit.

For compatibility, Red Hat Enterprise Linux includes the original font subsystem, called the core X font subsystem. This system, which is over 15 years old, is based around the *X Font Server* (*xfs*).

This section discusses how to configure fonts for X using both systems.

35.4.1. Fontconfig

The Fontconfig font subsystem allows applications to directly access fonts on the system and use Xft or other rendering mechanisms to render Fontconfig fonts with advanced anti-aliasing. Graphical applications can use the Xft library with Fontconfig to draw text to the screen.

Over time, the Fontconfig/Xft font subsystem replaces the core X font subsystem.

Important

The Fontconfig font subsystem does not yet work for **OpenOffice.org**, which uses its own font rendering technology.

It is important to note that Fontconfig uses the **`/etc/fonts/fonts.conf`** configuration file, which should not be edited by hand.

Note

Due to the transition to the new font system, GTK+ 1.2 applications are not affected by any changes made via the **Font Preferences** dialog (accessed by selecting System (on the panel) > **Preferences** > **Fonts**). For these applications, a font can be configured by adding the following lines to the file **`~/.gtkrc.mine`**:

```
style "user-font" {
 fontset = "<font-specification>"
}

widget_class "*" style "user-font"
```

Replace *<font-specification>* with a font specification in the style used by traditional X applications, such as **`-adobe-helvetica-medium-r-normal--*-120-*-*-*-*-*-*`**. A full list of core fonts can be obtained by running **`xlsfonts`** or created interactively using the **`xfontsel`** command.

35.4.1.1. Adding Fonts to Fontconfig

Adding new fonts to the Fontconfig subsystem is a straightforward process.

1. To add fonts system-wide, copy the new fonts into the **`/usr/share/fonts/`** directory. It is a good idea to create a new subdirectory, such as **`local/`** or similar, to help distinguish between user-installed and default fonts.

   To add fonts for an individual user, copy the new fonts into the **`.fonts/`** directory in the user's home directory.

2. Use the **`fc-cache`** command to update the font information cache, as in the following example:

   ```
   fc-cache <path-to-font-directory>
   ```

   In this command, replace *<path-to-font-directory>* with the directory containing the new fonts (either **`/usr/share/fonts/local/`** or **`/home/<user>/.fonts/`**).

Note

Individual users may also install fonts graphically, by typing **`fonts:///`** into the **Nautilus** address bar, and dragging the new font files there.

Important

If the font file name ends with a **`.gz`** extension, it is compressed and cannot be used until uncompressed. To do this, use the **`gunzip`** command or double-click the file and drag the font to a directory in **Nautilus**.

35.4.2. Core X Font System

For compatibility, Red Hat Enterprise Linux provides the core X font subsystem, which uses the X Font Serve (**`xfs`**) to provide fonts to X client applications.

The X server looks for a font server specified in the **`FontPath`** directive within the **`Files`** section of the **`/etc/X11/xorg.conf`** configuration file. Refer to Section 35.3.1.4, "**`Files`**" for more information about the **`FontPath`** entry.

The X server connects to the **`xfs`** server on a specified port to acquire font information. For this reason, the **`xfs`** service must be running for X to start. For more about configuring services for a particular runlevel, refer to Chapter 18, *Controlling Access to Services*.

35.4.2.1. `xfs` Configuration

The **`/etc/rc.d/init.d/xfs`** script starts the **`xfs`** server. Several options can be configured within its configuration file, **`/etc/X11/fs/config`**.

The following lists common options:

- **`alternate-servers`** — Specifies a list of alternate font servers to be used if this font server is not available. A comma must separate each font server in a list.
- **`catalogue`** — Specifies an ordered list of font paths to use. A comma must separate each font path in a list.

  Use the string **`:unscaled`** immediately after the font path to make the unscaled fonts in that path load first. Then specify the entire path again, so that other scaled fonts are also loaded.
- **`client-limit`** — Specifies the maximum number of clients the font server services. The default is **`10`**.
- **`clone-self`** — Allows the font server to clone a new version of itself when the **`client-limit`** is hit. By default, this option is **`on`**.
- **`default-point-size`** — Specifies the default point size for any font that does not specify this value. The value for this option is set in decipoints. The default of **`120`** corresponds to a 12 point font.
- **`default-resolutions`** — Specifies a list of resolutions supported by the X server. Each resolution in the list must be separated by a comma.

- **deferglyphs** — Specifies whether to defer loading *glyphs* (the graphic used to visually represent a font). To disable this feature use **none**, to enable this feature for all fonts use **all**, or to turn this feature on only for 16-bit fonts use **16**.
- **error-file** — Specifies the path and file name of a location where **xfs** errors are logged.
- **no-listen** — Prevents **xfs** from listening to particular protocols. By default, this option is set to **tcp** to prevent **xfs** from listening on TCP ports for security reasons.

Note

If **xfs** is used to serve fonts over the network, remove this line.

- **port** — Specifies the TCP port that **xfs** listens on if **no-listen** does not exist or is commented out.
- **use-syslog** — Specifies whether to use the system error log.

35.4.2.2. Adding Fonts to `xfs`

To add fonts to the core X font subsystem (**xfs**), follow these steps:

1. If it does not already exist, create a directory called **/usr/share/fonts/local/** using the following command as root:

   ```
   mkdir /usr/share/fonts/local/
   ```

   If creating the **/usr/share/fonts/local/** directory is necessary, it must be added to the **xfs** path using the following command as root:

   ```
   chkfontpath --add /usr/share/fonts/local/
   ```

2. Copy the new font file into the **/usr/share/fonts/local/** directory
3. Update the font information by issuing the following command as root:

   ```
   ttmkfdir -d /usr/share/fonts/local/ -o
   /usr/share/fonts/local/fonts.scale
   ```

4. Reload the **xfs** font server configuration file by issuing the following command as root:

   ```
   service xfs reload
   ```

35.5. Runlevels and X

In most cases, the Red Hat Enterprise Linux installer configures a machine to boot into a graphical login environment, known as *Runlevel 5*. It is possible, however, to boot into a text-only multi-user mode called *Runlevel 3* and begin an X session from there.

For more information about runlevels, refer to Section 18.1, “Runlevels”.

The following subsections review how X starts up in both runlevel 3 and runlevel 5.

35.5.1. Runlevel 3

When in runlevel 3, the best way to start an X session is to log in and type **`startx`**. The **`startx`** command is a front-end to the **`xinit`** command, which launches the X server (**`Xorg`**) and connects X client applications to it. Because the user is already logged into the system at runlevel 3, **`startx`** does not launch a display manager or authenticate users. Refer to Section 35.5.2, "Runlevel 5" for more information about display managers.

When the **`startx`** command is executed, it searches for the **`.xinitrc`** file in the user's home directory to define the desktop environment and possibly other X client applications to run. If no **`.xinitrc`** file is present, it uses the system default **`/etc/X11/xinit/xinitrc`** file instead.

The default **`xinitrc`** script then searches for user-defined files and default system files, including **`.Xresources`**, **`.Xmodmap`**, and **`.Xkbmap`** in the user's home directory, and **`Xresources`**, **`Xmodmap`**, and **`Xkbmap`** in the **`/etc/X11/`** directory. The **`Xmodmap`** and **`Xkbmap`** files, if they exist, are used by the **`xmodmap`** utility to configure the keyboard. The **`Xresources`** file is read to assign specific preference values to applications.

After setting these options, the **`xinitrc`** script executes all scripts located in the **`/etc/X11/xinit/xinitrc.d/`** directory. One important script in this directory is **`xinput.sh`**, which configures settings such as the default language.

Next, the **`xinitrc`** script attempts to execute **`.Xclients`** in the user's home directory and turns to **`/etc/X11/xinit/Xclients`** if it cannot be found. The purpose of the **`Xclients`** file is to start the desktop environment or, possibly, just a basic window manager. The **`.Xclients`** script in the user's home directory starts the user-specified desktop environment in the **`.Xclients-default`** file. If **`.Xclients`** does not exist in the user's home directory, the standard **`/etc/X11/xinit/Xclients`** script attempts to start another desktop environment, trying GNOME first and then KDE followed by **`twm`**.

When in runlevel 3, the user is returned to a text mode user session after ending an X session.

35.5.2. Runlevel 5

When the system boots into runlevel 5, a special X client application called a *display manager* is launched. A user must authenticate using the display manager before any desktop environment or window managers are launched.

Depending on the desktop environments installed on the system, three different display managers are available to handle user authentication.

- **`GNOME`** — The default display manager for Red Hat Enterprise Linux, **`GNOME`** allows the user to configure language settings, shutdown, restart or log in to the system.
- **`KDE`** — KDE's display manager which allows the user to shutdown, restart or log in to the system.
- **`xdm`** — A very basic display manager which only lets the user log in to the system.

When booting into runlevel 5, the **`prefdm`** script determines the preferred display manager by referencing the **`/etc/sysconfig/desktop`** file. A list of options for this file is available in this file:

```
/usr/share/doc/initscripts-<version-number>/sysconfig.txt
```

where <*version-number*> is the version number of the **`initscripts`** package.

Each of the display managers reference the **`/etc/X11/xdm/Xsetup_0`** file to set up the login screen. Once the user logs into the system, the **`/etc/X11/xdm/GiveConsole`** script runs to assign ownership of the console to the user. Then, the **`/etc/X11/xdm/Xsession`** script runs to accomplish many of the tasks normally performed by the **`xinitrc`** script when starting X from runlevel 3, including setting system and user resources, as well as running the scripts in the **`/etc/X11/xinit/xinitrc.d/`** directory.

Users can specify which desktop environment they want to utilize when they authenticate using the **`GNOME`** or **`KDE`** display managers by selecting it from the **Sessions** menu item (accessed by selecting System (on the panel) > **Preferences** > **More Preferences** > **Sessions**). If the desktop environment is not specified in the display manager, the **`/etc/X11/xdm/Xsession`** script checks the **`.xsession`** and **`.Xclients`** files in the user's home directory to decide which desktop environment to load. As a last resort, the **`/etc/X11/xinit/Xclients`** file is used to select a desktop environment or window manager to use in the same way as runlevel 3.

When the user finishes an X session on the default display (**`:0`**) and logs out, the **`/etc/X11/xdm/TakeConsole`** script runs and reassigns ownership of the console to the root user. The original display manager, which continues running after the user logged in, takes control by spawning a new display manager. This restarts the X server, displays a new login window, and starts the entire process over again.

The user is returned to the display manager after logging out of X from runlevel 5.

For more information on how display managers control user authentication, refer to the **`/usr/share/doc/gdm-<`*`version-number`*`>/README`** (where *<version-number>* is the version number for the **`gdm`** package installed) and the **`xdm`** man page.

35.6. Additional Resources

There is a large amount of detailed information available about the X server, the clients that connect to it, and the assorted desktop environments and window managers.

35.6.1. Installed Documentation

- **`/usr/share/X11/doc/`** — contains detailed documentation on the X Window System architecture, as well as how to get additional information about the Xorg project as a new user.
- **`man xorg.conf`** — Contains information about the **`xorg.conf`** configuration files, including the meaning and syntax for the different sections within the files.
- **`man Xorg`** — Describes the **`Xorg`** display server.

35.6.2. Useful Websites

- http://www.X.org/ — Home page of the X.Org Foundation, which produces the X11R7.1 release of the X Window System. The X11R7.1 release is bundled with Red Hat Enterprise Linux to control the necessary hardware and provide a GUI environment.
- http://dri.sourceforge.net/ — Home page of the DRI (Direct Rendering Infrastructure) project. The DRI is the core hardware 3D acceleration component of X.
- http://www.gnome.org/ — Home of the GNOME project.
- http://www.kde.org/ — Home of the KDE desktop environment.

Chapter 36. X Window System Configuration

During installation, the system's monitor, video card, and display settings are configured. To change any of these settings after installation, use the **X Configuration Tool**.

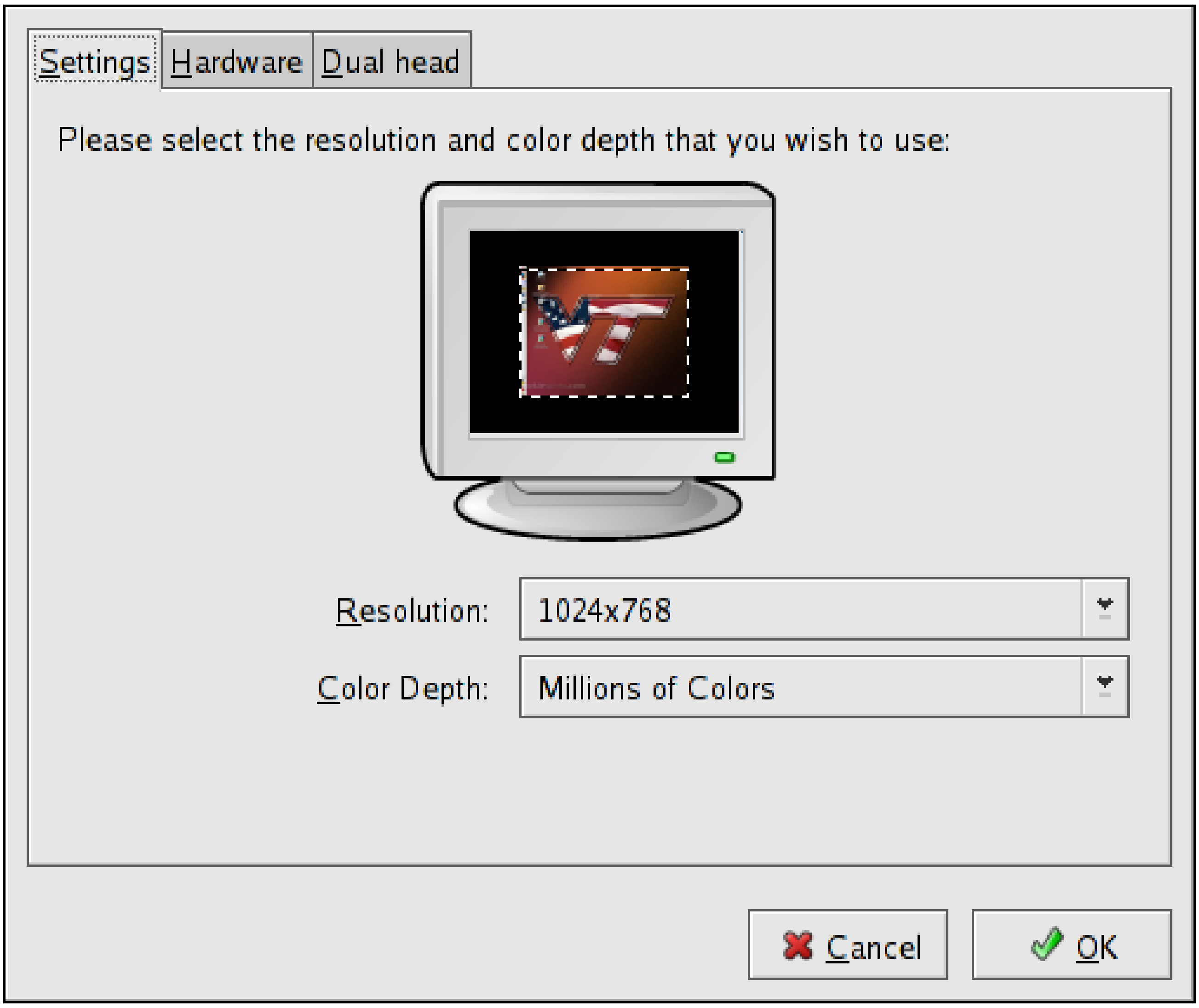
To start the **X Configuration Tool**, go to System (on the panel) > **Administration** > **Display**, or type the command **`system-config-display`** at a shell prompt (for example, in an XTerm or GNOME terminal). If the X Window System is not running, a small version of X is started to run the program.

After changing any of the settings, log out of the graphical desktop and log back in to enable the changes.

36.1. Display Settings

The **`Settings`** tab allows users to change the *resolution* and *color depth*. The display of a monitor consists of tiny dots called *pixels*. The number of pixels displayed at one time is called the resolution. For example, the resolution 1024x768 means that 1024 horizontal pixels and 768 vertical pixels are used. The higher the resolution values, the more images the monitor can display at one time.

The color depth of the display determines how many possible colors are displayed. A higher color depth means more contrast between colors.



Please select the resolution and color depth that you wish to use:

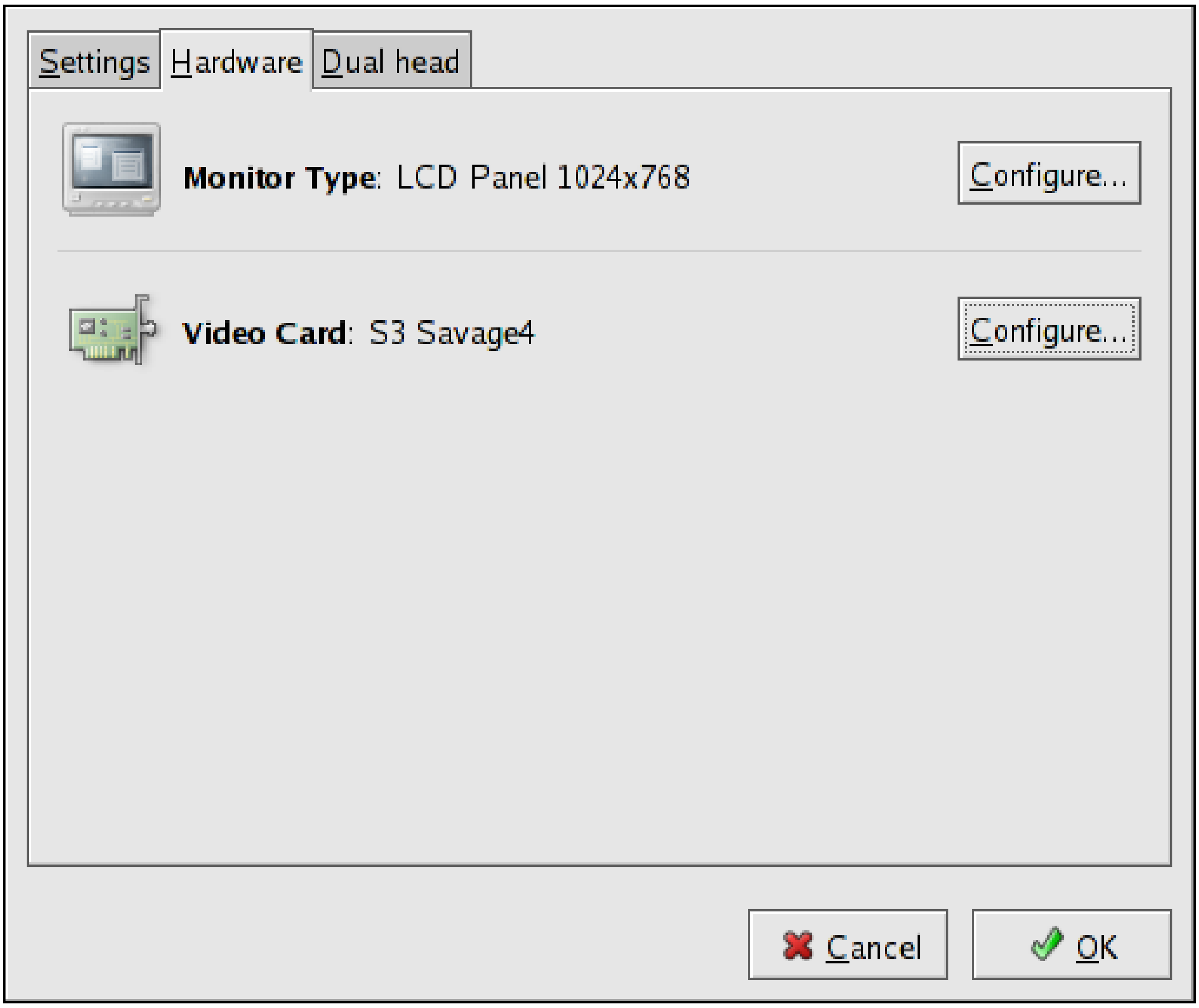
Resolution: 1024x768

Color Depth: Millions of Colors

Figure 36.1. Display Settings

36.2. Display Hardware Settings

When the **X Configuration Tool** is started, it probes the monitor and video card. If the hardware is probed properly, the information for it is shown on the **`Hardware`** tab as shown in Figure 36.2, "Display Hardware Settings".



Monitor Type: LCD Panel 1024x768

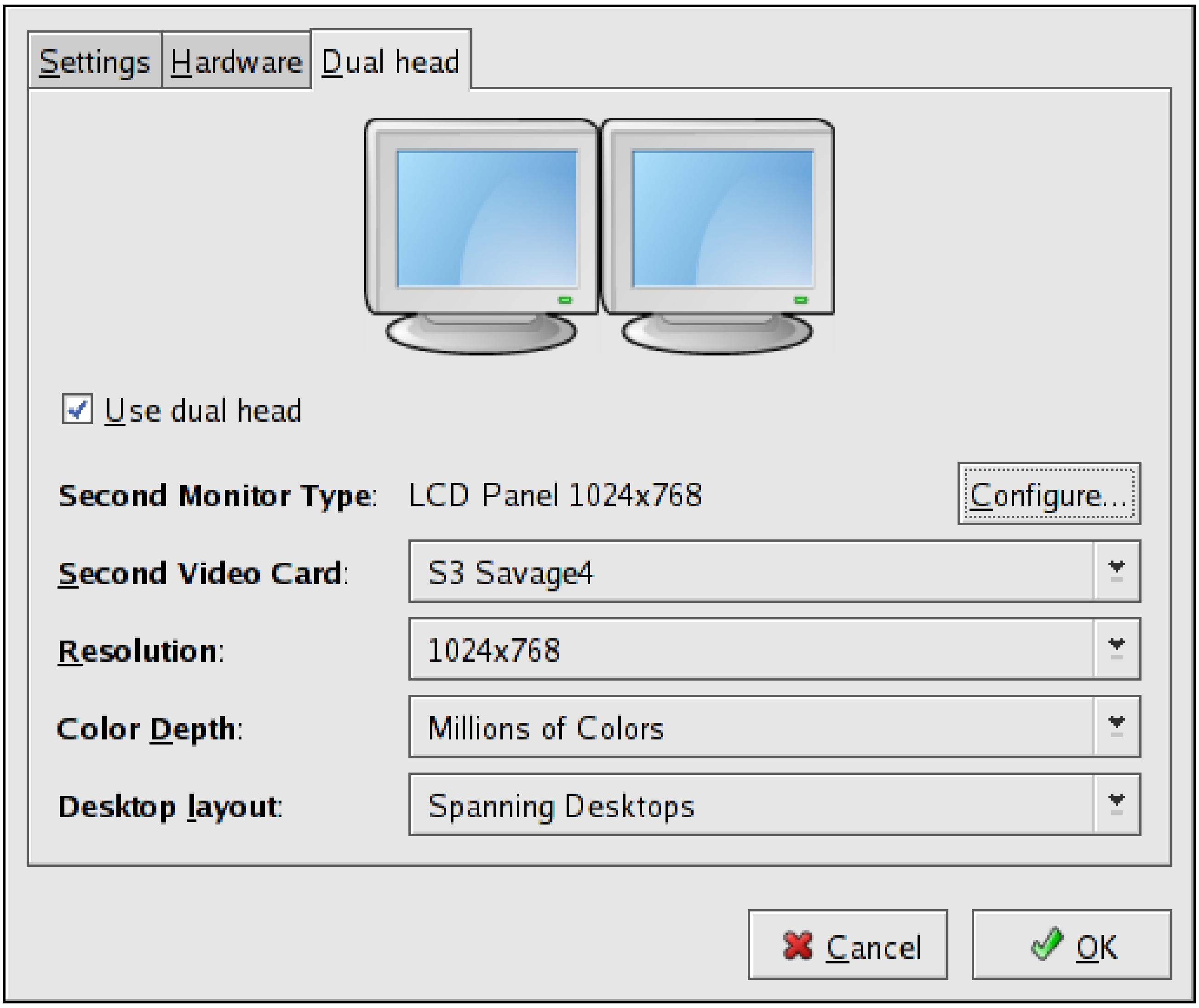
Video Card: S3 Savage4

Figure 36.2. Display Hardware Settings

To change the monitor type or any of its settings, click the corresponding **`Configure`** button. To change the video card type or any of its settings, click the **`Configure`** button beside its settings.

36.3. Dual Head Display Settings

If multiple video cards are installed on the system, dual head monitor support is available and is configured via the **`Dual head`** tab, as shown in Figure 36.3, "Dual Head Display Settings".



[✓] Use dual head

Second Monitor Type:	LCD Panel 1024x768
Second Video Card:	S3 Savage4
Resolution:	1024x768
Color Depth:	Millions of Colors
Desktop layout:	Spanning Desktops

Figure 36.3. Dual Head Display Settings

To enable use of Dual head, check the **`Use dual head`** checkbox.

To configure the second monitor type, click the corresponding **`Configure`** button. You can also configure the other Dual head settings by using the corresponding drop-down list.

For the **`Desktop layout`** option, selecting **`Spanning Desktops`** allows both monitors to use an enlarged usable workspace. Selecting **`Individual Desktops`** shares the mouse and keyboard among the displays, but restricts windows to a single display.

Chapter 37. Users and Groups

The control of *users* and *groups* is a core element of Red Hat Enterprise Linux system administration.

Users can be either people (meaning accounts tied to physical users) or accounts which exist for specific applications to use.

Groups are logical expressions of organization, tying users together for a common purpose. Users within a group can read, write, or execute files owned by that group.

Each user and group has a unique numerical identification number called a *userid* (*UID*) and a *groupid* (*GID*), respectively.

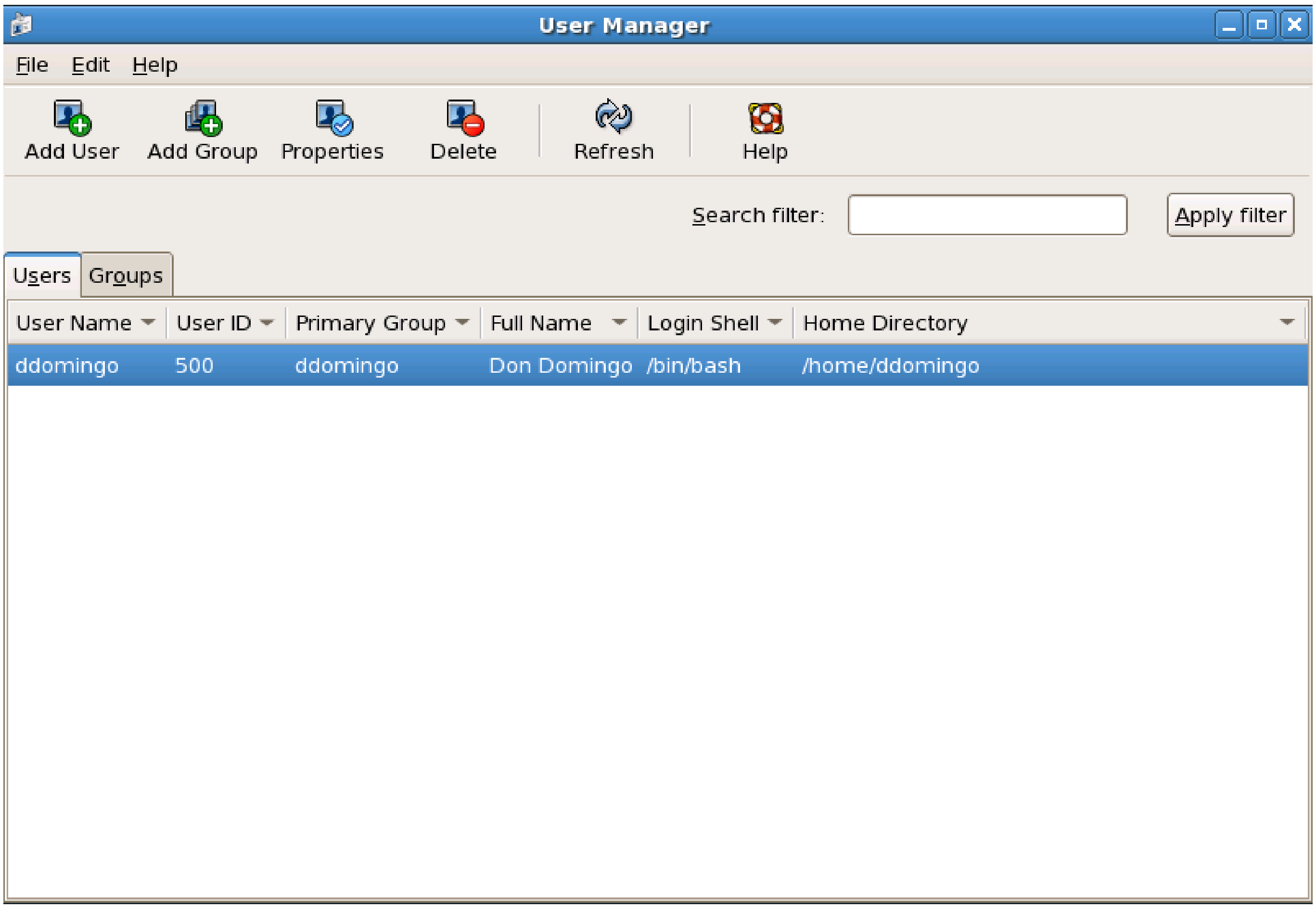
A user who creates a file is also the owner and group owner of that file. The file is assigned separate read, write, and execute permissions for the owner, the group, and everyone else. The file owner can be changed only by the root user, and access permissions can be changed by both the root user and file owner.

Red Hat Enterprise Linux also supports *access control lists* (*ACLs*) for files and directories which allow permissions for specific users outside of the owner to be set. For more information about ACLs, refer to Chapter 10, *Access Control Lists*.

37.1. User and Group Configuration

The **User Manager** allows you to view, modify, add, and delete local users and groups.

To use the **User Manager**, you must be running the X Window System, have root privileges, and have the **`system-config-users`** RPM package installed. To start the **User Manager** from the desktop, go to System (on the panel) > **Administration** > **Users & Groups**. You can also type the command **`system-config-users`** at a shell prompt (for example, in an XTerm or a GNOME terminal).



User Name	User ID	Primary Group	Full Name	Login Shell	Home Directory
ddomingo	500	ddomingo	Don Domingo	/bin/bash	/home/ddomingo

Figure 37.1. User Manager

To view a list of local users on the system, click the **`Users`** tab. To view a list of local groups on the system, click the **`Groups`** tab.

To find a specific user or group, type the first few letters of the name in the **`Search filter`** field. Press **`Enter`** or click the **`Apply filter`** button. The filtered list is displayed.

To sort the users or groups, click on the column name. The users or groups are sorted according to the value of that column.

Red Hat Enterprise Linux reserves user IDs below 500 for system users. By default, **User Manager** does not display system users. To view all users, including the system users, go to **Edit** > **Preferences** and uncheck **Hide system users and groups** from the dialog box.

37.1.1. Adding a New User

To add a new user, click the **`Add User`** button. A window as shown in Figure 37.2, "New User" appears. Type the username and full name for the new user in the appropriate fields. Type the user's password in the **`Password`** and **`Confirm Password`** fields. The password must be at least six characters.

Note

It is advisable to use a much longer password, as this makes it more difficult for an intruder to guess it and access the account without permission. It is also recommended that the password not be based on a dictionary term; use a combination of letters, numbers and special characters.

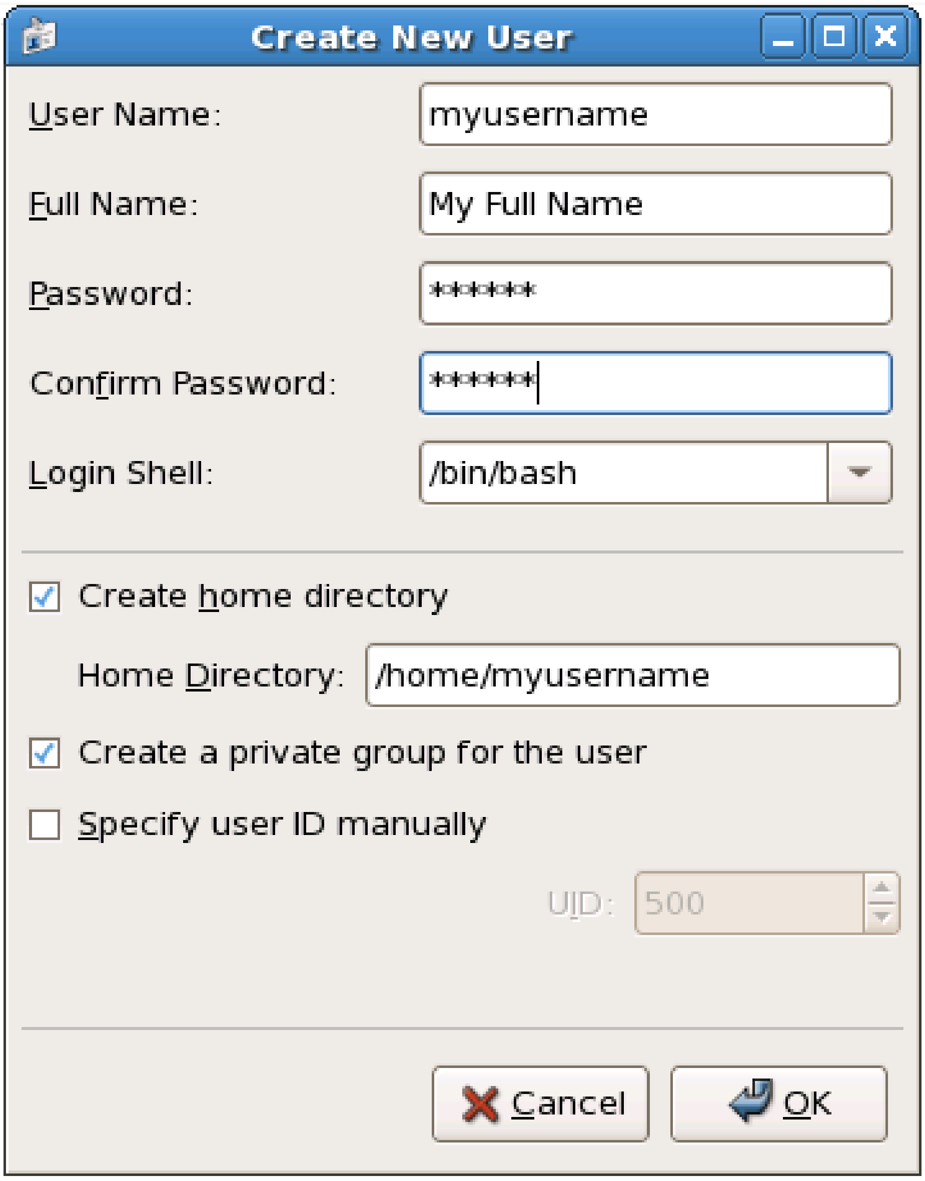
Select a login shell. If you are not sure which shell to select, accept the default value of **`/bin/bash`**. The default home directory is **`/home/<username>/`**. You can change the home directory that is created for the user, or you can choose not to create the home directory by unselecting **`Create home directory`**.

If you select to create the home directory, default configuration files are copied from the **`/etc/skel/`** directory into the new home directory.

Red Hat Enterprise Linux uses a *user private group* (UPG) scheme. The UPG scheme does not add or change anything in the standard UNIX way of handling groups; it offers a new convention. Whenever you create a new user, by default, a unique group with the same name as the user is created. If you do not want to create this group, unselect **`Create a private group for the user`**.

To specify a user ID for the user, select **`Specify user ID manually`**. If the option is not selected, the next available user ID above 500 is assigned to the new user. Because Red Hat Enterprise Linux reserves user IDs below 500 for system users, it is not advisable to manually assign user IDs 1-499.

Click **`OK`** to create the user.



Create New User

User Name: myusername

Full Name: My Full Name

Password: *******

Confirm Password: *******

Login Shell: /bin/bash

[✓] Create home directory

Home Directory: /home/myusername

[✓] Create a private group for the user

[ ] Specify user ID manually

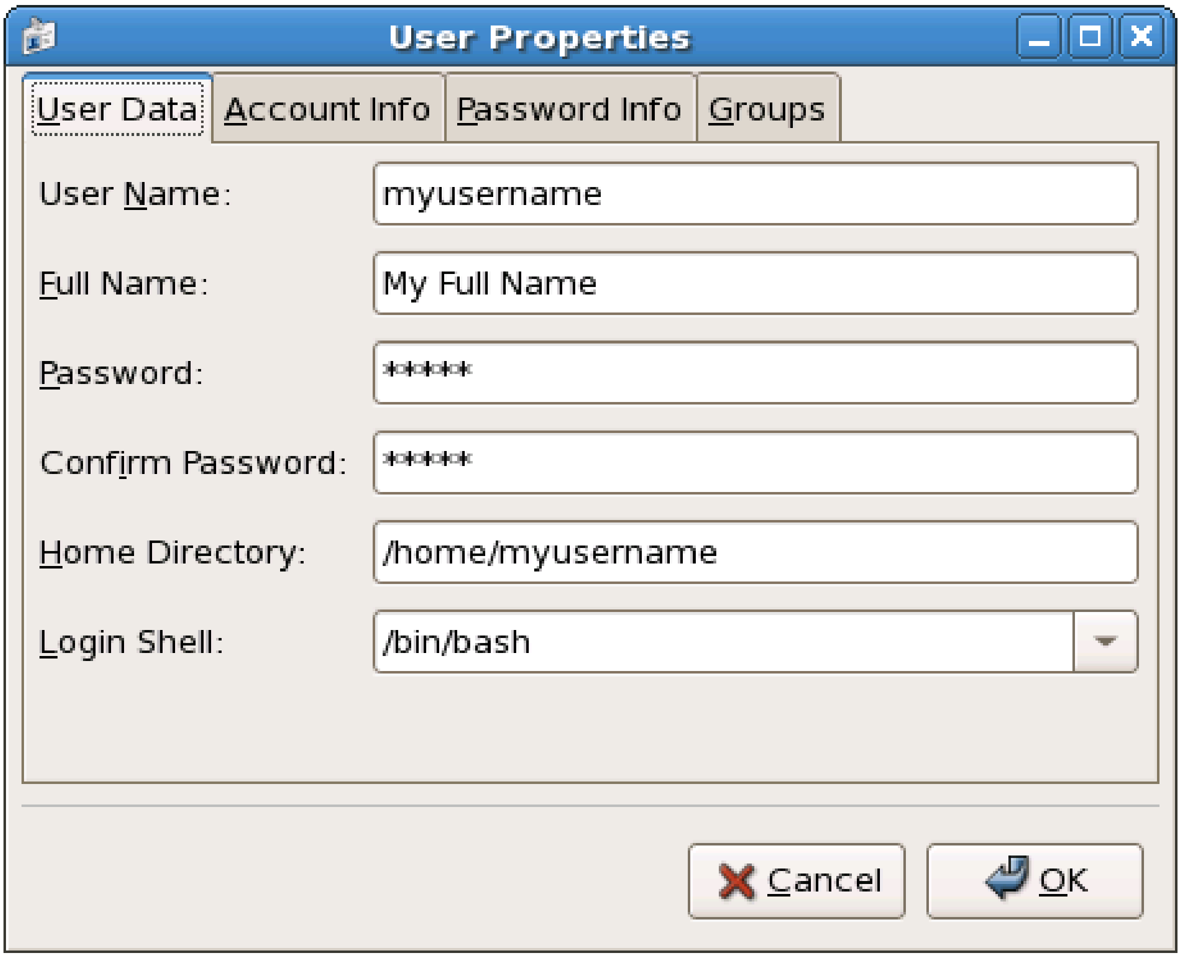
UID: 500

Figure 37.2. New User

To configure more advanced user properties, such as password expiration, modify the user's properties after adding the user. Refer to Section 37.1.2, “Modifying User Properties” for more information.

37.1.2. Modifying User Properties

To view the properties of an existing user, click on the **`Users`** tab, select the user from the user list, and click **`Properties`** from the menu (or choose **File** > **Properties** from the pulldown menu). A window similar to Figure 37.3, “User Properties” appears.



User Name: myusername
Full Name: My Full Name
Password: ******
Confirm Password: ******
Home Directory: /home/myusername
Login Shell: /bin/bash

Figure 37.3. User Properties

The **`User Properties`** window is divided into multiple tabbed pages:

- **`User Data`** — Shows the basic user information configured when you added the user. Use this tab to change the user's full name, password, home directory, or login shell.
- **`Account Info`** — Select **`Enable account expiration`** if you want the account to expire on a certain date. Enter the date in the provided fields. Select **`Local password is locked`** to lock the user account and prevent the user from logging into the system.
- **`Password Info`** — Displays the date that the user's password last changed. To force the user to change passwords after a certain number of days, select **`Enable password expiration`** and enter a desired value in the **`Days before change required:`** field. The number of days before the user's password expires, the number of days before the user is warned to change passwords, and days before the account becomes inactive can also be changed.
- **`Groups`** — Allows you to view and configure the Primary Group of the user, as well as other groups that you want the user to be a member of.

37.1.3. Adding a New Group

To add a new user group, click the **`Add Group`** button. A window similar to Figure 37.4, "New Group" appears. Type the name of the new group to create. To specify a group ID for the new group, select **`Specify group ID manually`** and select the GID. Note that Red Hat Enterprise Linux also reserves group IDs lower than 500 for system groups.

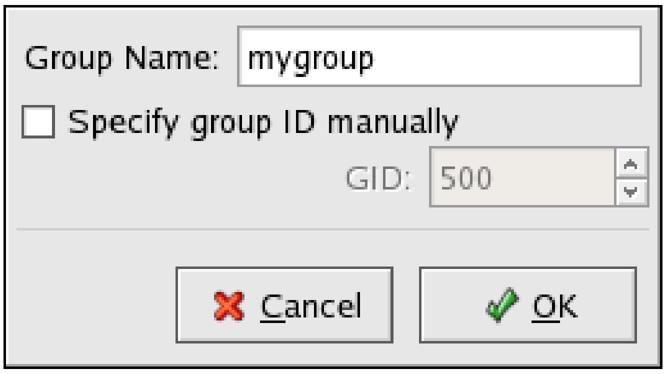


Figure 37.4. New Group

Click **OK** to create the group. The new group appears in the group list.

37.1.4. Modifying Group Properties

To view the properties of an existing group, select the group from the group list and click **`Properties`** from the menu (or choose **File** > **Properties** from the pulldown menu). A window similar to Figure 37.5, "Group Properties" appears.

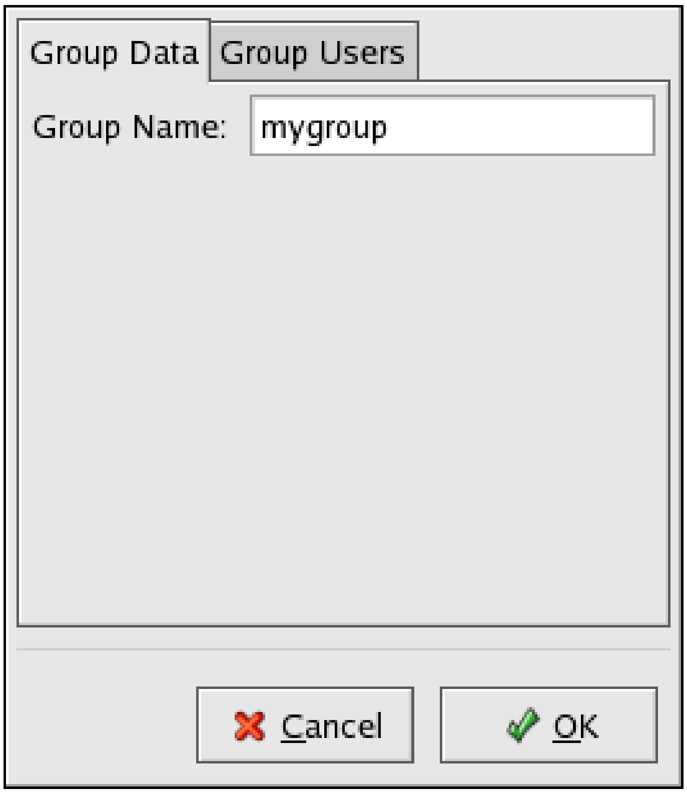


Figure 37.5. Group Properties

The **`Group Users`** tab displays which users are members of the group. Use this tab to add or remove users from the group. Click **OK** to save your changes.

37.2. User and Group Management Tools

Managing users and groups can be a tedious task; this is why Red Hat Enterprise Linux provides tools and conventions to make them easier to manage.

The easiest way to manage users and groups is through the graphical application, **User Manager** (**`system-config-users`**). For more information on **User Manager**, refer to Section 37.1, "User and Group Configuration".

The following command line tools can also be used to manage users and groups:

- **`useradd`**, **`usermod`**, and **`userdel`** — Industry-standard methods of adding, deleting and modifying user accounts
- **`groupadd`**, **`groupmod`**, and **`groupdel`** — Industry-standard methods of adding, deleting, and modifying user groups
- **`gpasswd`** — Industry-standard method of administering the **`/etc/group`** file
- **`pwck`**, **`grpck`** — Tools used for the verification of the password, group, and associated shadow files
- **`pwconv`**, **`pwunconv`** — Tools used for the conversion of passwords to shadow passwords and back to standard passwords

37.2.1. Command Line Configuration

If you prefer command line tools or do not have the X Window System installed, use this section to configure users and groups.

37.2.2. Adding a User

To add a user to the system:

1. Issue the **`useradd`** command to create a locked user account:

   ```
   useradd <username>
   ```

2. Unlock the account by issuing the **`passwd`** command to assign a password and set password aging guidelines:

   ```
   passwd <username>
   ```

Command line options for **`useradd`** are detailed in Table 37.1, "**`useradd`** Command Line Options".

Table 37.1. `useradd` Command Line Options

Option	Description
`-c` '*<comment>*'	*<comment>* can be replaced with any string. This option is generally used to specify the full name of a user.
`-d` *<home-dir>*	Home directory to be used instead of default **`/home/`*`<username>`*`/`**
`-e` *<date>*	Date for the account to be disabled in the format YYYY-MM-DD
`-f` *<days>*	Number of days after the password expires until the account is disabled. If **`0`** is specified, the account is disabled immediately after the password expires. If **`-1`** is specified, the account is not be disabled after the password expires.
`-g` *<group-name>*	Group name or group number for the user's default group. The group must exist prior to being specified here.

Option	Description
-G *<group-list>*	List of additional (other than default) group names or group numbers, separated by commas, of which the user is a member. The groups must exist prior to being specified here.
-m	Create the home directory if it does not exist.
-M	Do not create the home directory.
-n	Do not create a user private group for the user.
-r	Create a system account with a UID less than 500 and without a home directory
-p *<password>*	The password encrypted with **crypt**
-s	User's login shell, which defaults to **/bin/bash**
-u *<uid>*	User ID for the user, which must be unique and greater than 499

37.2.3. Adding a Group

To add a group to the system, use the command **groupadd**:

```
groupadd <group-name>
```

Command line options for **groupadd** are detailed in Table 37.2, "**groupadd** Command Line Options".

Table 37.2. groupadd Command Line Options

Option	Description
-g *<gid>*	Group ID for the group, which must be unique and greater than 499
-r	Create a system group with a GID less than 500
-f	When used with **-g** *<gid>* and *<gid>* already exists, **groupadd** will choose another unique *<gid>* for the group.

37.2.4. Password Aging

For security reasons, it is advisable to require users to change their passwords periodically. This can be done when adding or editing a user on the **Password Info** tab of the **User Manager**.

To configure password expiration for a user from a shell prompt, use the **chage** command with an option from Table 37.3, "**chage** Command Line Options", followed by the username.

Important

Shadow passwords must be enabled to use the **chage** command. For more information, see Section 37.6, "Shadow Passwords".

Table 37.3. chage Command Line Options

Option	Description
-m *<days>*	Specifies the minimum number of days between which the user must change passwords. If the value is 0, the password does not expire.

Option	Description
-M *<days>*	Specifies the maximum number of days for which the password is valid. When the number of days specified by this option plus the number of days specified with the **-d** option is less than the current day, the user must change passwords before using the account.
-d *<days>*	Specifies the number of days since January 1, 1970 the password was changed
-I *<days>*	Specifies the number of inactive days after the password expiration before locking the account. If the value is 0, the account is not locked after the password expires.
-E *<date>*	Specifies the date on which the account is locked, in the format YYYY-MM-DD. Instead of the date, the number of days since January 1, 1970 can also be used.
-W *<days>*	Specifies the number of days before the password expiration date to warn the user.
-l	Lists current account aging settings.

Note

If the **chage** command is followed directly by a username (with no options), it displays the current password aging values and allows them to be changed interactively.

You can configure a password to expire the first time a user logs in. This forces users to change passwords immediately.

1. *Set up an initial password* — There are two common approaches to this step. The administrator can assign a default password or assign a null password.

   To assign a default password, use the following steps:

   - Start the command line Python interpreter with the **python** command. It displays the following:

     ```
     Python 2.4.3 (#1, Jul 21 2006, 08:46:09)
     [GCC 4.1.1 20060718 (Red Hat 4.1.1-9)] on linux2
     Type "help", "copyright", "credits" or "license" for more
     information.
     >>>
     ```

   - At the prompt, type the following commands. Replace *<password>* with the password to encrypt and *<salt>* with a random combination of at least 2 of the following: any alphanumeric character, the slash (/) character or a dot (.):

     ```
     import crypt
     print crypt.crypt("<password>","<salt>")
     ```

     The output is the encrypted password, similar to **'12CsGd8FRcMSM'**.

   - Press **Ctrl-D** to exit the Python interpreter.

   - At the shell, enter the following command (replacing *<encrypted-password>* with the encrypted output of the Python interpreter):

```
usermod -p "<encrypted-password>" <username>
```

Alternatively, you can assign a null password instead of an initial password. To do this, use the following command:

```
usermod -p "" username
```

Warning

Using a null password, while convenient, is a highly unsecure practice, as any third party can log in first an access the system using the unsecure username. Always make sure that the user is ready to log in before unlocking an account with a null password.

2. *Force immediate password expiration* — Type the following command:

```
chage -d 0 username
```

This command sets the value for the date the password was last changed to the epoch (January 1, 1970). This value forces immediate password expiration no matter what password aging policy, if any, is in place.

Upon the initial log in, the user is now prompted for a new password.

37.2.5. Explaining the Process

The following steps illustrate what happens if the command **`useradd juan`** is issued on a system that has shadow passwords enabled:

1. A new line for **`juan`** is created in **`/etc/passwd`**. The line has the following characteristics:
   - It begins with the username **`juan`**.
   - There is an **`x`** for the password field indicating that the system is using shadow passwords.
   - A UID greater than 499 is created. (Under Red Hat Enterprise Linux, UIDs and GIDs below 500 are reserved for system use.)
   - A GID greater than 499 is created.
   - The optional GECOS information is left blank.
   - The home directory for **`juan`** is set to **`/home/juan/`**.
   - The default shell is set to **`/bin/bash`**.
2. A new line for **`juan`** is created in **`/etc/shadow`**. The line has the following characteristics:
   - It begins with the username **`juan`**.
   - Two exclamation points (**`!!`**) appear in the password field of the **`/etc/shadow`** file, which locks the account.

Note

If an encrypted password is passed using the **`-p`** flag, it is placed in the **`/etc/shadow`** file on the new line for the user.

- The password is set to never expire.

3. A new line for a group named **`juan`** is created in **`/etc/group`**. A group with the same name as a user is called a *user private group*. For more information on user private groups, refer to Section 37.1.1, "Adding a New User".

   The line created in **`/etc/group`** has the following characteristics:

   - It begins with the group name **`juan`**.
   - An **`x`** appears in the password field indicating that the system is using shadow group passwords.
   - The GID matches the one listed for user **`juan`** in **`/etc/passwd`**.

4. A new line for a group named **`juan`** is created in **`/etc/gshadow`**. The line has the following characteristics:

   - It begins with the group name **`juan`**.
   - An exclamation point (**`!`**) appears in the password field of the **`/etc/gshadow`** file, which locks the group.
   - All other fields are blank.

5. A directory for user **`juan`** is created in the **`/home/`** directory. This directory is owned by user **`juan`** and group **`juan`**. However, it has read, write, and execute privileges*only* for the user **`juan`**. All other permissions are denied.

6. The files within the **`/etc/skel/`** directory (which contain default user settings) are copied into the new **`/home/juan/`** directory.

At this point, a locked account called **`juan`** exists on the system. To activate it, the administrator must next assign a password to the account using the **`passwd`** command and, optionally, set password aging guidelines.

37.3. Standard Users

Table 37.4, "Standard Users" lists the standard users configured in the **`/etc/passwd`** file by an **`Everything`** installation. The groupid (GID) in this table is the *primary group* for the user. See Section 37.4, "Standard Groups" for a listing of standard groups.

Table 37.4. Standard Users

User	UID	GID	Home Directory	Shell
root	0	0	`/root`	`/bin/bash`
bin	1	1	`/bin`	`/sbin/nologin`
daemon	2	2	`/sbin`	`/sbin/nologin`
adm	3	4	`/var/adm`	`/sbin/nologin`

User	UID	GID	Home Directory	Shell
lp	4	7	`/var/spool/lpd`	`/sbin/nologin`
sync	5	0	`/sbin`	`/bin/sync`
shutdown	6	0	`/sbin`	`/sbin/shutdown`
halt	7	0	`/sbin`	`/sbin/halt`
mail	8	12	`/var/spool/mail`	`/sbin/nologin`
news	9	13	`/etc/news`	
uucp	10	14	`/var/spool/uucp`	`/sbin/nologin`
operator	11	0	`/root`	`/sbin/nologin`
games	12	100	`/usr/games`	`/sbin/nologin`
gopher	13	30	`/var/gopher`	`/sbin/nologin`
ftp	14	50	`/var/ftp`	`/sbin/nologin`
nobody	99	99	`/`	`/sbin/nologin`
rpm	37	37	`/var/lib/rpm`	`/sbin/nologin`
vcsa	69	69	`/dev`	`/sbin/nologin`
dbus	81	81	`/`	`/sbin/nologin`
ntp	38	38	`/etc/ntp`	`/sbin/nologin`
canna	39	39	`/var/lib/canna`	`/sbin/nologin`
nscd	28	28	`/`	`/sbin/nologin`
rpc	32	32	`/`	`/sbin/nologin`
postfix	89	89	`/var/spool/postfix`	`/sbin/nologin`
mailman	41	41	`/var/mailman`	`/sbin/nologin`
named	25	25	`/var/named`	`/bin/false`
amanda	33	6	`var/lib/amanda/`	`/bin/bash`
postgres	26	26	`/var/lib/pgsql`	`/bin/bash`
exim	93	93	`/var/spool/exim`	`/sbin/nologin`
sshd	74	74	`/var/empty/sshd`	`/sbin/nologin`
rpcuser	29	29	`/var/lib/nfs`	`/sbin/nologin`
nsfnobody	65534	65534	`/var/lib/nfs`	`/sbin/nologin`
pvm	24	24	`/usr/share/pvm3`	`/bin/bash`
apache	48	48	`/var/www`	`/sbin/nologin`
xfs	43	43	`/etc/X11/fs`	`/sbin/nologin`
gdm	42	42	`/var/gdm`	`/sbin/nologin`
htt	100	101	`/usr/lib/im`	`/sbin/nologin`
mysql	27	27	`/var/lib/mysql`	`/bin/bash`
webalizer	67	67	`/var/www/usage`	`/sbin/nologin`
mailnull	47	47	`/var/spool/mqueue`	`/sbin/nologin`
smmsp	51	51	`/var/spool/mqueue`	`/sbin/nologin`
squid	23	23	`/var/spool/squid`	`/sbin/nologin`
ldap	55	55	`/var/lib/ldap`	`/bin/false`
netdump	34	34	`/var/crash`	`/bin/bash`
pcap	77	77	`/var/arpwatch`	`/sbin/nologin`
radiusd	95	95	`/`	`/bin/false`
radvd	75	75	`/`	`/sbin/nologin`
quagga	92	92	`/var/run/quagga`	`/sbin/login`
wnn	49	49	`/var/lib/wnn`	`/sbin/nologin`

User	UID	GID	Home Directory	Shell
dovecot	97	97	`/usr/libexec/dovecot`	`/sbin/nologin`

37.4. Standard Groups

Table 37.5, “Standard Groups” lists the standard groups configured by an **`Everything`** installation. Groups are stored in the **`/etc/group`** file.

Table 37.5. Standard Groups

Group	GID	Members
root	0	root
bin	1	root, bin, daemon
daemon	2	root, bin, daemon
sys	3	root, bin, adm
adm	4	root, adm, daemon
tty	5	
disk	6	root
lp	7	daemon, lp
mem	8	
kmem	9	
wheel	10	root
mail	12	mail, postfix, exim
news	13	news
uucp	14	uucp
man	15	
games	20	
gopher	30	
dip	40	
ftp	50	
lock	54	
nobody	99	
users	100	
rpm	37	
utmp	22	
floppy	19	
vcsa	69	
dbus	81	
ntp	38	
canna	39	
nscd	28	
rpc	32	
postdrop	90	
postfix	89	
mailman	41	
exim	93	
named	25	
postgres	26	

Group	GID	Members
sshd	74	
rpcuser	29	
nfsnobody	65534	
pvm	24	
apache	48	
xfs	43	
gdm	42	
htt	101	
mysql	27	
webalizer	67	
mailnull	47	
smmsp	51	
squid	23	
ldap	55	
netdump	34	
pcap	77	
quaggavt	102	
quagga	92	
radvd	75	
slocate	21	
wnn	49	
dovecot	97	
radiusd	95	

37.5. User Private Groups

Red Hat Enterprise Linux uses a *user private group* (*UPG*) scheme, which makes UNIX groups easier to manage.

A UPG is created whenever a new user is added to the system. A UPG has the same name as the user for which it was created and that user is the only member of the UPG.

UPGs make it safe to set default permissions for a newly created file or directory, allowing both the user and *the group of that user* to make modifications to the file or directory.

The setting which determines what permissions are applied to a newly created file or directory is called a *umask* and is configured in the **`/etc/bashrc`** file. Traditionally on UNIX systems, the **`umask`** is set to **`022`**, which allows only the user who created the file or directory to make modifications. Under this scheme, all other users, *including members of the creator's group*, are not allowed to make any modifications. However, under the UPG scheme, this "group protection" is not necessary since every user has their own private group.

37.5.1. Group Directories

Many IT organizations like to create a group for each major project and then assign people to the group if they need to access that project's files. Using this traditional scheme, managing files has been difficult; when someone creates a file, it is associated with the primary group to which they belong. When a single person works on multiple projects, it is difficult to associate the right files with the right group. Using the UPG scheme, however, groups are automatically assigned to files created within a directory with the *setgid* bit set. The setgid bit makes managing group projects that share a common directory very simple because any files a user creates within the directory are owned by the group which owns the directory.

Let us say, for example, that a group of people need to work on files in the **`/usr/share/emacs/site-lisp/`** directory. Some people are trusted to modify the directory, but certainly not everyone is trusted. First create an **`emacs`** group, as in the following command:

```
groupadd emacs
```

To associate the contents of the directory with the **`emacs`** group, type:

```
chown -R root.emacs /usr/share/emacs/site-lisp
```

Now, it is possible to add the proper users to the group with the **`gpasswd`** command:

```
gpasswd -a <username> emacs
```

To allow users to create files within the directory, use the following command:

```
chmod 775 /usr/share/emacs/site-lisp
```

When a user creates a new file, it is assigned the group of the user's default private group. Next, set the setgid bit, which assigns everything created in the directory the same group permission as the directory itself (**`emacs`**). Use the following command:

```
chmod 2775 /usr/share/emacs/site-lisp
```

At this point, because the default umask of each user is 002, all members of the **`emacs`** group can create and edit files in the **`/usr/share/emacs/site-lisp/`** directory without the administrator having to change file permissions every time users write new files.

37.6. Shadow Passwords

In multiuser environments it is very important to use *shadow passwords* (provided by the **`shadow-utils`** package). Doing so enhances the security of system authentication files. For this reason, the installation program enables shadow passwords by default.

The following lists the advantages pf shadow passwords have over the traditional way of storing passwords on UNIX-based systems:

- Improves system security by moving encrypted password hashes from the world-readable **`/etc/passwd`** file to **`/etc/shadow`**, which is readable only by the root user.
- Stores information about password aging.
- Allows the use the **`/etc/login.defs`** file to enforce security policies.

Most utilities provided by the **`shadow-utils`** package work properly whether or not shadow passwords are enabled. However, since password aging information is stored exclusively in the **`/etc/shadow`** file, any commands which create or modify password aging information do not work.

The following is a list of commands which do not work without first enabling shadow passwords:

- **`chage`**
- **`gpasswd`**

- **`/usr/sbin/usermod -e`** or **`-f`** options
- **`/usr/sbin/useradd -e`** or **`-f`** options

37.7. Additional Resources

For more information about users and groups, and tools to manage them, refer to the following resources.

37.7.1. Installed Documentation

- Related man pages — There are a number of man pages for the various applications and configuration files involved with managing users and groups. Some of the more important man pages have been listed here:

  User and Group Administrative Applications

  - **`man chage`** — A command to modify password aging policies and account expiration.
  - **`man gpasswd`** — A command to administer the **`/etc/group`** file.
  - **`man groupadd`** — A command to add groups.
  - **`man grpck`** — A command to verify the **`/etc/group`** file.
  - **`man groupdel`** — A command to remove groups.
  - **`man groupmod`** — A command to modify group membership.
  - **`man pwck`** — A command to verify the **`/etc/passwd`** and **`/etc/shadow`** files.
  - **`man pwconv`** — A tool to convert standard passwords to shadow passwords.
  - **`man pwunconv`** — A tool to convert shadow passwords to standard passwords.
  - **`man useradd`** — A command to add users.
  - **`man userdel`** — A command to remove users.
  - **`man usermod`** — A command to modify users.

  Configuration Files

  - **`man 5 group`** — The file containing group information for the system.
  - **`man 5 passwd`** — The file containing user information for the system.
  - **`man 5 shadow`** — The file containing passwords and account expiration information for the system.

Chapter 38. Printer Configuration

Printer Configuration Tool allows users to configure a printer. This tool helps maintain the printer configuration file, print spool directories, print filters, and printer classes.

Red Hat Enterprise Linux 5.10 uses the Common Unix Printing System (CUPS). If a system was upgraded from a previous Red Hat Enterprise Linux version that used CUPS, the upgrade process preserves the configured queues.

Important

The **`cupsd.conf`** man page documents configuration of a CUPS server. It includes directives for enabling **SSL** support. However, CUPS does not allow control of the protocol versions used. Due to the vulnerability described in *Resolution for POODLE SSLv3.0 vulnerability (CVE-2014-3566) for components that do not allow SSLv3 to be disabled via configuration settings*, Red Hat recommends that you do not rely on this for security. It is recommend that you use **stunnel** to provide a secure tunnel and disable **SSLv3**.

For ad-hoc secure connections to a remote system's **Print Settings** tool, use X11 forwarding over **SSH** as described in Section 20.7.1, “X11 Forwarding”.

Using **Printer Configuration Tool** requires root privileges. To start the application, select System (on the panel) > **Administration** > **Printing**, or type the command **`system-config-printer`** at a shell prompt.

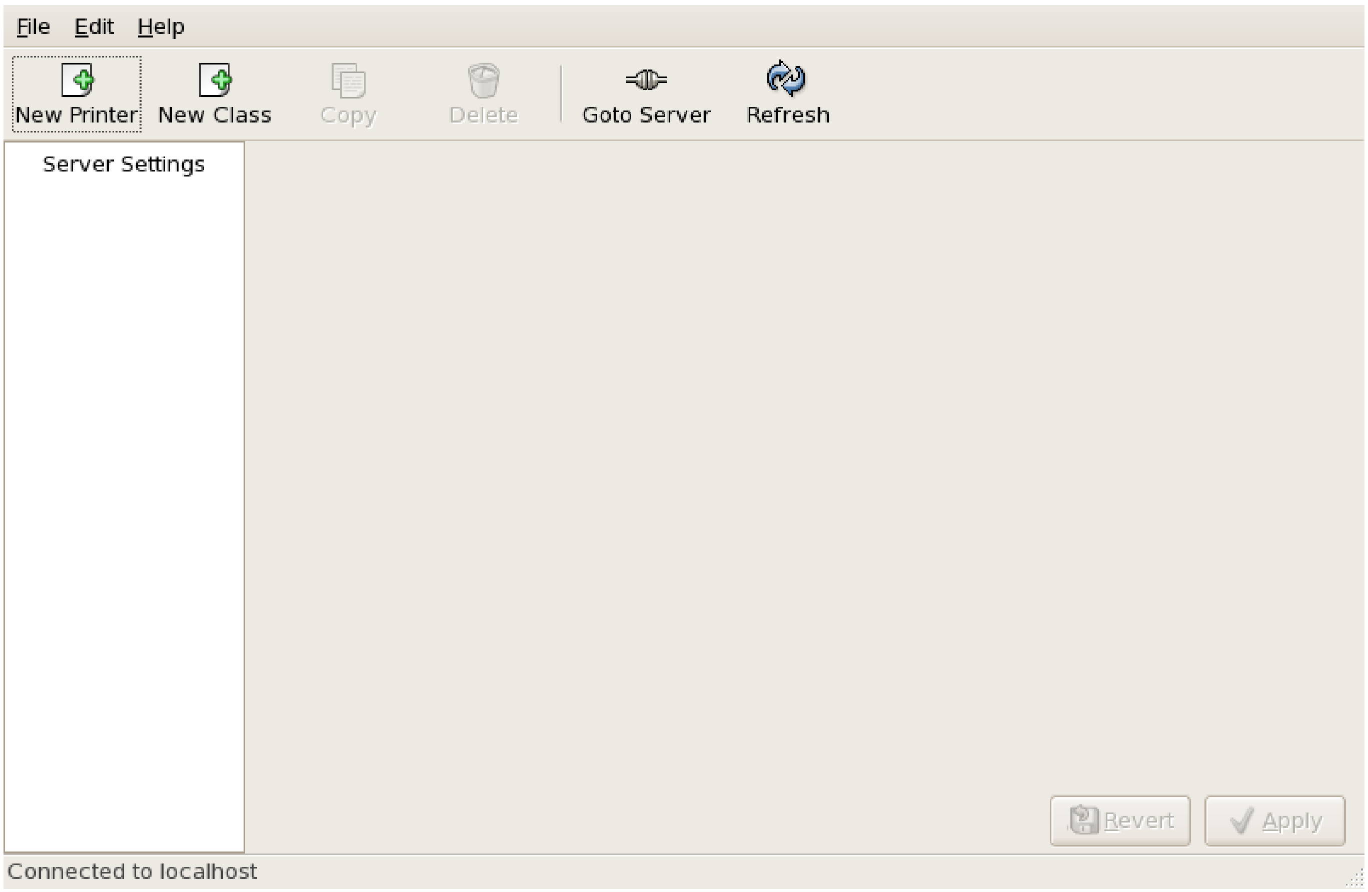


Figure 38.1. Printer Configuration Tool

The following types of print queues can be configured:

- **AppSocket/HP JetDirect** — a printer connected directly to the network through HP JetDirect or Appsocket interface instead of a computer.
- **Internet Printing Protocol (IPP)** — a printer that can be accessed over a TCP/IP network via the Internet Printing Protocol (for example, a printer attached to another Red Hat Enterprise Linux system running CUPS on the network).
- **LPD/LPR Host or Printer** — a printer attached to a different UNIX system that can be accessed over a TCP/IP network (for example, a printer attached to another Red Hat Enterprise Linux system running LPD on the network).
- **Networked Windows (SMB)** — a printer attached to a different system which is sharing a printer over an SMB network (for example, a printer attached to a Microsoft Windows™ machine).
- **Networked JetDirect** — a printer connected directly to the network through HP JetDirect instead of a computer.

Important

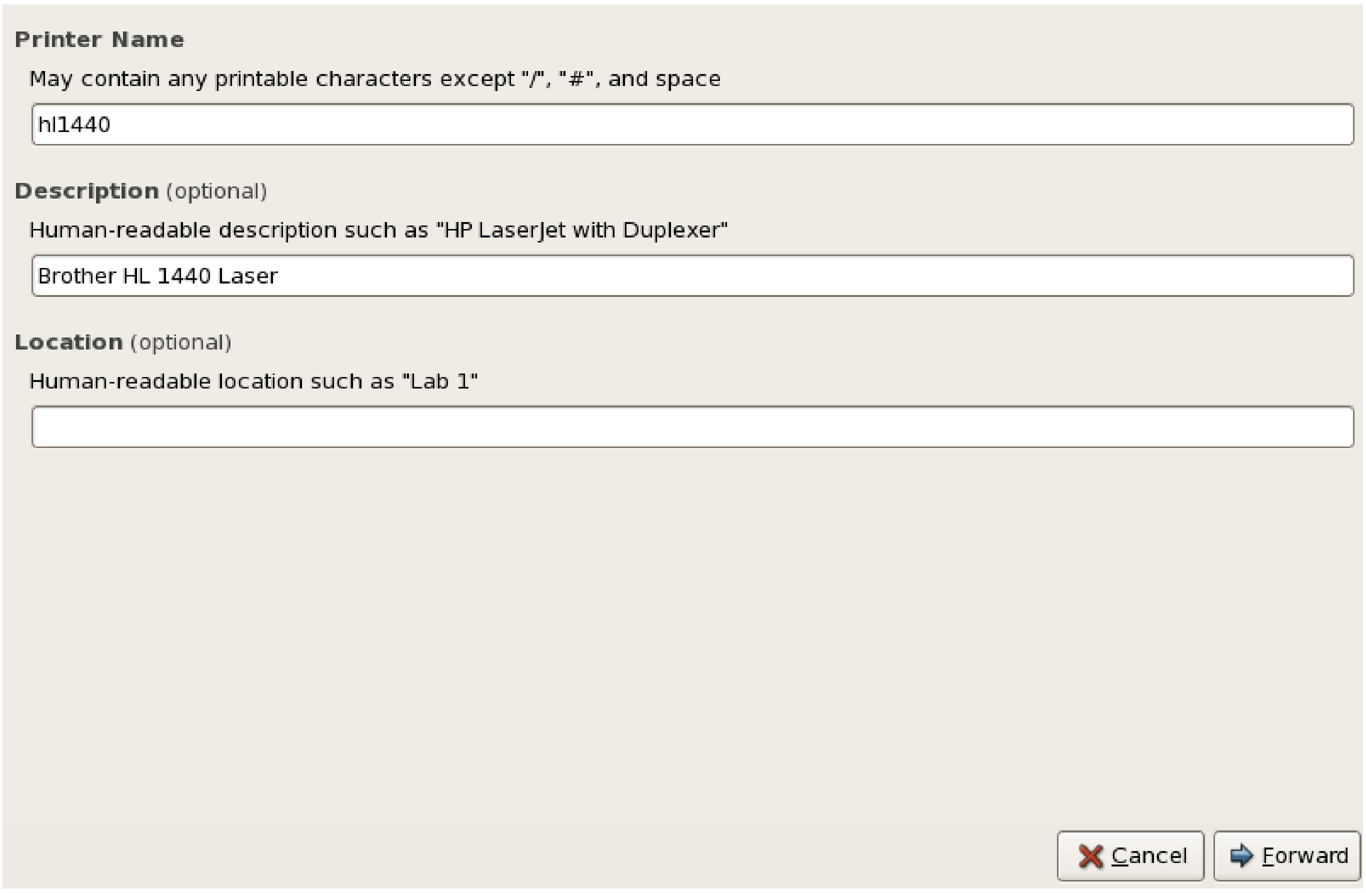
If you add a new print queue or modify an existing one, you must apply the changes for them to take effect.

Clicking the **`Apply`** button prompts the printer daemon to restart with the changes you have configured.

Clicking the **`Revert`** button discards unapplied changes.

38.1. Adding a Local Printer

To add a local printer, such as one attached through a parallel port or USB port on your computer, click the **`New Printer`** button in the main **Printer Configuration Tool** window to display the window in Figure 38.2, “Adding a Printer”.


Printer Name

May contain any printable characters except "/", "#", and space

hl1440

Description (optional)

Human-readable description such as "HP LaserJet with Duplexer"

Brother HL 1440 Laser

Location (optional)

Human-readable location such as "Lab 1"

Figure 38.2. Adding a Printer

Click **Forward** to proceed.

Enter a unique name for the printer in the **Printer Name** field. The printer name can contain letters, numbers, dashes (-), and underscores (_); it *must not* contain any spaces.

You can also use the **Description** and **Location** fields to further distinguish this printer from others that may be configured on your system. Both of these fields are optional, and may contain spaces.

Click **Forward** to open the **New Printer** dialogue (refer to Figure 38.3, “Adding a Local Printer”). If the printer has been automatically detected, the printer model appears in **Select Connection**. Select the printer model and click **Forward** to continue.

If the device does not automatically appear, select the device to which the printer is connected (such as **LPT #1** or **Serial Port #1**) in **Select Connection**.

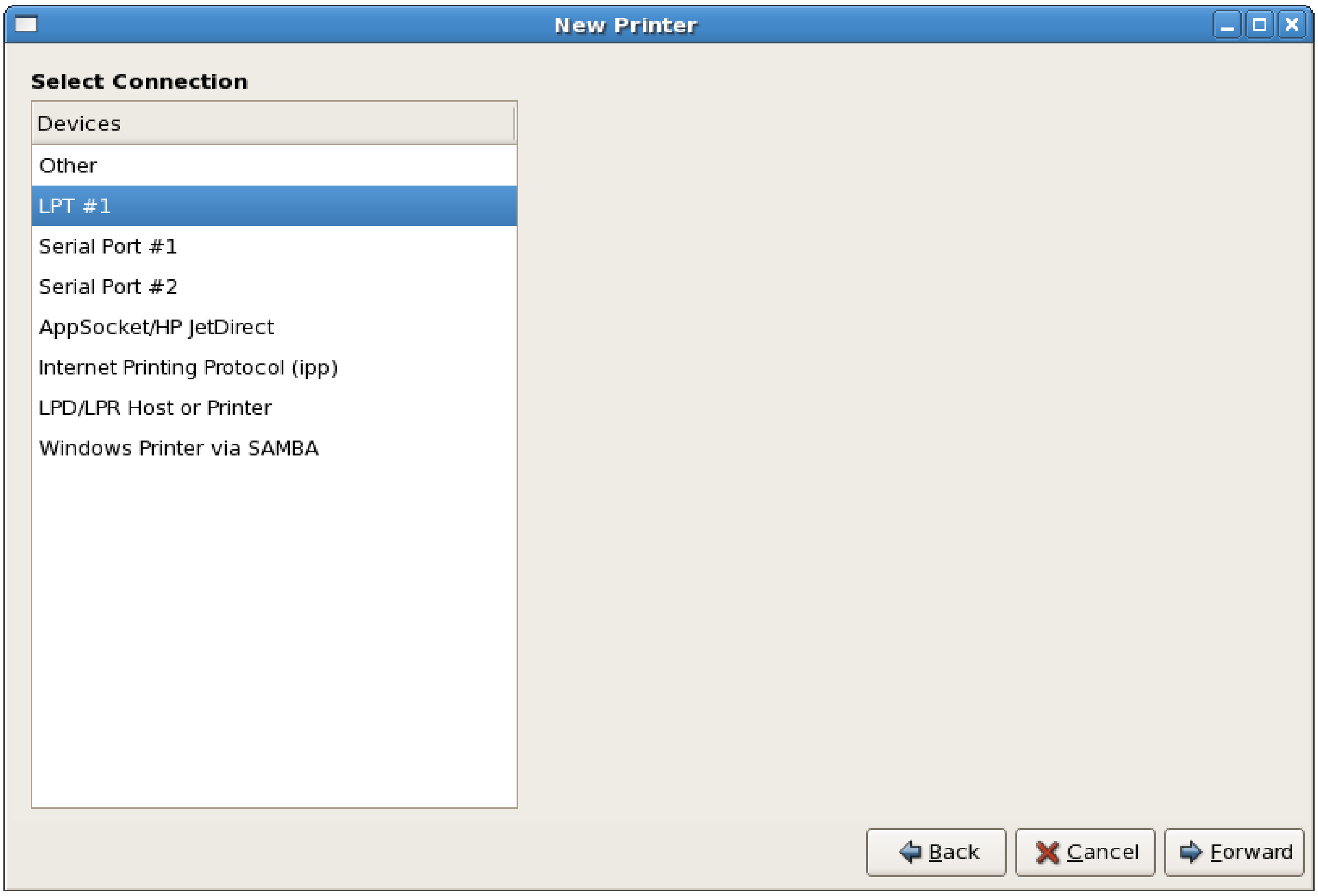


Figure 38.3. Adding a Local Printer

Next, select the printer type. Refer to Section 38.5, “Selecting the Printer Model and Finishing” for details.

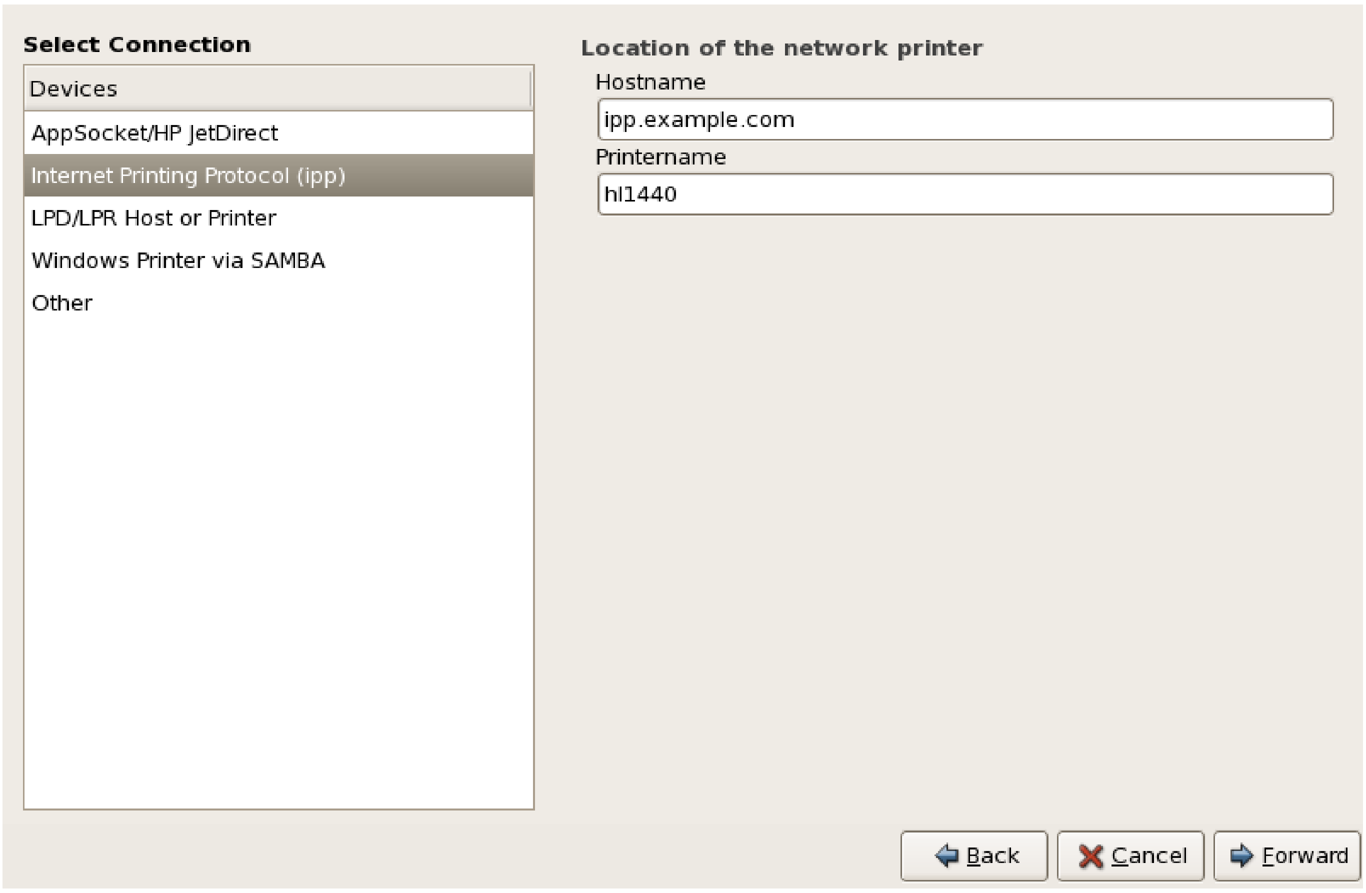
38.2. Adding an IPP Printer

An IPP printer is a printer attached to a different system on the same TCP/IP network. The system this printer is attached to may either be running CUPS or simply configured to use IPP.

If a firewall is enabled on the printer server, then the firewall should be configured to allow send / receive connections on the incoming UDP port 631. If a firewall is enabled on the client (the system sending the print request) then the firewall should be configured to allow accept and create connections through port 631.

You can add a networked IPP printer by clicking the **New Printer** button in the main **Printer Configuration Tool** window to display the window in Figure 38.2, “Adding a Printer”. Enter the **Printer Name** (printer names cannot contain spaces and may contain letters, numbers, dashes (-), and underscores (_)), **Description**, and **Location** to distinguish this printer from others that you may configure on your system. Click **Forward** to proceed.

In the window shown in Figure 38.4, “Adding an IPP Printer”, enter the hostname of the IPP printer in the **Hostname** field as well as a unique name for the printer in the **Printername** field.



Location of the network printer

Hostname: ipp.example.com

Printername: hl1440

Figure 38.4. Adding an IPP Printer

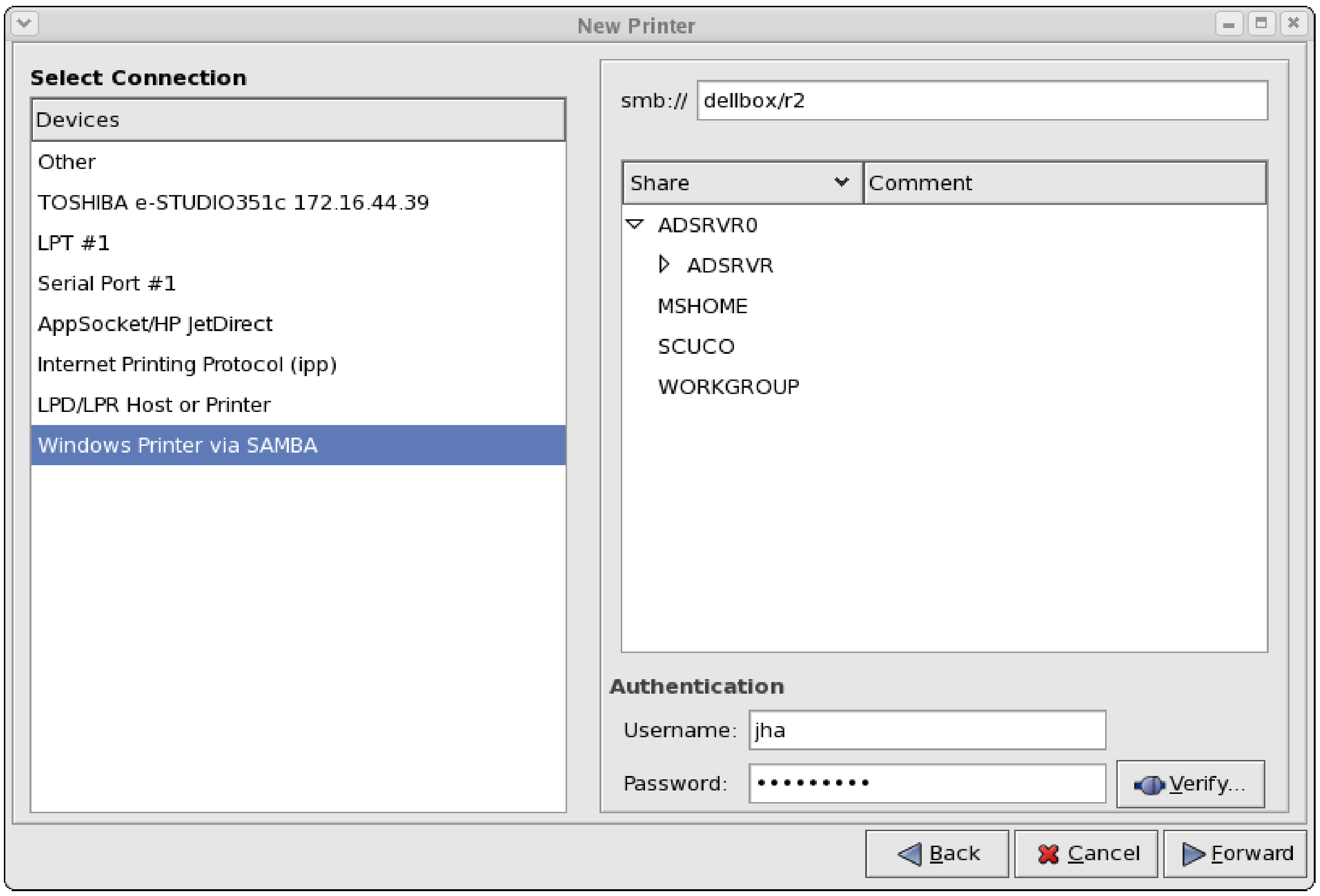
Click **`Forward`** to continue.

Next, select the printer type. Refer to Section 38.5, “Selecting the Printer Model and Finishing” for details.

38.3. Adding a Samba (SMB) Printer

You can add a Samba (SMB) based printer share by clicking the **`New Printer`** button in the main **Printer Configuration Tool** window to display the window in Figure 38.2, “Adding a Printer”. Enter a unique name for the printer in the **`Printer Name`** field. The printer name can contain letters, numbers, dashes (-), and underscores (_); it *must not* contain any spaces.

You can also use the **`Description`** and **`Location`** fields to further distinguish this printer from others that may be configured on your system. Both of these fields are optional, and may contain spaces.



smb:// dellbox/r2

Username: jha

Password: •••••••••

Figure 38.5. Adding a SMB Printer

As shown in Figure 38.5, "Adding a SMB Printer", available SMB shares are automatically detected and listed in the **`Share`** column. Click the arrow (▷) beside a Workgroup to expand it. From the expanded list, select a printer.

If the printer you are looking for does not appear in the list, enter the SMB address in the **`smb://`** field. Use the format *computer name/printer share*. In Figure 38.5, "Adding a SMB Printer", the *computer name* is **`dellbox`**, while the *printer share* is **`r2`**.

In the **`Username`** field, enter the username to access the printer. This user must exist on the SMB system, and the user must have permission to access the printer. The default user name is typically **`guest`** for Windows servers, or **nobody** for Samba servers.

Enter the **`Password`** (if required) for the user specified in the **`Username`** field.

You can then test the connection by clicking **`Verify`**. Upon successful verification, a dialog box appears confirming printer share accessibility.

Next, select the printer type. Refer to Section 38.5, "Selecting the Printer Model and Finishing" for details.

Warning

Samba printer usernames and passwords are stored in the printer server as unencrypted files readable by root and lpd. Thus, other users that have root access to the printer server can view the username and password you use to access the Samba printer.

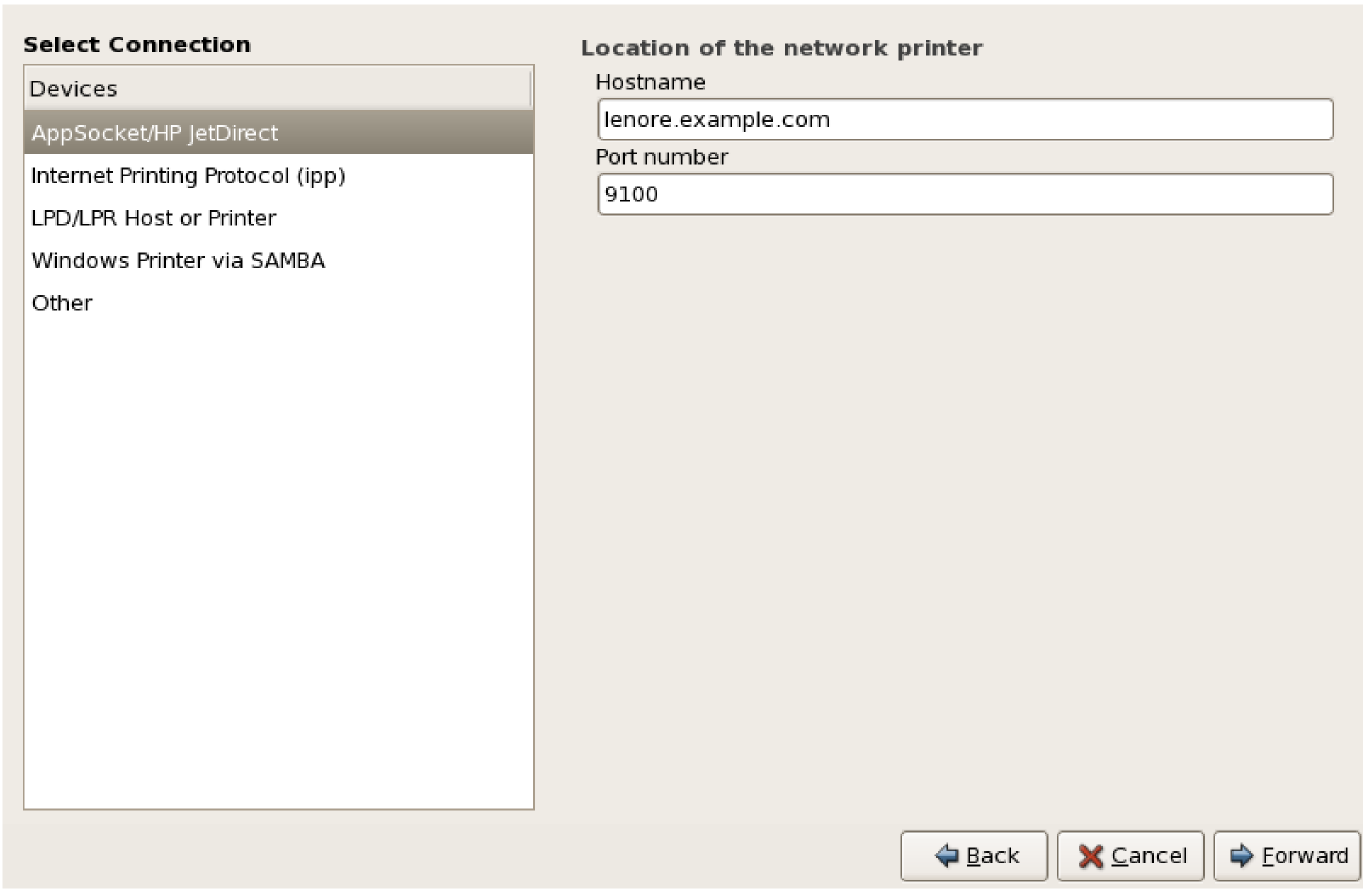
As such, when you choose a username and password to access a Samba printer, it is advisable that you choose a password that is different from what you use to access your local Red Hat Enterprise Linux system.

If there are files shared on the Samba print server, it is recommended that they also use a password different from what is used by the print queue.

38.4. Adding a JetDirect Printer

To add a JetDirect or AppSocket connected printer share, click the **`New Printer`** button in the main **Printer Configuration Tool** window to display the window in Figure 38.2, “Adding a Printer”. Enter a unique name for the printer in the **`Printer Name`** field. The printer name can contain letters, numbers, dashes (-), and underscores (_); it *must not* contain any spaces.

You can also use the **`Description`** and **`Location`** fields to further distinguish this printer from others that may be configured on your system. Both of these fields are optional, and may contain spaces.



Location of the network printer

Hostname: lenore.example.com

Port number: 9100

Figure 38.6. Adding a JetDirect Printer

Click **`Forward`** to continue.

Text fields for the following options appear:

- **`Hostname`** — The hostname or IP address of the JetDirect printer.
- **`Port Number`** — The port on the JetDirect printer that is listening for print jobs. The default port is 9100.

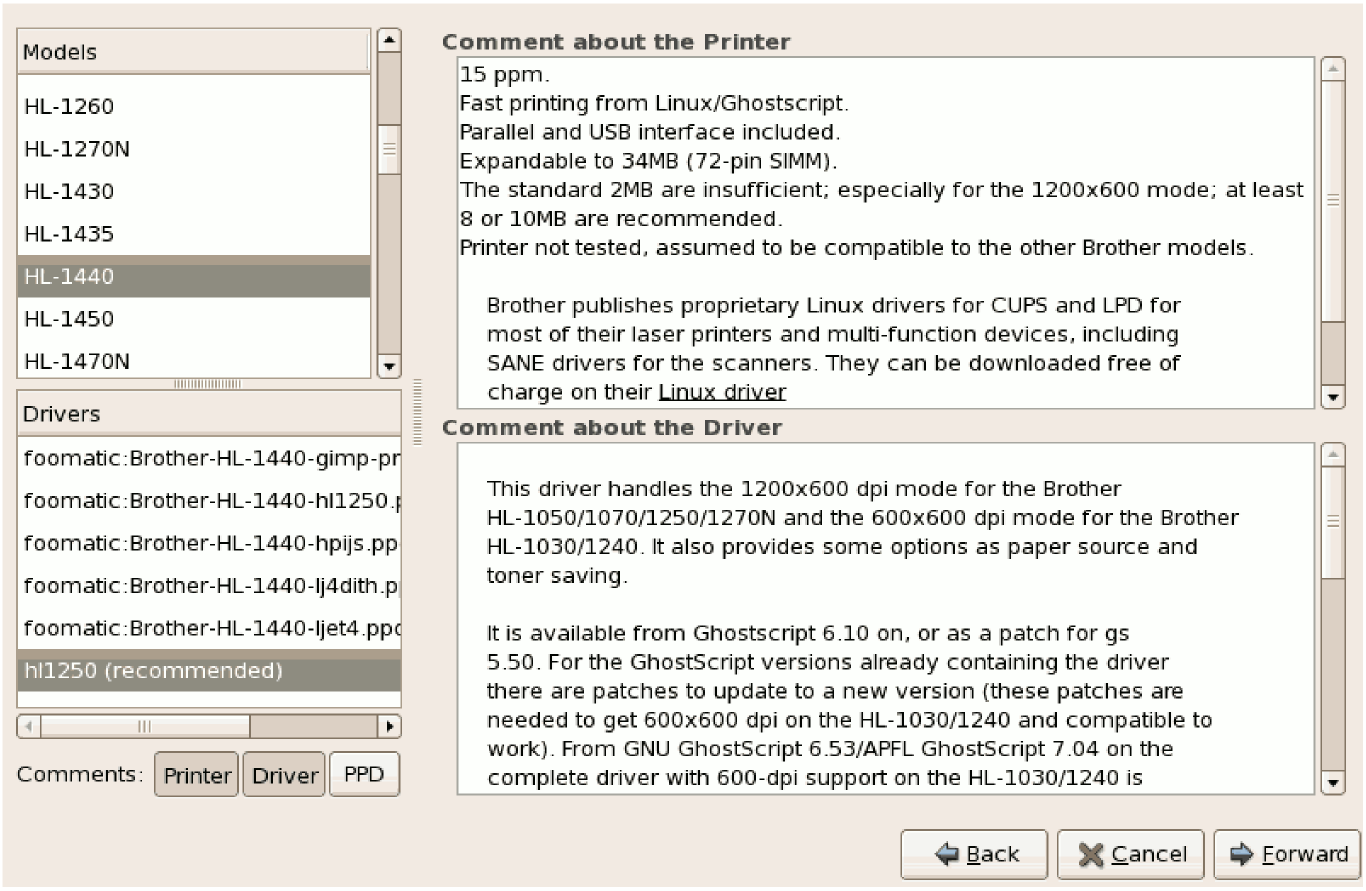
Next, select the printer type. Refer to Section 38.5, "Selecting the Printer Model and Finishing" for details.

38.5. Selecting the Printer Model and Finishing

Once you have properly selected a printer queue type, you can choose either option:

- Select a Printer from database - If you select this option, choose the make of your printer from the list of **`Makes`**. If your printer make is not listed, choose **`Generic`**.
- Provide PPD file - A PostScript Printer Description (PPD) file may also be provided with your printer. This file is normally provided by the manufacturer. If you are provided with a PPD file, you can choose this option and use the browser bar below the option description to select the PPD file.

Refer to Figure 38.7, "Selecting a Printer Model".


Models
HL-1260
HL-1270N
HL-1430
HL-1435
HL-1440
HL-1450
HL-1470N
Drivers
foomatic:Brother-HL-1440-gimp-pr
foomatic:Brother-HL-1440-hl1250.p
foomatic:Brother-HL-1440-hpijs.pp
foomatic:Brother-HL-1440-lj4dith.p
foomatic:Brother-HL-1440-ljet4.ppd
hl1250 (recommended)
Comments: Printer Driver PPD
Comment about the Printer
Comment about the Driver
Back
Cancel
Forward

15 ppm.
Fast printing from Linux/Ghostscript.
Parallel and USB interface included.
Expandable to 34MB (72-pin SIMM).
The standard 2MB are insufficient; especially for the 1200x600 mode; at least 8 or 10MB are recommended.
Printer not tested, assumed to be compatible to the other Brother models.

Brother publishes proprietary Linux drivers for CUPS and LPD for most of their laser printers and multi-function devices, including SANE drivers for the scanners. They can be downloaded free of charge on their Linux driver

This driver handles the 1200x600 dpi mode for the Brother HL-1050/1070/1250/1270N and the 600x600 dpi mode for the Brother HL-1030/1240. It also provides some options as paper source and toner saving.

It is available from Ghostscript 6.10 on, or as a patch for gs 5.50. For the GhostScript versions already containing the driver there are patches to update to a new version (these patches are needed to get 600x600 dpi on the HL-1030/1240 and compatible to work). From GNU GhostScript 6.53/APFL GhostScript 7.04 on the complete driver with 600-dpi support on the HL-1030/1240 is

Figure 38.7. Selecting a Printer Model

After choosing an option, click **`Forward`** to continue. Figure 38.7, "Selecting a Printer Model" appears. You now have to choose the corresponding model and driver for the printer.

The recommended printed driver is automatically selected based on the printer model you chose. The print driver processes the data that you want to print into a format the printer can understand. Since a local printer is attached directly to your computer, you need a printer driver to process the data that is sent to the printer.

If you have a PPD file for the device (usually provided by the manufacturer), you can select it by choosing

`Provide PPD file`. You can then browse the filesystem for the PPD file by clicking **`Browse`**.

38.5.1. Confirming Printer Configuration

The last step is to confirm your printer configuration. Click **`Apply`** to add the print queue if the settings are correct. Click **`Back`** to modify the printer configuration.

After applying the changes, print a test page to ensure the configuration is correct. Refer to Section 38.6, “Printing a Test Page” for details.

38.6. Printing a Test Page

After you have configured your printer, you should print a test page to make sure the printer is functioning properly. To print a test page, select the printer that you want to try out from the printer list, then click **`Print Test Page`** from the printer's **`Settings`** tab.

If you change the print driver or modify the driver options, you should print a test page to test the different configuration.

38.7. Modifying Existing Printers

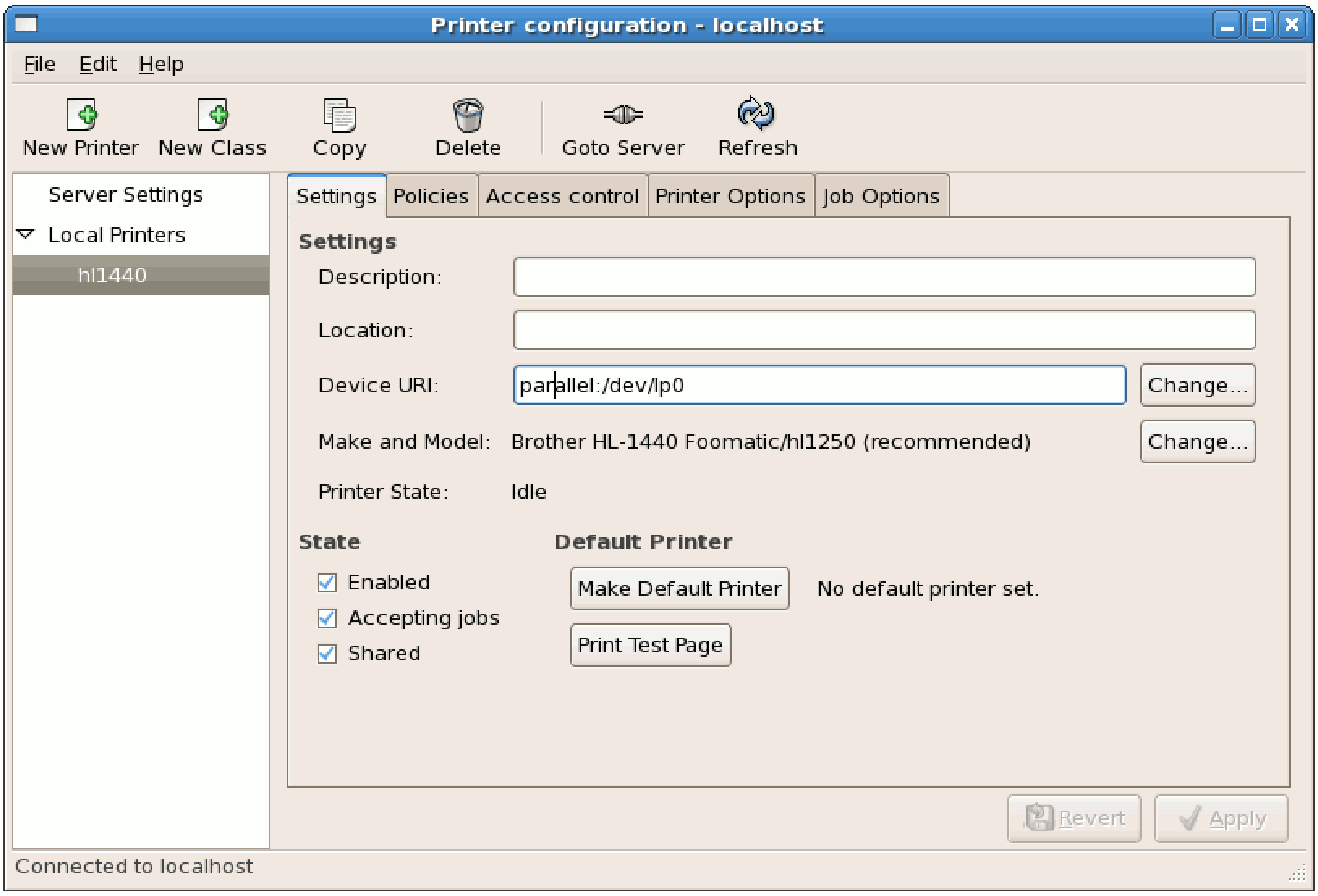
To delete an existing printer, select the printer and click the **`Delete`** button on the toolbar. The printer is removed from the printer list once you confirm deletion of the printer configuration.

To set the default printer, select the printer from the printer list and click the **`Make Default Printer`** button in the **`Settings`** tab.

38.7.1. The `Settings` Tab

To change printer driver configuration, click the corresponding name in the **`Printer`** list and click the **`Settings`** tab.

You can modify printer settings such as make and model, make a printer the default, print a test page, change the device location (URI), and more.



Settings

Description:

Location:

Device URI: parallel:/dev/lp0 Change...

Make and Model: Brother HL-1440 Foomatic/hl1250 (recommended) Change...

Printer State: Idle

State

- ✓ Enabled
- ✓ Accepting jobs
- ✓ Shared

Default Printer

Make Default Printer No default printer set.

Print Test Page

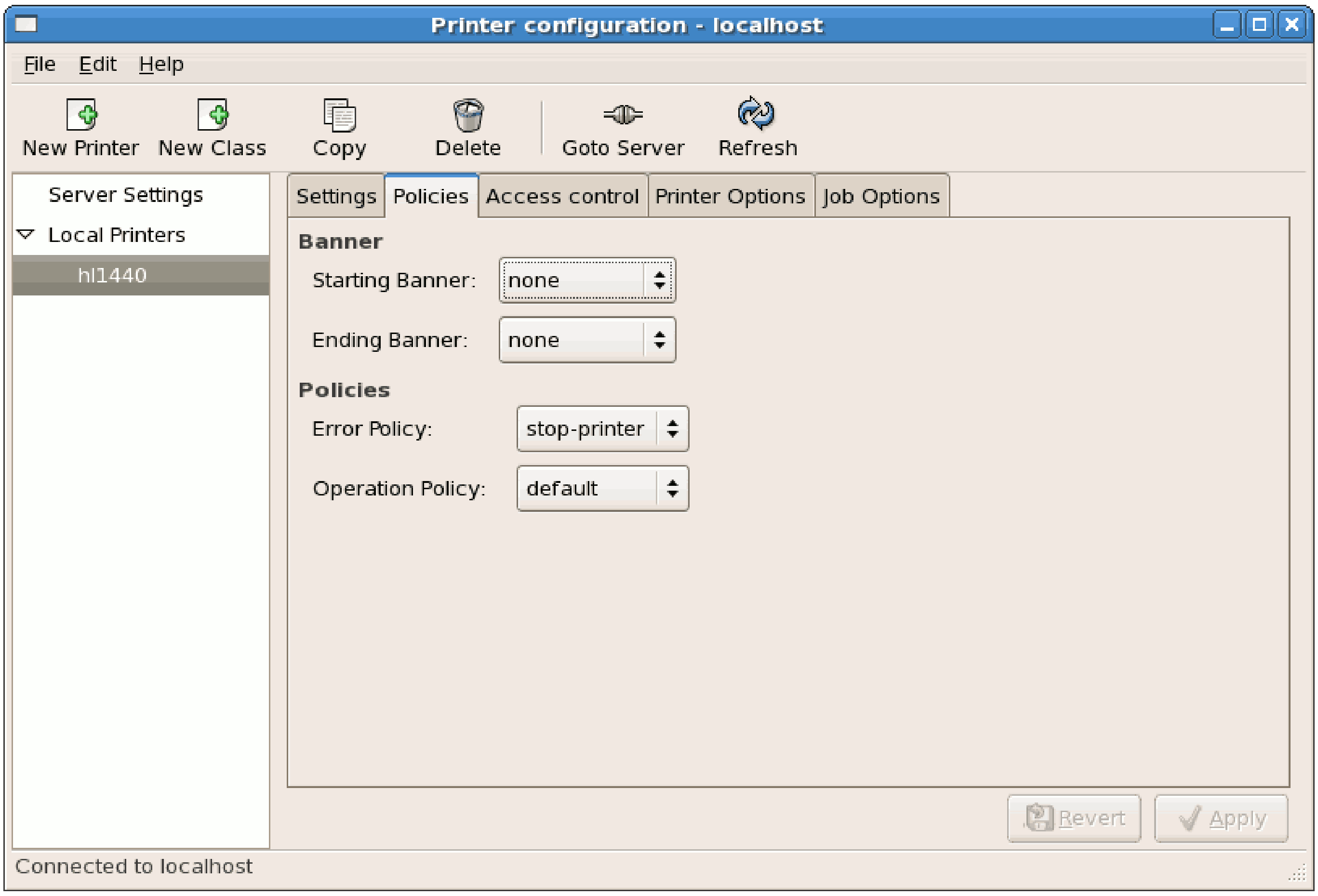
Connected to localhost

Figure 38.8. Settings Tab

38.7.2. The `Policies` Tab

To change settings in print output, click the **`Policies`** tab.

For example, to create a *banner page* (a page that describes aspects of the print job such as the originating printer, the username from the which the job originated, and the security status of the document being printed click the **`Starting Banner`** or **`Ending Banner`** drop-menu and choose the option that best describes the nature of the print jobs (such as **`topsecret`**, **`classified`**, or **`confidential`**).



Banner

Starting Banner: none

Ending Banner: none

Policies

Error Policy: stop-printer

Operation Policy: default

Connected to localhost

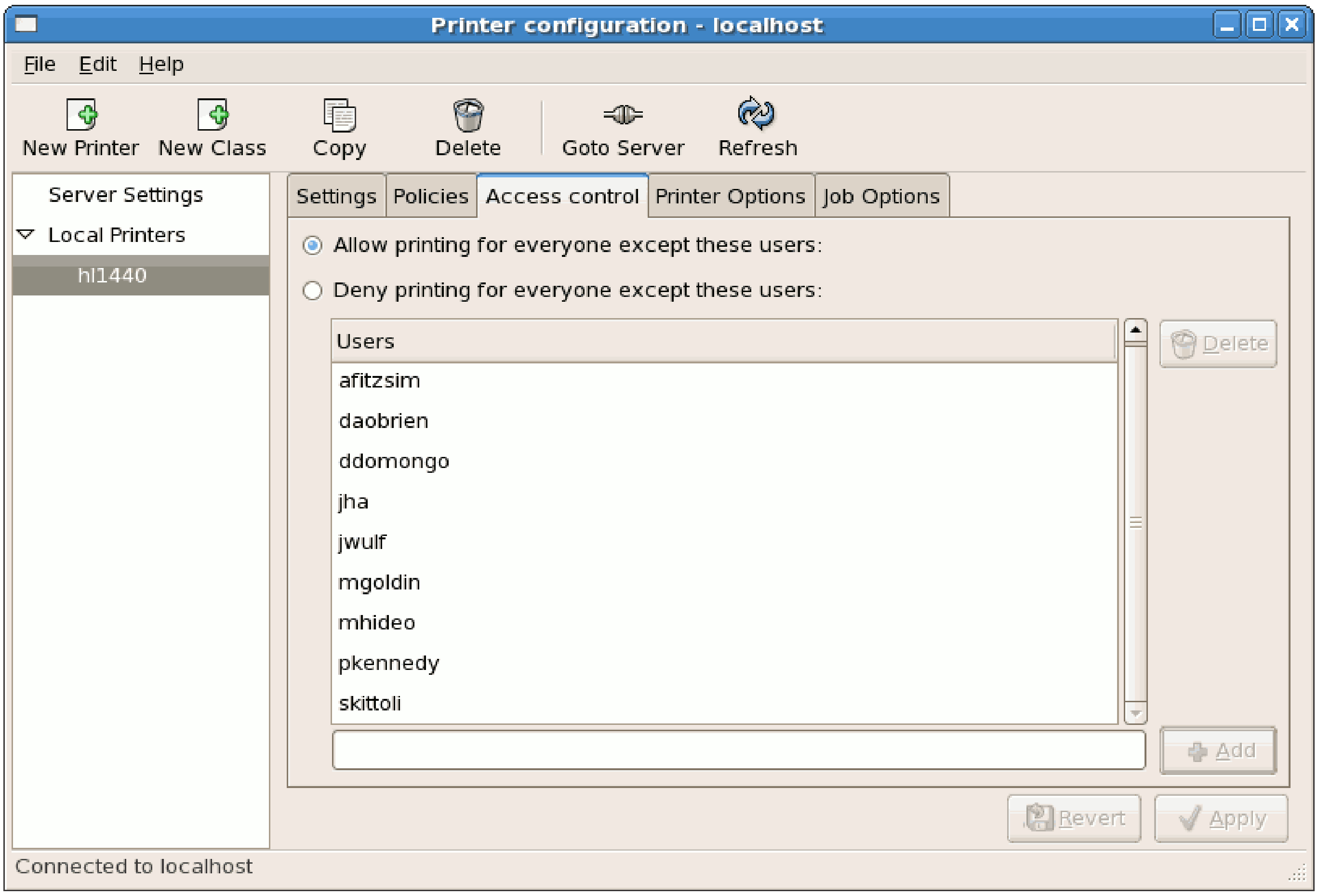
Figure 38.9. Policies Tab

You can also configure the **`Error Policy`** of the printer, by choosing an option from the drop-down menu. You can choose to abort the print job, retry, or stop it.

38.7.3. The `Access Control` Tab

You can change user-level access to the configured printer by clicking the **`Access Control`** tab.

Add users using the text box and click the **Add** button beside it. You can then choose to only allow use of the printer to that subset of users or deny use to those users.



Allow printing for everyone except these users: [✓]

Deny printing for everyone except these users:

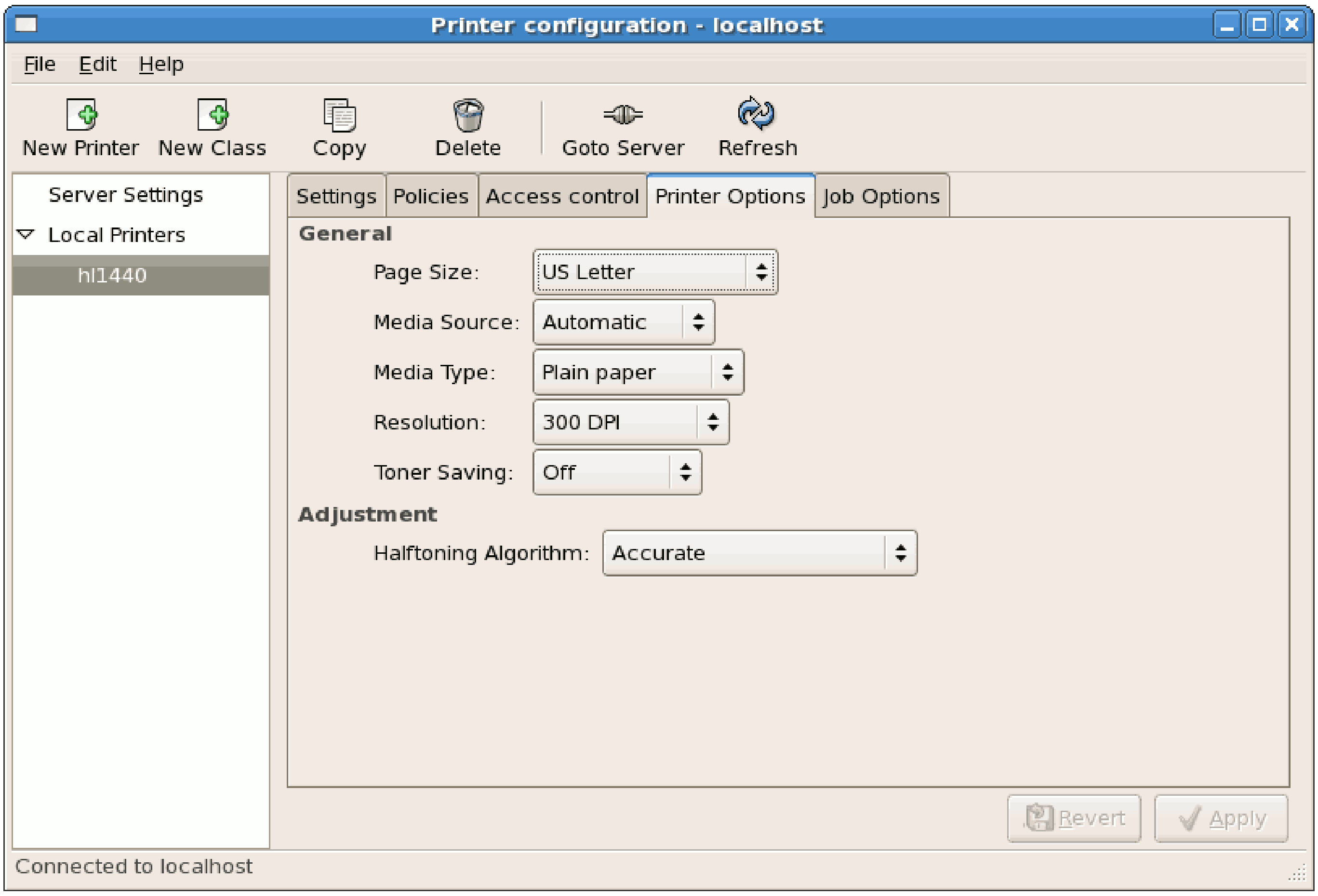
Users
afitzsim
daobrien
ddomongo
jha
jwulf
mgoldin
mhideo
pkennedy
skittoli

Connected to localhost

Figure 38.10. Access Control Tab

38.7.4. The `Printer` and `Job Options`Tab

The **`Printer Options`** tab contains various configuration options for the printer media and output.



General

Page Size: US Letter

Media Source: Automatic

Media Type: Plain paper

Resolution: 300 DPI

Toner Saving: Off

Adjustment

Halftoning Algorithm: Accurate

Figure 38.11. Printer Options Tab

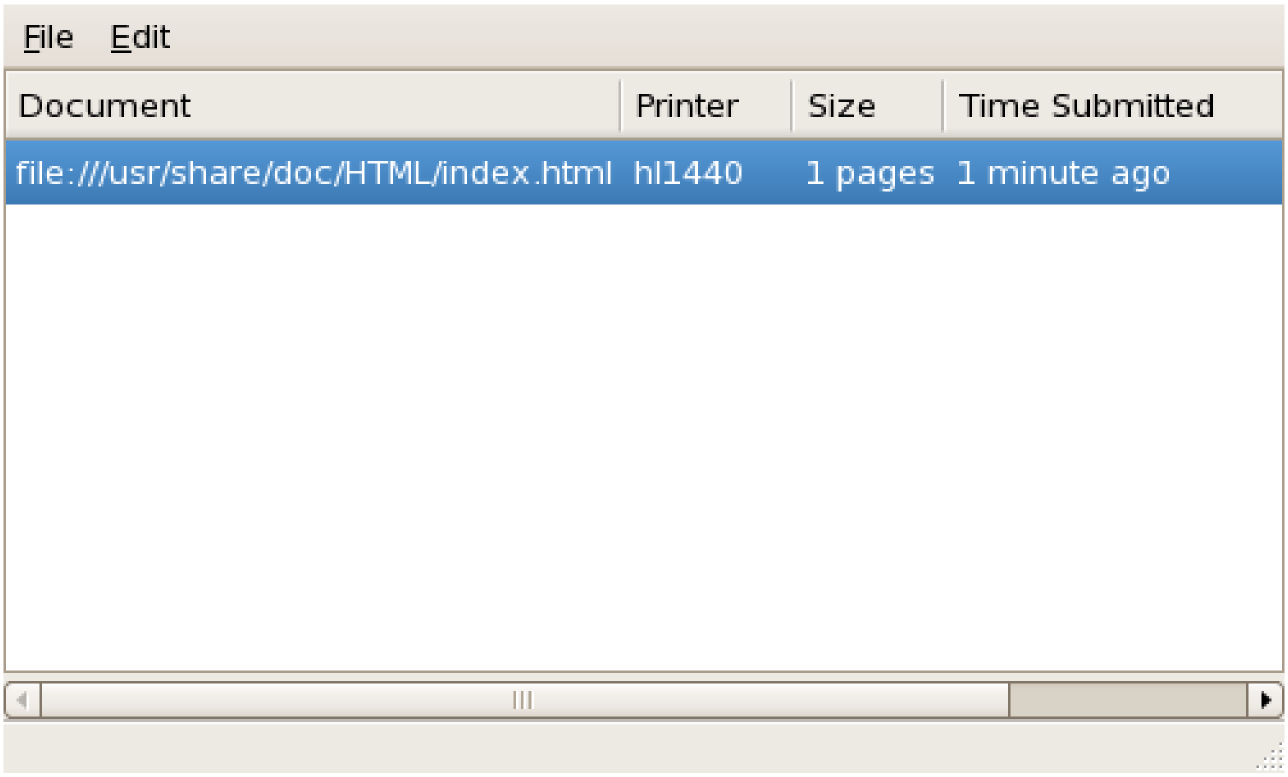
- **`Page Size`** — Allows the paper size to be selected. The options include US Letter, US Legal, A3, and A4
- **`Media Source`** — set to **`Automatic`** by default. Change this option to use paper from a different tray.
- **`Media Type`** — Allows you to change paper type. Options include: Plain, thick, bond, and transparency.
- **`Resolution`** — Configure the quality and detail of the printout. Default is 300 dots per inch (dpi).
- **`Toner Saving`** — Choose whether the printer uses less toner to conserve resources.

You can also configure printer job options using the **`Job Options`** tab. Use the drop-menu and choose the job options you wish to use, such as **`Landscape`** modes (horizontal or vertical printout), **`copies`**, or **`scaling`** (increase or decrease the size of the printable area, which can be used to fit an oversize print area onto a smaller physical sheet of print medium).

38.8. Managing Print Jobs

When you send a print job to the printer daemon, such as printing a text file from **Emacs** or printing an image from **The GIMP**, the print job is added to the print spool queue. The print spool queue is a list of print jobs that have been sent to the printer and information about each print request, such as the status of the request, the job number, and more.

During the printing process, the Printer Status icon appears in the **Notification Area** on the panel. To check the status of a print job, double click the Printer Status, which displays a window similar to Figure 38.12, “GNOME Print Status”.



Document	Printer	Size	Time Submitted
file:///usr/share/doc/HTML/index.html	hl1440	1 pages	1 minute ago

Figure 38.12. GNOME Print Status

To cancel a specific print job listed in the **GNOME Print Status**, select it from the list and select **Edit** > **Cancel Documents** from the pulldown menu.

To view the list of print jobs in the print spool from a shell prompt, type the command **`lpq`**. The last few lines look similar to the following:

Example 38.1. Example of `lpq` output

```
Rank   Owner/ID                   Class  Job Files        Size Time
active user@localhost+902          A      902 sample.txt  2050 01:20:46
```

If you want to cancel a print job, find the job number of the request with the command **`lpq`** and then use the command **`lprm`** ***`job number`***. For example, **`lprm 902`** would cancel the print job in Example 38.1, “Example of `lpq` output”. You must have proper permissions to cancel a print job. You can not cancel print jobs that were started by other users unless you are logged in as root on the machine to which the printer is attached.

You can also print a file directly from a shell prompt. For example, the command **`lpr sample.txt`** prints the text file **`sample.txt`**. The print filter determines what type of file it is and converts it into a format the printer can understand.

38.9. Additional Resources

To learn more about printing on Red Hat Enterprise Linux, refer to the following resources.

38.9.1. Installed Documentation

- **`map lpr`** — The manual page for the **`lpr`** command that allows you to print files from the command line.
- **`man lprm`** — The manual page for the command line utility to remove print jobs from the print queue.
- **`man mpage`** — The manual page for the command line utility to print multiple pages on one sheet of paper.
- **`man cupsd`** — The manual page for the CUPS printer daemon.
- **`man cupsd.conf`** — The manual page for the CUPS printer daemon configuration file.
- **`man classes.conf`** — The manual page for the class configuration file for CUPS.

38.9.2. Useful Websites

- http://www.linuxprinting.org — *GNU/Linux Printing* contains a large amount of information about printing in Linux.
- http://www.cups.org/ — Documentation, FAQs, and newsgroups about CUPS.

Chapter 39. Automated Tasks

In Linux, tasks can be configured to run automatically within a specified period of time, on a specified date, or when the system load average is below a specified number. Red Hat Enterprise Linux is pre-configured to run important system tasks to keep the system updated. For example, the slocate database used by the **`locate`** command is updated daily. A system administrator can use automated tasks to perform periodic backups, monitor the system, run custom scripts, and more.

Red Hat Enterprise Linux comes with several automated tasks utilities: **`cron`**, **`at`**, and **`batch`**.

39.1. Cron

Cron is a daemon that can be used to schedule the execution of recurring jobs according to a combination of the time, day of the month, month, day of the week, and week.

Cron assumes that the system is on continuously. If the system is not on when a job is scheduled, it is not executed. To schedule one-time jobs, refer to Section 39.2, "At and Batch".

To use the cron service, the **`vixie-cron`** RPM package must be installed and the **`crond`** service must be running. To determine if the package is installed, use the **`rpm -q vixie-cron`** command. To determine if the service is running, use the command **`/sbin/service crond status`**.

39.1.1. Configuring Cron Jobs

The main configuration file for cron, **`/etc/crontab`**, contains the following lines:

```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/

# run-parts
01 * * * * root run-parts /etc/cron.hourly
02 4 * * * root run-parts /etc/cron.daily
22 4 * * 0 root run-parts /etc/cron.weekly
42 4 1 * * root run-parts /etc/cron.monthly
```

The first four lines are variables used to configure the environment in which the cron jobs are run. The **`SHELL`** variable tells the system which shell environment to use (in this example the bash shell), while the **`PATH`** variable defines the path used to execute commands. The output of the cron jobs are emailed to the username defined with the **`MAILTO`** variable. If the **`MAILTO`** variable is defined as an empty string (**`MAILTO=""`**), email is not sent. The **`HOME`** variable can be used to set the home directory to use when executing commands or scripts.

Each line in the **`/etc/crontab`** file represents a job and has the following format:

```
minute   hour   day   month   dayofweek   command
```

- **`minute`** — any integer from 0 to 59
- **`hour`** — any integer from 0 to 23
- **`day`** — any integer from 1 to 31 (must be a valid day if a month is specified)

- **`month`** — any integer from 1 to 12 (or the short name of the month such as jan or feb)
- **`dayofweek`** — any integer from 0 to 7, where 0 or 7 represents Sunday (or the short name of the week such as sun or mon)
- **`command`** — the command to execute (the command can either be a command such as **`ls /proc >> /tmp/proc`** or the command to execute a custom script)

For any of the above values, an asterisk (*) can be used to specify all valid values. For example, an asterisk for the month value means execute the command every month within the constraints of the other values.

A hyphen (-) between integers specifies a range of integers. For example, **`1-4`** means the integers 1, 2, 3, and 4.

A list of values separated by commas (,) specifies a list. For example, **`3, 4, 6, 8`** indicates those four specific integers.

The forward slash (/) can be used to specify step values. The value of an integer can be skipped within a range by following the range with **`/<`*`integer`*`>`**. For example, **`0-59/2`** can be used to define every other minute in the minute field. Step values can also be used with an asterisk. For instance, the value **`*/3`** can be used in the month field to run the job every third month.

Any lines that begin with a hash mark (#) are comments and are not processed.

As shown in the **`/etc/crontab`** file, the **`run-parts`** script executes the scripts in the **`/etc/cron.hourly/`**, **`/etc/cron.daily/`**, **`/etc/cron.weekly/`**, and **`/etc/cron.monthly/`** directories on an hourly, daily, weekly, or monthly basis respectively. The files in these directories should be shell scripts.

If a cron job is required to be executed on a schedule other than hourly, daily, weekly, or monthly, it can be added to the **`/etc/cron.d/`** directory. All files in this directory use the same syntax as **`/etc/crontab`**. Refer to Example 39.1, “Sample of /etc/crontab” for examples.

Example 39.1. Sample of /etc/crontab

```
# record the memory usage of the system every monday
# at 3:30AM in the file /tmp/meminfo
30 3 * * mon cat /proc/meminfo >> /tmp/meminfo
# run custom script the first day of every month at 4:10AM
10 4 1 * * /root/scripts/backup.sh
```

Users other than root can configure cron jobs by using the **`crontab`** utility. All user-defined crontabs are stored in the **`/var/spool/cron/`** directory and are executed using the usernames of the users that created them. To create a crontab as a user, login as that user and type the command **`crontab -e`** to edit the user's crontab using the editor specified by the **`VISUAL`** or **`EDITOR`** environment variable. The file uses the same format as **`/etc/crontab`**. When the changes to the crontab are saved, the crontab is stored according to username and written to the file **`/var/spool/cron/`*`username`***.

The cron daemon checks the **`/etc/crontab`** file, the **`/etc/cron.d/`** directory, and the **`/var/spool/cron/`** directory every minute for any changes. If any changes are found, they are loaded into memory. Thus, the daemon does not need to be restarted if a crontab file is changed.

Cron jobs can be run at random intervals, which is useful for highly loaded shared networks in order to avoid overloading the network. Job randomization is disabled by default but it can be configured in the **`/etc/sysconfig/run-parts`** file by specifying the following parameters:

- **`RANDOMIZE`** — When set to **`1`**, it enables randomize functionality. When set to **`0`**, cron job randomization is disabled.
- **`RANDOM`** — Specifies the initial random seed. It has to be set to an integer value greater than or equal to **`1`**.
- **`RANDOMTIME`** — When set to an integer value greater than or equal to **`1`**, it provides an additional level of randomization.

Example 39.2. Sample of /etc/sysconfig/run-parts - Job Randomization Setting

```
RANDOMIZE=1
RANDOM=4
RANDOMTIME=8
```

39.1.2. Controlling Access to Cron

The **`/etc/cron.allow`** and **`/etc/cron.deny`** files are used to restrict access to cron. The format of both access control files is one username on each line. Whitespace is not permitted in either file. The cron daemon (**`crond`**) does not have to be restarted if the access control files are modified. The access control files are read each time a user tries to add or delete a cron job.

The root user can always use cron, regardless of the usernames listed in the access control files.

If the file **`cron.allow`** exists, only users listed in it are allowed to use cron, and the **`cron.deny`** file is ignored.

If **`cron.allow`** does not exist, users listed in **`cron.deny`** are not allowed to use cron.

39.1.3. Starting and Stopping the Service

To start the cron service, use the command **`/sbin/service crond start`**. To stop the service, use the command **`/sbin/service crond stop`**. It is recommended that you start the service at boot time. Refer to Chapter 18, *Controlling Access to Services* for details on starting the cron service automatically at boot time.

39.2. At and Batch

While cron is used to schedule recurring jobs, the **`at`** command is used to schedule a one-time job at a specific time and the **`batch`** command is used to schedule a one-time job to be executed when the systems load average drops below 0.8.

To use **`at`** or **`batch`**, the **`at`** RPM package must be installed, and the **`atd`** service must be running. To determine if the package is installed, use the **`rpm -q at`** command. To determine if the service is running, use the command **`/sbin/service atd status`**.

39.2.1. Configuring At Jobs

To schedule a one-time job at a specific time, type the command **`at`** ***`time`***, where ***`time`*** is the time to execute the command.

The argument *time* can be one of the following:

- HH:MM format — For example, 04:00 specifies 4:00 a.m. If the time is already past, it is executed at the specified time the next day.
- midnight — Specifies 12:00 a.m.
- noon — Specifies 12:00 p.m.
- teatime — Specifies 4:00 p.m.
- month-name day year format — For example, January 15 2002 specifies the 15th day of January in the year 2002. The year is optional.
- MMDDYY, MM/DD/YY, or MM.DD.YY formats — For example, 011502 for the 15th day of January in the year 2002.
- now + time — time is in minutes, hours, days, or weeks. For example, now + 5 days specifies that the command should be executed at the same time five days from now.

The time must be specified first, followed by the optional date. For more information about the time format, read the **`/usr/share/doc/at-<`*`version`*`>/timespec`** text file.

After typing the **`at`** command with the time argument, the **`at>`** prompt is displayed. Type the command to execute, press **`Enter`**, and type **`Ctrl`**+**`D`** . Multiple commands can be specified by typing each command followed by the **`Enter`** key. After typing all the commands, press **`Enter`** to go to a blank line and type **`Ctrl`**+**`D`** . Alternatively, a shell script can be entered at the prompt, pressing **`Enter`** after each line in the script, and typing **`Ctrl`**+**`D`** on a blank line to exit. If a script is entered, the shell used is the shell set in the user's **`SHELL`** environment, the user's login shell, or **`/bin/sh`** (whichever is found first).

If the set of commands or script tries to display information to standard out, the output is emailed to the user.

Use the command **`atq`** to view pending jobs. Refer to Section 39.2.3, “Viewing Pending Jobs” for more information.

Usage of the **`at`** command can be restricted. For more information, refer to Section 39.2.5, “Controlling Access to At and Batch” for details.

39.2.2. Configuring Batch Jobs

To execute a one-time job when the load average is below 0.8, use the **`batch`** command.

After typing the **`batch`** command, the **`at>`** prompt is displayed. Type the command to execute, press **`Enter`**, and type **`Ctrl`**+**`D`** . Multiple commands can be specified by typing each command followed by the **`Enter`** key. After typing all the commands, press **`Enter`** to go to a blank line and type **`Ctrl`**+**`D`** . Alternatively, a shell script can be entered at the prompt, pressing **`Enter`** after each line in the script, and typing **`Ctrl`**+**`D`** on a blank line to exit. If a script is entered, the shell used is the shell set in the user's **`SHELL`** environment, the user's login shell, or **`/bin/sh`** (whichever is found first). As soon as the load average is below 0.8, the set of commands or script is executed.

If the set of commands or script tries to display information to standard out, the output is emailed to the user.

Use the command **`atq`** to view pending jobs. Refer to Section 39.2.3, “Viewing Pending Jobs” for more information.

Usage of the **`batch`** command can be restricted. For more information, refer to Section 39.2.5, “Controlling Access to At and Batch” for details.

39.2.3. Viewing Pending Jobs

To view pending **`at`** and **`batch`** jobs, use the **`atq`** command. The **`atq`** command displays a list of pending jobs, with each job on a line. Each line follows the job number, date, hour, job class, and username format. Users can only view their own jobs. If the root user executes the **`atq`** command, all jobs for all users are displayed.

39.2.4. Additional Command Line Options

Additional command line options for **`at`** and **`batch`** include:

Table 39.1. `at` and `batch` Command Line Options

Option	Description
`-f`	Read the commands or shell script from a file instead of specifying them at the prompt.
`-m`	Send email to the user when the job has been completed.
`-v`	Display the time that the job is executed.

39.2.5. Controlling Access to At and Batch

The **`/etc/at.allow`** and **`/etc/at.deny`** files can be used to restrict access to the **`at`** and **`batch`** commands. The format of both access control files is one username on each line. Whitespace is not permitted in either file. The **`at`** daemon (**`atd`**) does not have to be restarted if the access control files are modified. The access control files are read each time a user tries to execute the **`at`** or **`batch`** commands.

The root user can always execute **`at`** and **`batch`** commands, regardless of the access control files.

If the file **`at.allow`** exists, only users listed in it are allowed to use **`at`** or **`batch`**, and the **`at.deny`** file is ignored.

If **`at.allow`** does not exist, users listed in **`at.deny`** are not allowed to use **`at`** or **`batch`**.

39.2.6. Starting and Stopping the Service

To start the **`at`** service, use the command **`/sbin/service atd start`**. To stop the service, use the command **`/sbin/service atd stop`**. It is recommended that you start the service at boot time. Refer to Chapter 18, *Controlling Access to Services* for details on starting the cron service automatically at boot time.

39.3. Additional Resources

To learn more about configuring automated tasks, refer to the following resources.

39.3.1. Installed Documentation

- **`cron`** man page — overview of cron.
- **`crontab`** man pages in sections 1 and 5 — The man page in section 1 contains an overview of the **`crontab`** file. The man page in section 5 contains the format for the file and some example entries.
- **`/usr/share/doc/at-<`*`version`*`>/timespec`** contains more detailed information about the times that can be specified for cron jobs.
- **`at`** man page — description of **`at`** and **`batch`** and their command line options.

Chapter 40. Log Files

Log files are files that contain messages about the system, including the kernel, services, and applications running on it. There are different log files for different information. For example, there is a default system log file, a log file just for security messages, and a log file for cron tasks.

Log files can be very useful when trying to troubleshoot a problem with the system such as trying to load a kernel driver or when looking for unauthorized log in attempts to the system. This chapter discusses where to find log files, how to view log files, and what to look for in log files.

Some log files are controlled by a daemon called **`syslogd`**. A list of log messages maintained by **`syslogd`** can be found in the **`/etc/syslog.conf`** configuration file.

40.1. Locating Log Files

Most log files are located in the **`/var/log/`** directory. Some applications such as **`httpd`** and **`samba`** have a directory within **`/var/log/`** for their log files.

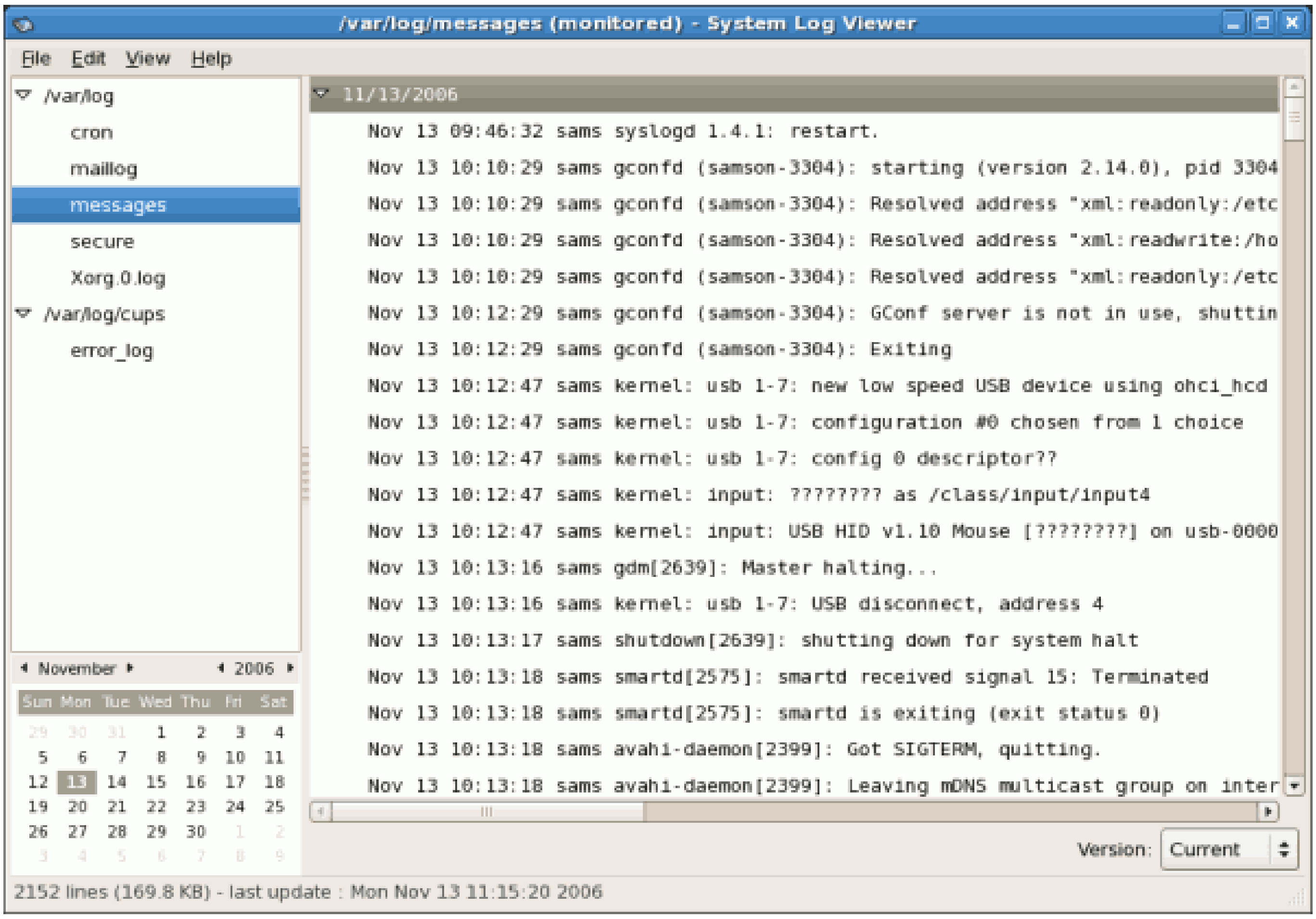
You may notice multiple files in the log file directory with numbers after them. These are created when the log files are rotated. Log files are rotated so their file sizes do not become too large. The **`logrotate`** package contains a cron task that automatically rotates log files according to the **`/etc/logrotate.conf`** configuration file and the configuration files in the **`/etc/logrotate.d/`** directory. By default, it is configured to rotate every week and keep four weeks worth of previous log files.

40.2. Viewing Log Files

Most log files are in plain text format. You can view them with any text editor such as **`Vi`** or **Emacs**. Some log files are readable by all users on the system; however, root privileges are required to read most log files.

To view system log files in an interactive, real-time application, use the **System Log Viewer**. To start the application, go to **Applications** (the main menu on the panel) > **System** > **System Logs**, or type the command **`gnome-system-log`** at a shell prompt.

The application only displays log files that exist; thus, the list might differ from the one shown in Figure 40.1, “System Log Viewer”.

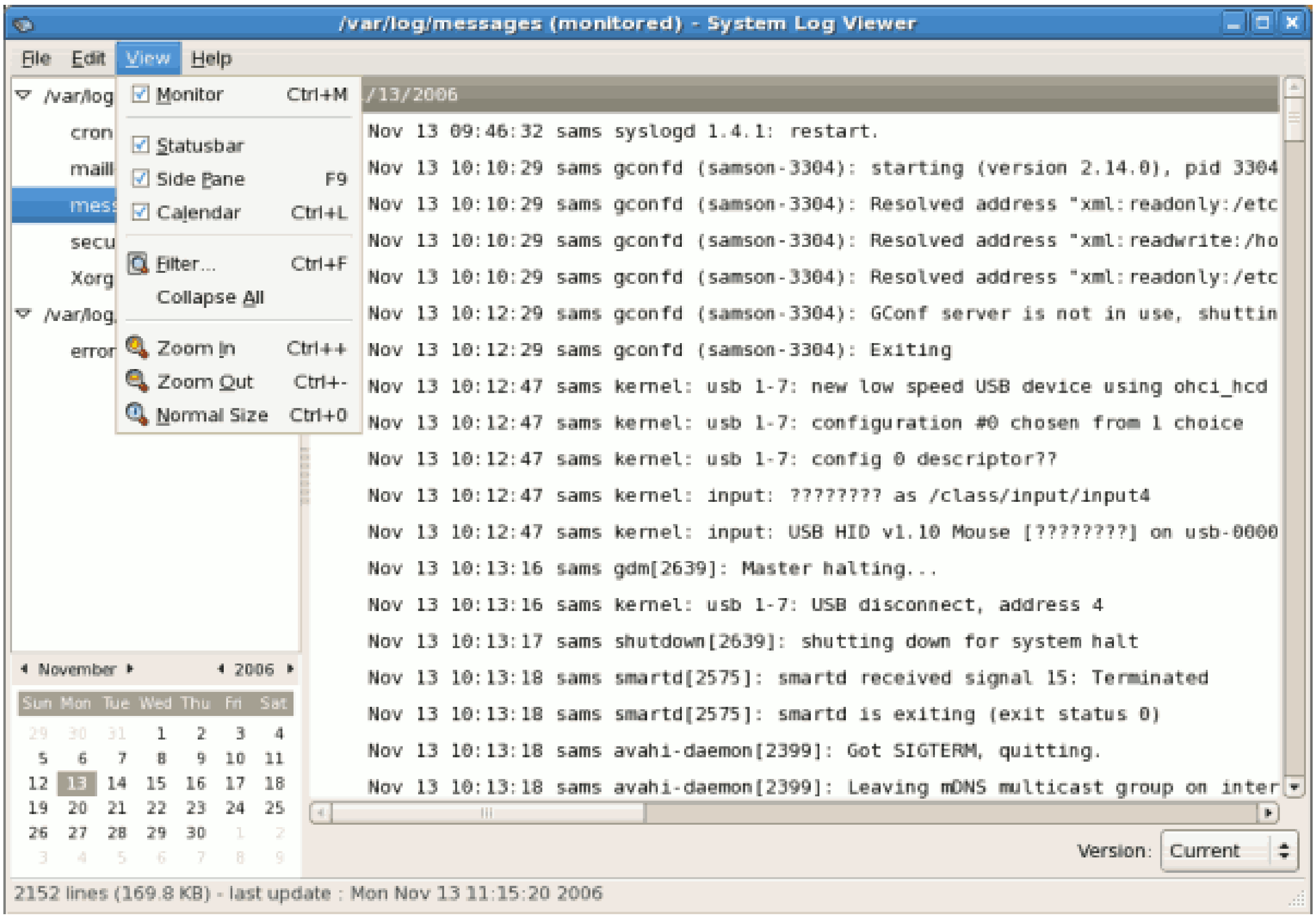


```
Nov 13 09:46:32 sams syslogd 1.4.1: restart.
Nov 13 10:10:29 sams gconfd (samson-3304): starting (version 2.14.0), pid 3304
Nov 13 10:10:29 sams gconfd (samson-3304): Resolved address "xml:readonly:/etc
Nov 13 10:10:29 sams gconfd (samson-3304): Resolved address "xml:readwrite:/ho
Nov 13 10:10:29 sams gconfd (samson-3304): Resolved address "xml:readonly:/etc
Nov 13 10:12:29 sams gconfd (samson-3304): GConf server is not in use, shuttin
Nov 13 10:12:29 sams gconfd (samson-3304): Exiting
Nov 13 10:12:47 sams kernel: usb 1-7: new low speed USB device using ohci_hcd
Nov 13 10:12:47 sams kernel: usb 1-7: configuration #0 chosen from 1 choice
Nov 13 10:12:47 sams kernel: usb 1-7: config 0 descriptor??
Nov 13 10:12:47 sams kernel: input: ???????? as /class/input/input4
Nov 13 10:12:47 sams kernel: input: USB HID v1.10 Mouse [????????] on usb-0000
Nov 13 10:13:16 sams gdm[2639]: Master halting...
Nov 13 10:13:16 sams kernel: usb 1-7: USB disconnect, address 4
Nov 13 10:13:17 sams shutdown[2639]: shutting down for system halt
Nov 13 10:13:18 sams smartd[2575]: smartd received signal 15: Terminated
Nov 13 10:13:18 sams smartd[2575]: smartd is exiting (exit status 0)
Nov 13 10:13:18 sams avahi-daemon[2399]: Got SIGTERM, quitting.
Nov 13 10:13:18 sams avahi-daemon[2399]: Leaving mDNS multicast group on inter
```

2152 lines (169.8 KB) - last update : Mon Nov 13 11:15:20 2006

Figure 40.1. System Log Viewer

To filter the contents of the selected log file, click on **View** from the menu and select **Filter** as illustrated below.

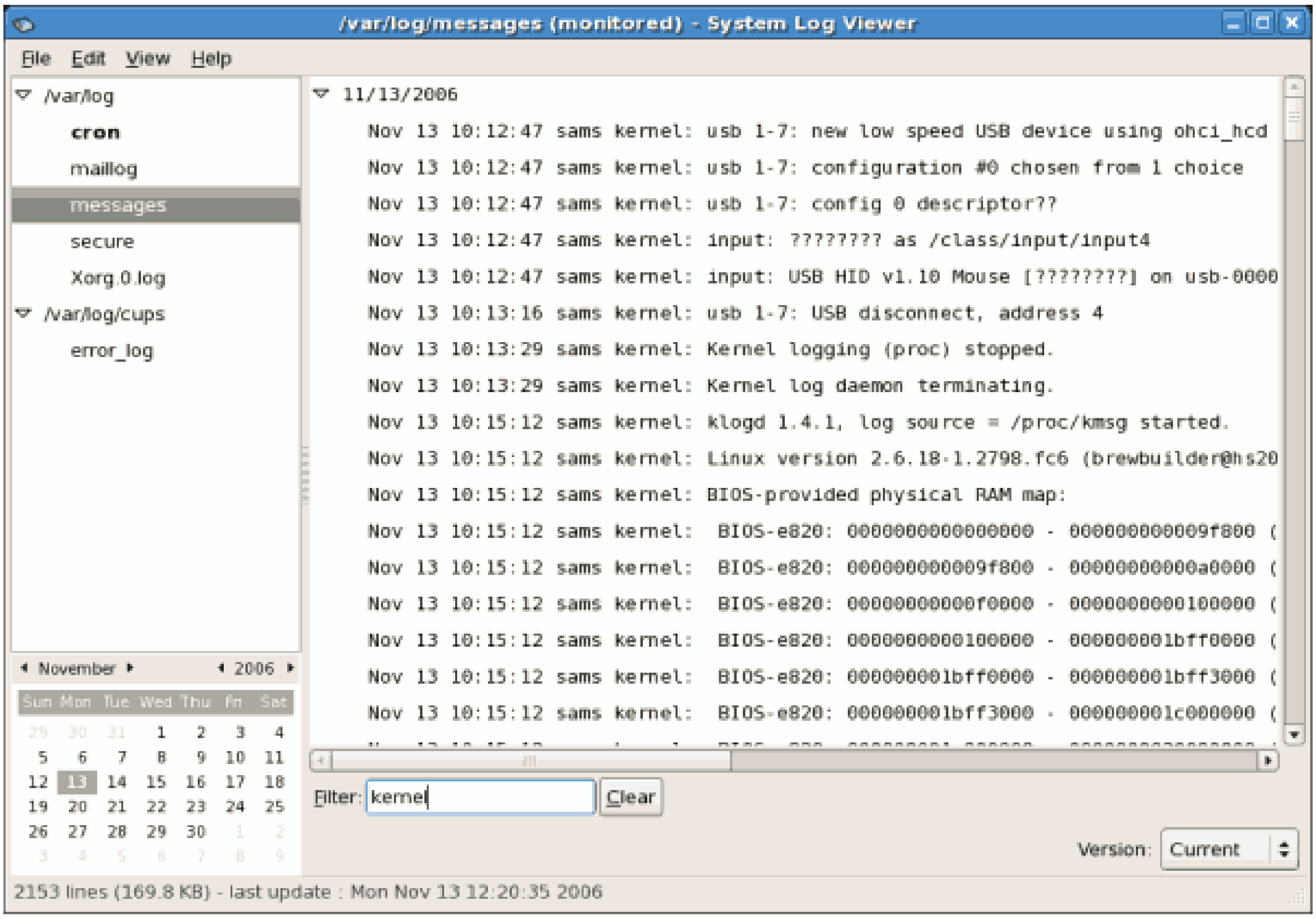


```
/13/2006
Nov 13 09:46:32 sams syslogd 1.4.1: restart.
Nov 13 10:10:29 sams gconfd (samson-3304): starting (version 2.14.0), pid 3304
Nov 13 10:10:29 sams gconfd (samson-3304): Resolved address "xml:readonly:/etc
Nov 13 10:10:29 sams gconfd (samson-3304): Resolved address "xml:readwrite:/ho
Nov 13 10:10:29 sams gconfd (samson-3304): Resolved address "xml:readonly:/etc
Nov 13 10:12:29 sams gconfd (samson-3304): GConf server is not in use, shuttin
Nov 13 10:12:29 sams gconfd (samson-3304): Exiting
Nov 13 10:12:47 sams kernel: usb 1-7: new low speed USB device using ohci_hcd
Nov 13 10:12:47 sams kernel: usb 1-7: configuration #0 chosen from 1 choice
Nov 13 10:12:47 sams kernel: usb 1-7: config 0 descriptor??
Nov 13 10:12:47 sams kernel: input: ???????? as /class/input/input4
Nov 13 10:12:47 sams kernel: input: USB HID v1.10 Mouse [????????] on usb-0000
Nov 13 10:13:16 sams gdm[2639]: Master halting...
Nov 13 10:13:16 sams kernel: usb 1-7: USB disconnect, address 4
Nov 13 10:13:17 sams shutdown[2639]: shutting down for system halt
Nov 13 10:13:18 sams smartd[2575]: smartd received signal 15: Terminated
Nov 13 10:13:18 sams smartd[2575]: smartd is exiting (exit status 0)
Nov 13 10:13:18 sams avahi-daemon[2399]: Got SIGTERM, quitting.
Nov 13 10:13:18 sams avahi-daemon[2399]: Leaving mDNS multicast group on inter
```

2152 lines (169.8 KB) - last update : Mon Nov 13 11:15:20 2006

Figure 40.2. System Log Viewer - View Menu

Selecting the **`Filter`** menu item will display the **`Filter`** text field where you can type the keywords you wish to use for your filter. To clear your filter click on the **`Clear`** button.The figure below illustrates a sample filter.



```
11/13/2006
    Nov 13 10:12:47 sams kernel: usb 1-7: new low speed USB device using ohci_hcd
    Nov 13 10:12:47 sams kernel: usb 1-7: configuration #0 chosen from 1 choice
    Nov 13 10:12:47 sams kernel: usb 1-7: config 0 descriptor??
    Nov 13 10:12:47 sams kernel: input: ???????? as /class/input/input4
    Nov 13 10:12:47 sams kernel: input: USB HID v1.10 Mouse [????????] on usb-0000
    Nov 13 10:13:16 sams kernel: usb 1-7: USB disconnect, address 4
    Nov 13 10:13:29 sams kernel: Kernel logging (proc) stopped.
    Nov 13 10:13:29 sams kernel: Kernel log daemon terminating.
    Nov 13 10:15:12 sams kernel: klogd 1.4.1, log source = /proc/kmsg started.
    Nov 13 10:15:12 sams kernel: Linux version 2.6.18-1.2798.fc6 (brewbuilder@hs20
    Nov 13 10:15:12 sams kernel: BIOS-provided physical RAM map:
    Nov 13 10:15:12 sams kernel:  BIOS-e820: 0000000000000000 - 000000000009f800 (
    Nov 13 10:15:12 sams kernel:  BIOS-e820: 000000000009f800 - 00000000000a0000 (
    Nov 13 10:15:12 sams kernel:  BIOS-e820: 00000000000f0000 - 0000000000100000 (
    Nov 13 10:15:12 sams kernel:  BIOS-e820: 0000000000100000 - 000000001bff0000 (
    Nov 13 10:15:12 sams kernel:  BIOS-e820: 000000001bff0000 - 000000001bff3000 (
    Nov 13 10:15:12 sams kernel:  BIOS-e820: 000000001bff3000 - 000000001c000000 (
```

Filter: kernel

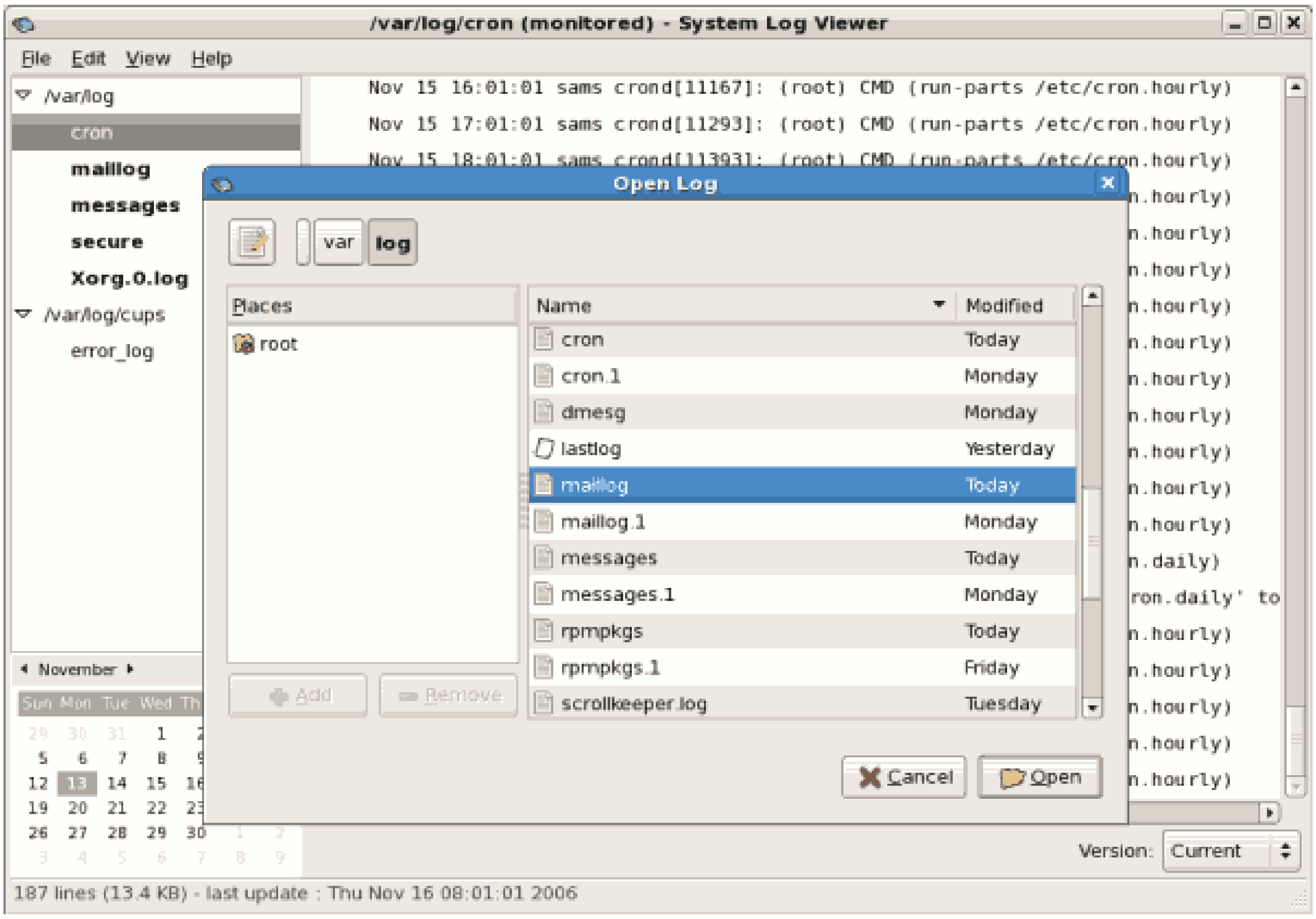
Version: Current

2153 lines (169.8 KB) - last update : Mon Nov 13 12:20:35 2006

Figure 40.3. System Log Viewer - Filter

40.3. Adding a Log File

To add a log file you wish to view in the list, select **File** > **Open**. This will display the **Open Log** window where you can select the directory and filename of the log file you wish to view.The figure below illustrates the **Open Log** window.



```
Nov 15 16:01:01 sams crond[11167]: (root) CMD (run-parts /etc/cron.hourly)
Nov 15 17:01:01 sams crond[11293]: (root) CMD (run-parts /etc/cron.hourly)
Nov 15 18:01:01 sams crond[11393]: (root) CMD (run-parts /etc/cron.hourly)
```

Name	Modified
cron	Today
cron.1	Monday
dmesg	Monday
lastlog	Yesterday
maillog	Today
maillog.1	Monday
messages	Today
messages.1	Monday
rpmpkgs	Today
rpmpkgs.1	Friday
scrollkeeper.log	Tuesday

```
187 lines (13.4 KB) - last update : Thu Nov 16 08:01:01 2006
```

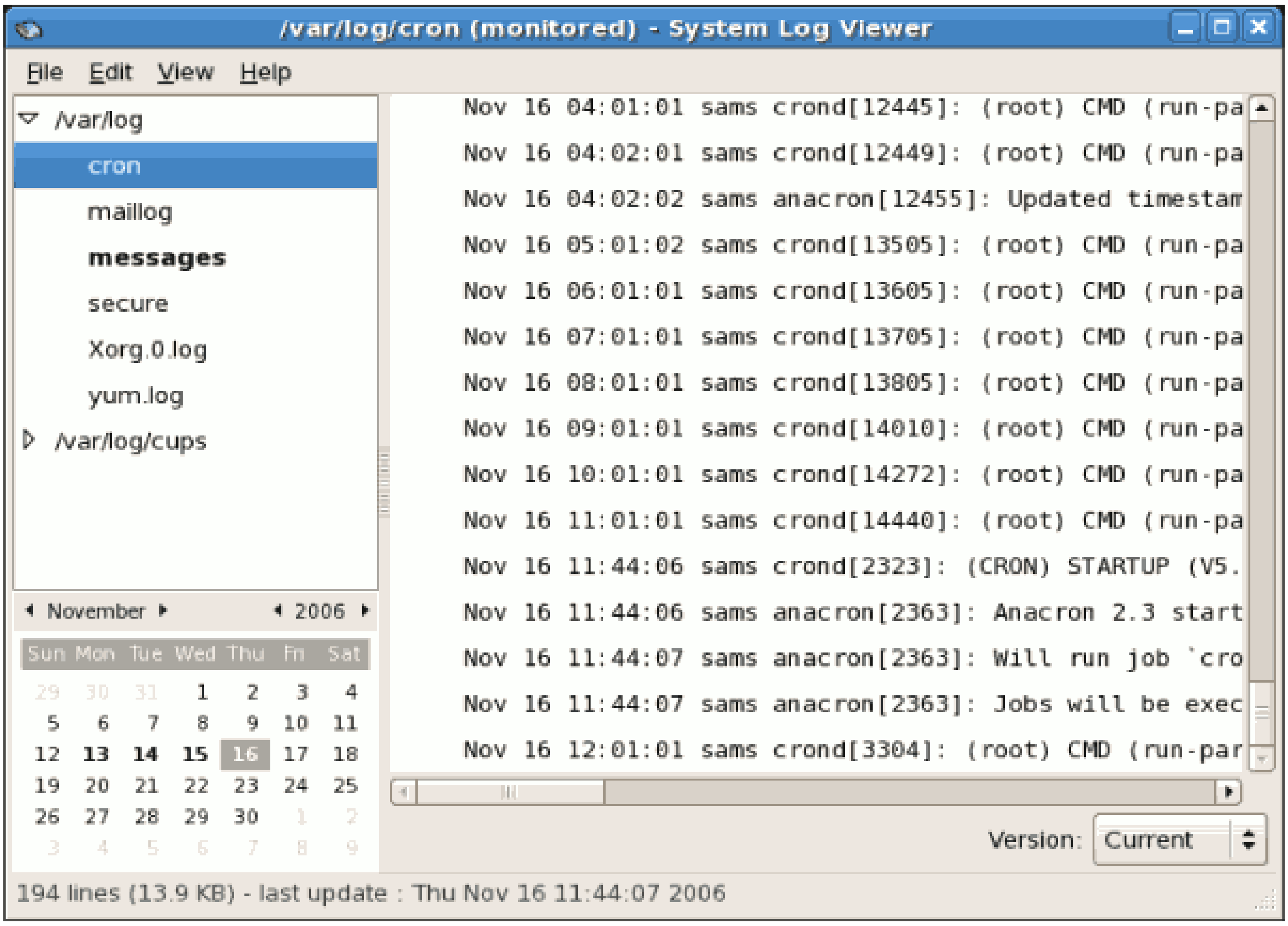
Figure 40.4. Adding a Log File

Click on the **Open** button to open the file. The file is immediately added to the viewing list where you can select it and view the contents.

Please also note that the System Log Viewer also allows you to open zipped logs whose filenames end in ".gz".

40.4. Monitoring Log Files

System Log Viewer monitors all opened logs by default. If a new line is added to a monitored log file, the log name appears in bold in the log list. If the log file is selected or displayed, the new lines appear in bold at the bottom of the log file and after five seconds are displayed in normal format. This is illustrated in the figures below. The figure below illustrates a new alert in the **messages** log file. The log file is listed in bold text.

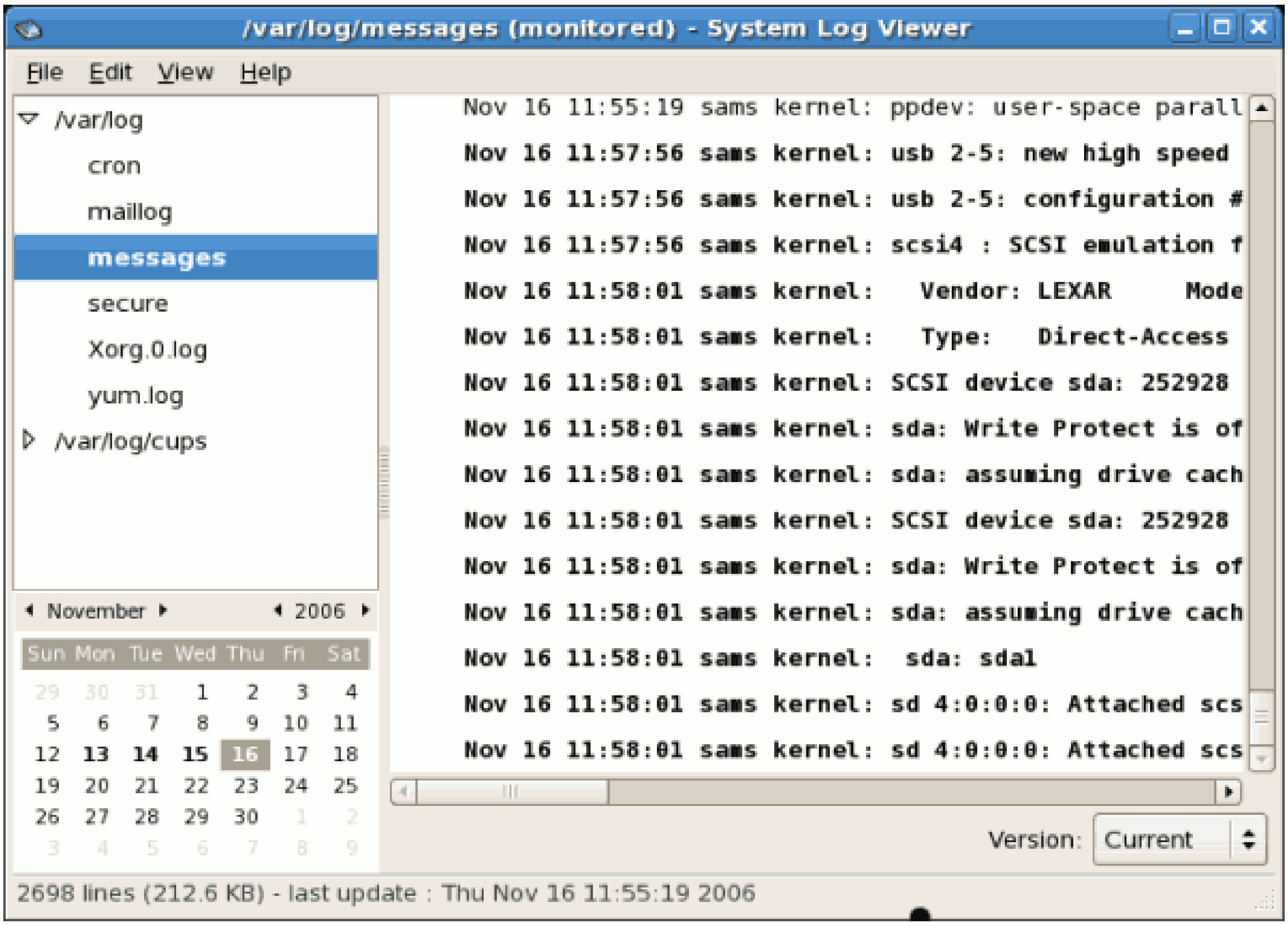


```
Nov 16 04:01:01 sams crond[12445]: (root) CMD (run-pa
Nov 16 04:02:01 sams crond[12449]: (root) CMD (run-pa
Nov 16 04:02:02 sams anacron[12455]: Updated timestam
Nov 16 05:01:02 sams crond[13505]: (root) CMD (run-pa
Nov 16 06:01:01 sams crond[13605]: (root) CMD (run-pa
Nov 16 07:01:01 sams crond[13705]: (root) CMD (run-pa
Nov 16 08:01:01 sams crond[13805]: (root) CMD (run-pa
Nov 16 09:01:01 sams crond[14010]: (root) CMD (run-pa
Nov 16 10:01:01 sams crond[14272]: (root) CMD (run-pa
Nov 16 11:01:01 sams crond[14440]: (root) CMD (run-pa
Nov 16 11:44:06 sams crond[2323]: (CRON) STARTUP (V5.
Nov 16 11:44:06 sams anacron[2363]: Anacron 2.3 start
Nov 16 11:44:07 sams anacron[2363]: Will run job `cro
Nov 16 11:44:07 sams anacron[2363]: Jobs will be exec
Nov 16 12:01:01 sams crond[3304]: (root) CMD (run-par
```

194 lines (13.9 KB) - last update : Thu Nov 16 11:44:07 2006

Figure 40.5. Log File Alert

Clicking on the **messages** log file displays the logs in the file with the new lines in bold as illustrated below.

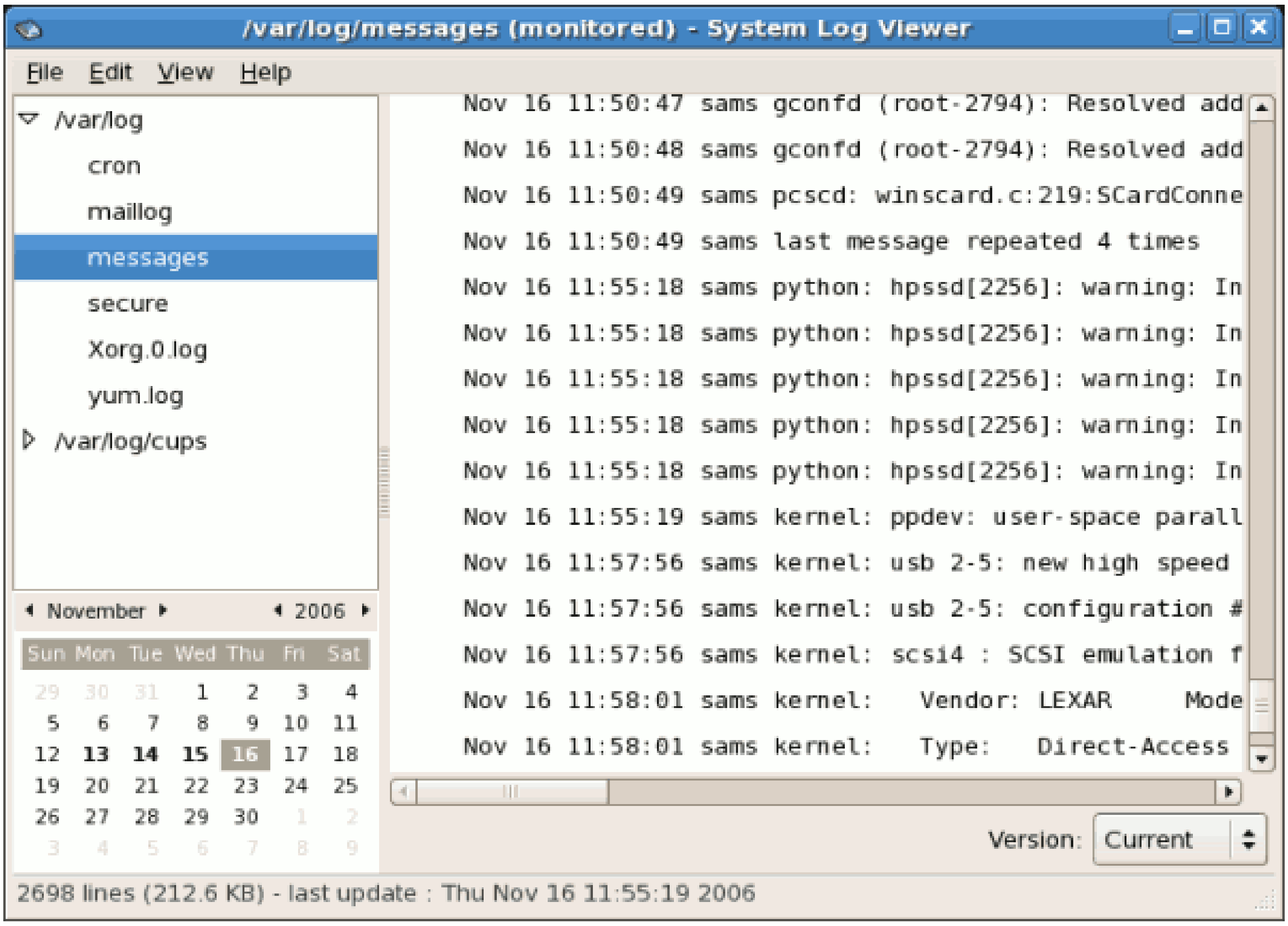


```
Nov 16 11:55:19 sams kernel: ppdev: user-space parall
Nov 16 11:57:56 sams kernel: usb 2-5: new high speed
Nov 16 11:57:56 sams kernel: usb 2-5: configuration #
Nov 16 11:57:56 sams kernel: scsi4 : SCSI emulation f
Nov 16 11:58:01 sams kernel:   Vendor: LEXAR     Mode
Nov 16 11:58:01 sams kernel:   Type:   Direct-Access
Nov 16 11:58:01 sams kernel: SCSI device sda: 252928
Nov 16 11:58:01 sams kernel: sda: Write Protect is of
Nov 16 11:58:01 sams kernel: sda: assuming drive cach
Nov 16 11:58:01 sams kernel: SCSI device sda: 252928
Nov 16 11:58:01 sams kernel: sda: Write Protect is of
Nov 16 11:58:01 sams kernel: sda: assuming drive cach
Nov 16 11:58:01 sams kernel:  sda: sda1
Nov 16 11:58:01 sams kernel: sd 4:0:0:0: Attached scs
Nov 16 11:58:01 sams kernel: sd 4:0:0:0: Attached scs
```

2698 lines (212.6 KB) - last update : Thu Nov 16 11:55:19 2006

Figure 40.6. Log file contents

The new lines are displayed in bold for five seconds after which they are displayed in normal font.



```
Nov 16 11:50:47 sams gconfd (root-2794): Resolved add
Nov 16 11:50:48 sams gconfd (root-2794): Resolved add
Nov 16 11:50:49 sams pcscd: winscard.c:219:SCardConne
Nov 16 11:50:49 sams last message repeated 4 times
Nov 16 11:55:18 sams python: hpssd[2256]: warning: In
Nov 16 11:55:18 sams python: hpssd[2256]: warning: In
Nov 16 11:55:18 sams python: hpssd[2256]: warning: In
Nov 16 11:55:18 sams python: hpssd[2256]: warning: In
Nov 16 11:55:18 sams python: hpssd[2256]: warning: In
Nov 16 11:55:19 sams kernel: ppdev: user-space parall
Nov 16 11:57:56 sams kernel: usb 2-5: new high speed
Nov 16 11:57:56 sams kernel: usb 2-5: configuration #
Nov 16 11:57:56 sams kernel: scsi4 : SCSI emulation f
Nov 16 11:58:01 sams kernel:   Vendor: LEXAR     Mode
Nov 16 11:58:01 sams kernel:   Type:   Direct-Access
```

2698 lines (212.6 KB) - last update : Thu Nov 16 11:55:19 2006

Figure 40.7. Log file contents after five seconds

Part V. System Monitoring

System administrators also monitor system performance. Red Hat Enterprise Linux contains tools to assist administrators with these tasks.

Chapter 41. SystemTap

41.1. Introduction

SystemTap provides a simple command line interface and scripting language to simplify the gathering of information about the running Linux kernel so that it can be further analyzed. Data may be extracted, filtered, and summarized quickly and safely, to enable diagnoses of complex performance or functional problems.

SystemTap allows scripts to be written in the SystemTap scripting language, which are then compiled to C-code kernel modules and inserted into the kernel.

The essential idea behind a systemtap script is to name events, and to give them handlers. Whenever a specified event occurs, the Linux kernel runs the handler as if it were a quick subroutine, then resumes. There are several kind of events, such as entering or exiting a function, a timer expiring, or the entire systemtap session starting or stopping. A handler is a series of script language statements that specify the work to be done whenever the event occurs. This work normally includes extracting data from the event context, storing them into internal variables, or printing results.

41.2. Implementation

SystemTap takes a compiler-oriented approach to generating instrumentation. Refer to Figure 41.1, “Flow of Data in SystemTap” "Flow of data in SystemTap" for an overall diagram of SystemTap used in this discussion. In the upper right hand corner of the diagram is the probe.stp, the probe script the developer has written. This is parsed by the translator into parse trees. During this time the input is checked for syntax errors. The translator then performs elaboration, pulling in additional code from the script library and determining locations of probe points and variables from the debug information. After the elaboration is complete the translator can generate the probe.c, the kernel module in C.

The probe.c file is compiled into a regular kernel module, probe.ko, using the GCC compiler. The compilation may pull in support code from the runtime libraries. After GCC has generated the probe.ko, the SystemTap daemon is started to collect the output of the instrumentation module. The instrumentation module is loaded into the kernel, and data collection is started. Data from the instrumentation module is transferred to user-space via relayfs and displayed by the daemon. When the user hits Control-C the daemon unloads the module, which also shuts down the data collection process.

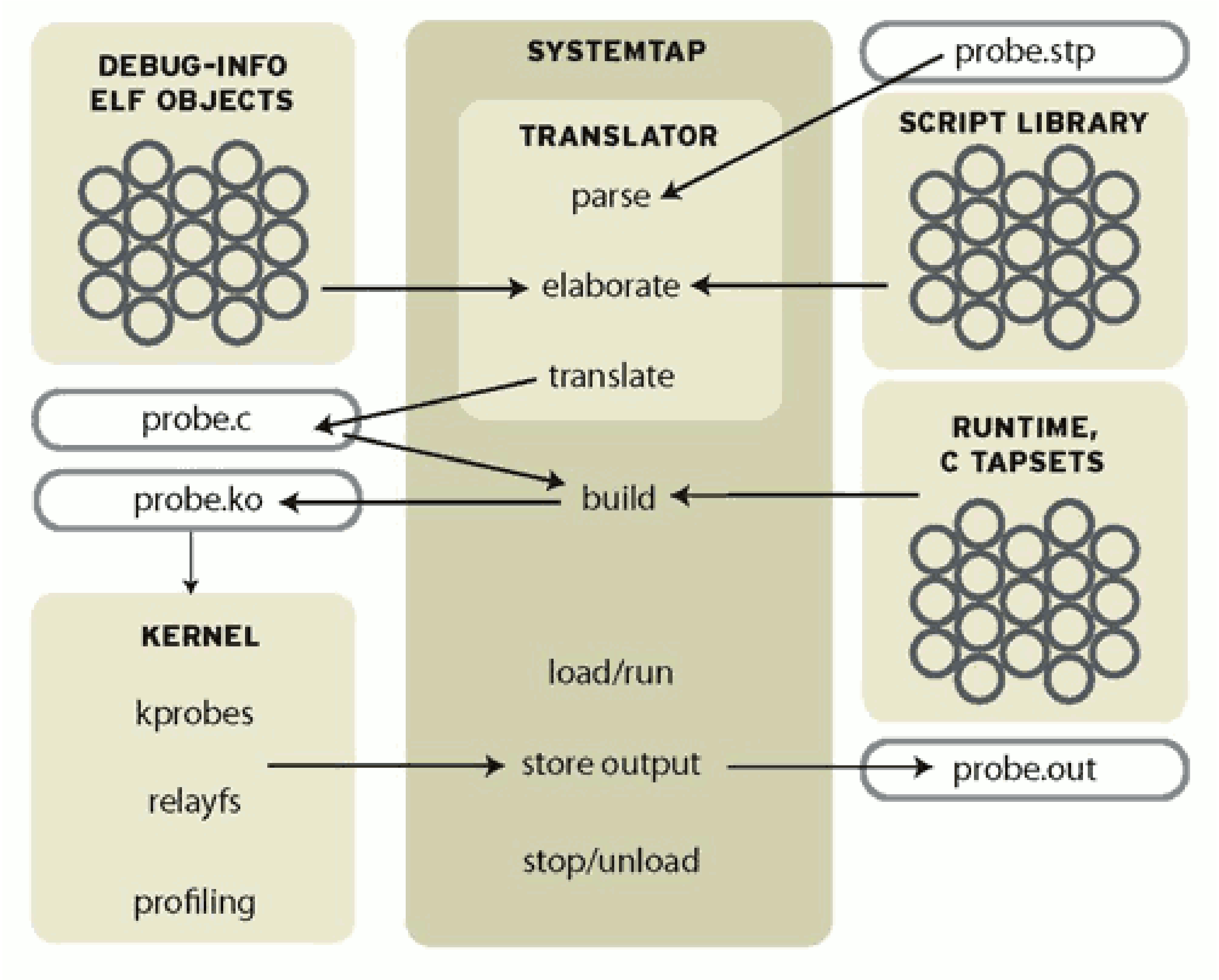


Figure 41.1. Flow of Data in SystemTap

41.3. Using SystemTap

Systemtap works by translating a SystemTap script to C, running the system C compiler to create a kernel module from that. When the module is loaded, it activates all the probed events by hooking into the kernel. Then, as events occur on any processor, the compiled handlers run. Eventually, the session stops, the hooks are disconnected, and the module removed. This entire process is driven from a single command-line program, `stap`.

41.3.1. Tracing

The simplest kind of probe is simply to trace an event. This is the effect of inserting strategically located print statements into a program. This is often the first step of problem solving: explore by seeing a history of what has happened.

This style of instrumentation is the simplest. It just asks systemtap to print something at each event. To express this in the script language, you need to say where to probe and what to print there.

41.3.1.1. Where to Probe

Systemtap supports a number of built-in events. The library of scripts that comes with systemtap, each called a "tapset", may define additional ones defined in terms of the built-in family. See the `stapprobes` man page for details. All these events are named using a unified syntax that looks like dot-separated parameterized identifiers:

Table 41.1. SystemTap Events

Event	Description
`begin`	The startup of the systemtap session.
`end`	The end of the systemtap session.
`kernel.function("sys_open")`	The entry to the function named sys_open in the kernel.
`syscall.close.return`	The return from the close system call..
`module("ext3").statement(0xdeadbeef)`	The addressed instruction in the ext3 filesystem driver.
`timer.ms(200)`	A timer that fires every 200 milliseconds.

We will use as a demonstration case that you would like to trace all function entries and exits in a source file, for example **`net/socket.c`** in the kernel. The **`kernel.function`** probe point lets you express that easily, since systemtap examines the kernel's debugging information to relate object code to source code. It works like a debugger: if you can name or place it, you can probe it. Use **`kernel.function("*@net/socket.c")`** for the function entries, and **`kernel.function("*@net/socket.c").return`** for the exits. Note the use of wildcards in the function name part, and the subsequent **`@FILENAME`** part. You can also put wildcards into the file name, and even add a colon (:) and a line number, if you want to restrict the search that precisely. Since systemtap will put a separate probe in every place that matches a probe point, a few wildcards can expand to hundreds or thousands of probes, so be careful what you ask for.

Once you identify the probe points, the skeleton of the systemtap script appears. The **`probe`** keyword introduces a probe point, or a comma-separated list of them. The following { and } braces enclose the handler for all listed probe points.

You can run this script as is, though with empty handlers there will be no output. Put the two lines into a new file. Run **`stap -v FILE`**. Terminate it any time with **`^C`**. (The **`-v`** option tells systemtap to print more verbose messages during its processing. Try the **`-h`** option to see more options.)

41.3.1.2. What to Print

Since you are interested in each function that was entered and exited, a line should be printed for each, containing the function name. In order to make that list easy to read, systemtap should indent the lines so that functions called by other traced functions are nested deeper. To tell each single process apart from any others that may be running concurrently, systemtap should also print the process ID in the line.

Chapter 42. Gathering System Information

Before you learn how to configure your system, you should learn how to gather essential system information. For example, you should know how to find the amount of free memory, the amount of available hard drive space, how your hard drive is partitioned, and what processes are running. This chapter discusses how to retrieve this type of information from your Red Hat Enterprise Linux system using simple commands and a few simple programs.

42.1. System Processes

The **`ps ax`** command displays a list of current system processes, including processes owned by other users. To display the owner alongside each process, use the **`ps aux`** command. This list is a static list; in other words, it is a snapshot of what was running when you invoked the command. If you want a constantly updated list of running processes, use **`top`** as described below.

The **`ps`** output can be long. To prevent it from scrolling off the screen, you can pipe it through less:

```
ps aux | less
```

You can use the **`ps`** command in combination with the **`grep`** command to see if a process is running. For example, to determine if **Emacs** is running, use the following command:

```
ps ax | grep emacs
```

The **`top`** command displays currently running processes and important information about them including their memory and CPU usage. The list is both real-time and interactive. An example of output from the **`top`** command is provided as follows:

```
top - 15:02:46 up 35 min,  4 users,  load average: 0.17, 0.65, 1.00
Tasks: 110 total,   1 running, 107 sleeping,   0 stopped,   2 zombie
Cpu(s): 41.1% us,  2.0% sy,  0.0% ni, 56.6% id,  0.0% wa,  0.3% hi,  0.0% si
Mem:    775024k total,   772028k used,     2996k free,    68468k buffers
Swap:  1048568k total,      176k used,  1048392k free,   441172k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 4624 root      15   0 40192  18m 7228 S 28.4  2.4   1:23.21 X
 4926 mhideo    15   0 55564  33m 9784 S 13.5  4.4   0:25.96 gnome-terminal
 6475 mhideo    16   0  3612  968  760 R  0.7  0.1   0:00.11 top
 4920 mhideo    15   0 20872  10m 7808 S  0.3  1.4   0:01.61 wnck-applet
    1 root      16   0  1732  548  472 S  0.0  0.1   0:00.23 init
    2 root      34  19     0    0    0 S  0.0  0.0   0:00.00 ksoftirqd/0
    3 root       5 -10     0    0    0 S  0.0  0.0   0:00.03 events/0
    4 root       6 -10     0    0    0 S  0.0  0.0   0:00.02 khelper
    5 root       5 -10     0    0    0 S  0.0  0.0   0:00.00 kacpid
   29 root       5 -10     0    0    0 S  0.0  0.0   0:00.00 kblockd/0
   47 root      16   0     0    0    0 S  0.0  0.0   0:01.74 pdflush
   50 root      11 -10     0    0    0 S  0.0  0.0   0:00.00 aio/0
   30 root      15   0     0    0    0 S  0.0  0.0   0:00.05 khubd
   49 root      16   0     0    0    0 S  0.0  0.0   0:01.44 kswapd0
```

To exit **`top`**, press the **`q`** key.

Table 42.1, “Interactive `top` commands” contains useful interactive commands that you can use with **`top`**. For more information, refer to the **`top`**(1) manual page.

Table 42.1. Interactive `top` commands

Command	Description
`Space`	Immediately refresh the display
`h`	Display a help screen
`k`	Kill a process. You are prompted for the process ID and the signal to send to it.
`n`	Change the number of processes displayed. You are prompted to enter the number.
`u`	Sort by user.
`M`	Sort by memory usage.
`P`	Sort by CPU usage.

If you prefer a graphical interface for **`top`**, you can use the **GNOME System Monitor**. To start it from the desktop, select **System** > **Administration** > **System Monitor** or type **`gnome-system-monitor`** at a shell prompt (such as an XTerm). Select the **`Process Listing`** tab.

The **GNOME System Monitor** allows you to search for a process in the list of running processes. Using the Gnome System Monitor, you can also view all processes, your processes, or active processes.

The **Edit** menu item allows you to:

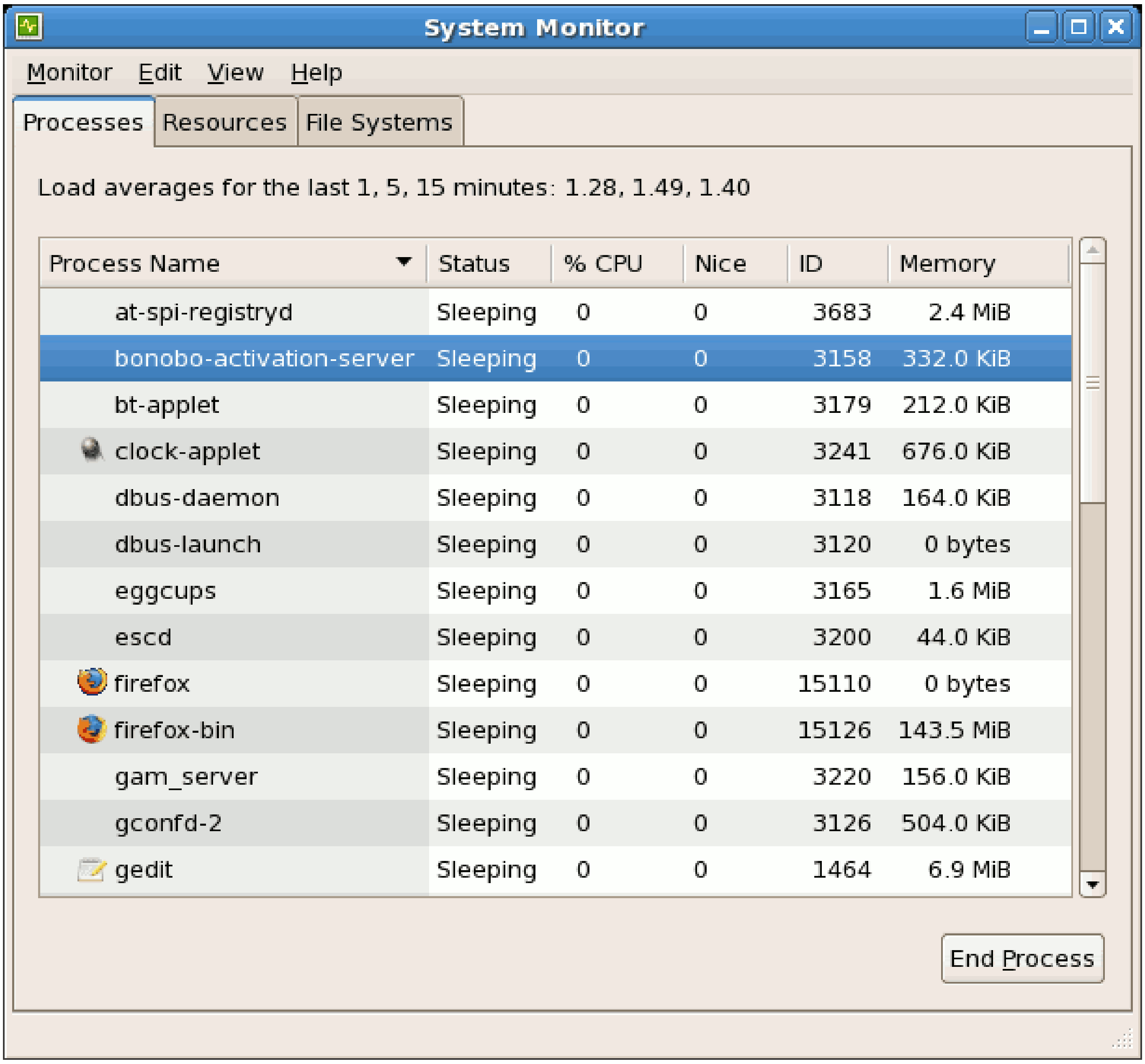
- Stop a process.
- Continue or start a process.
- End a processes.
- Kill a process.
- Change the priority of a selected process.
- Edit the System Monitor preferences. These include changing the interval seconds to refresh the list and selecting process fields to display in the System Monitor window.

The **View** menu item allows you to:

- View only active processes.
- View all processes.
- View my processes.
- View process dependencies.
- Hide a process.
- View hidden processes.
- View memory maps.
- View the files opened by the selected process.

To stop a process, select it and click **`End Process`**. Alternatively you can also stop a process by selecting it, clicking **Edit** on your menu and selecting **Stop Process**.

To sort the information by a specific column, click on the name of the column. This sorts the information by the selected column in ascending order. Click on the name of the column again to toggle the sort between ascending and descending order.



Load averages for the last 1, 5, 15 minutes: 1.28, 1.49, 1.40

Process Name	Status	% CPU	Nice	ID	Memory
at-spi-registryd	Sleeping	0	0	3683	2.4 MiB
bonobo-activation-server	Sleeping	0	0	3158	332.0 KiB
bt-applet	Sleeping	0	0	3179	212.0 KiB
clock-applet	Sleeping	0	0	3241	676.0 KiB
dbus-daemon	Sleeping	0	0	3118	164.0 KiB
dbus-launch	Sleeping	0	0	3120	0 bytes
eggcups	Sleeping	0	0	3165	1.6 MiB
escd	Sleeping	0	0	3200	44.0 KiB
firefox	Sleeping	0	0	15110	0 bytes
firefox-bin	Sleeping	0	0	15126	143.5 MiB
gam_server	Sleeping	0	0	3220	156.0 KiB
gconfd-2	Sleeping	0	0	3126	504.0 KiB
gedit	Sleeping	0	0	1464	6.9 MiB

Figure 42.1. GNOME System Monitor

42.2. Memory Usage

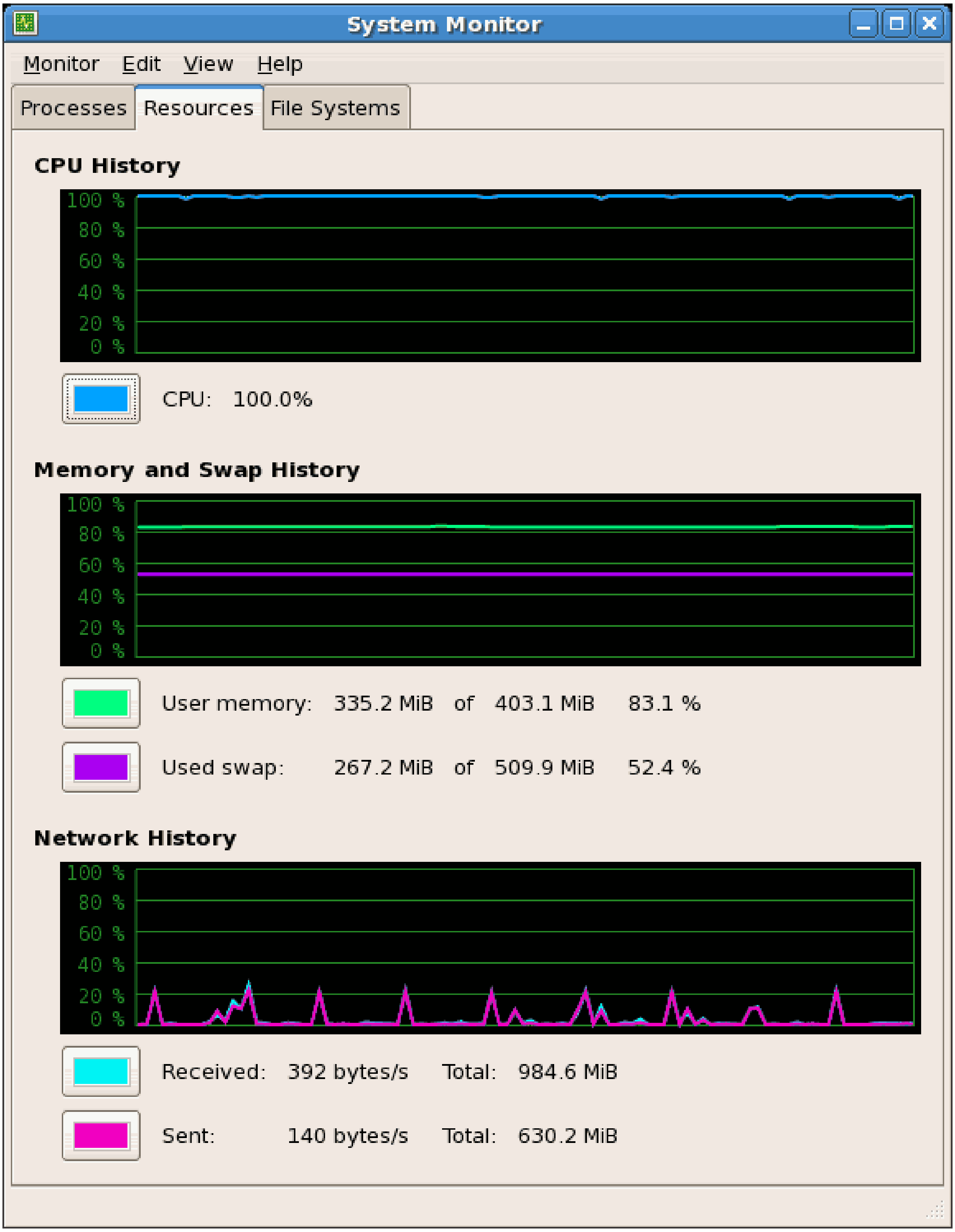
The **`free`** command displays the total amount of physical memory and swap space for the system as well as the amount of memory that is used, free, shared, in kernel buffers, and cached.

```
            total       used       free     shared    buffers     cached
Mem:       645712     549720      95992          0     176248     224452
-/+ buffers/cache:     149020     496692
Swap:     1310712          0    1310712
```

The command **`free -m`** shows the same information in megabytes, which are easier to read.

```
             total       used       free     shared    buffers     cached
Mem:           630        536         93          0        172        219
-/+ buffers/cache:        145        485
Swap:         1279          0       1279
```

If you prefer a graphical interface for **`free`**, you can use the **GNOME System Monitor**. To start it from the desktop, go to **System** > **Administration** > **System Monitor** or type **`gnome-system-monitor`** at a shell prompt (such as an XTerm). Click on the **`Resources`** tab.



CPU: 100.0%

User memory: 335.2 MiB of 403.1 MiB 83.1 %

Used swap: 267.2 MiB of 509.9 MiB 52.4 %

Received: 392 bytes/s Total: 984.6 MiB

Sent: 140 bytes/s Total: 630.2 MiB

Figure 42.2. GNOME System Monitor - Resources tab

42.3. File Systems

The **`df`** command reports the system's disk space usage. If you type the command **`df`** at a shell prompt, the output looks similar to the following:

```
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol00
                       11675568   6272120   4810348  57% / /dev/sda1
                  100691      9281     86211  10% /boot
none                     322856         0    322856   0% /dev/shm
```

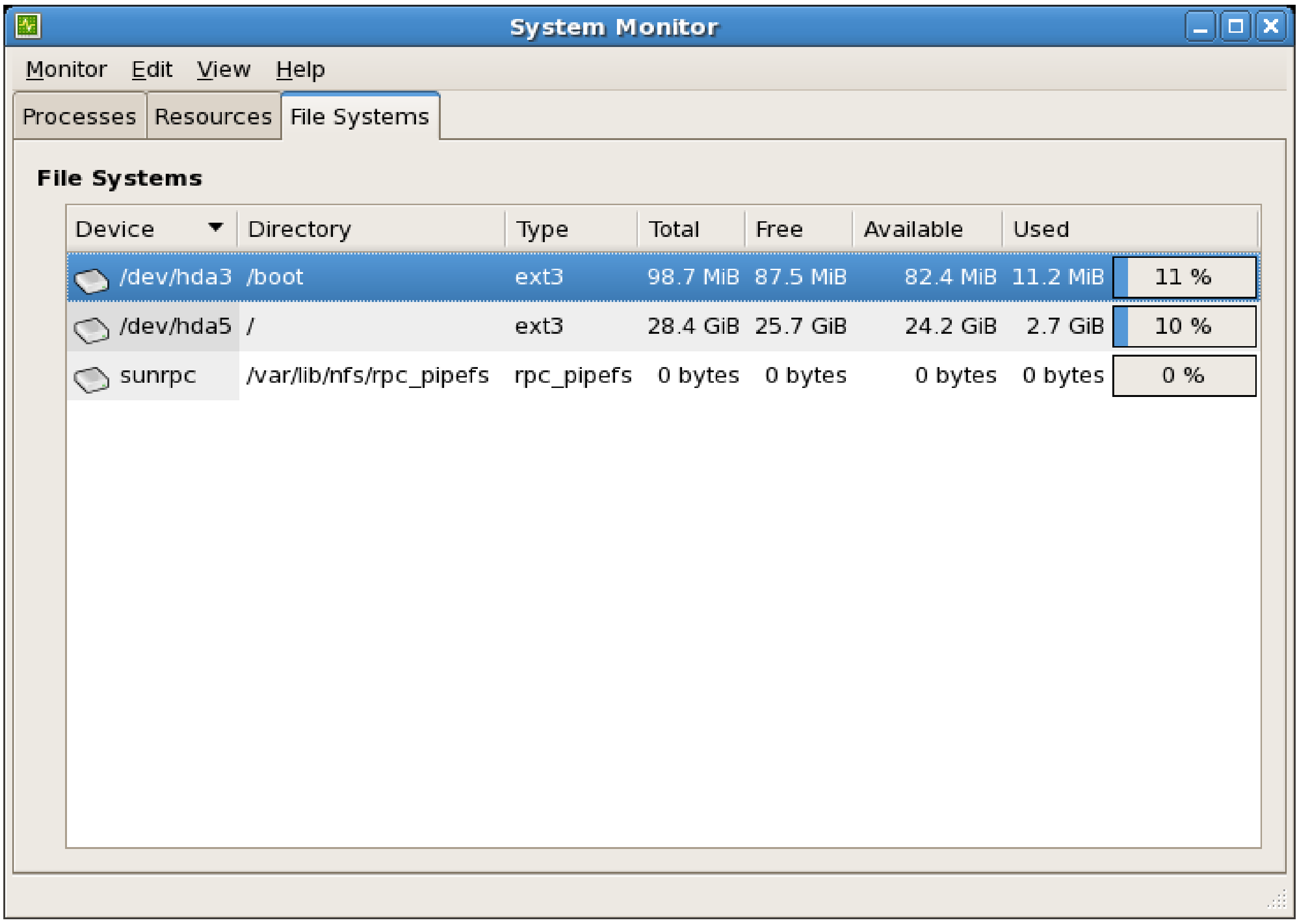
By default, this utility shows the partition size in 1 kilobyte blocks and the amount of used and available disk space in kilobytes. To view the information in megabytes and gigabytes, use the command **`df -h`**. The **`-h`** argument stands for human-readable format. The output looks similar to the following:

```
Filesystem            Size  Used Avail Use% Mounted on
/dev/mapper/VolGroup00-LogVol00
                        12G  6.0G  4.6G  57% / /dev/sda1
   99M  9.1M   85M  10% /boot
none    316M     0  316M   0% /dev/shm
```

In the list of mounted partitions, there is an entry for **`/dev/shm`**. This entry represents the system's virtual memory file system.

The **`du`** command displays the estimated amount of space being used by files in a directory. If you type **`du`** at a shell prompt, the disk usage for each of the subdirectories is displayed in a list. The grand total for the current directory and subdirectories are also shown as the last line in the list. If you do not want to see the totals for all the subdirectories, use the command **`du -hs`** to see only the grand total for the directory in human-readable format. Use the **`du --help`** command to see more options.

To view the system's partitions and disk space usage in a graphical format, use the **`Gnome System Monitor`** by clicking on **System** > **Administration** > **System Monitor** or type **`gnome-system-monitor`** at a shell prompt (such as an XTerm). Select the File Systems tab to view the system's partitions. The figure below illustrates the File Systems tab.

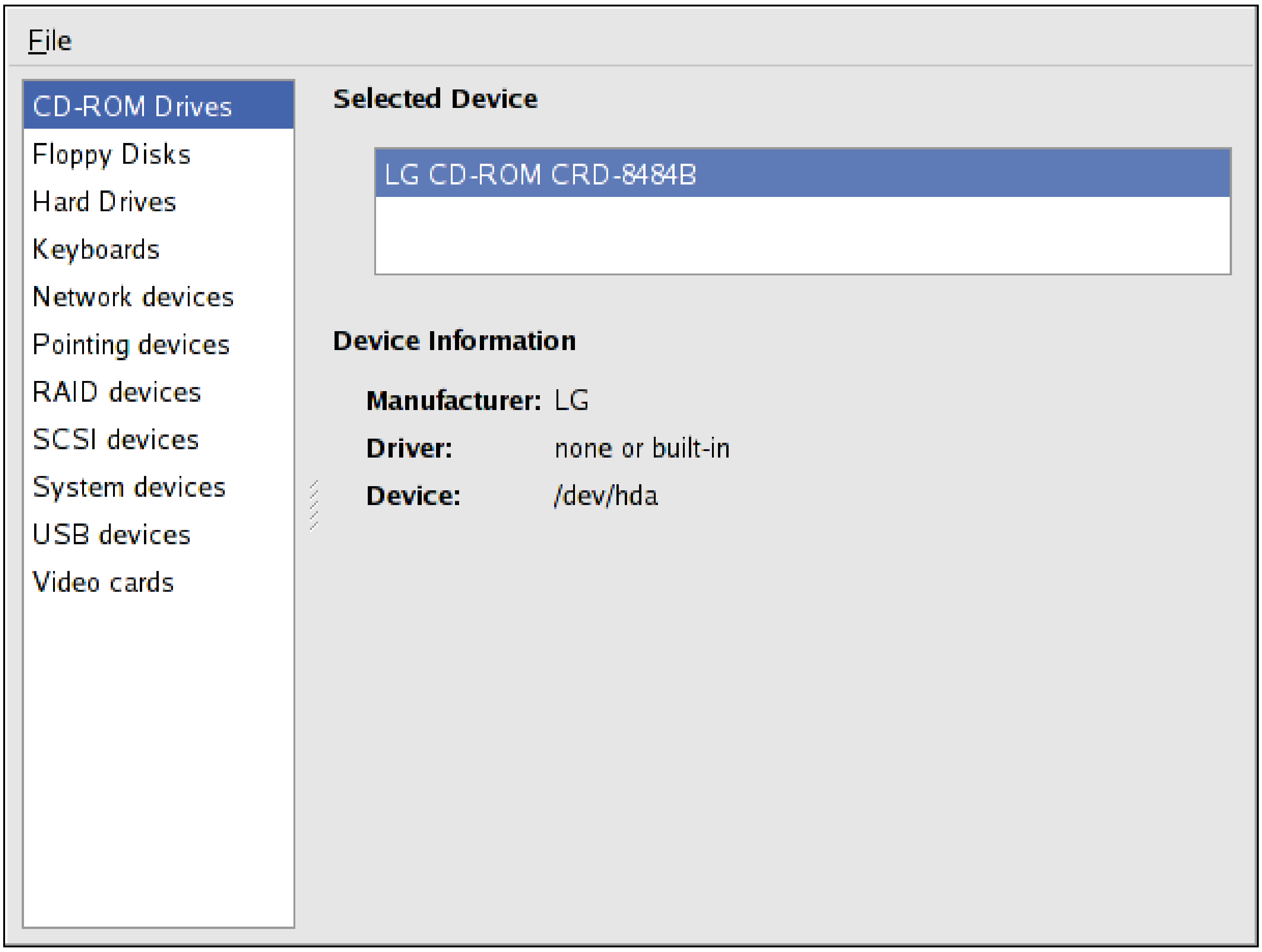


Device	Directory	Type	Total	Free	Available	Used	
/dev/hda3	/boot	ext3	98.7 MiB	87.5 MiB	82.4 MiB	11.2 MiB	11 %
/dev/hda5	/	ext3	28.4 GiB	25.7 GiB	24.2 GiB	2.7 GiB	10 %
sunrpc	/var/lib/nfs/rpc_pipefs	rpc_pipefs	0 bytes	0 bytes	0 bytes	0 bytes	0 %

Figure 42.3. GNOME System Monitor - File Systems

42.4. Hardware

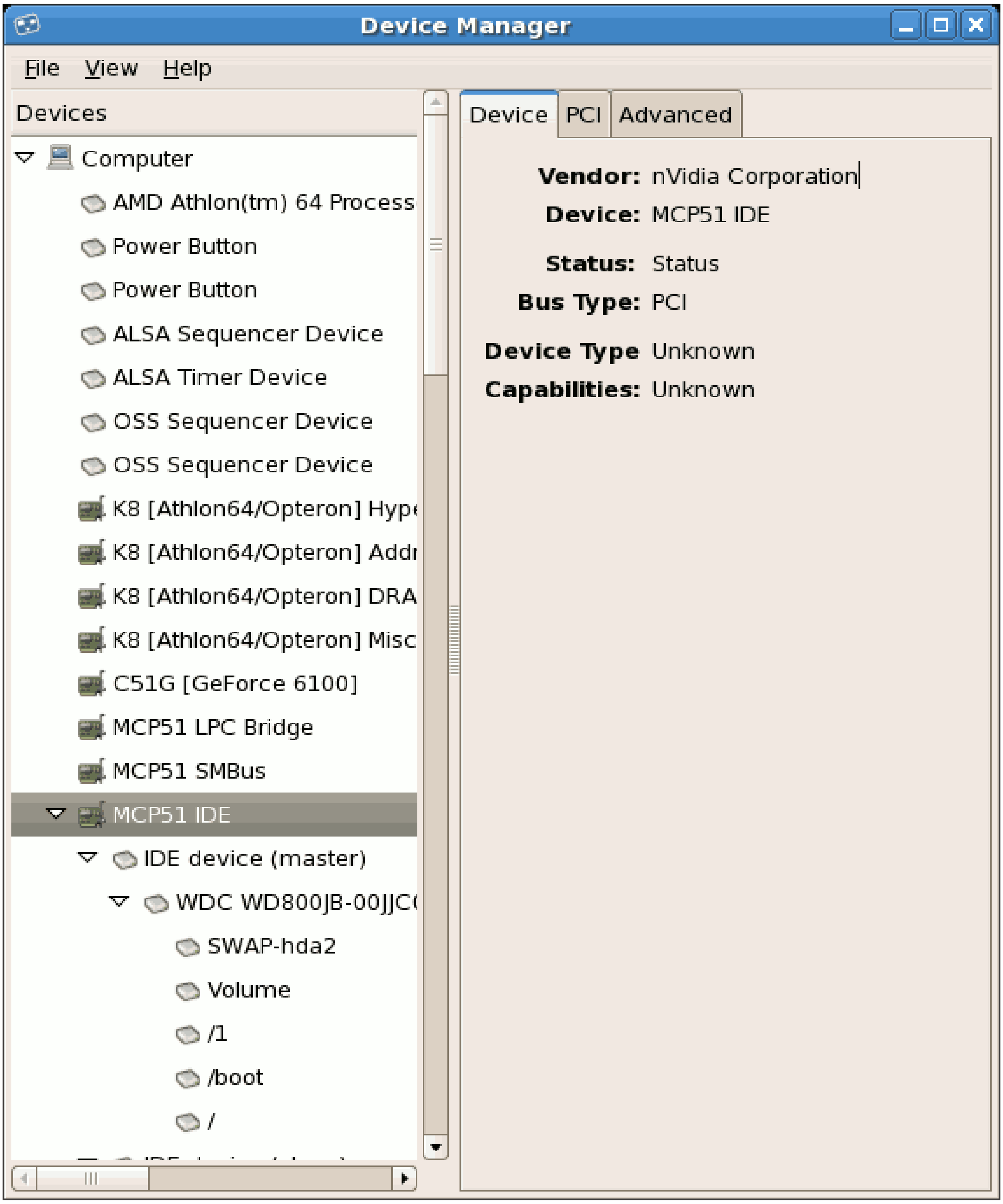
If you are having trouble configuring your hardware or just want to know what hardware is in your system, you can use the **Hardware Browser** application to display the hardware that can be probed. To start the program from the desktop, select **System** (the main menu on the panel) > **Administration** > **Hardware** or type **`hwbrowser`** at a shell prompt. As shown in Figure 42.4, “Hardware Browser”, it displays your CD-ROM devices, diskette drives, hard drives and their partitions, network devices, pointing devices, system devices, and video cards. Click on the category name in the left menu, and the information is displayed.



Manufacturer: LG
Driver: none or built-in
Device: /dev/hda

Figure 42.4. Hardware Browser

The **Device Manager** application can also be used to display your system hardware. This application can be started by selecting **System** (the main menu on the panel) > **Administration** > **Hardware** like the **Hardware Browser**. To start the application from a terminal, type **`hal-device-manager`**. Depending on your installation preferences, the graphical menu above may start this application or the **Hardware Browser** when clicked. The figure below illustrates the **Device Manager** window.



Vendor: nVidia Corporation
Device: MCP51 IDE
Status: Status
Bus Type: PCI
Device Type Unknown
Capabilities: Unknown

Figure 42.5. Device Manager

You can also use the **`lspci`** command to list all PCI devices. Use the command **`lspci -v`** for more verbose information or **`lspci -vv`** for very verbose output.

For example, **`lspci`** can be used to determine the manufacturer, model, and memory size of a system's video card:

```
00:00.0 Host bridge: ServerWorks CNB20LE Host Bridge (rev 06)
00:00.1 Host bridge: ServerWorks CNB20LE Host Bridge (rev 06)
00:01.0 VGA compatible controller: S3 Inc. Savage 4 (rev 04)
00:02.0 Ethernet controller: Intel Corp. 82557/8/9 [Ethernet Pro 100] (rev
08)
00:0f.0 ISA bridge: ServerWorks OSB4 South Bridge (rev 50)
00:0f.1 IDE interface: ServerWorks OSB4 IDE Controller
00:0f.2 USB Controller: ServerWorks OSB4/CSB5 OHCI USB Controller (rev 04)
01:03.0 SCSI storage controller: Adaptec AIC-7892P U160/m (rev 02)
01:05.0 RAID bus controller: IBM ServeRAID Controller
```

The **`lspci`** is also useful to determine the network card in your system if you do not know the manufacturer or model number.

42.5. Additional Resources

To learn more about gathering system information, refer to the following resources.

42.5.1. Installed Documentation

- **`ps --help`** — Displays a list of options that can be used with **`ps`**.
- **`top`** manual page — Type **`man top`** to learn more about **`top`** and its many options.
- **`free`** manual page — type **`man free`** to learn more about **`free`** and its many options.
- **`df`** manual page — Type **`man df`** to learn more about the **`df`** command and its many options.
- **`du`** manual page — Type **`man du`** to learn more about the **`du`** command and its many options.
- **`lspci`** manual page — Type **`man lspci`** to learn more about the **`lspci`** command and its many options.
- **`/proc/`** directory — The contents of the **`/proc/`** directory can also be used to gather more detailed system information.

Chapter 43. OProfile

OProfile is a low overhead, system-wide performance monitoring tool. It uses the performance monitoring hardware on the processor to retrieve information about the kernel and executables on the system, such as when memory is referenced, the number of L2 cache requests, and the number of hardware interrupts received. On a Red Hat Enterprise Linux system, the **`oprofile`** RPM package must be installed to use this tool.

Many processors include dedicated performance monitoring hardware. This hardware makes it possible to detect when certain events happen (such as the requested data not being in cache). The hardware normally takes the form of one or more *counters* that are incremented each time an event takes place. When the counter value, essentially rolls over, an interrupt is generated, making it possible to control the amount of detail (and therefore, overhead) produced by performance monitoring.

OProfile uses this hardware (or a timer-based substitute in cases where performance monitoring hardware is not present) to collect *samples* of performance-related data each time a counter generates an interrupt. These samples are periodically written out to disk; later, the data contained in these samples can then be used to generate reports on system-level and application-level performance.

OProfile is a useful tool, but be aware of some limitations when using it:

- *Use of shared libraries* — Samples for code in shared libraries are not attributed to the particular application unless the **`--separate=library`** option is used.
- *Performance monitoring samples are inexact* — When a performance monitoring register triggers a sample, the interrupt handling is not precise like a divide by zero exception. Due to the out-of-order execution of instructions by the processor, the sample may be recorded on a nearby instruction.
- ***`opreport`** does not associate samples for inline functions' properly* — **`opreport`** uses a simple address range mechanism to determine which function an address is in. Inline function samples are not attributed to the inline function but rather to the function the inline function was inserted into.
- *OProfile accumulates data from multiple runs* — OProfile is a system-wide profiler and expects processes to start up and shut down multiple times. Thus, samples from multiple runs accumulate. Use the command **`opcontrol --reset`** to clear out the samples from previous runs.
- *Non-CPU-limited performance problems* — OProfile is oriented to finding problems with CPU-limited processes. OProfile does not identify processes that are asleep because they are waiting on locks or for some other event to occur (for example an I/O device to finish an operation).

43.1. Overview of Tools

Table 43.1, “OProfile Commands” provides a brief overview of the tools provided with the **`oprofile`** package.

Table 43.1. OProfile Commands

Command	Description
`ophelp`	Displays available events for the system's processor along with a brief description of each.
`opimport`	Converts sample database files from a foreign binary format to the native format for the system. Only use this option when analyzing a sample database from a different architecture.

Command	Description
`opannotate`	Creates annotated source for an executable if the application was compiled with debugging symbols. Refer to Section 43.5.4, "Using **`opannotate`**" for details.
`opcontrol`	Configures what data is collected. Refer to Section 43.2, "Configuring OProfile" for details.
`opreport`	Retrieves profile data. Refer to Section 43.5.1, "Using **`opreport`**" for details.
`oprofiled`	Runs as a daemon to periodically write sample data to disk.

43.2. Configuring OProfile

Before OProfile can be run, it must be configured. At a minimum, selecting to monitor the kernel (or selecting not to monitor the kernel) is required. The following sections describe how to use the **`opcontrol`** utility to configure OProfile. As the **`opcontrol`** commands are executed, the setup options are saved to the **`/root/.oprofile/daemonrc`** file.

43.2.1. Specifying the Kernel

First, configure whether OProfile should monitor the kernel. This is the only configuration option that is required before starting OProfile. All others are optional.

To monitor the kernel, execute the following command as root:

```
opcontrol --setup --vmlinux=/usr/lib/debug/lib/modules/`uname -r`/vmlinux
```

Note

The **`debuginfo`** package must be installed (which contains the uncompressed kernel) in order to monitor the kernel.

To configure OProfile not to monitor the kernel, execute the following command as root:

```
opcontrol --setup --no-vmlinux
```

This command also loads the **`oprofile`** kernel module, if it is not already loaded, and creates the **`/dev/oprofile/`** directory, if it does not already exist. Refer to Section 43.6, "Understanding `/dev/oprofile/`" for details about this directory.

Note

Even if OProfile is configured not to profile the kernel, the SMP kernel still must be running so that the **`oprofile`** module can be loaded from it.

Setting whether samples should be collected within the kernel only changes what data is collected, not how or where the collected data is stored. To generate different sample files for the kernel and application libraries, refer to Section 43.2.3, "Separating Kernel and User-space Profiles".

43.2.2. Setting Events to Monitor

Most processors contain *counters*, which are used by OProfile to monitor specific events. As shown in Table 43.2, "OProfile Processors and Counters", the number of counters available depends on the processor.

Table 43.2. OProfile Processors and Counters

Processor	`cpu_type`	Number of Counters
Pentium Pro	i386/ppro	2
Pentium II	i386/pii	2
Pentium III	i386/piii	2
Pentium 4 (non-hyper-threaded)	i386/p4	8
Pentium 4 (hyper-threaded)	i386/p4-ht	4
Athlon	i386/athlon	4
AMD64	x86-64/hammer	4
Itanium	ia64/itanium	4
Itanium 2	ia64/itanium2	4
TIMER_INT	timer	1
IBM eServer iSeries and pSeries	timer	1
	ppc64/power4	8
	ppc64/power5	6
	ppc64/970	8
IBM eServer S/390 and S/390x	timer	1
IBM eServer zSeries	timer	1

Use Table 43.2, "OProfile Processors and Counters" to verify that the correct processor type was detected and to determine the number of events that can be monitored simultaneously. **`timer`** is used as the processor type if the processor does not have supported performance monitoring hardware.

If **`timer`** is used, events cannot be set for any processor because the hardware does not have support for hardware performance counters. Instead, the timer interrupt is used for profiling.

If **`timer`** is not used as the processor type, the events monitored can be changed, and counter 0 for the processor is set to a time-based event by default. If more than one counter exists on the processor, the counters other than counter 0 are not set to an event by default. The default events monitored are shown in Table 43.3, "Default Events".

Table 43.3. Default Events

Processor	Default Event for Counter	Description
Pentium Pro, Pentium II, Pentium III, Athlon, AMD64	CPU_CLK_UNHALTED	The processor's clock is not halted
Pentium 4 (HT and non-HT)	GLOBAL_POWER_EVENTS	The time during which the processor is not stopped
Itanium 2	CPU_CYCLES	CPU Cycles
TIMER_INT	(none)	Sample for each timer interrupt
ppc64/power4	CYCLES	Processor Cycles
ppc64/power5	CYCLES	Processor Cycles
ppc64/970	CYCLES	Processor Cycles

The number of events that can be monitored at one time is determined by the number of counters for the processor. However, it is not a one-to-one correlation; on some processors, certain events must be mapped to specific counters. To determine the number of counters available, execute the following command:

```
ls -d /dev/oprofile/[0-9]*
```

The events available vary depending on the processor type. To determine the events available for profiling, execute the following command as root (the list is specific to the system's processor type):

```
ophelp
```

The events for each counter can be configured via the command line or with a graphical interface. For more information on the graphical interface, refer to Section 43.8, “Graphical Interface”. If the counter cannot be set to a specific event, an error message is displayed.

To set the event for each configurable counter via the command line, use **opcontrol**:

```
opcontrol --event=<event-name>:<sample-rate>
```

Replace <*event-name*> with the exact name of the event from **ophelp**, and replace <*sample-rate*> with the number of events between samples.

43.2.2.1. Sampling Rate

By default, a time-based event set is selected. It creates a sample every 100,000 clock cycles per processor. If the timer interrupt is used, the timer is set to whatever the jiffy rate is and is not user-settable. If the **cpu_type** is not **timer**, each event can have a *sampling rate* set for it. The sampling rate is the number of events between each sample snapshot.

When setting the event for the counter, a sample rate can also be specified:

```
opcontrol --event=<event-name>:<sample-rate>
```

Replace <*sample-rate*> with the number of events to wait before sampling again. The smaller the count, the more frequent the samples. For events that do not happen frequently, a lower count may be needed to capture the event instances.

Warning

Be extremely careful when setting sampling rates. Sampling too frequently can overload the system, causing the system to appear as if it is frozen or causing the system to actually freeze.

43.2.2.2. Unit Masks

Some user performance monitoring events may also require unit masks to further define the event.

Unit masks for each event are listed with the **ophelp** command. The values for each unit mask are listed in hexadecimal format. To specify more than one unit mask, the hexadecimal values must be combined using a bitwise *or* operation.

```
opcontrol --event=<event-name>:<sample-rate>:<unit-mask>
```

43.2.3. Separating Kernel and User-space Profiles

By default, kernel mode and user mode information is gathered for each event. To configure OProfile to ignore events in kernel mode for a specific counter, execute the following command:

```
opcontrol --event=<event-name>:<sample-rate>:<unit-mask>:0
```

Execute the following command to start profiling kernel mode for the counter again:

```
opcontrol --event=<event-name>:<sample-rate>:<unit-mask>:1
```

To configure OProfile to ignore events in user mode for a specific counter, execute the following command:

```
opcontrol --event=<event-name>:<sample-rate>:<unit-mask>:<kernel>:0
```

Execute the following command to start profiling user mode for the counter again:

```
opcontrol --event=<event-name>:<sample-rate>:<unit-mask>:<kernel>:1
```

When the OProfile daemon writes the profile data to sample files, it can separate the kernel and library profile data into separate sample files. To configure how the daemon writes to sample files, execute the following command as root:

```
opcontrol --separate=<choice>
```

<*choice*> can be one of the following:

- **none** — do not separate the profiles (default)
- **`library`** — generate per-application profiles for libraries
- **`kernel`** — generate per-application profiles for the kernel and kernel modules
- **`all`** — generate per-application profiles for libraries and per-application profiles for the kernel and kernel modules

If **`--separate=library`** is used, the sample file name includes the name of the executable as well as the name of the library.

Note

These configuration changes will take effect when **`oprofile`** is restarted.

43.3. Starting and Stopping OProfile

To start monitoring the system with OProfile, execute the following command as root:

```
opcontrol --start
```

Output similar to the following is displayed:

```
Using log file /var/lib/oprofile/oprofiled.log Daemon started. Profiler
running.
```

The settings in **`/root/.oprofile/daemonrc`** are used.

The OProfile daemon, **`oprofiled`**, is started; it periodically writes the sample data to the **`/var/lib/oprofile/samples/`** directory. The log file for the daemon is located at **`/var/lib/oprofile/oprofiled.log`**.

To stop the profiler, execute the following command as root:

```
opcontrol --shutdown
```

43.4. Saving Data

Sometimes it is useful to save samples at a specific time. For example, when profiling an executable, it may be useful to gather different samples based on different input data sets. If the number of events to be monitored exceeds the number of counters available for the processor, multiple runs of OProfile can be used to collect data, saving the sample data to different files each time.

To save the current set of sample files, execute the following command, replacing <*name*> with a unique descriptive name for the current session.

```
opcontrol --save=<name>
```

The directory **`/var/lib/oprofile/samples/`*`name`*`/`** is created and the current sample files are copied to it.

43.5. Analyzing the Data

Periodically, the OProfile daemon, **`oprofiled`**, collects the samples and writes them to the **`/var/lib/oprofile/samples/`** directory. Before reading the data, make sure all data has been written to this directory by executing the following command as root:

```
opcontrol --dump
```

Each sample file name is based on the name of the executable. For example, the samples for the default event on a Pentium III processor for **`/bin/bash`** becomes:

```
\{root\}/bin/bash/\{dep\}/\{root\}/bin/bash/CPU_CLK_UNHALTED.100000
```

The following tools are available to profile the sample data once it has been collected:

- **`opreport`**
- **`opannotate`**

Use these tools, along with the binaries profiled, to generate reports that can be further analyzed.

Warning

The executable being profiled must be used with these tools to analyze the data. If it must change after the data is collected, backup the executable used to create the samples as well as the sample files. Please note that the sample file and the binary have to agree. Making a backup isn't going to work if they do not match. **`oparchive`** can be used to address this problem.

Samples for each executable are written to a single sample file. Samples from each dynamically linked library are also written to a single sample file. While OProfile is running, if the executable being monitored changes and a sample file for the executable exists, the existing sample file is automatically deleted. Thus, if the existing sample file is needed, it must be backed up, along with the executable used to create it before replacing the executable with a new version. The oprofile analysis tools use the executable file that created the samples during analysis. If the executable changes the analysis tools will be unable to analyze the associated samples. Refer to Section 43.4, “Saving Data” for details on how to backup the sample file.

43.5.1. Using `opreport`

The **`opreport`** tool provides an overview of all the executables being profiled.

The following is part of a sample output:

```
Profiling through timer interrupt
TIMER:0|
samples|      %|
------------------
25926 97.5212 no-vmlinux
359  1.3504 pi
65  0.2445 Xorg
62  0.2332 libvte.so.4.4.0
56  0.2106 libc-2.3.4.so
34  0.1279 libglib-2.0.so.0.400.7
19  0.0715 libXft.so.2.1.2
17  0.0639 bash
8  0.0301 ld-2.3.4.so
8  0.0301 libgdk-x11-2.0.so.0.400.13
6  0.0226 libgobject-2.0.so.0.400.7
5  0.0188 oprofiled
4  0.0150 libpthread-2.3.4.so
4  0.0150 libgtk-x11-2.0.so.0.400.13
3  0.0113 libXrender.so.1.2.2
3  0.0113 du
1  0.0038 libcrypto.so.0.9.7a
1  0.0038 libpam.so.0.77
1  0.0038 libtermcap.so.2.0.8
1  0.0038 libX11.so.6.2
1  0.0038 libgthread-2.0.so.0.400.7
1  0.0038 libwnck-1.so.4.9.0
```

Each executable is listed on its own line. The first column is the number of samples recorded for the executable. The second column is the percentage of samples relative to the total number of samples. The third column is the name of the executable.

Refer to the **`opreport`** man page for a list of available command line options, such as the **`-r`** option used to sort the output from the executable with the smallest number of samples to the one with the largest number of samples.

43.5.2. Using `opreport` on a Single Executable

To retrieve more detailed profiled information about a specific executable, use **`opreport`**:

```
opreport <mode> <executable>
```

<executable> must be the full path to the executable to be analyzed. *<mode>* must be one of the following:

`-l`

List sample data by symbols. For example, the following is part of the output from running the command **`opreport -l /lib/tls/libc-<version>.so`**:

```
samples  %        symbol name
12       21.4286  __gconv_transform_utf8_internal
5         8.9286  _int_malloc
4         7.1429  malloc
3         5.3571  __i686.get_pc_thunk.bx
3         5.3571  _dl_mcount_wrapper_check
3         5.3571  mbrtowc
3         5.3571  memcpy
2         3.5714  _int_realloc
2         3.5714  _nl_intern_locale_data
2         3.5714  free
2         3.5714  strcmp
1         1.7857  __ctype_get_mb_cur_max
1         1.7857  __unregister_atfork
1         1.7857  __write_nocancel
1         1.7857  _dl_addr
1         1.7857  _int_free
1         1.7857  _itoa_word
1         1.7857  calc_eclosure_iter
1         1.7857  fopen@@GLIBC_2.1
1         1.7857  getpid
1         1.7857  memmove
1         1.7857  msort_with_tmp
1         1.7857  strcpy
1         1.7857  strlen
1         1.7857  vfprintf
1         1.7857  write
```

The first column is the number of samples for the symbol, the second column is the percentage of samples for this symbol relative to the overall samples for the executable, and the third column is the symbol name.

To sort the output from the largest number of samples to the smallest (reverse order), use **`-r`** in conjunction with the **`-l`** option.

`-i <symbol-name>`

List sample data specific to a symbol name. For example, the following output is from the command **`opreport -l -i __gconv_transform_utf8_internal /lib/tls/libc-<version>.so`**:

```
samples  %        symbol name
12       100.000  __gconv_transform_utf8_internal
```

The first line is a summary for the symbol/executable combination.

The first column is the number of samples for the memory symbol. The second column is the percentage of samples for the memory address relative to the total number of samples for the symbol. The third column is the symbol name.

`-d`

List sample data by symbols with more detail than **`-l`**. For example, the following output is from the command **`opreport -l -d __gconv_transform_utf8_internal /lib/tls/libc-<version>.so`**:

```
vma      samples  %        symbol name
00a98640 12       100.000  __gconv_transform_utf8_internal
00a98640 1         8.3333
00a9868c 2        16.6667
00a9869a 1         8.3333
00a986c1 1         8.3333
00a98720 1         8.3333
00a98749 1         8.3333
00a98753 1         8.3333
00a98789 1         8.3333
00a98864 1         8.3333
00a98869 1         8.3333
00a98b08 1         8.3333
```

The data is the same as the **`-l`** option except that for each symbol, each virtual memory address used is shown. For each virtual memory address, the number of samples and percentage of samples relative to the number of samples for the symbol is displayed.

`-x`*<symbol-name>*

Exclude the comma-separated list of symbols from the output.

`session`:*<name>*

Specify the full path to the session or a directory relative to the **`/var/lib/oprofile/samples/`** directory.

43.5.3. Getting more detailed output on the modules

OProfile collects data on a system-wide basis for kernel- and user-space code running on the machine. However, once a module is loaded into the kernel, the information about the origin of the kernel module is lost. The module could have come from the initrd file on boot up, the directory with the various kernel modules, or a locally created kernel module. As a result when OProfile records sample for a module, it just lists the samples for the modules for an executable in the root directory, but this is unlikely to be the place with the actual code for the module. You will need to take some steps to make sure that analysis tools get the executable.

For example on an AMD64 machine the sampling is set up to record "Data cache accesses" and "Data cache misses" and assuming you would like to see the data for the ext3 module:

```
~]$ opreport /ext3
```

```
CPU: AMD64 processors, speed 797.948 MHz (estimated)
Counted DATA_CACHE_ACCESSES events (Data cache accesses) with a unit mask of
0x00 (No unit mask) count 500000
Counted DATA_CACHE_MISSES events (Data cache misses) with a unit mask of
0x00 (No unit mask) count 500000
DATA_CACHE_ACC...|DATA_CACHE_MIS...|
samples|      %|  samples|      %|
------------------------------------
148721 100.000      1493 100.000 ext3
```

To get a more detailed view of the actions of the module, you will need to either have the module unstripped (e.g. installed from a custom build) or have the debuginfo RPM installed for the kernel.

Find out which kernel is running, "uname -a", get the appropriate debuginfo rpm, and install on the machine.

Then make a symbolic link so oprofile finds the code for the module in the correct place:

```
~]# ln -s /lib/modules/`uname -r`/kernel/fs/ext3/ext3.ko /ext3
```

Then the detailed information can be obtained with:

```
~]# opreport image:/ext3 -l|more
warning: could not check that the binary file /ext3 has not been modified
since the profile was taken. Results may be inaccurate.
CPU: AMD64 processors, speed 797.948 MHz (estimated)
Counted DATA_CACHE_ACCESSES events (Data cache accesses) with a unit mask of
0x00 (No unit mask) count 500000
Counted DATA_CACHE_MISSES events (Data cache misses) with a unit mask of
0x00 (No unit mask) count 500000
samples  %        samples  %        symbol name
16728    11.2479  7         0.4689  ext3_group_sparse
16454    11.0637  4         0.2679  ext3_count_free_blocks
14583     9.8056  51        3.4159  ext3_fill_super
8281      5.5681  129       8.6403  ext3_ioctl
7810      5.2514  62        4.1527  ext3_write_info
7286      4.8991  67        4.4876  ext3_ordered_writepage
6509      4.3767  130       8.7073  ext3_new_inode
6378      4.2886  156      10.4488  ext3_new_block
5932      3.9887  87        5.8272  ext3_xattr_block_list
...
```

43.5.4. Using `opannotate`

The **`opannotate`** tool tries to match the samples for particular instructions to the corresponding lines in the source code. The resulting files generated should have the samples for the lines at the left. It also puts in a comment at the beginning of each function listing the total samples for the function.

For this utility to work, the executable must be compiled with GCC's **`-g`** option. By default, Red Hat Enterprise Linux packages are not compiled with this option.

The general syntax for **`opannotate`** is as follows:

```
opannotate --search-dirs <src-dir> --source <executable>
```

The directory containing the source code and the executable to be analyzed must be specified. Refer to the **`opannotate`** man page for a list of additional command line options.

43.6. Understanding `/dev/oprofile/`

The **`/dev/oprofile/`** directory contains the file system for OProfile. Use the **`cat`** command to display the values of the virtual files in this file system. For example, the following command displays the type of processor OProfile detected:

```
cat /dev/oprofile/cpu_type
```

A directory exists in **`/dev/oprofile/`** for each counter. For example, if there are 2 counters, the directories **`/dev/oprofile/0/`** and **`dev/oprofile/1/`** exist.

Each directory for a counter contains the following files:

- **`count`** — The interval between samples.
- **`enabled`** — If 0, the counter is off and no samples are collected for it; if 1, the counter is on and samples are being collected for it.
- **`event`** — The event to monitor.
- **`kernel`** — If 0, samples are not collected for this counter event when the processor is in kernel-space; if 1, samples are collected even if the processor is in kernel-space.
- **`unit_mask`** — Defines which unit masks are enabled for the counter.
- **`user`** — If 0, samples are not collected for the counter event when the processor is in user-space; if 1, samples are collected even if the processor is in user-space.

The values of these files can be retrieved with the **`cat`** command. For example:

```
cat /dev/oprofile/0/count
```

43.7. Example Usage

While OProfile can be used by developers to analyze application performance, it can also be used by system administrators to perform system analysis. For example:

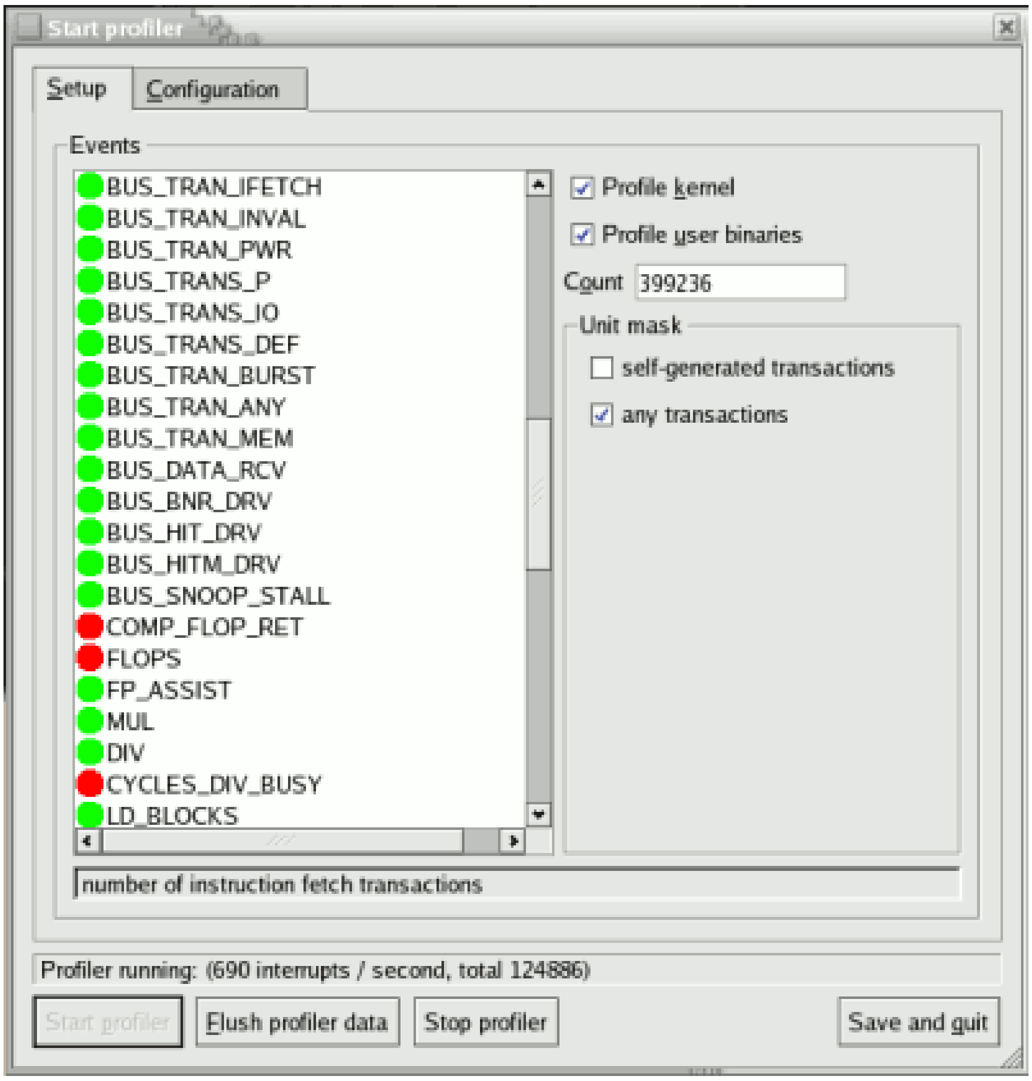
- *Determine which applications and services are used the most on a system* — **`opreport`** can be used to determine how much processor time an application or service uses. If the system is used for multiple services but is under performing, the services consuming the most processor time can be moved to dedicated systems.
- *Determine processor usage* — The **`CPU_CLK_UNHALTED`** event can be monitored to determine the processor load over a given period of time. This data can then be used to determine if additional processors or a faster processor might improve system performance.

43.8. Graphical Interface

Some OProfile preferences can be set with a graphical interface. To start it, execute the **`oprof_start`** command as root at a shell prompt. To use the graphical interface, you will need to have the **`oprofile-gui`** package installed.

After changing any of the options, save them by clicking the **`Save and quit`** button. The preferences are written to **`/root/.oprofile/daemonrc`**, and the application exits. Exiting the application does not stop OProfile from sampling.

On the **`Setup`** tab, to set events for the processor counters as discussed in Section 43.2.2, “Setting Events to Monitor”, select the counter from the pulldown menu and select the event from the list. A brief description of the event appears in the text box below the list. Only events available for the specific counter and the specific architecture are displayed. The interface also displays whether the profiler is running and some brief statistics about it.



🟢 BUS_TRAN_IFETCH
🟢 BUS_TRAN_INVAL
🟢 BUS_TRAN_PWR
🟢 BUS_TRANS_P
🟢 BUS_TRANS_IO
🟢 BUS_TRANS_DEF
🟢 BUS_TRAN_BURST
🟢 BUS_TRAN_ANY
🟢 BUS_TRAN_MEM
🟢 BUS_DATA_RCV
🟢 BUS_BNR_DRV
🟢 BUS_HIT_DRV
🟢 BUS_HITM_DRV
🟢 BUS_SNOOP_STALL
🔴 COMP_FLOP_RET
🔴 FLOPS
🟢 FP_ASSIST
🟢 MUL
🟢 DIV
🔴 CYCLES_DIV_BUSY
🟢 LD_BLOCKS

[✓] Profile kernel
[✓] Profile user binaries
Count 399236
[ ] self-generated transactions
[✓] any transactions

number of instruction fetch transactions

Profiler running: (690 interrupts / second, total 124886)

Figure 43.1. OProfile Setup

On the right side of the tab, select the **`Profile kernel`** option to count events in kernel mode for the currently selected event, as discussed in Section 43.2.3, “Separating Kernel and User-space Profiles”. If this option is unselected, no samples are collected for the kernel.

Select the **`Profile user binaries`** option to count events in user mode for the currently selected event, as discussed in Section 43.2.3, “Separating Kernel and User-space Profiles”. If this option is unselected, no samples are collected for user applications.

Use the **Count** text field to set the sampling rate for the currently selected event as discussed in Section 43.2.2.1, "Sampling Rate".

If any unit masks are available for the currently selected event, as discussed in Section 43.2.2.2, "Unit Masks", they are displayed in the **Unit Masks** area on the right side of the **Setup** tab. Select the checkbox beside the unit mask to enable it for the event.

On the **Configuration** tab, to profile the kernel, enter the name and location of the **vmlinux** file for the kernel to monitor in the **Kernel image file** text field. To configure OProfile not to monitor the kernel, select **No kernel image**.

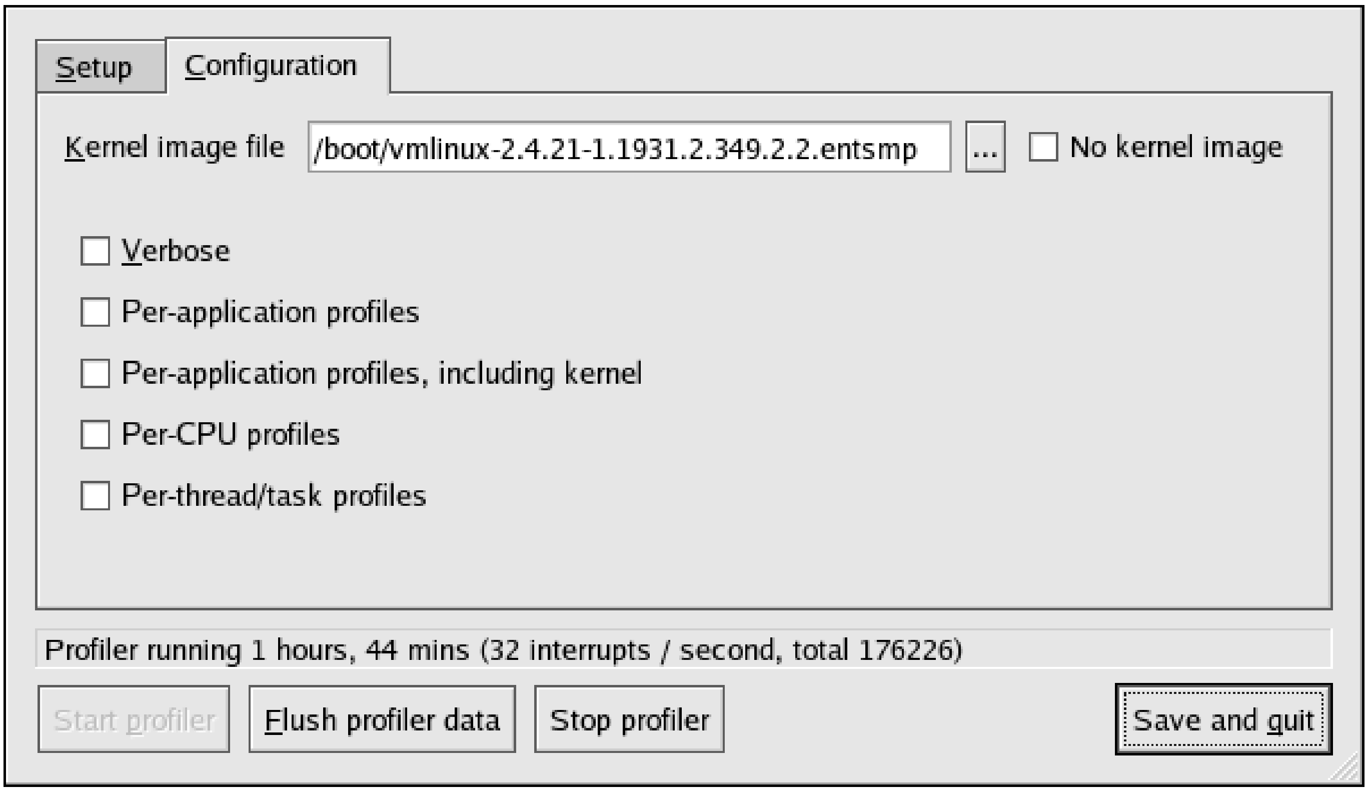

Setup
Configuration
Kernel image file: /boot/vmlinux-2.4.21-1.1931.2.349.2.2.entsmp
☐ No kernel image
☐ Verbose
☐ Per-application profiles
☐ Per-application profiles, including kernel
☐ Per-CPU profiles
☐ Per-thread/task profiles
Profiler running 1 hours, 44 mins (32 interrupts / second, total 176226)

Figure 43.2. OProfile Configuration

If the **Verbose** option is selected, the **oprofiled** daemon log includes more information.

If **Per-application kernel samples files** is selected, OProfile generates per-application profiles for the kernel and kernel modules as discussed in Section 43.2.3, "Separating Kernel and User-space Profiles". This is equivalent to the **opcontrol --separate=kernel** command. If **Per-application shared libs samples files** is selected, OProfile generates per-application profiles for libraries. This is equivalent to the **opcontrol --separate=library** command.

To force data to be written to samples files as discussed in Section 43.5, "Analyzing the Data", click the **Flush profiler data** button. This is equivalent to the **opcontrol --dump** command.

To start OProfile from the graphical interface, click **Start profiler**. To stop the profiler, click **Stop profiler**. Exiting the application does not stop OProfile from sampling.

43.9. Additional Resources

This chapter only highlights OProfile and how to configure and use it. To learn more, refer to the following resources.

43.9.1. Installed Docs

- **`/usr/share/doc/oprofile-<version>/oprofile.html`** — *OProfile Manual*
- **`oprofile`** man page — Discusses **`opcontrol`**, **`opreport`**, **`opannotate`**, and **`ophelp`**

43.9.2. Useful Websites

- http://oprofile.sourceforge.net/ — Contains the latest documentation, mailing lists, IRC channels, and more.

Part VI. Kernel and Driver Configuration

System administrators can learn about and customize their kernels. Red Hat Enterprise Linux contains kernel tools to assist administrators with their customizations.

Chapter 44. Manually Upgrading the Kernel

The Red Hat Enterprise Linux kernel is custom built by the Red Hat Enterprise Linux kernel team to ensure its integrity and compatibility with supported hardware. Before Red Hat releases a kernel, it must first pass a rigorous set of quality assurance tests.

Red Hat Enterprise Linux kernels are packaged in RPM format so that they are easy to upgrade and verify using the **Package Management Tool**, or the **yum** command. The **Package Management Tool** automatically queries the Red Hat Enterprise Linux servers and determines which packages need to be updated on your machine, including the kernel. This chapter is *only* useful for those individuals that require manual updating of kernel packages, without using the **yum** command.

Warning

Building a custom kernel is not supported by the Red Hat Global Services Support team, and therefore is not explored in this manual.

Note

The use of **yum** is *highly* recommended by Red Hat for installing upgraded kernels.

For more information on Red Hat Network, the **Package Management Tool**, and **yum**, refer to Chapter 15, *Registering a System and Managing Subscriptions*.

44.1. Overview of Kernel Packages

Red Hat Enterprise Linux contains the following kernel packages (some may not apply to your architecture):

- **kernel** — Contains the kernel for multi-processor systems. For x86 system, only the first 4GB of RAM is used. As such, x86 systems with over 4GB of RAM should use the **kernel-PAE**.
- **kernel-devel** — Contains the kernel headers and makefiles sufficient to build modules against the **kernel** package.
- **kernel-PAE** (only for i686 systems) — This package offers the following key configuration option (in addition to the options already enabled for the **kernel** package):
  - PAE (Physical Address Extension) support for systems with more than 4GB of RAM, and reliably up to 16GB.

Important

Physical Address Extension allows x86 processors to address up to 64GB of physical RAM, but due to differences between the Red Hat Enterprise Linux 4 and 5 kernels, only Red Hat Enterprise Linux 4 (with the **`kernel-hugemem`** package) is able to reliably address all 64GB of memory. Additionally, the Red Hat Enterprise Linux 5 PAE variant does not allow 4GB of addressable memory per-process like the Red Hat Enterprise Linux 4 **`kernel-hugemem`** variant does. However, the x86_64 kernel does not suffer from any of these limitations, and is the suggested Red Hat Enterprise Linux 5 architecture to use with large-memory systems.

- **`kernel-PAE-devel`** — Contains the kernel headers and makefiles required to build modules against the **`kernel-PAE`** package.
- **`kernel-doc`** — Contains documentation files from the kernel source. Various portions of the Linux kernel and the device drivers shipped with it are documented in these files. Installation of this package provides a reference to the options that can be passed to Linux kernel modules at load time.

  By default, these files are placed in the **`/usr/share/doc/kernel-doc-<`*`version`*`>/`** directory.
- **`kernel-headers`** — Includes the C header files that specify the interface between the Linux kernel and userspace libraries and programs. The header files define structures and constants that are needed for building most standard programs.
- **`kernel-xen`** — Includes a version of the Linux kernel which is needed to run Virtualization.
- **`kernel-xen-devel`** — Contains the kernel headers and makefiles required to build modules against the **`kernel-xen`** package

Note

The **`kernel-source`** package has been removed and replaced with an RPM that can only be retrieved from Red Hat Network. This **`*.src.rpm`** package must then be rebuilt locally using the **`rpmbuild`** command. For more information on obtaining and installing the kernel source package, refer to the latest updated Release Notes (including all updates) at http://www.redhat.com/docs/manuals/enterprise/

44.2. Preparing to Upgrade

Before upgrading the kernel, it is recommended that you take some precautionary steps. The first step is to make sure working boot media exists for the system in case a problem occurs. If the boot loader is not configured properly to boot the new kernel, the system cannot be booted into Red Hat Enterprise Linux without working boot media.

To create a boot diskette, login as root, and run the command **``/sbin/mkbootdisk `uname -r` ``** at a shell prompt.

Note

Refer to the **mkbootdisk** man page for more options. You can create bootable media via CD-Rs, CD-RWs, and USB flash drives, provided that your system BIOS also supports it.

Reboot the machine with the boot media and verify that it works before continuing.

To determine which kernel packages are installed, execute the command **rpm -qa | grep kernel** at a shell prompt:

The output contains some or all of the following packages, depending on the system's architecture (the version numbers and packages may differ):

```
kernel-2.6.9-5.EL
kernel-devel-2.6.9-5.EL
kernel-utils-2.6.9-5.EL
kernel-doc-2.6.9-5.EL
kernel-smp-2.6.9-5.EL
kernel-smp-devel-2.6.9-5.EL
kernel-hugemem-devel-2.6.9-5.EL
```

From the output, determine which packages need to be download for the kernel upgrade. For a single processor system, the only required package is the **kernel** package. Refer to Section 44.1, “Overview of Kernel Packages” for descriptions of the different packages.

In the file name, each kernel package contains the architecture for which the package was built. The format is kernel-<*variant*>-<*version*>.<*arch*>.rpm, where <*variant*> is one of either **PAE**, **xen**, and so forth. The <*arch*> is one of the following:

- **x86_64** for the AMD64 and Intel EM64T architectures
- **ia64** for the Intel® Itanium™ architecture
- **ppc64** for the IBM® eServer™ pSeries™ architecture
- **s390** for the IBM® S/390® architecture
- **s390x** for the IBM® eServer™ System z® architecture
- **i686** for Intel® Pentium® II, Intel® Pentium® III, Intel® Pentium® 4, AMD Athlon®, and AMD Duron® systems

44.3. Downloading the Upgraded Kernel

There are several ways to determine if an updated kernel is available for the system.

- Security Errata — Refer to http://www.redhat.com/security/updates/ for information on security errata, including kernel upgrades that fix security issues.
- Via Red Hat Network — Download and install the kernel RPM packages. Red Hat Network can download the latest kernel, upgrade the kernel on the system, create an initial RAM disk image if needed, and configure the boot loader to boot the new kernel. For more information, refer to http://www.redhat.com/docs/manuals/RHNetwork/.

If Red Hat Network was used to download and install the updated kernel, follow the instructions in Section 44.5, "Verifying the Initial RAM Disk Image" and Section 44.6, "Verifying the Boot Loader", only *do not* change the kernel to boot by default. Red Hat Network automatically changes the default kernel to the latest version. To install the kernel manually, continue to Section 44.4, "Performing the Upgrade".

44.4. Performing the Upgrade

After retrieving all of the necessary packages, it is time to upgrade the existing kernel.

Important

It is strongly recommended that you keep the old kernel in case there are problems with the new kernel.

At a shell prompt, change to the directory that contains the kernel RPM packages. Use **`-i`** argument with the **`rpm`** command to keep the old kernel. Do *not* use the **`-U`** option, since it overwrites the currently installed kernel, which creates boot loader problems. For example:

```
rpm -ivh kernel-<kernel version>.<arch>.rpm
```

The next step is to verify that the initial RAM disk image has been created. Refer to Section 44.5, "Verifying the Initial RAM Disk Image" for details.

44.5. Verifying the Initial RAM Disk Image

If the system uses the ext3 file system, a SCSI controller, or labels to reference partitions in **`/etc/fstab`**, an initial RAM disk is needed. The initial RAM disk allows a modular kernel to have access to modules that it might need to boot from before the kernel has access to the device where the modules normally reside.

On architectures other than IBM eServer iSeries, the initial RAM disk can be created with the **`mkinitrd`** command. However, this step is performed automatically if the kernel and its associated packages are installed or upgraded from the RPM packages distributed by Red Hat; in such cases, you do not need to create the initial RAM disk manually. To verify that an initial RAM disk already exists, use the command **`ls -l /boot`** to make sure the **`initrd-<version>.img`** file was created (the version should match the version of the kernel just installed).

On iSeries systems, the initial RAM disk file and **`vmlinux`** file are combined into one file, which is created with the **`addRamDisk`** command. This step is performed automatically if the kernel and its associated packages are installed or upgraded from the RPM packages distributed by Red Hat, Inc.; thus, it does not need to be executed manually. To verify that it was created, use the command **`ls -l /boot`** to make sure the **`/boot/vmlinitrd-<kernel-version>`** file already exists (the **`<kernel-version>`** should match the version of the kernel just installed).

The next step is to verify that the boot loader has been configured to boot the new kernel. Refer to Section 44.6, "Verifying the Boot Loader" for details.

44.6. Verifying the Boot Loader

The **`kernel`** RPM package configures the boot loader to boot the newly installed kernel (except for IBM eServer iSeries systems). However, it does not configure the boot loader to boot the new kernel by default.

It is always a good idea to confirm that the boot loader has been configured correctly. This is a crucial step. If the boot loader is configured incorrectly, the system will not boot into Red Hat Enterprise Linux properly. If this happens, boot the system with the boot media created earlier and try configuring the boot loader again.

44.6.1. x86 Systems

All x86 systems (including all AMD64 systems) use GRUB as the boot loader.

44.6.1.1. GRUB

Confirm that the file **`/boot/grub/grub.conf`** contains a **`title`** section with the same version as the **`kernel`** package just installed

```
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,0)
#          kernel /vmlinuz-version ro root=/dev/hda2
#          initrd /initrd-version.img
#boot=/dev/hda
default=1 timeout=10
splashimage=(hd0,0)/grub/splash.xpm.gz
title Red Hat Enterprise Linux (2.6.9-5.EL)
         root (hd0,0)
  kernel /vmlinuz-2.6.9-5.EL ro root=LABEL=/
  initrd /initrd-2.6.9-5.EL.img
title Red Hat Enterprise Linux (2.6.9-1.906_EL)
         root (hd0,0)
  kernel /vmlinuz-2.6.9-1.906_EL ro root=LABEL=/
  initrd /initrd-2.6.9-1.906_EL.img
```

If a separate **`/boot/`** partition was created, the paths to the kernel and **`initrd`** image are relative to **`/boot/`**.

Notice that the default is not set to the new kernel. To configure GRUB to boot the new kernel by default, change the value of the **`default`** variable to the title section number for the title section that contains the new kernel. The count starts with 0. For example, if the new kernel is the first title section, set **`default`** to **`0`**.

Begin testing the new kernel by rebooting the computer and watching the messages to ensure that the hardware is detected properly.

44.6.2. Itanium Systems

Itanium systems use ELILO as the boot loader, which uses **`/boot/efi/EFI/redhat/elilo.conf`** as the configuration file. Confirm that this file contains an **`image`** section with the same version as the **`kernel`** package just installed:

```
prompt timeout=50 default=old  image=vmlinuz-2.6.9-5.EL
         label=linux
  initrd=initrd-2.6.9-5.EL.img         read-only
  append="root=LABEL=/" image=vmlinuz-2.6.9-1.906_EL
  label=old
  initrd=initrd-2.6.9-1.906.img         read-only
  append="root=LABEL=/"
```

Notice that the default is not set to the new kernel. To configure ELILO to boot the new kernel, change the value of the **`default`** variable to the value of the **`label`** for the **`image`** section that contains the new kernel.

Begin testing the new kernel by rebooting the computer and watching the messages to ensure that the hardware is detected properly.

44.6.3. IBM S/390 and IBM System z Systems

The IBM S/390 and IBM System z systems use z/IPL as the boot loader, which uses **`/etc/zipl.conf`** as the configuration file. Confirm that the file contains a section with the same version as the kernel package just installed:

```
[defaultboot] default=old target=/boot/
[linux]
          image=/boot/vmlinuz-2.6.9-5.EL
  ramdisk=/boot/initrd-2.6.9-5.EL.img
  parameters="root=LABEL=/"
[old]
          image=/boot/vmlinuz-2.6.9-1.906_EL
  ramdisk=/boot/initrd-2.6.9-1.906_EL.img
  parameters="root=LABEL=/"
```

Notice that the default is not set to the new kernel. To configure z/IPL to boot the new kernel by default, change the value of the **`default`** variable to the name of the section that contains the new kernel. The first line of each section contains the name in brackets.

After modifying the configuration file, run **`/sbin/zipl`** as root to enable the changes.

Begin testing the new kernel by rebooting the computer and watching the messages to ensure that the hardware is detected properly.

44.6.4. IBM eServer iSeries Systems

The **`/boot/vmlinitrd-<`*`kernel-version`*`>`** file is installed when you upgrade the kernel. However, you must use the **dd** command to configure the system to boot the new kernel:

1. As root, issue the command **`cat /proc/iSeries/mf/side`** to determine the default side (either A, B, or C).

2. As root, issue the following command, where <*kernel-version*> is the version of the new kernel and <*side*> is the side from the previous command:

   `dd if=/boot/vmlinitrd-<`*`kernel-version`*`> of=/proc/iSeries/mf/<`*`side`*`>/vmlinux bs=8k`

Begin testing the new kernel by rebooting the computer and watching the messages to ensure that the hardware is detected properly.

44.6.5. IBM eServer pSeries Systems

IBM eServer pSeries systems use YABOOT as the boot loader, which uses **`/etc/aboot.conf`** as the configuration file. Confirm that the file contains an **`image`** section with the same version as the **`kernel`** package just installed:

```
boot=/dev/sda1 init-message=Welcome to Red Hat Enterprise Linux! Hit <TAB>
```

```
for boot options
partition=2 timeout=30 install=/usr/lib/yaboot/yaboot delay=10 nonvram
image=/vmlinux--2.6.9-5.EL
         label=old
  read-only
  initrd=/initrd--2.6.9-5.EL.img
  append="root=LABEL=/"
image=/vmlinux-2.6.9-5.EL
  label=linux
  read-only
  initrd=/initrd-2.6.9-5.EL.img
  append="root=LABEL=/"
```

Notice that the default is not set to the new kernel. The kernel in the first image is booted by default. To change the default kernel to boot either move its image stanza so that it is the first one listed or add the directive **`default`** and set it to the **`label`** of the image stanza that contains the new kernel.

Begin testing the new kernel by rebooting the computer and watching the messages to ensure that the hardware is detected properly.

Chapter 45. General Parameters and Modules

This chapter is provided to illustrate *some* of the possible parameters available for common hardware device *drivers* [9], which under Red Hat Enterprise Linux are called kernel *modules*. In most cases, the default parameters do work. However, there may be times when extra module parameters are necessary for a device to function properly or to override the module's default parameters for the device.

During installation, Red Hat Enterprise Linux uses a limited subset of device drivers to create a stable installation environment. Although the installation program supports installation on many different types of hardware, some drivers (including those for SCSI adapters and network adapters) are not included in the installation kernel. Rather, they must be loaded as modules by the user at boot time.

Once installation is completed, support exists for a large number of devices through kernel modules.

Important

Red Hat provides a large number of unsupported device drivers in groups of packages called **kernel-smp-unsupported-*<kernel-version>*** and **kernel-hugemem-unsupported-*<kernel-version>*** . Replace *<kernel-version>* with the version of the kernel installed on the system. These packages are not installed by the Red Hat Enterprise Linux installation program, and the modules provided are not supported by Red Hat, Inc.

45.1. Kernel Module Utilities

A group of commands for managing kernel modules is available if the **module-init-tools** package is installed. Use these commands to determine if a module has been loaded successfully or when trying different modules for a piece of new hardware.

The command **/sbin/lsmod** displays a list of currently loaded modules. For example:

```
Module                  Size  Used by
tun                    11585  1
autofs4                21573  1
hidp                   16193  2
rfcomm                 37849  0
l2cap                  23873  10 hidp,rfcomm
bluetooth              50085  5 hidp,rfcomm,l2cap
sunrpc                153725  1
dm_mirror              29073  0
dm_mod                 57433  1 dm_mirror
video                  17221  0
sbs                    16257  0
i2c_ec                  5569  1 sbs
container               4801  0
button                  7249  0
battery                10565  0
asus_acpi              16857  0
ac                      5701  0
ipv6                  246113  12
lp                     13065  0
parport_pc             27493  1
parport                37001  2 lp,parport_pc
```

```
uhci_hcd               23885  0
floppy                 57317  1
sg                     34653  0
snd_ens1371            26721  1
gameport               16073  1 snd_ens1371
snd_rawmidi            24897  1 snd_ens1371
snd_ac97_codec         91360  1 snd_ens1371
snd_ac97_bus            2753  1 snd_ac97_codec
snd_seq_dummy           4293  0
snd_seq_oss            32705  0
serio_raw               7493  0
snd_seq_midi_event      8001  1 snd_seq_oss
snd_seq                51633  5
snd_seq_dummy,snd_seq_oss,snd_seq_midi_event
snd_seq_device          8781  4
snd_rawmidi,snd_seq_dummy,snd_seq_oss,snd_seq
snd_pcm_oss            42849  0
snd_mixer_oss          16833  1 snd_pcm_oss
snd_pcm                76485  3 snd_ens1371,snd_ac97_codec,snd_pcm_oss
snd_timer              23237  2 snd_seq,snd_pcm
snd                    52933  12
snd_ens1371,snd_rawmidi,snd_ac97_codec,snd_seq_oss,snd_seq,snd_seq_device,sn
d_pcm_oss,snd_mixer_oss,snd_pcm,snd_timer
soundcore              10145  1 snd
i2c_piix4               8909  0
ide_cd                 38625  3
snd_page_alloc         10569  1 snd_pcm
i2c_core               21697  2 i2c_ec,i2c_piix4
pcnet32                34117  0
cdrom                  34913  1 ide_cd
mii                     5825  1 pcnet32
pcspkr                  3521  0
ext3                  129737  2
jbd                    58473  1 ext3
mptspi                 17353  3
scsi_transport_spi     25025  1 mptspi
mptscsih               23361  1 mptspi
sd_mod                 20929  16
scsi_mod              134121  5 sg,mptspi,scsi_transport_spi,mptscsih,sd_mod
mptbase                52193  2 mptspi,mptscsih
```

For each line, the first column is the name of the module, the second column is the size of the module, and the third column is the use count.

The **`/sbin/lsmod`** output is less verbose and easier to read than the output from viewing **`/proc/modules`**.

To load a kernel module, use the **`/sbin/modprobe`** command followed by the kernel module name. By default, **`modprobe`** attempts to load the module from the **`/lib/modules/`*`<kernel-version>`*`/kernel/drivers/`** subdirectories. There is a subdirectory for each type of module, such as the **`net/`** subdirectory for network interface drivers. Some kernel modules have module dependencies, meaning that other modules must be loaded first for it to load. The **`/sbin/modprobe`** command checks for these dependencies and loads the module dependencies before loading the specified module.

For example, the command

```
modprobe e100
```

loads any module dependencies and then the **e100** module.

To print to the screen all commands as **/sbin/modprobe** executes them, use the **-v** option. For example:

```
modprobe -v e100
```

Output similar to the following is displayed:

```
insmod /lib/modules/2.6.9-5.EL/kernel/drivers/net/e100.ko
Using /lib/modules/2.6.9-5.EL/kernel/drivers/net/e100.ko
Symbol version prefix 'smp_'
```

The **/sbin/insmod** command also exists to load kernel modules; however, it does not resolve dependencies. Thus, it is recommended that the **/sbin/modprobe** command be used.

To unload kernel modules, use the **/sbin/rmmod** command followed by the module name. The **rmmod** utility only unloads modules that are not in use and that are not a dependency of other modules in use.

For example, the command

```
rmmod e100
```

unloads the **e100** kernel module.

Another useful kernel module utility is **modinfo**. Use the command **/sbin/modinfo** to display information about a kernel module. The general syntax is:

```
modinfo [options] <module>
```

Options include **-d**, which displays a brief description of the module, and **-p**, which lists the parameters the module supports. For a complete list of options, refer to the **modinfo** man page (**man modinfo**).

45.2. Persistent Module Loading

Kernel modules are usually loaded directly by the facility that requires them, which is given correct settings in the **/etc/modprobe.conf** file. However, it is sometimes necessary to explicitly force the loading of a module at boot time.

Red Hat Enterprise Linux checks for the existence of the **/etc/rc.modules** file at boot time, which contains various commands to load modules. The **rc.modules** should be used, and *not* **rc.local** because **rc.modules** is executed earlier in the boot process.

For example, the following commands configure loading of the **foo** module at boot time (as root):

```
echo modprobe foo >> /etc/rc.modules
chmod +x /etc/rc.modules
```

Note

This approach is not necessary for network and SCSI interfaces because they have their own specific mechanisms.

45.3. Specifying Module Parameters

In some situations, it may be necessary to supply parameters to a module as it is loaded for it to function properly.

For instance, to enable full duplex at 100Mbps connection speed for an Intel Ether Express/100 card, load the **`e100`** driver with the **`e100_speed_duplex=4`** option.

Warning

When a parameter has commas, be sure *not* to put a space after a comma.

Note

The **`modinfo`** command is also useful for listing various information about a kernel module, such as version, dependencies, parameter options, and aliases.

45.4. Storage parameters

Table 45.1. Storage Module Parameters

Hardware	Module	Parameters
3ware Storage Controller and 9000 series	**`3w-xxxx.ko, 3w-9xxx.ko`**	

Hardware	Module	Parameters
Adaptec Advanced Raid Products, Dell PERC2, 2/Si, 3/Si, 3/Di, HP NetRAID-4M, IBM ServeRAID, and ICP SCSI driver	`aacraid.ko`	***`nondasd`*** — Control scanning of hba for nondasd devices. 0=off, 1=on ***`dacmode`*** — Control whether dma addressing is using 64 bit DAC. 0=off, 1=on ***`commit`*** — Control whether a COMMIT_CONFIG is issued to the adapter for foreign arrays. This is typically needed in systems that do not have a BIOS. 0=off, 1=on ***`startup_timeout`*** — The duration of time in seconds to wait for adapter to have it's kernel up and running. This is typically adjusted for large systems that do not have a BIOS ***`aif_timeout`*** — The duration of time in seconds to wait for applications to pick up AIFs before deregistering them. This is typically adjusted for heavily burdened systems. ***`numacb`*** — Request a limit to the number of adapter control blocks (FIB) allocated. Valid values are 512 and down. Default is to use suggestion from Firmware. ***`acbsize`*** — Request a specific adapter control block (FIB) size. Valid values are 512, 2048, 4096 and 8192. Default is to use suggestion from Firmware.

Hardware	Module	Parameters
Adaptec 28xx, R9xx, 39xx AHA-284x, AHA-29xx, AHA-394x, AHA-398x, AHA-274x, AHA-274xT, AHA-2842, AHA-2910B, AHA-2920C, AHA-2930/U/U2, AHA-2940/W/U/UW/AU/, U2W/U2/U2B/, U2BOEM, AHA-2944D/WD/UD/UWD, AHA-2950U2/W/B, AHA-3940/U/W/UW/, AUW/U2W/U2B, AHA-3950U2D, AHA-3985/U/W/UW, AIC-777x, AIC-785x, AIC-786x, AIC-787x, AIC-788x , AIC-789x, AIC-3860	`aic7xxx.ko`	***`verbose`*** — Enable verbose/diagnostic logging ***`allow_memio`*** — Allow device registers to be memory mapped ***`debug`*** — Bitmask of debug values to enable ***`no_probe`*** — Toggle EISA/VLB controller probing ***`probe_eisa_vl`*** — Toggle EISA/VLB controller probing ***`no_reset`*** — Supress initial bus resets ***`extended`*** — Enable extended geometry on all controllers ***`periodic_otag`*** — Send an ordered tagged transaction periodically to prevent tag starvation. This may be required by some older disk drives or RAID arrays. ***`tag_info:<tag_str>`*** — Set per-target tag depth ***`global_tag_depth:<int>`*** — Global tag depth for every target on every bus ***`seltime:<int>`*** — Selection Timeout (0/256ms,1/128ms,2/64ms,3/32ms)
IBM ServeRAID	`ips.ko`	

Hardware	Module	Parameters
LSI Logic MegaRAID Mailbox Driver	**`megaraid_mbox.ko`**	***`unconf_disks`*** — Set to expose unconfigured disks to kernel (default=0) ***`busy_wait`*** — Max wait for mailbox in microseconds if busy (default=10) ***`max_sectors`*** — Maximum number of sectors per IO command (default=128) ***`cmd_per_lun`*** — Maximum number of commands per logical unit (default=64) ***`fast_load`*** — Faster loading of the driver, skips physical devices! (default=0) ***`debug_level`*** — Debug level for driver (default=0)
Emulex LightPulse Fibre Channel SCSI driver	**`lpfc.ko`**	***`lpfc_poll`*** — FCP ring polling mode control: 0 - none, 1 - poll with interrupts enabled 3 - poll and disable FCP ring interrupts ***`lpfc_log_verbose`*** — Verbose logging bit-mask ***`lpfc_lun_queue_depth`*** — Max number of FCP commands we can queue to a specific LUN ***`lpfc_hba_queue_depth`*** — Max number of FCP commands we can queue to a lpfc HBA ***`lpfc_scan_down`*** — Start scanning for devices from highest ALPA to lowest ***`lpfc_nodev_tmo`*** — Seconds driver will hold I/O waiting for a device to come back ***`lpfc_topology`*** — Select Fibre Channel topology ***`lpfc_link_speed`*** — Select link speed ***`lpfc_fcp_class`*** — Select Fibre Channel class of service for FCP sequences

Hardware	Module	Parameters
		lpfc_use_adisc — Use ADISC on rediscovery to authenticate FCP devices ***lpfc_ack0*** — Enable ACK0 support ***lpfc_cr_delay*** — A count of milliseconds after which an interrupt response is generated ***lpfc_cr_count*** — A count of I/O completions after which an interrupt response is generated ***lpfc_multi_ring_support*** — Determines number of primary SLI rings to spread IOCB entries across ***lpfc_fdmi_on*** — Enable FDMI support ***lpfc_discovery_threads*** — Maximum number of ELS commands during discovery ***lpfc_max_luns*** — Maximum allowed LUN ***lpfc_poll_tmo*** — Milliseconds driver will wait between polling FCP ring
HP Smart Array	cciss.ko	
LSI Logic MPT Fusion	mptbase.ko mptctl.ko mptfc.ko mptlan.ko mptsas.ko mptscsih.ko mptspi.ko	***mpt_msi_enable*** — MSI Support Enable ***mptfc_dev_loss_tmo*** — Initial time the driver programs the transport to wait for an rport to return following a device loss event. ***mpt_pt_clear*** — Clear persistency table ***mpt_saf_te*** — Force enabling SEP Processor

Hardware	Module	Parameters
QLogic Fibre Channel Driver	qla2xxx.ko	***ql2xlogintimeout*** — Login timeout value in seconds. ***qlport_down_retry*** — Maximum number of command retries to a port that returns a PORT-DOWN status ***ql2xplogiabsentdevice*** — Option to enable PLOGI to devices that are not present after a Fabric scan. ***ql2xloginretrycount*** — Specify an alternate value for the NVRAM login retry count. ***ql2xallocfwdump*** — Option to enable allocation of memory for a firmware dump during HBA initialization. Default is 1 - allocate memory. ***extended_error_logging*** — Option to enable extended error logging. ***ql2xfdmienable*** — Enables FDMI registrations.

Hardware	Module	Parameters
NCR, Symbios and LSI 8xx and 1010	sym53c8xx	***cmd_per_lun*** — The maximum number of tags to use by default ***tag_ctrl*** — More detailed control over tags per LUN ***burst*** — Maximum burst. 0 to disable, 255 to read from registers ***led*** — Set to 1 to enable LED support ***diff*** — 0 for no differential mode, 1 for BIOS, 2 for always, 3 for not GPIO3 ***irqm*** — 0 for open drain, 1 to leave alone, 2 for totem pole ***buschk*** — 0 to not check, 1 for detach on error, 2 for warn on error ***hostid*** — The SCSI ID to use for the host adapters ***verb*** — 0 for minimal verbosity, 1 for normal, 2 for excessive ***debug*** — Set bits to enable debugging ***settle*** — Settle delay in seconds. Default 3 ***nvram*** — Option currently not used ***excl*** — List ioport addresses here to prevent controllers from being attached ***safe*** — Set other settings to a "safe mode"

45.5. Ethernet Parameters

Important

Most modern Ethernet-based network interface cards (NICs), do not require module parameters to alter settings. Instead, they can be configured using **`ethtool`** or **`mii-tool`**. Only after these tools fail to work should module parameters be adjusted. Module parameters can be viewed using the **`modinfo`** command.

Note

For information about using these tools, consult the man pages for **`ethtool`**, **`mii-tool`**, and **`modinfo`**.

Table 45.2. Ethernet Module Parameters

Hardware	Module	Parameters
3Com EtherLink PCI III/XL Vortex (3c590, 3c592, 3c595, 3c597) Boomerang (3c900, 3c905, 3c595)	**`3c59x.ko`**	***`debug`*** — 3c59x debug level (0-6) ***`options`*** — 3c59x: Bits 0-3: media type, bit 4: bus mastering, bit 9: full duplex ***`global_options`*** — 3c59x: same as options, but applies to all NICs if options is unset ***`full_duplex`*** — 3c59x full duplex setting(s) (1) ***`global_full_duplex`*** — 3c59x: same as full_duplex, but applies to all NICs if full_duplex is unset ***`hw_checksums`*** — 3c59x Hardware checksum checking by adapter(s) (0-1) ***`flow_ctrl`*** — 3c59x 802.3x flow control usage (PAUSE only) (0-1) ***`enable_wol`*** — 3c59x: Turn on Wake-on-LAN for adapter(s) (0-1) ***`global_enable_wol`*** — 3c59x: same as enable_wol, but applies to all NICs if enable_wol is unset ***`rx_copybreak`*** — 3c59x copy breakpoint for copy-only-tiny-frames

Hardware	Module	Parameters
		max_interrupt_work — 3c59x maximum events handled per interrupt ***compaq_ioaddr*** — 3c59x PCI I/O base address (Compaq BIOS problem workaround) ***compaq_irq*** — 3c59x PCI IRQ number (Compaq BIOS problem workaround) ***compaq_device_id*** — 3c59x PCI device ID (Compaq BIOS problem workaround) ***watchdog*** — 3c59x transmit timeout in milliseconds ***global_use_mmio*** — 3c59x: same as use_mmio, but applies to all NICs if options is unset ***use_mmio*** — 3c59x: use memory-mapped PCI I/O resource (0-1)
RTL8139, SMC EZ Card Fast Ethernet, RealTek cards using RTL8129, or RTL8139 Fast Ethernet chipsets	**8139too.ko**	
Broadcom 4400 10/100 PCI ethernet driver	**b44.ko**	***b44_debug*** — B44 bitmapped debugging message enable value
Broadcom NetXtreme II BCM5706/5708 Driver	**bnx2.ko**	***disable_msi*** — Disable Message Signaled Interrupt (MSI)
Intel Ether Express/100 driver	**e100.ko**	***debug*** — Debug level (0=none,...,16=all) ***eeprom_bad_csum_allow*** — Allow bad eeprom checksums

Hardware	Module	Parameters
Intel EtherExpress/1000 Gigabit	**`e1000.ko`**	***`TxDescriptors`*** — Number of transmit descriptors ***`RxDescriptors`*** — Number of receive descriptors ***`Speed`*** — Speed setting ***`Duplex`*** — Duplex setting ***`AutoNeg`*** — Advertised auto-negotiation setting ***`FlowControl`*** — Flow Control setting ***`XsumRX`*** — Disable or enable Receive Checksum offload ***`TxIntDelay`*** — Transmit Interrupt Delay ***`TxAbsIntDelay`*** — Transmit Absolute Interrupt Delay ***`RxIntDelay`*** — Receive Interrupt Delay ***`RxAbsIntDelay`*** — Receive Absolute Interrupt Delay ***`InterruptThrottleRate`*** — Interrupt Throttling Rate ***`SmartPowerDownEnable`*** — Enable PHY smart power down ***`KumeranLockLoss`*** — Enable Kumeran lock loss workaround

Hardware	Module	Parameters
Myricom 10G driver (10GbE)	**`myri10ge.ko`**	***`myri10ge_fw_name`*** — Firmware image name ***`myri10ge_ecrc_enable`*** — Enable Extended CRC on PCI-E ***`myri10ge_max_intr_slots`*** — Interrupt queue slots ***`myri10ge_small_bytes`*** — Threshold of small packets ***`myri10ge_msi`*** — Enable Message Signalled Interrupts ***`myri10ge_intr_coal_delay`*** — Interrupt coalescing delay ***`myri10ge_flow_control`*** — Pause parameter ***`myri10ge_deassert_wait`*** — Wait when deasserting legacy interrupts ***`myri10ge_force_firmware`*** — Force firmware to assume aligned completions ***`myri10ge_skb_cross_4k`*** — Can a small skb cross a 4KB boundary? ***`myri10ge_initial_mtu`*** — Initial MTU ***`myri10ge_napi_weight`*** — Set NAPI weight ***`myri10ge_watchdog_timeout`*** — Set watchdog timeout ***`myri10ge_max_irq_loops`*** — Set stuck legacy IRQ detection threshold

Hardware	Module	Parameters
NatSemi DP83815 Fast Ethernet	**`natsemi.ko`**	***`mtu`*** — DP8381x MTU (all boards) ***`debug`*** — DP8381x default debug level ***`rx_copybreak`*** — DP8381x copy breakpoint for copy-only-tiny-frames ***`options`*** — DP8381x: Bits 0-3: media type, bit 17: full duplex ***`full_duplex`*** — DP8381x full duplex setting(s) (1)
AMD PCnet32 and AMD PCnetPCI	**`pcnet32.ko`**	
PCnet32 and PCnetPCI	**`pcnet32.ko`**	***`debug`*** — pcnet32 debug level ***`max_interrupt_work`*** — pcnet32 maximum events handled per interrupt ***`rx_copybreak`*** — pcnet32 copy breakpoint for copy-only-tiny-frames ***`tx_start_pt`*** — pcnet32 transmit start point (0-3) ***`pcnet32vlb`*** — pcnet32 Vesa local bus (VLB) support (0/1) ***`options`*** — pcnet32 initial option setting(s) (0-15) ***`full_duplex`*** — pcnet32 full duplex setting(s) (1) ***`homepna`*** — pcnet32 mode for 79C978 cards (1 for HomePNA, 0 for Ethernet, default Ethernet
RealTek RTL-8169 Gigabit Ethernet driver	**`r8169.ko`**	***`media`*** — force phy operation. Deprecated by ethtool (8). ***`rx_copybreak`*** — Copy breakpoint for copy-only-tiny-frames ***`use_dac`*** — Enable PCI DAC. Unsafe on 32 bit PCI slot. ***`debug`*** — Debug verbosity level (0=none, ..., 16=all)

Hardware	Module	Parameters
Neterion Xframe 10GbE Server Adapter	**`s2io.ko`**	
SIS 900/701G PCI Fast Ethernet	**`sis900.ko`**	***`multicast_filter_limit`*** — SiS 900/7016 maximum number of filtered multicast addresses ***`max_interrupt_work`*** — SiS 900/7016 maximum events handled per interrupt ***`sis900_debug`*** — SiS 900/7016 bitmapped debugging message level
Adaptec Starfire Ethernet driver	**`starfire.ko`**	***`max_interrupt_work`*** — Maximum events handled per interrupt ***`mtu`*** — MTU (all boards) ***`debug`*** — Debug level (0-6) ***`rx_copybreak`*** — Copy breakpoint for copy-only-tiny-frames ***`intr_latency`*** — Maximum interrupt latency, in microseconds ***`small_frames`*** — Maximum size of receive frames that bypass interrupt latency (0,64,128,256,512) ***`options`*** — Deprecated: Bits 0-3: media type, bit 17: full duplex ***`full_duplex`*** — Deprecated: Forced full-duplex setting (0/1) ***`enable_hw_cksum`*** — Enable/disable hardware cksum support (0/1)
Broadcom Tigon3	**`tg3.ko`**	***`tg3_debug`*** — Tigon3 bitmapped debugging message enable value

Hardware	Module	Parameters
ThunderLAN PCI	**`tlan.ko`**	***`aui`*** — ThunderLAN use AUI port(s) (0-1) ***`duplex`*** — ThunderLAN duplex setting(s) (0-default, 1-half, 2-full) ***`speed`*** — ThunderLAN port speen setting(s) (0,10,100) ***`debug`*** — ThunderLAN debug mask ***`bbuf`*** — ThunderLAN use big buffer (0-1)
Digital 21x4x Tulip PCI Ethernet cards SMC EtherPower 10 PCI(8432T/8432BT) SMC EtherPower 10/100 PCI(9332DST) DEC EtherWorks 100/10 PCI(DE500-XA) DEC EtherWorks 10 PCI(DE450) DEC QSILVER's, Znyx 312 etherarray Allied Telesis LA100PCI-T Danpex EN-9400, Cogent EM110	**`tulip.ko`**	***`io`*** *io_port*
VIA Rhine PCI Fast Ethernet cards with either the VIA VT86c100A Rhine-II PCI or 3043 Rhine-I D-Link DFE-930-TX PCI 10/100	**`via-rhine.ko`**	***`max_interrupt_work`*** — VIA Rhine maximum events handled per interrupt ***`debug`*** — VIA Rhine debug level (0-7) ***`rx_copybreak`*** — VIA Rhine copy breakpoint for copy-only-tiny-frames ***`avoid_D3`*** — Avoid power state D3 (work-around for broken BIOSes)

45.5.1. The Channel Bonding Module

Red Hat Enterprise Linux allows administrators to bind NICs together into a single channel using the **`bonding`** kernel module and a special network interface, called a *channel bonding interface*. Channel bonding enables two or more network interfaces to act as one, simultaneously increasing the bandwidth and providing redundancy.

To channel bond multiple network interfaces, the administrator must perform the following steps:

1. Add the following line to **`/etc/modprobe.conf`**:

```
alias bond<N> bonding
```

Replace <*N*> with the interface number, such as **`0`**. For each configured channel bonding interface, there must be a corresponding entry in **`/etc/modprobe.conf`**.

2. Configure a channel bonding interface as outlined in Section 16.2.3, “Channel Bonding Interfaces”.

3. To enhance performance, adjust available module options to ascertain what combination works best. Pay particular attention to the **`miimon`** or **`arp_interval`** and the **`arp_ip_target`** parameters. Refer to Section 45.5.1.1, “bonding Module Directives” for a list of available options and how to quickly determine the best ones for your bonded interface.

45.5.1.1. bonding Module Directives

It is a good idea to test which channel bonding module parameters work best for your bonded interfaces before adding them to the ***`BONDING_OPTS="<bonding parameters>"`*** directive in your bonding interface configuration file (**`ifcfg-bond0`** for example). Parameters to bonded interfaces can be configured without unloading (and reloading) the bonding module by manipulating files in the **`sysfs`** file system.

`sysfs` is a virtual file system that represents kernel objects as directories, files and symbolic links. **`sysfs`** can be used to query for information about kernel objects, and can also manipulate those objects through the use of normal file system commands. The **`sysfs`** virtual file system has a line in **`/etc/fstab`**, and is mounted under **`/sys`**. All bonded interfaces can be configured dynamically by interacting with and manipulating files under the **`/sys/class/net/`** directory.

After you have created a channel bonding interface file such as **`ifcfg-bond0`** and inserted ***`SLAVE=yes`*** and ***`MASTER=bond0`*** directives in the bonded interfaces following the instructions in Section 16.2.3, “Channel Bonding Interfaces”, you can proceed to testing and determining the best parameters for your bonded interface.

First, bring up the bond you created by running **`ifconfig bond<N> up`** as root:

```
ifconfig bond0 up
```

If you have correctly created the **`ifcfg-bond0`** bonding interface file, you will be able to see **`bond0`** listed in the output of running **`ifconfig`** (without any options):

```
~]# ifconfig
bond0     Link encap:Ethernet  HWaddr 00:00:00:00:00:00
          UP BROADCAST RUNNING MASTER MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
eth0      Link encap:Ethernet  HWaddr 52:54:00:26:9E:F1
          inet addr:192.168.122.251  Bcast:192.168.122.255
Mask:255.255.255.0
          inet6 addr: fe80::5054:ff:fe26:9ef1/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:207 errors:0 dropped:0 overruns:0 frame:0
          TX packets:205 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:70374 (68.7 KiB)  TX bytes:25298 (24.7 KiB)
[output truncated]
```

To view all existing bonds, even if they are not up, run:

```
~]# cat /sys/class/net/bonding_masters
bond0
```

You can configure each bond individually by manipulating the files located in the **`/sys/class/net/bond<N>/bonding/`** directory. First, the bond you are configuring must be taken down:

```
ifconfig bond0 down
```

As an example, to enable MII monitoring on bond0 with a 1 second interval, you could run (as root):

```
echo 1000 > /sys/class/net/bond0/bonding/miimon
```

To configure bond0 for ***`balance-alb`*** mode, you could run either:

```
echo 6 > /sys/class/net/bond0/bonding/mode
```

...or, using the name of the mode:

```
echo balance-alb > /sys/class/net/bond0/bonding/mode
```

After configuring some options for the bond in question, you can bring it up and test it by running **`ifconfig bond<N> up`** . If you decide to change the options, take the interface down, modify its parameters using **`sysfs`**, bring it back up, and re-test.

Once you have determined the best set of parameters for your bond, add those parameters as a space-separated list to the ***`BONDING_OPTS=`*** directive of the **`/etc/sysconfig/network-scripts/ifcfg-bond<N>`** file for the bonded interface you are configuring. Whenever that bond is brought up (for example, by the system during the boot sequence if the ***`ONBOOT=yes`*** directive is set), the bonding options specified in the ***`BONDING_OPTS`*** will take effect for that bond. For more information on configuring bonded interfaces (and ***`BONDING_OPTS`***), refer to Section 16.2.3, “Channel Bonding Interfaces”.

The following is a list of available channel bonding module parameters for the **`bonding`** module. For more in-depth information on configuring channel bonding and the exhaustive list of bonding module parameters, install the *kernel-doc* package and then locating and opening the included **`bonding.txt`** file:

```
yum -y install kernel-doc
nano -w $(rpm -ql kernel-doc | grep bonding.txt)
```

Bonding Interface Parameters

`arp_interval=<`*`time_in_milliseconds`*`>`

Specifies (in milliseconds) how often ARP monitoring occurs.

Important

It is essential that both **`arp_interval`** and **`arp_ip_target`** parameters are specified, or, alternatively, the **`miimon`** parameter is specified. Failure to do so can cause degradation of network performance in the event that a link fails.

If using this setting while in **`mode=0`** or **`mode=1`** (the two load-balancing modes), the network switch must be configured to distribute packets evenly across the NICs. For more information on how to accomplish this, refer to **`/usr/share/doc/kernel-doc-<kernel_version>/Documentation/networking/bonding.txt`**

The value is set to **`0`** by default, which disables it.

`arp_ip_target=<ip_address> [,<ip_address_2>,...<ip_address_16> ]`

Specifies the target IP address of ARP requests when the **`arp_interval`** parameter is enabled. Up to 16 IP addresses can be specified in a comma separated list.

`arp_validate=<value>`

Validate source/distribution of ARP probes; default is **none**. Other valid values are **`active`**, **`backup`**, and **`all`**.

`debug=<number>`

Enables debug messages. Possible values are:

- **`0`** — Debug messages are disabled. This is the default.
- **`1`** — Debug messages are enabled.

`downdelay=<time_in_milliseconds>`

Specifies (in milliseconds) how long to wait after link failure before disabling the link. The value must be a multiple of the value specified in the **`miimon`** parameter. The value is set to **`0`** by default, which disables it.

lacp_rate=<*value*>

Specifies the rate at which link partners should transmit LACPDU packets in 802.3ad mode. Possible values are:

- **`slow`** or **`0`** — Default setting. This specifies that partners should transmit LACPDUs every 30 seconds.
- **`fast`** or **`1`** — Specifies that partners should transmit LACPDUs every 1 second.

`miimon=<time_in_milliseconds>`

Specifies (in milliseconds) how often MII link monitoring occurs. This is useful if high availability is required because MII is used to verify that the NIC is active. To verify that the driver for a particular NIC supports the MII tool, type the following command as root:

```
ethtool <interface_name> | grep "Link detected:"
```

In this command, replace <*interface_name*> with the name of the device interface, such as **`eth0`**, not the bond interface. If MII is supported, the command returns:

```
Link detected: yes
```

If using a bonded interface for high availability, the module for each NIC must support MII. Setting the value to **`0`** (the default), turns this feature off. When configuring this setting, a good starting point for this parameter is **`100`**.

Important

It is essential that both **`arp_interval`** and **`arp_ip_target`** parameters are specified, or, alternatively, the **`miimon`** parameter is specified. Failure to do so can cause degradation of network performance in the event that a link fails.

`mode=<`*`value`*`>`

...where <*value*> is one of:

- **`balance-rr`** or **`0`** — Sets a round-robin policy for fault tolerance and load balancing. Transmissions are received and sent out sequentially on each bonded slave interface beginning with the first one available.
- **`active-backup`** or **`1`** — Sets an active-backup policy for fault tolerance. Transmissions are received and sent out via the first available bonded slave interface. Another bonded slave interface is only used if the active bonded slave interface fails.
- **`balance-xor`** or **`2`** — Sets an XOR (exclusive-or) policy for fault tolerance and load balancing. Using this method, the interface matches up the incoming request's MAC address with the MAC address for one of the slave NICs. Once this link is established, transmissions are sent out sequentially beginning with the first available interface.
- **`broadcast`** or **`3`** — Sets a broadcast policy for fault tolerance. All transmissions are sent on all slave interfaces.
- **`802.3ad`** or **`4`** — Sets an IEEE 802.3ad dynamic link aggregation policy. Creates aggregation groups that share the same speed and duplex settings. Transmits and receives on all slaves in the active aggregator. Requires a switch that is 802.3ad compliant.
- **`balance-tlb`** or **`5`** — Sets a Transmit Load Balancing (TLB) policy for fault tolerance and load balancing. The outgoing traffic is distributed according to the current load on each slave interface. Incoming traffic is received by the current slave. If the receiving slave fails, another slave takes over the MAC address of the failed slave.
- **`balance-alb`** or **`6`** — Sets an Active Load Balancing (ALB) policy for fault tolerance and load balancing. Includes transmit and receive load balancing for IPV4 traffic. Receive load balancing is achieved through ARP negotiation.

`num_unsol_na=<`*`number`*`>`

Specifies the number of unsolicited IPv6 Neighbor Advertisements to be issued after a failover event. One unsolicited NA is issued immediately after the failover.

The valid range is **`0 - 255`**; the default value is **`1`**. This option affects only the active-backup mode.

`primary=<`*`interface_name`*`>`

Specifies the interface name, such as **`eth0`**, of the primary device. The **`primary`** device is the first of the bonding interfaces to be used and is not abandoned unless it fails. This setting is particularly useful when one NIC in the bonding interface is faster and, therefore, able to handle a bigger load.

This setting is only valid when the bonding interface is in **`active-backup`** mode. Refer to **`/usr/share/doc/kernel-doc-<`*`kernel-version`*`>/Documentation/networking/bonding.txt`** for more information.

primary_reselect=<*value*>

Specifies the reselection policy for the primary slave. This affects how the primary slave is chosen to become the active slave when failure of the active slave or recovery of the primary slave occurs. This option is designed to prevent flip-flopping between the primary slave and other slaves. Possible values are:

- **always** or **0** (default) — The primary slave becomes the active slave whenever it comes back up.
- **better** or **1** — The primary slave becomes the active slave when it comes back up, if the speed and duplex of the primary slave is better than the speed and duplex of the current active slave.
- **failure** or **2** — The primary slave becomes the active slave only if the current active slave fails and the primary slave is up.

The **primary_reselect** setting is ignored in two cases:

- If no slaves are active, the first slave to recover is made the active slave.
- When initially enslaved, the primary slave is always made the active slave.

Changing the **primary_reselect** policy via **sysfs** will cause an immediate selection of the best active slave according to the new policy. This may or may not result in a change of the active slave, depending upon the circumstances

updelay=<*time_in_milliseconds*>

Specifies (in milliseconds) how long to wait before enabling a link. The value must be a multiple of the value specified in the **miimon** parameter. The value is set to **0** by default, which disables it.

use_carrier=<*number*>

Specifies whether or not **miimon** should use MII/ETHTOOL ioctls or **netif_carrier_ok()** to determine the link state. The **netif_carrier_ok()** function relies on the device driver to maintains its state with **netif_carrier_*on/off*** ; most device drivers support this function.

The MII/ETHROOL ioctls tools utilize a deprecated calling sequence within the kernel. However, this is still configurable in case your device driver does not support **netif_carrier_*on/off*** .

Valid values are:

- **1** — Default setting. Enables the use of **netif_carrier_ok()**.
- **0** — Enables the use of MII/ETHTOOL ioctls.

Note

If the bonding interface insists that the link is up when it should not be, it is possible that your network device driver does not support **netif_carrier_*on/off*** .

xmit_hash_policy=<*value*>

Selects the transmit hash policy used for slave selection in **balance-xor** and **802.3ad** modes. Possible values are:

- **`0`** or **`layer2`** — Default setting. This option uses the XOR of hardware MAC addresses to generate the hash. The formula used is:

  ```
  (<source_MAC_address> XOR <destination_MAC>) MODULO <slave_count>
  ```

  This algorithm will place all traffic to a particular network peer on the same slave, and is 802.3ad compliant.

- **`1`** or **`layer3+4`** — Uses upper layer protocol information (when available) to generate the hash. This allows for traffic to a particular network peer to span multiple slaves, although a single connection will not span multiple slaves.

  The formula for unfragmented TCP and UDP packets used is:

  ```
  ((<source_port> XOR <dest_port>) XOR
    ((<source_IP> XOR <dest_IP>) AND 0xffff)
      MODULO <slave_count>
  ```

  For fragmented TCP or UDP packets and all other IP protocol traffic, the source and destination port information is omitted. For non-IP traffic, the formula is the same as the **`layer2`** transmit hash policy.

  This policy intends to mimic the behavior of certain switches; particularly, Cisco switches with PFC2 as well as some Foundry and IBM products.

  The algorithm used by this policy is not 802.3ad compliant.

- **`2`** or **`layer2+3`** — Uses a combination of layer2 and layer3 protocol information to generate the hash.

  Uses XOR of hardware MAC addresses and IP addresses to generate the hash. The formula is:

  ```
  (((<source_IP> XOR <dest_IP>) AND 0xffff) XOR
    ( <source_MAC> XOR <destination_MAC> ))
      MODULO <slave_count>
  ```

  This algorithm will place all traffic to a particular network peer on the same slave. For non-IP traffic, the formula is the same as for the layer2 transmit hash policy.

  This policy is intended to provide a more balanced distribution of traffic than layer2 alone, especially in environments where a layer3 gateway device is required to reach most destinations.

  This algorithm is 802.3ad compliant.

45.6. Additional Resources

For more information on kernel modules and their utilities, refer to the following resources.

45.6.1. Installed Documentation

- **`lsmod`** man page — description and explanation of its output.
- **`insmod`** man page — description and list of command line options.

- **`modprobe`** man page — description and list of command line options.
- **`rmmod`** man page — description and list of command line options.
- **`modinfo`** man page — description and list of command line options.
- **`/usr/share/doc/kernel-doc-<`*`version`*`>/Documentation/kbuild/modules.txt`** — how to compile and use kernel modules. Note you must have the **`kernel-doc`** package installed to read this file.

45.6.2. Useful Websites

- http://tldp.org/HOWTO/Module-HOWTO/ — *Linux Loadable Kernel Module HOWTO* from the Linux Documentation Project.

[9] A driver is software which enables Linux to use a particular hardware device. Without a driver, the kernel cannot communicate with attached devices.

Chapter 46. The kdump Crash Recovery Service

kdump is an advanced crash dumping mechanism. When enabled, the system is booted from the context of another kernel. This second kernel reserves a small amount of memory and its only purpose is to capture the core dump image in case the system crashes. The ability to analyze the core dump significantly helps to determine the exact cause of the system failure, and as a consequence, it is strongly recommended to have this feature enabled.

This chapter explains how to configure, test, and use the **kdump** service in Red Hat Enterprise Linux, and provides a brief overview of how to analyze the resulting core dump using the **crash** debugging utility.

46.1. Installing the kdump Service

In order to use the **kdump** service on your system, make sure you have the *kexec-tools* package installed. To do so, type the following at a shell prompt as **root**:

```
~]# yum install kexec-tools
```

For more information on how to install new packages in Red Hat Enterprise Linux, refer to Part II, “Package Management”.

46.2. Configuring the kdump Service

There are three common means of configuring the **kdump** service: you can enable and configure it at the first boot, use the **Kernel Dump Configuration** utility for the graphical user interface, or do so manually on the command line.

Important

A limitation in the current implementation of the **Intel IOMMU** driver can occasionally prevent the **kdump** service from capturing the core dump image. To use **kdump** on Intel architectures reliably, it is advised that the IOMMU support is disabled.

Warning

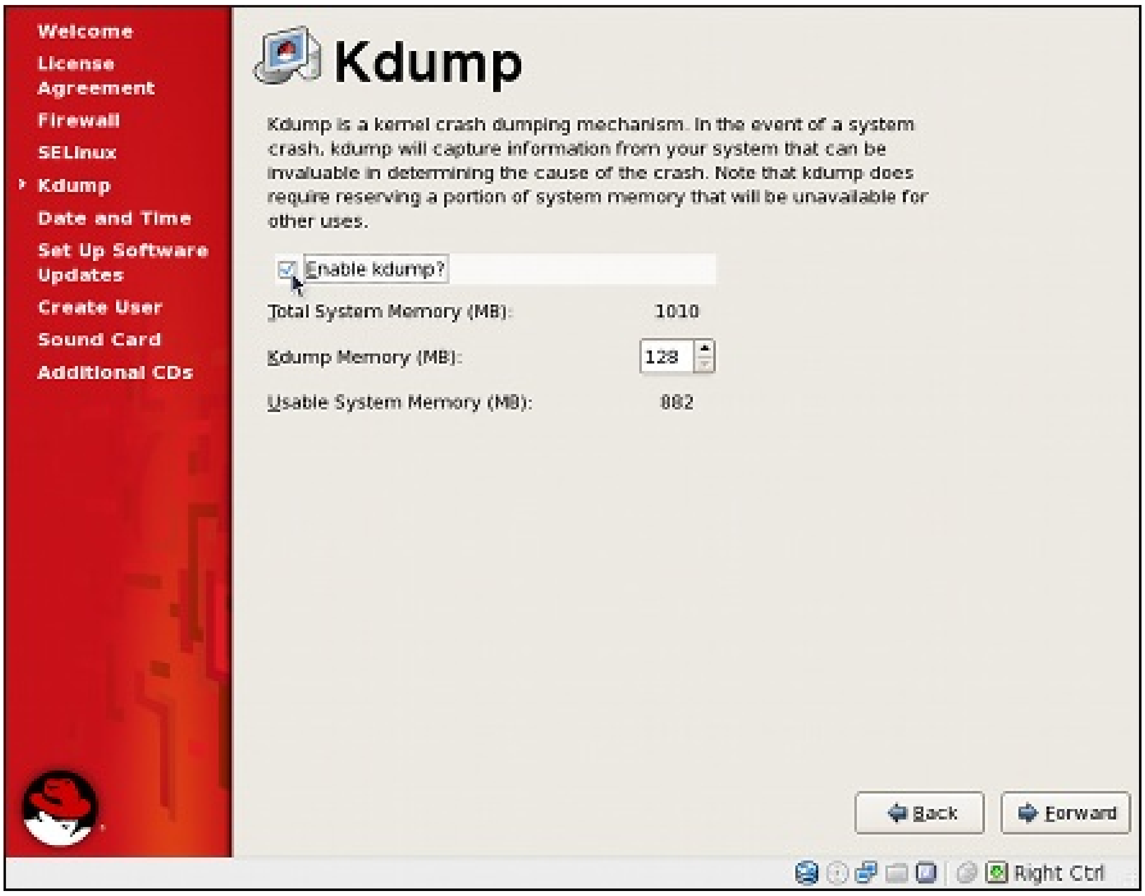
It is known that the **kdump** service does not work reliably on certain combinations of HP Smart Array devices and system boards from the same vendor. Consequent to this, users are strongly advised to test the configuration before using it in production environment, and if necessary, configure **kdump** to store the kernel crash dump to a remote machine over a network. For more information on how to test the **kdump** configuration, refer to Section 46.2.4, “Testing the Configuration”.

46.2.1. Configuring kdump at First Boot

When the system boots for the first time, the **firstboot** application is launched to guide the user through the initial configuration of the freshly installed system. To configure **kdump**, navigate to the **Kdump** page and follow the instructions below.

Important

Unless the system has enough memory, the **`Kdump`** page will not be available. For information on minimum memory requirements, refer to the Red Hat Enterprise Linux comparison chart. When the **`kdump`** crash recovery is enabled, the minimum memory requirements increase by the amount of memory reserved for it. This value is determined by the user and on x86, AMD64, and Intel 64 architectures, it defaults to 128 MB plus 64 MB for each TB of physical memory (that is, a total of 192 MB for a system with 1 TB of physical memory).



Kdump

Kdump is a kernel crash dumping mechanism. In the event of a system crash, kdump will capture information from your system that can be invaluable in determining the cause of the crash. Note that kdump does require reserving a portion of system memory that will be unavailable for other uses.

[✓] Enable kdump?

Total System Memory (MB): 1010

Kdump Memory (MB): 128

Usable System Memory (MB): 882

Figure 46.1. The kdump configuration screen

46.2.1.1. Enabling the Service

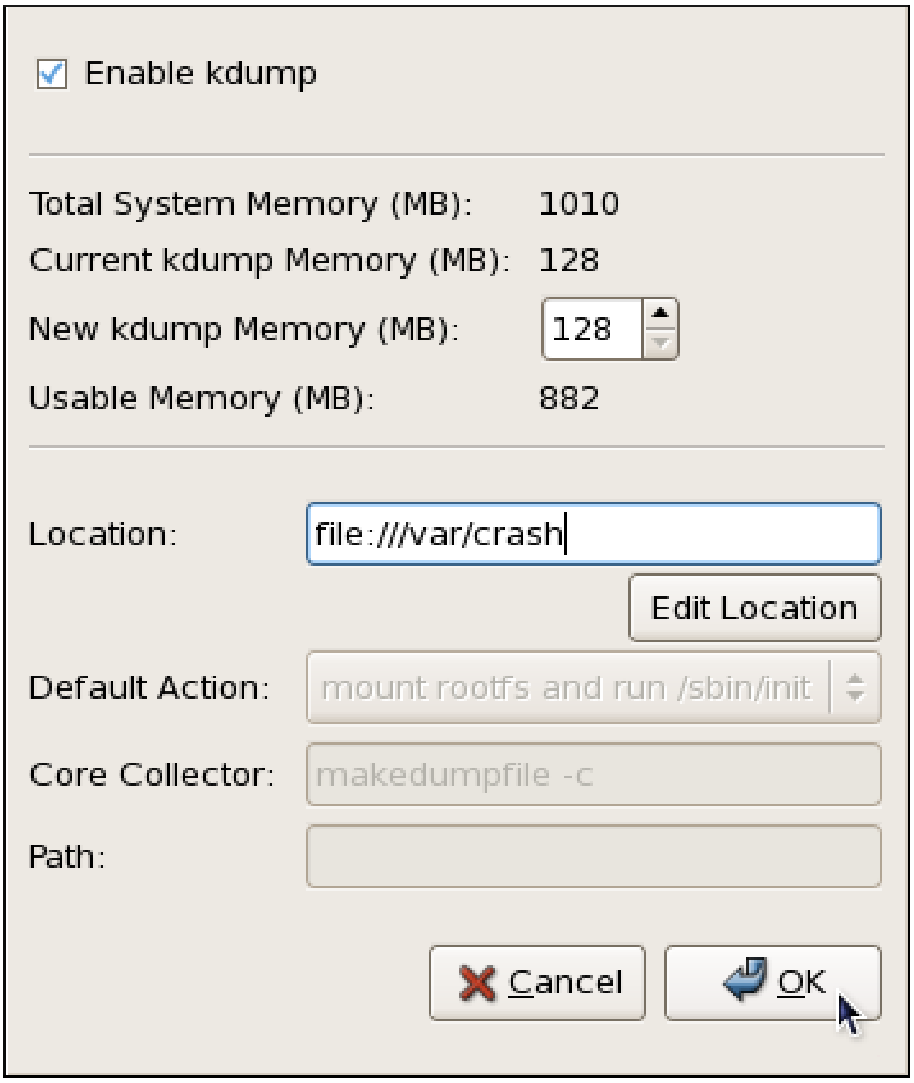
To start the **`kdump`** daemon at boot time, select the **`Enable kdump?`** checkbox. This will enable the service for runlevels **`2`**, **`3`**, **`4`**, and **`5`**, and start it for the current session. Similarly, clearing the checkbox will disable it for all runlevels and stop the service immediately.

46.2.1.2. Configuring the Memory Usage

To configure the amount of memory that is reserved for the **`kdump`** kernel, click the up and down arrow buttons next to the **`Kdump Memory`** field to increase or decrease the value. Notice that the **`Usable System Memory`** field changes accordingly showing you the remaining memory that will be available to the system.

46.2.2. Using the Kernel Dump Configuration Utility

To start the **Kernel Dump Configuration** utility, select **Applications** → **System Tools** → **Kdump** from the panel, or type **`system-config-kdump`** at a shell prompt. Unless you are already authenticated, you will be prompted to enter the **`root`** password.



Enable kdump: ✓

Total System Memory (MB): 1010
Current kdump Memory (MB): 128
New kdump Memory (MB): 128
Usable Memory (MB): 882

Location: file:///var/crash

Default Action: mount rootfs and run /sbin/init
Core Collector: makedumpfile -c
Path:

Figure 46.2. The Kernel Dump Configuration utility

The utility allows you to configure **kdump** as well as to enable or disable starting the service at boot time. When you are done, click **`OK`** to save the changes. The system reboot will be requested.

Important

Unless the system has enough memory, the **Kernel Dump Configuration** utility will not start and you will be presented with an error message. For information on minimum memory requirements, refer to the Red Hat Enterprise Linux comparison chart. When the **kdump** crash recovery is enabled, the minimum memory requirements increase by the amount of memory reserved for it. This value is determined by the user and on x86, AMD64, and Intel 64 architectures, it defaults to 128 MB plus 64 MB for each TB of physical memory (that is, a total of 192 MB for a system with 1 TB of physical memory).

46.2.2.1. Enabling the Service

To start the **kdump** daemon at boot time, select the **Enable kdump** checkbox. This will enable the service for runlevels **2**, **3**, **4**, and **5**, and start it for the current session. Similarly, clearing the checkbox will disable it for all runlevels and stop the service immediately.

46.2.2.2. Configuring the Memory Usage

To configure the amount of memory that is reserved for the **kdump** kernel, click the up and down arrow buttons next to the **New kdump Memory** field to increase or decrease the value. Notice that the **Usable Memory** field changes accordingly showing you the remaining memory that will be available to the system.

46.2.2.3. Configuring the Target Type

When a kernel crash is captured, the core dump can be either stored as a file in a local file system, written directly to a device, or sent over a network using the NFS (Network File System) or SSH (Secure Shell) protocol. To change this, click the **Edit Location** button, and select a location type as described below.

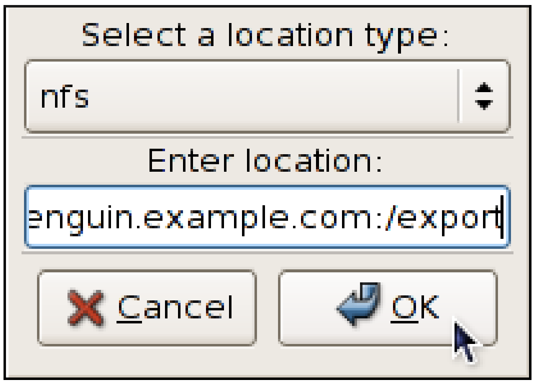


Figure 46.3. The Edit Location dialog

To save the dump to the local file system, select **file** from the pulldown list. Optionally, if you wish to write the file to a different partition, select **ext3** or **ext2** from the pulldown list according to the file system you are using, and enter a valid device name to the **Enter location** field. Note that after clicking **OK**, you can then customize the destination directory by changing the value in the **Path** field at the bottom.

To write the dump directly to a device, select **raw** from the pulldown list, and enter a valid device name (for example, **/dev/sdb1**). When you are done, click **OK** to confirm your choice.

To store the dump to a remote machine using the NFS protocol, select **nfs** from the pulldown list, and enter a valid target in the *hostname:directory* form (for example, **penguin.example.com:/export**). Clicking the **OK** button will confirm your changes. Finally, edit the value of the **Path** field to customize the destination directory (for instance, **cores**).

To store the dump to a remote machine using the SSH protocol, select **ssh** from the pulldown list, and enter a valid username and hostname in the *username@hostname* form (for example, **john@penguin.example.com**). Clicking the **OK** button will confirm your changes. Finally, edit the value of the **Path** field to customize the destination directory (for instance, **/export/cores**).

Refer to Chapter 20, *OpenSSH* for information on how to configure an SSH server, and how to set up a key-based authentication.

46.2.2.4. Configuring the Core Collector

To reduce the size of the **`vmcore`** dump file, **`kdump`** allows you to specify an external application (that is, a core collector) to compress the data, and optionally leave out all irrelevant information. Currently, the only fully supported core collector is **`makedumpfile`**.

To enable the dump file compression, make sure the **`-c`** parameter is listed after the **`makedumpfile`** command in the **`Core Collector`** field (for example, **`makedumpfile -c`**).

To remove certain pages from the dump, add the **`-d`** ***`value`*** parameter after the **`makedumpfile`** command in the **`Core Collector`** field. The *value* is a sum of values of pages you want to omit as described in Table 46.1, “Supported filtering levels”. For example, to remove both zero and free pages, use **`makedumpfile -d 17`**.

Refer to the manual page for **`makedumpfile`** for a complete list of available options.

46.2.2.5. Changing the Default Action

To choose what action to perform when **kdump** fails to create a core dump, select the appropriate option from the **`Default Action`** pulldown list. Available options are **mount rootfs and run /sbin/init** (the default action), **reboot** (to reboot the system), **shell** (to present a user with an interactive shell prompt), and **halt** (to halt the system).

46.2.3. Configuring `kdump` on the Command Line

46.2.3.1. Configuring the Memory Usage

To configure the amount of memory that is reserved for the **`kdump`** kernel on x86, AMD64, and Intel 64 architectures, open the **`/boot/grub/grub.conf`** file as **`root`** and add the **`crashkernel=<`*`size`*`>M@16M`** parameter to the list of kernel options as shown in Example 46.1, “Sample `/boot/grub/grub.conf` file”.

Important

Unless the system has enough memory, the **`kdump`** crash recovery service will not be operational. For information on minimum memory requirements, refer to the Red Hat Enterprise Linux comparison chart. When **`kdump`** is enabled, the minimum memory requirements increase by the amount of memory reserved for it. This value is determined by the user and on x86, AMD64, and Intel 64 architectures, it defaults to 128 MB plus 64 MB for each TB of physical memory (that is, a total of 192 MB for a system with 1 TB of physical memory).

Example 46.1. Sample `/boot/grub/grub.conf` file

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this
file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,0)
#          kernel /vmlinuz-version ro root=/dev/sda3
#          initrd /initrd-version.img
#boot=/dev/sda
default=0
```

```
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title Red Hat Enterprise Linux Server (2.6.18-274.3.1.el5)
        root (hd0,0)
        kernel /vmlinuz-2.6.18-274.3.1.el5 ro root=/dev/sda3
crashkernel=128M@16M
        initrd /initrd-2.6.18-274.3.1.el5.img
```

46.2.3.2. Configuring the Target Type

When a kernel crash is captured, the core dump can be either stored as a file in a local file system, written directly to a device, or sent over a network using the NFS (Network File System) or SSH (Secure Shell) protocol. Note that only one of these options can be set at the moment. The default option is to store the **`vmcore`** file in the **`/var/crash/`** directory of the local file system. To change this, open the **`/etc/kdump.conf`** configuration file as **`root`** and edit the options as described below.

To change the local directory in which the core dump is to be saved, remove the hash sign ("#") from the beginning of the **`#path /var/crash`** line, and replace the value with a desired directory path. Optionally, if you wish to write the file to a different partition, follow the same procedure with the **`#ext3 /dev/sda3`** line as well, and change both the file system type and the device (a device name, a file system label, and UUID are all supported) accordingly. For example:

```
ext3 /dev/sda4
path /usr/local/cores
```

To write the dump directly to a device, remove the hash sign ("#") from the beginning of the **`#raw /dev/sda5`** line, and replace the value with a desired device name. For example:

```
raw /dev/sdb1
```

To store the dump to a remote machine using the NFS protocol, remove the hash sign ("#") from the beginning of the **`#net my.server.com:/export/tmp`** line, and replace the value with a valid hostname and directory path. For example:

```
net penguin.example.com:/export/cores
```

To store the dump to a remote machine using the SSH protocol, remove the hash sign ("#") from the beginning of the **`#net user@my.server.com`** line, and replace the value with a valid username and hostname. For example:

```
net john@penguin.example.com
```

Refer to Chapter 20, *OpenSSH* for information on how to configure an SSH server, and how to set up a key-based authentication.

46.2.3.3. Configuring the Core Collector

To reduce the size of the **`vmcore`** dump file, **`kdump`** allows you to specify an external application (that is, a core collector) to compress the data, and optionally leave out all irrelevant information. Currently, the only fully supported core collector is **`makedumpfile`**.

To enable the core collector, open the **`/etc/kdump.conf`** configuration file as **`root`**, remove the hash sign

("#") from the beginning of the **#core_collector makedumpfile -c --message-level 1** line, and edit the command line options as described below.

To enable the dump file compression, add the **-c** parameter. For example:

```
core_collector makedumpfile -c
```

To remove certain pages from the dump, add the **-d *value*** parameter, where *value* is a sum of values of pages you want to omit as described in Table 46.1, "Supported filtering levels". For example, to remove both zero and free pages, use the following:

```
core_collector makedumpfile -d 17 -c
```

Refer to the manual page for **makedumpfile** for a complete list of available options.

Table 46.1. Supported filtering levels

Option	Description
1	Zero pages
2	Cache pages
4	Cache private
8	User pages
16	Free pages

46.2.3.4. Changing the Default Action

By default, when **kdump** fails to create a core dump, the root file system is mounted and **/sbin/init** is run. To change this behavior, open the **/etc/kdump.conf** configuration file as **root**, remove the hash sign ("#") from the beginning of the **#default shell** line, and replace the value with a desired action as described in Table 46.2, "Supported actions". For example:

```
default halt
```

Table 46.2. Supported actions

Option	Action
reboot	Reboot the system, losing the core in the process.
halt	After failing to capture a core, halt the system.
shell	Run the **msh** session from within the initramfs, allowing a user to record the core manually.

46.2.3.5. Enabling the Service

To start the **kdump** daemon at boot time, type the following at a shell prompt as **root**:

```
~]# chkconfig kdump on
```

This will enable the service for runlevels **2**, **3**, **4**, and **5**. Similarly, typing **chkconfig kdump off** will disable it for all runlevels. To start the service in the current session, use the following command as **root**:

```
~]# service kdump start
```

```
No kdump initial ramdisk found.                                [WARNING]
Rebuilding /boot/initrd-2.6.18-194.8.1.el5kdump.img
Starting kdump:                                                [  OK  ]
```

For more information on runlevels and configuring services in general, refer to *Chapter 18, Controlling Access to Services*.

46.2.4. Testing the Configuration

Warning

The commands below will cause the kernel to crash. Use caution when following these steps, and by no means use them on a production machine.

To test the configuration, reboot the system with **kdump** enabled, and as **root**, make sure that the service is running:

```
~]# service kdump status
Kdump is operational
```

Then type the following commands at a shell prompt as **root**:

```
~]# echo 1 > /proc/sys/kernel/sysrq
~]# echo c > /proc/sysrq-trigger
```

This will force the Linux kernel to crash, and the ***YYYY-MM-DD-HH:MM*/vmcore** file will be copied to the location you have selected in the configuration (that is, to **/var/crash/** by default).

46.3. Analyzing the Core Dump

Note

To analyze the **vmcore** dump file, you must have the *crash* and *kernel-debuginfo* packages installed. To do so, type the following at a shell prompt as **root**:

```
~]# yum install --enablerepo=rhel-debuginfo crash kernel-debuginfo
```

Refer to Part II, “Package Management” for more information on how to install new packages in Red Hat Enterprise Linux.

To determine the cause of the system crash, you can use the **crash** utility. This utility allows you to interactively analyze a running Linux system as well as a core dump created by **netdump**, **diskdump**, **xendump**, or **kdump**. When started, it presents you with an interactive prompt very similar to the GNU Debugger (GDB).

To start the utility, type the command in the following form at a shell prompt:

```
crash /var/crash/timestamp/vmcore /usr/lib/debug/lib/modules/kernel/vmlinux
```

Note that the *kernel* version should be the same as the one that was captured by **kdump**. To find out which kernel you are currently running, use the **uname -r** command.

Example 46.2. Running the crash utility

```
~]# crash /var/crash/2010-08-04-17\:55/vmcore \
/usr/lib/debug/lib/modules/2.6.18-194.8.1.el5/vmlinux

crash 4.1.2-4.el5_5.1
Copyright (C) 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009  Red Hat,
Inc.
Copyright (C) 2004, 2005, 2006  IBM Corporation
Copyright (C) 1999-2006  Hewlett-Packard Co
Copyright (C) 2005, 2006  Fujitsu Limited
Copyright (C) 2006, 2007  VA Linux Systems Japan K.K.
Copyright (C) 2005  NEC Corporation
Copyright (C) 1999, 2002, 2007  Silicon Graphics, Inc.
Copyright (C) 1999, 2000, 2001, 2002  Mission Critical Linux, Inc.
This program is free software, covered by the GNU General Public License,
and you are welcome to change it and/or distribute copies of it under
certain conditions.  Enter "help copying" to see the conditions.
This program has absolutely no warranty.  Enter "help warranty" for
details.

GNU gdb 6.1
Copyright 2004 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you
are
welcome to change it and/or distribute copies of it under certain
conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB.  Type "show warranty" for
details.
This GDB was configured as "i686-pc-linux-gnu"...

      KERNEL: /usr/lib/debug/lib/modules/2.6.18-194.8.1.el5/vmlinux
    DUMPFILE: /var/crash/2010-08-04-17:55/vmcore
        CPUS: 1
        DATE: Wed Aug  4 17:50:41 2010
      UPTIME: 00:56:53
LOAD AVERAGE: 0.47, 0.47, 0.55
       TASKS: 128
    NODENAME: localhost.localdomain
     RELEASE: 2.6.18-194.el5
     VERSION: #1 SMP Tue Mar 16 21:52:43 EDT 2010
     MACHINE: i686  (2702 Mhz)
      MEMORY: 1 GB
       PANIC: "SysRq : Trigger a crashdump"
         PID: 6042
     COMMAND: "bash"
        TASK: f09c7000  [THREAD_INFO: e1ba9000]
```

```
          CPU: 0
        STATE: TASK_RUNNING (SYSRQ)

crash>
```

To exit the interactive prompt and terminate **`crash`**, type **`exit`**.

46.3.1. Displaying the Message Buffer

To display the kernel message buffer, type the **`log`** command at the interactive prompt.

Example 46.3. Displaying the kernel message buffer

```
crash> log
Linux version 2.6.18-194.el5 (mockbuild@x86-007.build.bos.redhat.com) (gcc
version 4.1.2 20080704 (Red Hat 4.1.2-48)) #1 SMP Tue Mar 16 21:52:43 EDT
2010
BIOS-provided physical RAM map:
 BIOS-e820: 0000000000010000 - 000000000009fc00 (usable)
 BIOS-e820: 000000000009fc00 - 00000000000a0000 (reserved)
 BIOS-e820: 00000000000f0000 - 0000000000100000 (reserved)
 BIOS-e820: 0000000000100000 - 000000003fff0000 (usable)
 BIOS-e820: 000000003fff0000 - 0000000040000000 (ACPI data)
 BIOS-e820: 00000000fffc0000 - 0000000100000000 (reserved)
127MB HIGHMEM available.
896MB LOWMEM available.
Using x86 segment limits to approximate NX protection
On node 0 totalpages: 262128
  DMA zone: 4096 pages, LIFO batch:0
  Normal zone: 225280 pages, LIFO batch:31
  HighMem zone: 32752 pages, LIFO batch:7
DMI 2.5 present.
Using APIC driver default
... several lines omitted ...
SysRq : Trigger a crashdump
```

Type **`help log`** for more information on the command usage.

46.3.2. Displaying a Backtrace

To display the kernel stack trace, type the **`bt`** command at the interactive prompt. You can use **`bt`** ***`pid`*** to display the backtrace of the selected process.

Example 46.4. Displaying the kernel stack trace

```
crash> bt
PID: 6042   TASK: f09c7000  CPU: 0   COMMAND: "bash"
 #0 [e1ba9d10] schedule at c061c738
 #1 [e1ba9d28] netlink_getsockopt at c05d50bb
 #2 [e1ba9d34] netlink_queue_skip at c05d40d5
 #3 [e1ba9d40] netlink_sock_destruct at c05d506d
 #4 [e1ba9d84] sock_recvmsg at c05b6cc8
 #5 [e1ba9dd4] enqueue_task at c041eed5
 #6 [e1ba9dec] try_to_wake_up at c041f798
 #7 [e1ba9e10] vsnprintf at c04efef2
 #8 [e1ba9ec0] machine_kexec at c0419bf0
 #9 [e1ba9f04] sys_kexec_load at c04448a1
#10 [e1ba9f4c] tty_audit_exit at c0549f06
#11 [e1ba9f50] tty_audit_add_data at c0549d5d
#12 [e1ba9f84] do_readv_writev at c0476055
#13 [e1ba9fb8] system_call at c0404f10
    EAX: ffffffda  EBX: 00000001  ECX: b7f7f000  EDX: 00000002
    DS:  007b      ESI: 00000002  ES:  007b      EDI: b7f7f000
    SS:  007b      ESP: bf83f478  EBP: bf83f498
    CS:  0073      EIP: 009ac402  ERR: 00000004  EFLAGS: 00000246
```

Type **help bt** for more information on the command usage.

46.3.3. Displaying a Process Status

To display a status of processes in the system, type the **ps** command at the interactive prompt. You can use **ps *pid*** to display the status of the selected process.

Example 46.5. Displaying status of processes in the system

```
crash> ps
   PID    PPID  CPU   TASK    ST  %MEM     VSZ    RSS  COMM
      0      0   0  c068a3c0  RU   0.0       0      0  [swapper]
      1      0   0  f7c81aa0  IN   0.1    2152    616  init
... several lines omitted ...
   6017      1   0  e39f6550  IN   1.2   40200  13000  gnome-terminal
   6019   6017   0  e39f6000  IN   0.1    2568    708  gnome-pty-helpe
   6020   6017   0  f0421550  IN   0.1    4620   1480  bash
   6021      1   0  f7f69aa0  ??   1.2   40200  13000  gnome-terminal
   6039   6020   0  e7e84aa0  IN   0.1    5004   1300  su
>  6042   6039   0  f09c7000  RU   0.1    4620   1464  bash
```

Type **help ps** for more information on the command usage.

46.3.4. Displaying Virtual Memory Information

To display basic virtual memory information, type the **vm** command at the interactive prompt. You can use **vm *pid*** to display information on the selected process.

Example 46.6. Displaying virtual memory information of the current context

```
crash> vm
PID: 6042   TASK: f09c7000  CPU: 0   COMMAND: "bash"
   MM        PGD       RSS    TOTAL_VM
e275ee40  e2b08000  1464k    4620k
  VMA       START      END    FLAGS  FILE
e315d764    1fe000    201000     75  /lib/libtermcap.so.2.0.8
```

```
e315de9c    201000    202000 100073  /lib/libtermcap.so.2.0.8
c9b040d4    318000    46a000     75  /lib/libc-2.5.so
e315da04    46a000    46c000 100071  /lib/libc-2.5.so
e315d7b8    46c000    46d000 100073  /lib/libc-2.5.so
e315de48    46d000    470000 100073
e315dba8    9ac000    9ad000 8040075
c9b04a04    a2f000    a4a000    875  /lib/ld-2.5.so
c9b04374    a4a000    a4b000 100871  /lib/ld-2.5.so
e315d6bc    a4b000    a4c000 100873  /lib/ld-2.5.so
e315d908    fa1000    fa4000     75  /lib/libdl-2.5.so
e315db00    fa4000    fa5000 100071  /lib/libdl-2.5.so
e315df44    fa5000    fa6000 100073  /lib/libdl-2.5.so
e315d320    ff0000    ffa000     75  /lib/libnss_files-2.5.so
e315d668    ffa000    ffb000 100071  /lib/libnss_files-2.5.so
e315def0    ffb000    ffc000 100073  /lib/libnss_files-2.5.so
e315d374   8048000   80f5000   1875  /bin/bash
c9b045c0   80f5000   80fa000 101873  /bin/bash
... several lines omitted ...
```

Type **help vm** for more information on the command usage.

46.3.5. Displaying Open Files

To display information about open files, type the **files** command at the interactive prompt. You can use **files *pid*** to display files opened by the selected process.

Example 46.7. Displaying information about open files of the current context

```
crash> files
PID: 6042   TASK: f09c7000  CPU: 0   COMMAND: "bash"
ROOT: /    CWD: /root
 FD    FILE      DENTRY    INODE    TYPE  PATH
  0  e33be480  e609bf70  f0e1d880  CHR   /dev/pts/1
  1  e424d8c0  d637add8  f7809b78  REG   /proc/sysrq-trigger
  2  e33be480  e609bf70  f0e1d880  CHR   /dev/pts/1
 10  e33be480  e609bf70  f0e1d880  CHR   /dev/pts/1
255  e33be480  e609bf70  f0e1d880  CHR   /dev/pts/1
```

Type **help files** for more information on the command usage.

46.4. Additional Resources

46.4.1. Installed Documentation

man kdump.conf

The manual page for the **/etc/kdump.conf** configuration file containing the full documentation of available options.

man kexec

The manual page for **`kexec`** containing the full documentation on its usage.

`man crash`

The manual page for the **`crash`** utility containing the full documentation on its usage.

`/usr/share/doc/kexec-tools-`*`version`*`/kexec-kdump-howto.txt`

An overview of the **`kdump`** and **`kexec`** installation and usage.

46.4.2. Useful Websites

https://access.redhat.com/kb/docs/DOC-6039

The Red Hat Knowledgebase article about the **`kexec`** and **`kdump`** configuration.

http://people.redhat.com/anderson/

The **`crash`** utility homepage.

Part VII. Security And Authentication

Whether system administrators need to secure their mission-critical systems, services, or data, Red Hat Enterprise Linux provides a range of tools and methods to serve as part of a comprehensive security strategy.

This chapter provides a general introduction to security, and from the perspective of Red Hat Enterprise Linux in particular. It provides conceptual information in the areas of security assessment, common exploits, and intrusion and incident response. It also provides conceptual and specific configuration information on how to use SELinux to harden Workstation, Server, VPN, firewall and other implementations.

This chapter assumes a basic knowledge of IT security, and consequently provides only minimal coverage of common security practices such as controlling physical access, sound account-keeping policies and procedures, auditing, etc. Where appropriate, reference is made to external resources for this and related information.

Chapter 47. Security Overview

Because of the increased reliance on powerful, networked computers to help run businesses and keep track of our personal information, industries have been formed around the practice of network and computer security. Enterprises have solicited the knowledge and skills of security experts to properly audit systems and tailor solutions to fit the operating requirements of the organization. Because most organizations are dynamic in nature, with workers accessing company IT resources locally and remotely, the need for secure computing environments has become more pronounced.

Unfortunately, most organizations (as well as individual users) regard security as an afterthought, a process that is overlooked in favor of increased power, productivity, and budgetary concerns. Proper security implementation is often enacted *postmortem* — after an unauthorized intrusion has already occurred. Security experts agree that the right measures taken prior to connecting a site to an untrusted network, such as the Internet, is an effective means of thwarting most attempts at intrusion.

47.1. Introduction to Security

47.1.1. What is Computer Security?

Computer security is a general term that covers a wide area of computing and information processing. Industries that depend on computer systems and networks to conduct daily business transactions and access crucial information regard their data as an important part of their overall assets. Several terms and metrics have entered our daily business vocabulary, such as total cost of ownership (TCO) and quality of service (QoS). In these metrics, industries calculate aspects such as data integrity and high-availability as part of their planning and process management costs. In some industries, such as electronic commerce, the availability and trustworthiness of data can be the difference between success and failure.

47.1.1.1. How did Computer Security Come about?

Information security has evolved over the years due to the increasing reliance on public networks not to disclose personal, financial, and other restricted information. There are numerous instances such as the Mitnick and the Vladimir Levin cases that prompted organizations across all industries to rethink the way they handle information transmission and disclosure. The popularity of the Internet was one of the most important developments that prompted an intensified effort in data security.

An ever-growing number of people are using their personal computers to gain access to the resources that th Internet has to offer. From research and information retrieval to electronic mail and commerce transaction, the Internet has been regarded as one of the most important developments of the 20th century.

The Internet and its earlier protocols, however, were developed as a *trust-based* system. That is, the Internet Protocol was not designed to be secure in itself. There are no approved security standards built into the TCP/IP communications stack, leaving it open to potentially malicious users and processes across the network. Modern developments have made Internet communication more secure, but there are still several incidents that gain national attention and alert us to the fact that nothing is completely safe.

47.1.1.2. Security Today

In February of 2000, a Distributed Denial of Service (DDoS) attack was unleashed on several of the most heavily-trafficked sites on the Internet. The attack rendered yahoo.com, cnn.com, amazon.com, fbi.gov, and several other sites completely unreachable to normal users, as it tied up routers for several hours with large-byte ICMP packet transfers, also called a *ping flood*. The attack was brought on by unknown assailants using specially created, widely available programs that scanned vulnerable network servers, installed client applications called *Trojans* on the servers, and timed an attack with every infected server flooding the victim sites and rendering them unavailable. Many blame the attack on fundamental flaws in the way routers and the protocols used are structured to accept all incoming data, no matter where or for what purpose the packets are sent.

Currently, an estimated 945 million people use or have used the Internet worldwide (Computer Industry Almanac, 2004). At the same time:

- On any given day, there are approximately 225 major incidences of security breach reported to the CERT Coordination Center at Carnegie Mellon University. [10]
- In 2003, the number of CERT reported incidences jumped to 137,529 from 82,094 in 2002 and from 52,658 in 2001. [11]
- The worldwide economic impact of the three most dangerous Internet Viruses of the last three years was estimated at US$13.2 Billion. [12]

Computer security has become a quantifiable and justifiable expense for all IT budgets. Organizations that require data integrity and high availability elicit the skills of system administrators, developers, and engineers to ensure 24x7 reliability of their systems, services, and information. Falling victim to malicious users, processes, or coordinated attacks is a direct threat to the success of the organization.

Unfortunately, system and network security can be a difficult proposition, requiring an intricate knowledge of how an organization regards, uses, manipulates, and transmits its information. Understanding the way an organization (and the people that make up the organization) conducts business is paramount to implementing a proper security plan.

47.1.1.3. Standardizing Security

Enterprises in every industry rely on regulations and rules that are set by standards making bodies such as the American Medical Association (AMA) or the Institute of Electrical and Electronics Engineers (IEEE). The same ideals hold true for information security. Many security consultants and vendors agree upon the standard security model known as CIA, or *Confidentiality, Integrity, and Availability*. This three-tiered model is a generally accepted component to assessing risks of sensitive information and establishing security policy. The following describes the CIA model in further detail:

- Confidentiality — Sensitive information must be available only to a set of pre-defined individuals. Unauthorized transmission and usage of information should be restricted. For example, confidentiality of information ensures that a customer's personal or financial information is not obtained by an unauthorized individual for malicious purposes such as identity theft or credit fraud.
- Integrity — Information should not be altered in ways that render it incomplete or incorrect. Unauthorized users should be restricted from the ability to modify or destroy sensitive information.
- Availability — Information should be accessible to authorized users any time that it is needed. Availability is a warranty that information can be obtained with an agreed-upon frequency and timeliness. This is often measured in terms of percentages and agreed to formally in Service Level Agreements (SLAs) used by network service providers and their enterprise clients.

47.1.2. Security Controls

Computer security is often divided into three distinct master categories, commonly referred to as *controls*:

- Physical
- Technical

- Administrative

These three broad categories define the main objectives of proper security implementation. Within these controls are sub-categories that further detail the controls and how to implement them.

47.1.2.1. Physical Controls

Physical control is the implementation of security measures in a defined structure used to deter or prevent unauthorized access to sensitive material. Examples of physical controls are:

- Closed-circuit surveillance cameras
- Motion or thermal alarm systems
- Security guards
- Picture IDs
- Locked and dead-bolted steel doors
- Biometrics (includes fingerprint, voice, face, iris, handwriting, and other automated methods used to recognize individuals)

47.1.2.2. Technical Controls

Technical controls use technology as a basis for controlling the access and usage of sensitive data throughout a physical structure and over a network. Technical controls are far-reaching in scope and encompass such technologies as:

- Encryption
- Smart cards
- Network authentication
- Access control lists (ACLs)
- File integrity auditing software

47.1.2.3. Administrative Controls

Administrative controls define the human factors of security. It involves all levels of personnel within an organization and determines which users have access to what resources and information by such means as:

- Training and awareness
- Disaster preparedness and recovery plans
- Personnel recruitment and separation strategies
- Personnel registration and accounting

47.1.3. Conclusion

Now that you have learned about the origins, reasons, and aspects of security, you can determine the appropriate course of action with regard to Red Hat Enterprise Linux. It is important to know what factors and conditions make up security in order to plan and implement a proper strategy. With this information in mind, the process can be formalized and the path becomes clearer as you delve deeper into the specifics of the security process.

47.2. Vulnerability Assessment

Given time, resources, and motivation, a cracker can break into nearly any system. At the end of the day, all of the security procedures and technologies currently available cannot guarantee that any systems are safe from intrusion. Routers help secure gateways to the Internet. Firewalls help secure the edge of the network. Virtual Private Networks safely pass data in an encrypted stream. Intrusion detection systems warn you of malicious activity. However, the success of each of these technologies is dependent upon a number of variables, including:

- The expertise of the staff responsible for configuring, monitoring, and maintaining the technologies.
- The ability to patch and update services and kernels quickly and efficiently.
- The ability of those responsible to keep constant vigilance over the network.

Given the dynamic state of data systems and technologies, securing corporate resources can be quite complex. Due to this complexity, it is often difficult to find expert resources for all of your systems. While it is possible to have personnel knowledgeable in many areas of information security at a high level, it is difficult to retain staff who are experts in more than a few subject areas. This is mainly because each subject area of information security requires constant attention and focus. Information security does not stand still.

47.2.1. Thinking Like the Enemy

Suppose that you administer an enterprise network. Such networks are commonly comprised of operating systems, applications, servers, network monitors, firewalls, intrusion detection systems, and more. Now imagine trying to keep current with each of these. Given the complexity of today's software and networking environments, exploits and bugs are a certainty. Keeping current with patches and updates for an entire network can prove to be a daunting task in a large organization with heterogeneous systems.

Combine the expertise requirements with the task of keeping current, and it is inevitable that adverse incidents occur, systems are breached, data is corrupted, and service is interrupted.

To augment security technologies and aid in protecting systems, networks, and data, you must think like a cracker and gauge the security of your systems by checking for weaknesses. Preventative vulnerability assessments against your own systems and network resources can reveal potential issues that can be addressed before a cracker exploits it.

A vulnerability assessment is an internal audit of your network and system security; the results of which indicate the confidentiality, integrity, and availability of your network (as explained in Section 47.1.1.3, “Standardizing Security”). Typically, vulnerability assessment starts with a reconnaissance phase, during which important data regarding the target systems and resources is gathered. This phase leads to the system readiness phase, whereby the target is essentially checked for all known vulnerabilities. The readiness phase culminates in the reporting phase, where the findings are classified into categories of high, medium, and low risk; and methods for improving the security (or mitigating the risk of vulnerability) of the target are discussed.

If you were to perform a vulnerability assessment of your home, you would likely check each door to your home to see if they are closed and locked. You would also check every window, making sure that they closed completely and latch correctly. This same concept applies to systems, networks, and electronic data. Malicious users are the thieves and vandals of your data. Focus on their tools, mentality, and motivations, anc you can then react swiftly to their actions.

47.2.2. Defining Assessment and Testing

Vulnerability assessments may be broken down into one of two types: *Outside looking in* and *inside looking around*.

When performing an outside looking in vulnerability assessment, you are attempting to compromise your systems from the outside. Being external to your company provides you with the cracker's viewpoint. You see what a cracker sees — publicly-routable IP addresses, systems on your *DMZ*, external interfaces of your firewall, and more. DMZ stands for "demilitarized zone", which corresponds to a computer or small subnetwork that sits between a trusted internal network, such as a corporate private LAN, and an untrusted external network, such as the public Internet. Typically, the DMZ contains devices accessible to Internet traffic, such as Web (HTTP) servers, FTP servers, SMTP (e-mail) servers and DNS servers.

When you perform an inside looking around vulnerability assessment, you are somewhat at an advantage since you are internal and your status is elevated to trusted. This is the viewpoint you and your co-workers have once logged on to your systems. You see print servers, file servers, databases, and other resources.

There are striking distinctions between these two types of vulnerability assessments. Being internal to your company gives you elevated privileges more so than any outsider. Still today in most organizations, security is configured in such a manner as to keep intruders out. Very little is done to secure the internals of the organization (such as departmental firewalls, user-level access controls, authentication procedures for internal resources, and more). Typically, there are many more resources when looking around inside as most systems are internal to a company. Once you set yourself outside of the company, you immediately are given an untrusted status. The systems and resources available to you externally are usually very limited.

Consider the difference between vulnerability assessments and *penetration tests*. Think of a vulnerability assessment as the first step to a penetration test. The information gleaned from the assessment is used for testing. Whereas, the assessment is checking for holes and potential vulnerabilities, the penetration testing actually attempts to exploit the findings.

Assessing network infrastructure is a dynamic process. Security, both information and physical, is dynamic. Performing an assessment shows an overview, which can turn up false positives and false negatives.

Security administrators are only as good as the tools they use and the knowledge they retain. Take any of the assessment tools currently available, run them against your system, and it is almost a guarantee that there are some false positives. Whether by program fault or user error, the result is the same. The tool may find vulnerabilities which in reality do not exist (false positive); or, even worse, the tool may not find vulnerabilities that actually do exist (false negative).

Now that the difference between a vulnerability assessment and a penetration test is defined, take the findings of the assessment and review them carefully before conducting a penetration test as part of your new best practices approach.

Warning

Attempting to exploit vulnerabilities on production resources can have adverse effects to the productivity and efficiency of your systems and network.

The following list examines some of the benefits to performing vulnerability assessments.

- Creates proactive focus on information security
- Finds potential exploits before crackers find them
- Results in systems being kept up to date and patched
- Promotes growth and aids in developing staff expertise
- Abates Financial loss and negative publicity

47.2.2.1. Establishing a Methodology

To aid in the selection of tools for a vulnerability assessment, it is helpful to establish a vulnerability assessment methodology. Unfortunately, there is no predefined or industry approved methodology at this time; however, common sense and best practices can act as a sufficient guide.

What is the target? Are we looking at one server, or are we looking at our entire network and everything within the network? Are we external or internal to the company? The answers to these questions are important as they help determine not only which tools to select but also the manner in which they are used.

To learn more about establishing methodologies, refer to the following websites:

- http://www.isecom.org/projects/osstmm.htm *The Open Source Security Testing Methodology Manual* (OSSTMM)
- http://www.owasp.org/ *The Open Web Application Security Project*

47.2.3. Evaluating the Tools

An assessment can start by using some form of an information gathering tool. When assessing the entire network, map the layout first to find the hosts that are running. Once located, examine each host individually. Focusing on these hosts requires another set of tools. Knowing which tools to use may be the most crucial step in finding vulnerabilities.

Just as in any aspect of everyday life, there are many different tools that perform the same job. This concept applies to performing vulnerability assessments as well. There are tools specific to operating systems, applications, and even networks (based on the protocols used). Some tools are free; others are not. Some tools are intuitive and easy to use, while others are cryptic and poorly documented but have features that other tools do not.

Finding the right tools may be a daunting task and in the end, experience counts. If possible, set up a test lab and try out as many tools as you can, noting the strengths and weaknesses of each. Review the README file or man page for the tool. Additionally, look to the Internet for more information, such as articles, step-by-step guides, or even mailing lists specific to a tool.

The tools discussed below are just a small sampling of the available tools.

47.2.3.1. Scanning Hosts with Nmap

Nmap is a popular tool included in Red Hat Enterprise Linux that can be used to determine the layout of a network. Nmap has been available for many years and is probably the most often used tool when gathering information. An excellent man page is included that provides a detailed description of its options and usage. Administrators can use Nmap on a network to find host systems and open ports on those systems.

Nmap is a competent first step in vulnerability assessment. You can map out all the hosts within your network and even pass an option that allows Nmap to attempt to identify the operating system running on a particular host. Nmap is a good foundation for establishing a policy of using secure services and stopping unused services.

47.2.3.1.1. Using Nmap

Nmap can be run from a shell prompt by typing the **`nmap`** command followed by the hostname or IP address of the machine to scan.

```
nmap foo.example.com
```

The results of the scan (which could take up to a few minutes, depending on where the host is located) shoulc look similar to the following:

```
Starting nmap V. 3.50 ( www.insecure.org/nmap/ )
Interesting ports on localhost.localdomain (127.0.0.1):
(The 1591 ports scanned but not shown below are in state: closed)
Port       State       Service
22/tcp     open        ssh
25/tcp     open        smtp
111/tcp    open        sunrpc
443/tcp    open        https
515/tcp    open        printer
950/tcp    open        oftep-rpc
6000/tcp   open        X11

Nmap run completed -- 1 IP address (1 host up) scanned in 71.825 seconds
```

Nmap tests the most common network communication ports for listening or waiting services. This knowledge can be helpful to an administrator who wants to close down unnecessary or unused services.

For more information about using Nmap, refer to the official homepage at the following URL:

http://www.insecure.org/

47.2.3.2. Nessus

Nessus is a full-service security scanner. The plug-in architecture of Nessus allows users to customize it for their systems and networks. As with any scanner, Nessus is only as good as the signature database it relies upon. Fortunately, Nessus is frequently updated and features full reporting, host scanning, and real-time vulnerability searches. Remember that there could be false positives and false negatives, even in a tool as powerful and as frequently updated as Nessus.

Note

Nessus is not included with Red Hat Enterprise Linux and is not supported. It has been included in this document as a reference to users who may be interested in using this popular application.

For more information about Nessus, refer to the official website at the following URL:

http://www.nessus.org/

47.2.3.3. Nikto

Nikto is an excellent common gateway interface (CGI) script scanner. Nikto not only checks for CGI vulnerabilities but does so in an evasive manner, so as to elude intrusion detection systems. It comes with thorough documentation which should be carefully reviewed prior to running the program. If you have Web servers serving up CGI scripts, Nikto can be an excellent resource for checking the security of these servers.

Note

Nikto is not included with Red Hat Enterprise Linux and is not supported. It has been included in this document as a reference to users who may be interested in using this popular application.

More information about Nikto can be found at the following URL:

http://www.cirt.net/code/nikto.shtml

47.2.3.4. VLAD the Scanner

VLAD is a vulnerabilities scanner developed by the RAZOR team at Bindview, Inc., which checks for the SANS Top Ten list of common security issues (SNMP issues, file sharing issues, etc.). While not as full-featured as Nessus, VLAD is worth investigating.

Note

VLAD is not included with Red Hat Enterprise Linux and is not supported. It has been included in this document as a reference to users who may be interested in using this popular application.

More information about VLAD can be found on the RAZOR team website at the following URL:

http://www.bindview.com/Support/Razor/Utilities/

47.2.3.5. Anticipating Your Future Needs

Depending upon your target and resources, there are many tools available. There are tools for wireless networks, Novell networks, Windows systems, Linux systems, and more. Another essential part of performing assessments may include reviewing physical security, personnel screening, or voice/PBX network assessment. New concepts, such as *war walking* scanning the perimeter of your enterprise's physical structures for wireless network vulnerabilities are some emerging concepts that you can investigate and, if needed, incorporate into your assessments. Imagination and exposure are the only limits of planning and conducting vulnerability assessments.

47.3. Attackers and Vulnerabilities

To plan and implement a good security strategy, first be aware of some of the issues which determined, motivated attackers exploit to compromise systems. But before detailing these issues, the terminology used when identifying an attacker must be defined.

47.3.1. A Quick History of Hackers

The modern meaning of the term *hacker* has origins dating back to the 1960s and the Massachusetts Institute of Technology (MIT) Tech Model Railroad Club, which designed train sets of large scale and intricate detail. Hacker was a name used for club members who discovered a clever trick or workaround for a problem.

The term hacker has since come to describe everything from computer buffs to gifted programmers. A common trait among most hackers is a willingness to explore in detail how computer systems and networks function with little or no outside motivation. Open source software developers often consider themselves and their colleagues to be hackers, and use the word as a term of respect.

Typically, hackers follow a form of the *hacker ethic* which dictates that the quest for information and expertise is essential, and that sharing this knowledge is the hackers duty to the community. During this quest for knowledge, some hackers enjoy the academic challenges of circumventing security controls on computer systems. For this reason, the press often uses the term hacker to describe those who illicitly access systems and networks with unscrupulous, malicious, or criminal intent. The more accurate term for this type of computer hacker is *cracker* — a term created by hackers in the mid-1980s to differentiate the two communities.

47.3.1.1. Shades of Gray

Within the community of individuals who find and exploit vulnerabilities in systems and networks are several distinct groups. These groups are often described by the shade of hat that they "wear" when performing their security investigations and this shade is indicative of their intent.

The *white hat hacker* is one who tests networks and systems to examine their performance and determine how vulnerable they are to intrusion. Usually, white hat hackers crack their own systems or the systems of a client who has specifically employed them for the purposes of security auditing. Academic researchers and professional security consultants are two examples of white hat hackers.

A *black hat hacker* is synonymous with a cracker. In general, crackers are less focused on programming and the academic side of breaking into systems. They often rely on available cracking programs and exploit well known vulnerabilities in systems to uncover sensitive information for personal gain or to inflict damage on the target system or network.

The *gray hat hacker*, on the other hand, has the skills and intent of a white hat hacker in most situations but uses his knowledge for less than noble purposes on occasion. A gray hat hacker can be thought of as a white hat hacker who wears a black hat at times to accomplish his own agenda.

Gray hat hackers typically subscribe to another form of the hacker ethic, which says it is acceptable to break into systems as long as the hacker does not commit theft or breach confidentiality. Some would argue, however, that the act of breaking into a system is in itself unethical.

Regardless of the intent of the intruder, it is important to know the weaknesses a cracker may likely attempt to exploit. The remainder of the chapter focuses on these issues.

47.3.2. Threats to Network Security

Bad practices when configuring the following aspects of a network can increase the risk of attack.

47.3.2.1. Insecure Architectures

A misconfigured network is a primary entry point for unauthorized users. Leaving a trust-based, open local network vulnerable to the highly-insecure Internet is much like leaving a door ajar in a crime-ridden neighborhood — nothing may happen for an arbitrary amount of time, but *eventually* someone exploits the opportunity.

47.3.2.1.1. Broadcast Networks

System administrators often fail to realize the importance of networking hardware in their security schemes. Simple hardware such as hubs and routers rely on the broadcast or non-switched principle; that is, whenever a node transmits data across the network to a recipient node, the hub or router sends a broadcast of the data packets until the recipient node receives and processes the data. This method is the most vulnerable to address resolution protocol (*arp*) or media access control (*MAC*) address spoofing by both outside intruders and unauthorized users on local hosts.

47.3.2.1.2. Centralized Servers

Another potential networking pitfall is the use of centralized computing. A common cost-cutting measure for many businesses is to consolidate all services to a single powerful machine. This can be convenient as it is easier to manage and costs considerably less than multiple-server configurations. However, a centralized server introduces a single point of failure on the network. If the central server is compromised, it may render the network completely useless or worse, prone to data manipulation or theft. In these situations, a central server becomes an open door which allows access to the entire network.

47.3.3. Threats to Server Security

Server security is as important as network security because servers often hold a great deal of an organization's vital information. If a server is compromised, all of its contents may become available for the cracker to steal or manipulate at will. The following sections detail some of the main issues.

47.3.3.1. Unused Services and Open Ports

A full installation of Red Hat Enterprise Linux contains 1000+ application and library packages. However, most server administrators do not opt to install every single package in the distribution, preferring instead to install a base installation of packages, including several server applications.

A common occurrence among system administrators is to install the operating system without paying attention to what programs are actually being installed. This can be problematic because unneeded services may be installed, configured with the default settings, and possibly turned on. This can cause unwanted services, such as Telnet, DHCP, or DNS, to run on a server or workstation without the administrator realizing it, which in turn can cause unwanted traffic to the server, or even, a potential pathway into the system for crackers. Refer To Section 48.2, “Server Security” for information on closing ports and disabling unused services.

47.3.3.2. Unpatched Services

Most server applications that are included in a default installation are solid, thoroughly tested pieces of software. Having been in use in production environments for many years, their code has been thoroughly refined and many of the bugs have been found and fixed.

However, there is no such thing as perfect software and there is always room for further refinement. Moreover, newer software is often not as rigorously tested as one might expect, because of its recent arrival to production environments or because it may not be as popular as other server software.

Developers and system administrators often find exploitable bugs in server applications and publish the information on bug tracking and security-related websites such as the Bugtraq mailing list (http://www.securityfocus.com) or the Computer Emergency Response Team (CERT) website (http://www.cert.org). Although these mechanisms are an effective way of alerting the community to security vulnerabilities, it is up to system administrators to patch their systems promptly. This is particularly true because crackers have access to these same vulnerability tracking services and will use the information to crack unpatched systems whenever they can. Good system administration requires vigilance, constant bug tracking, and proper system maintenance to ensure a more secure computing environment.

Refer to Section 47.5, “Security Updates” for more information about keeping a system up-to-date.

47.3.3.3. Inattentive Administration

Administrators who fail to patch their systems are one of the greatest threats to server security. According to the *System Administration Network and Security Institute* (*SANS*), the primary cause of computer security vulnerability is to "assign untrained people to maintain security and provide neither the training nor the time to make it possible to do the job." [13] This applies as much to inexperienced administrators as it does to overconfident or amotivated administrators.

Some administrators fail to patch their servers and workstations, while others fail to watch log messages from the system kernel or network traffic. Another common error is when default passwords or keys to services are left unchanged. For example, some databases have default administration passwords because the database developers assume that the system administrator changes these passwords immediately after installation. If a database administrator fails to change this password, even an inexperienced cracker can use a widely-known default password to gain administrative privileges to the database. These are only a few examples of how inattentive administration can lead to compromised servers.

47.3.3.4. Inherently Insecure Services

Even the most vigilant organization can fall victim to vulnerabilities if the network services they choose are inherently insecure. For instance, there are many services developed under the assumption that they are used over trusted networks; however, this assumption fails as soon as the service becomes available over the Internet — which is itself inherently untrusted.

One category of insecure network services are those that require unencrypted usernames and passwords for authentication. Telnet and FTP are two such services. If packet sniffing software is monitoring traffic between the remote user and such a service usernames and passwords can be easily intercepted.

Inherently, such services can also more easily fall prey to what the security industry terms the *man-in-the-middle* attack. In this type of attack, a cracker redirects network traffic by tricking a cracked name server on the network to point to his machine instead of the intended server. Once someone opens a remote session to the server, the attacker's machine acts as an invisible conduit, sitting quietly between the remote service and the unsuspecting user capturing information. In this way a cracker can gather administrative passwords and raw data without the server or the user realizing it.

Another category of insecure services include network file systems and information services such as NFS or NIS, which are developed explicitly for LAN usage but are, unfortunately, extended to include WANs (for remote users). NFS does not, by default, have any authentication or security mechanisms configured to prevent a cracker from mounting the NFS share and accessing anything contained therein. NIS, as well, has vital information that must be known by every computer on a network, including passwords and file permissions, within a plain text ASCII or DBM (ASCII-derived) database. A cracker who gains access to this database can then access every user account on a network, including the administrator's account.

By default, Red Hat Enterprise Linux is released with all such services turned off. However, since administrators often find themselves forced to use these services, careful configuration is critical. Refer to Section 48.2, “Server Security” for more information about setting up services in a safe manner.

47.3.4. Threats to Workstation and Home PC Security

Workstations and home PCs may not be as prone to attack as networks or servers, but since they often contain sensitive data, such as credit card information, they are targeted by system crackers. Workstations can also be co-opted without the user's knowledge and used by attackers as "slave" machines in coordinated attacks. For these reasons, knowing the vulnerabilities of a workstation can save users the headache of reinstalling the operating system, or worse, recovering from data theft.

47.3.4.1. Bad Passwords

Bad passwords are one of the easiest ways for an attacker to gain access to a system. For more on how to avoid common pitfalls when creating a password, refer to Section 48.1.3, “Password Security”.

47.3.4.2. Vulnerable Client Applications

Although an administrator may have a fully secure and patched server, that does not mean remote users are secure when accessing it. For instance, if the server offers Telnet or FTP services over a public network, an attacker can capture the plain text usernames and passwords as they pass over the network, and then use the account information to access the remote user's workstation.

Even when using secure protocols, such as SSH, a remote user may be vulnerable to certain attacks if they do not keep their client applications updated. For instance, v.1 SSH clients are vulnerable to an X-forwarding attack from malicious SSH servers. Once connected to the server, the attacker can quietly capture any keystrokes and mouse clicks made by the client over the network. This problem was fixed in the v.2 SSH protocol, but it is up to the user to keep track of what applications have such vulnerabilities and update them as necessary.

Section 48.1, “Workstation Security” discusses in more detail what steps administrators and home users should take to limit the vulnerability of computer workstations.

47.4. Common Exploits and Attacks

Table 47.1, “Common Exploits” details some of the most common exploits and entry points used by intruders to access organizational network resources. Key to these common exploits are the explanations of how they are performed and how administrators can properly safeguard their network against such attacks.

Table 47.1. Common Exploits

Exploit	Description	Notes
Null or Default Passwords	Leaving administrative passwords blank or using a default password set by the product vendor. This is most common in hardware such as routers and firewalls, though some services that run on Linux can contain default administrator passwords (though Red Hat Enterprise Linux 5 does not ship with them).	Commonly associated with networking hardware such as routers, firewalls, VPNs, and network attached storage (NAS) appliances. Common in many legacy operating systems, especially OSes that bundle services (such as UNIX and Windows.) Administrators sometimes create privileged user accounts in a rush and leave the password null, a perfect entry point for malicious users who discover the account.
Default Shared Keys	Secure services sometimes package default security keys for development or evaluation testing purposes. If these keys are left unchanged and are placed in a production environment on the Internet, *all* users with the same default keys have access to that shared-key resource, and any sensitive information that it contains.	Most common in wireless access points and preconfigured secure server appliances.
IP Spoofing	A remote machine acts as a node on your local network, finds vulnerabilities with your servers, and installs a backdoor program or Trojan horse to gain control over your network resources.	Spoofing is quite difficult as it involves the attacker predicting TCP/IP SYN-ACK numbers to coordinate a connection to target systems, but several tools are available to assist crackers in performing such a vulnerability. Depends on target system running services (such as **`rsh`**, **`telnet`**, FTP and others) that use *source-based* authentication techniques, which are not recommended when compared to PKI or other forms of encrypted authentication used in **`ssh`** or SSL/TLS.

Exploit	Description	Notes
Eavesdropping	Collecting data that passes between two active nodes on a network by eavesdropping on the connection between the two nodes.	This type of attack works mostly with plain text transmission protocols such as Telnet, FTP, and HTTP transfers. Remote attacker must have access to a compromised system on a LAN in order to perform such an attack; usually the cracker has used an active attack (such as IP spoofing or man-in-the-middle) to compromise a system on the LAN. Preventative measures include services with cryptographic key exchange, one-time passwords, or encrypted authentication to prevent password snooping; strong encryption during transmission is also advised.
Service Vulnerabilities	An attacker finds a flaw or loophole in a service run over the Internet; through this vulnerability, the attacker compromises the entire system and any data that it may hold, and could possibly compromise other systems on the network.	HTTP-based services such as CGI are vulnerable to remote command execution and even interactive shell access. Even if the HTTP service runs as a non-privileged user such as "nobody", information such as configuration files and network maps can be read, or the attacker can start a denial of service attack which drains system resources or renders it unavailable to other users. Services sometimes can have vulnerabilities that go unnoticed during development and testing; these vulnerabilities (such as *buffer overflows*, where attackers crash a service using arbitrary values that fill the memory buffer of an application, giving the attacker an interactive command prompt from which they may execute arbitrary commands) can give complete administrative control to an attacker. Administrators should make sure that services do not run as the root user, and should stay vigilant of patches and errata updates for applications from vendors or security organizations such as CERT and CVE.

Exploit	Description	Notes
Application Vulnerabilities	Attackers find faults in desktop and workstation applications (such as e-mail clients) and execute arbitrary code, implant Trojan horses for future compromise, or crash systems. Further exploitation can occur if the compromised workstation has administrative privileges on the rest of the network.	Workstations and desktops are more prone to exploitation as workers do not have the expertise or experience to prevent or detect a compromise; it is imperative to inform individuals of the risks they are taking when they install unauthorized software or open unsolicited email attachments. Safeguards can be implemented such that email client software does not automatically open or execute attachments. Additionally, the automatic update of workstation software via Red Hat Network or other system management services can alleviate the burdens of multi-seat security deployments.
Denial of Service (DoS) Attacks	Attacker or group of attackers coordinate against an organization's network or server resources by sending unauthorized packets to the target host (either server, router, or workstation). This forces the resource to become unavailable to legitimate users.	The most reported DoS case in the US occurred in 2000. Several highly-trafficked commercial and government sites were rendered unavailable by a coordinated ping flood attack using several compromised systems with high bandwidth connections acting as *zombies*, or redirected broadcast nodes. Source packets are usually forged (as well as rebroadcasted), making investigation as to the true source of the attack difficult. Advances in ingress filtering (IETF rfc2267) using **`iptables`** and Network IDSes such as **`snort`** assist administrators in tracking down and preventing distributed DoS attacks.

47.5. Security Updates

As security vulnerabilities are discovered, the affected software must be updated in order to limit any potential security risks. If the software is part of a package within a Red Hat Enterprise Linux distribution that is currently supported, Red Hat, Inc. is committed to releasing updated packages that fix the vulnerability as soon as possible. Often, announcements about a given security exploit are accompanied with a patch (or source code that fixes the problem). This patch is then applied to the Red Hat Enterprise Linux package, tested by the Red Hat quality assurance team, and released as an errata update. However, if an announcement does not include a patch, a Red Hat developer works with the maintainer of the software to fix the problem. Once the problem is fixed, the package is tested and released as an errata update.

If an errata update is released for software used on your system, it is highly recommended that you update the effected packages as soon as possible to minimize the amount of time the system is potentially vulnerable.

47.5.1. Updating Packages

When updating software on a system, it is important to download the update from a trusted source. An attacker can easily rebuild a package with the same version number as the one that is supposed to fix the problem but with a different security exploit and release it on the Internet. If this happens, using security measures such as verifying files against the original RPM does not detect the exploit. Thus, it is very important to only download RPMs from trusted sources, such as from Red Hat, Inc. and check the signature of the package to verify its integrity.

Red Hat offers two ways to find information on errata updates:

1. Listed and available for download on Red Hat Network
2. Listed and unlinked on the Red Hat Errata website

Note

Beginning with the Red Hat Enterprise Linux product line, updated packages can be downloaded only from Red Hat Network. Although the Red Hat Errata website contains updated information, it does not contain the actual packages for download.

47.5.1.1. Using Automatic Updates with RHN Classic

Warning

Automatic system updates are only available using RHN Classic, which basis subscription consumption on access to content repository channels. RHN Classic is available as a convenience for customer environments with legacy systems which have not updated to Certificate-Based Red Hat Network.

The update and content stream is different for Certificate-Based Red Hat Network, so automatic updates are not used.

The new Certificate-Based Red Hat Network and the differences between Certificate-Based Red Hat Network and RHN Classic are described in Chapter 15, *Registering a System and Managing Subscriptions*.

RHN Classic allows the majority of the update process to be automated. It determines which RPM packages are necessary for the system, downloads them from a secure repository, verifies the RPM signature to make sure they have not been tampered with, and updates them. The package install can occur immediately or can be scheduled during a certain time period.

RHN Classic requires a *system profile* for each machine, which contains hardware and software information about the system. This information is kept confidential and is not given to anyone else. It is only used to determine which errata updates are applicable to each system, and, without it, RHN Classic can not determine whether a given system needs updates. When a security errata (or any type of errata) is released, RHN Classic sends an email with a description of the errata as well as a list of systems which are affected. To apply the update, use the **Red Hat Update Agent** or schedule the package to be updated through the RHN Classic Subscription Management area of the Customer Portal.

Important

Before installing any security errata, be sure to read any special instructions contained in the errata report and execute them accordingly. Refer to Section 47.5.1.5, "Applying the Changes" for general instructions about applying the changes made by an errata update.

47.5.1.2. Using the Red Hat Errata Website

When security errata reports are released, they are published on the Red Hat Errata website available at http://www.redhat.com/security/. From this page, select the product and version for your system, and then select **`security`** at the top of the page to display only Red Hat Enterprise Linux Security Advisories. If the synopsis of one of the advisories describes a package used on your system, click on the synopsis for more details.

The details page describes the security exploit and any special instructions that must be performed in addition to updating the package to fix the security hole.

To download the updated package(s), click on the link to login to Red Hat Network, click the package name(s) and save to the hard drive. It is highly recommended that you create a new directory, such as **`/tmp/updates`**, and save all the downloaded packages to it.

47.5.1.3. Verifying Signed Packages

All Red Hat Enterprise Linux packages are signed with the Red Hat, Inc. *GPG* key. GPG stands for GNU Privacy Guard, or GnuPG, a free software package used for ensuring the authenticity of distributed files. For example, a private key (secret key) held by Red Hat locks the package while the public key unlocks and verifies the package. If the public key distributed by Red Hat does not match the private key during RPM verification, the package may have been altered and therefore cannot be trusted.

The RPM utility within Red Hat Enterprise Linux automatically tries to verify the GPG signature of an RPM package before installing it. If the Red Hat GPG key is not installed, install it from a secure, static location, such as an Red Hat Enterprise Linux installation CD-ROM.

Assuming the CD-ROM is mounted in **`/mnt/cdrom`**, use the following command to import it into the *keyring* (a database of trusted keys on the system):

```
rpm --import /mnt/cdrom/RPM-GPG-KEY-redhat-release
```

To display a list of all keys installed for RPM verification, execute the following command:

```
rpm -qa gpg-pubkey*
```

For the Red Hat key, the output includes the following:

```
gpg-pubkey-37017186-45761324
```

To display details about a specific key, use the **`rpm -qi`** command followed by the output from the previous command, as in this example:

```
rpm -qi gpg-pubkey-37017186-45761324
```

It is extremely important to verify the signature of the RPM files before installing them to ensure that they have not been altered from the Red Hat, Inc. release of the packages. To verify all the downloaded packages at once, issue the following command:

```
rpm -K /tmp/updates/*.rpm
```

For each package, if the GPG key verifies successfully, the command returns **`gpg OK`**. If it doesn't, make sure you are using the correct Red Hat public key, as well as verifying the source of the content. Packages that do not pass GPG verifications should not be installed, as they may have been altered by a third party.

After verifying the GPG key and downloading all the packages associated with the errata report, install the packages as root at a shell prompt.

47.5.1.4. Installing Signed Packages

Installation for most packages can be done safely (except kernel packages) by issuing the following command:

```
rpm -Uvh /tmp/updates/*.rpm
```

For kernel packages use the following command:

```
rpm -ivh /tmp/updates/<kernel-package>
```

Replace <*kernel-package*> in the previous example with the name of the kernel RPM.

Once the machine has been safely rebooted using the new kernel, the old kernel may be removed using the following command:

```
rpm -e <old-kernel-package>
```

Replace <*old-kernel-package*> in the previous example with the name of the older kernel RPM.

Note

It is not a requirement that the old kernel be removed. The default boot loader, GRUB, allows for multiple kernels to be installed, then chosen from a menu at boot time.

Important

Before installing any security errata, be sure to read any special instructions contained in the errata report and execute them accordingly. Refer to Section 47.5.1.5, “Applying the Changes” for general instructions about applying the changes made by an errata update.

47.5.1.5. Applying the Changes

After downloading and installing security errata via Red Hat Network or the Red Hat errata website, it is important to halt usage of the older software and begin using the new software. How this is done depends on the type of software that has been updated. The following list itemizes the general categories of software and provides instructions for using the updated versions after a package upgrade.

Note

In general, rebooting the system is the surest way to ensure that the latest version of a software package is used; however, this option is not always available to the system administrator.

Applications

User-space applications are any programs that can be initiated by a system user. Typically, such applications are used only when a user, script, or automated task utility launches them and they do not persist for long periods of time.

Once such a user-space application is updated, halt any instances of the application on the system and launch the program again to use the updated version.

Kernel

The kernel is the core software component for the Red Hat Enterprise Linux operating system. It manages access to memory, the processor, and peripherals as well as schedules all tasks.

Because of its central role, the kernel cannot be restarted without also stopping the computer. Therefore, an updated version of the kernel cannot be used until the system is rebooted.

Shared Libraries

Shared libraries are units of code, such as **`glibc`**, which are used by a number of applications and services. Applications utilizing a shared library typically load the shared code when the application is initialized, so any applications using the updated library must be halted and relaunched.

To determine which running applications link against a particular library, use the **`lsof`** command as in the following example:

```
lsof /usr/lib/libwrap.so*
```

This command returns a list of all the running programs which use TCP wrappers for host access control. Therefore, any program listed must be halted and relaunched if the **`tcp_wrappers`** package is updated.

SysV Services

SysV services are persistent server programs launched during the boot process. Examples of SysV services include **`sshd`**, **`vsftpd`**, and **`xinetd`**.

Because these programs usually persist in memory as long as the machine is booted, each updated SysV service must be halted and relaunched after the package is upgraded. This can be done using the **Services Configuration Tool** or by logging into a root shell prompt and issuing the **`/sbin/service`** command as in the following example:

```
service <service-name> restart
```

In the previous example, replace *<service-name>* with the name of the service, such as **sshd**.

Refer to [Chapter 17, *Network Configuration*] for more information on the **Services Configuration Tool**.

xinetd Services

Services controlled by the **xinetd** super service only run when a there is an active connection. Examples of services controlled by **xinetd** include Telnet, IMAP, and POP3.

Because new instances of these services are launched by **xinetd** each time a new request is received, connections that occur after an upgrade are handled by the updated software. However, if there are active connections at the time the **xinetd** controlled service is upgraded, they are serviced by the older version of the software.

To kill off older instances of a particular **xinetd** controlled service, upgrade the package for the service then halt all processes currently running. To determine if the process is running, use the **ps** command and then use the **kill** or **killall** command to halt current instances of the service.

For example, if security errata **imap** packages are released, upgrade the packages, then type the following command as root into a shell prompt:

```
ps -aux | grep imap
```

This command returns all active IMAP sessions. Individual sessions can then be terminated by issuing the following command:

```
kill <PID>
```

If this fails to terminate the session, use the following command instead:

```
kill -9 <PID>
```

In the previous examples, replace *<PID>* with the process identification number (found in the second column of the **ps** command) for an IMAP session.

To kill all active IMAP sessions, issue the following command:

```
killall imapd
```

[10] Source: http://www.cert.org

[11] Source: http://www.cert.org/stats/

[12] Source: http://www.newsfactor.com/perl/story/16407.html

[13] Source: http://www.sans.org/security-resources/mistakes.php

Chapter 48. Securing Your Network

48.1. Workstation Security

Securing a Linux environment begins with the workstation. Whether locking down a personal machine or securing an enterprise system, sound security policy begins with the individual computer. A computer network is only as secure as its weakest node.

48.1.1. Evaluating Workstation Security

When evaluating the security of a Red Hat Enterprise Linux workstation, consider the following:

- *BIOS and Boot Loader Security* — Can an unauthorized user physically access the machine and boot into single user or rescue mode without a password?
- *Password Security* — How secure are the user account passwords on the machine?
- *Administrative Controls* — Who has an account on the system and how much administrative control do they have?
- *Available Network Services* — What services are listening for requests from the network and should they be running at all?
- *Personal Firewalls* — What type of firewall, if any, is necessary?
- *Security Enhanced Communication Tools* — Which tools should be used to communicate between workstations and which should be avoided?

48.1.2. BIOS and Boot Loader Security

Password protection for the BIOS (or BIOS equivalent) and the boot loader can prevent unauthorized users who have physical access to systems from booting using removable media or obtaining root privileges through single user mode. The security measures you should take to protect against such attacks depends both on the sensitivity of the information on the workstation and the location of the machine.

For example, if a machine is used in a trade show and contains no sensitive information, then it may not be critical to prevent such attacks. However, if an employee's laptop with private, unencrypted SSH keys for the corporate network is left unattended at that same trade show, it could lead to a major security breach with ramifications for the entire company.

If the workstation is located in a place where only authorized or trusted people have access, however, then securing the BIOS or the boot loader may not be necessary.

48.1.2.1. BIOS Passwords

The two primary reasons for password protecting the BIOS of a computer are [14]:

1. *Preventing Changes to BIOS Settings* — If an intruder has access to the BIOS, they can set it to boot from a diskette or CD-ROM. This makes it possible for them to enter rescue mode or single user mode, which in turn allows them to start arbitrary processes on the system or copy sensitive data.
2. *Preventing System Booting* — Some BIOSes allow password protection of the boot process. When activated, an attacker is forced to enter a password before the BIOS launches the boot loader.

Because the methods for setting a BIOS password vary between computer manufacturers, consult the computer's manual for specific instructions.

If you forget the BIOS password, it can either be reset with jumpers on the motherboard or by disconnecting the CMOS battery. For this reason, it is good practice to lock the computer case if possible. However, consult the manual for the computer or motherboard before attempting to disconnect the CMOS battery.

48.1.2.1.1. Securing Non-x86 Platforms

Other architectures use different programs to perform low-level tasks roughly equivalent to those of the BIOS on x86 systems. For instance, Intel® Itanium™ computers use the *Extensible Firmware Interface* (*EFI*) shell.

For instructions on password protecting BIOS-like programs on other architectures, refer to the manufacturer's instructions.

48.1.2.2. Boot Loader Passwords

The primary reasons for password protecting a Linux boot loader are as follows:

1. *Preventing Access to Single User Mode* — If attackers can boot the system into single user mode, they are logged in automatically as root without being prompted for the root password.

2. *Preventing Access to the GRUB Console* — If the machine uses GRUB as its boot loader, an attacker can use the GRUB editor interface to change its configuration or to gather information using the **`cat`** command.

3. *Preventing Access to Insecure Operating Systems* — If it is a dual-boot system, an attacker can select an operating system at boot time (for example, DOS), which ignores access controls and file permissions.

Red Hat Enterprise Linux ships with the GRUB boot loader on the x86 platform. For a detailed look at GRUB, refer to the Red Hat Installation Guide.

48.1.2.2.1. Password Protecting GRUB

You can configure GRUB to address the first two issues listed in Section 48.1.2.2, “Boot Loader Passwords” by adding a password directive to its configuration file. To do this, first choose a strong password, open a shell, log in as root, and then type the following command:

```
grub-md5-crypt
```

When prompted, type the GRUB password and press **`Enter`**. This returns an MD5 hash of the password.

Next, edit the GRUB configuration file **`/boot/grub/grub.conf`**. Open the file and below the **`timeout`** line in the main section of the document, add the following line:

```
password --md5 <password-hash>
```

Replace *<password-hash>* with the value returned by **`/sbin/grub-md5-crypt`** [15].

The next time the system boots, the GRUB menu prevents access to the editor or command interface without first pressing **`p`** followed by the GRUB password.

Unfortunately, this solution does not prevent an attacker from booting into an insecure operating system in a dual-boot environment. For this, a different part of the **`/boot/grub/grub.conf`** file must be edited.

Look for the **`title`** line of the operating system that you want to secure, and add a line with the **`lock`** directive immediately beneath it.

For a DOS system, the stanza should begin similar to the following:

```
title DOS lock
```

Warning

A **`password`** line must be present in the main section of the **`/boot/grub/grub.conf`** file for this method to work properly. Otherwise, an attacker can access the GRUB editor interface and remove the lock line.

To create a different password for a particular kernel or operating system, add a **`lock`** line to the stanza, followed by a password line.

Each stanza protected with a unique password should begin with lines similar to the following example:

```
title DOS lock password --md5 <password-hash>
```

48.1.3. Password Security

Passwords are the primary method that Red Hat Enterprise Linux uses to verify a user's identity. This is why password security is so important for protection of the user, the workstation, and the network.

For security purposes, the installation program configures the system to use *Message-Digest Algorithm* (*MD5*) and shadow passwords. It is highly recommended that you do not alter these settings.

If MD5 passwords are deselected during installation, the older *Data Encryption Standard* (*DES*) format is used. This format limits passwords to eight alphanumeric characters (disallowing punctuation and other special characters), and provides a modest 56-bit level of encryption.

If shadow passwords are deselected during installation, all passwords are stored as a one-way hash in the world-readable **`/etc/passwd`** file, which makes the system vulnerable to offline password cracking attacks. If an intruder can gain access to the machine as a regular user, they can copy the **`/etc/passwd`** file to their own machine and run any number of password cracking programs against it. If there is an insecure password in the file, it is only a matter of time before the password cracker discovers it.

Shadow passwords eliminate this type of attack by storing the password hashes in the file **`/etc/shadow`**, which is readable only by the root user.

This forces a potential attacker to attempt password cracking remotely by logging into a network service on the machine, such as SSH or FTP. This sort of brute-force attack is much slower and leaves an obvious trail as hundreds of failed login attempts are written to system files. Of course, if the cracker starts an attack in the middle of the night on a system with weak passwords, the cracker may have gained access before dawn and edited the log files to cover their tracks.

In addition to format and storage considerations is the issue of content. The single most important thing a user can do to protect their account against a password cracking attack is create a strong password.

48.1.3.1. Creating Strong Passwords

When creating a secure password, it is a good idea to follow these guidelines:

- *Do Not Use Only Words or Numbers* — Never use only numbers or words in a password.

  Some insecure examples include the following:

  - 8675309

  - juan
  - hackme

- *Do Not Use Recognizable Words* — Words such as proper names, dictionary words, or even terms from television shows or novels should be avoided, even if they are bookended with numbers.

  Some insecure examples include the following:

  - john1
  - DS-9
  - mentat123

- *Do Not Use Words in Foreign Languages* — Password cracking programs often check against word lists that encompass dictionaries of many languages. Relying on foreign languages for secure passwords is not secure.

  Some insecure examples include the following:

  - cheguevara
  - bienvenido1
  - 1dumbKopf

- *Do Not Use Hacker Terminology* — If you think you are elite because you use hacker terminology — also called l337 (LEET) speak — in your password, think again. Many word lists include LEET speak.

  Some insecure examples include the following:

  - H4X0R
  - 1337

- *Do Not Use Personal Information* — Avoid using any personal information in your passwords. If the attacker knows your identity, the task of deducing your password becomes easier. The following is a list of the types of information to avoid when creating a password:

  Some insecure examples include the following:

  - Your name
  - The names of pets
  - The names of family members
  - Any birth dates
  - Your phone number or zip code

- *Do Not Invert Recognizable Words* — Good password checkers always reverse common words, so inverting a bad password does not make it any more secure.

  Some insecure examples include the following:

  - R0X4H
  - nauj
  - 9-DS

- *Do Not Write Down Your Password* — Never store a password on paper. It is much safer to memorize it.
- *Do Not Use the Same Password For All Machines* — It is important to make separate passwords for each machine. This way if one system is compromised, all of your machines are not immediately at risk.

The following guidelines will help you to create a strong password:

- *Make the Password at Least Eight Characters Long* — The longer the password, the better. If using MD5 passwords, it should be 15 characters or longer. With DES passwords, use the maximum length (eight characters).
- *Mix Upper and Lower Case Letters* — Red Hat Enterprise Linux is case sensitive, so mix cases to enhance the strength of the password.
- *Mix Letters and Numbers* — Adding numbers to passwords, especially when added to the middle (not just at the beginning or the end), can enhance password strength.
- *Include Non-Alphanumeric Characters* — Special characters such as &, $, and > can greatly improve the strength of a password (this is not possible if using DES passwords).
- *Pick a Password You Can Remember* — The best password in the world does little good if you cannot remember it; use acronyms or other mnemonic devices to aid in memorizing passwords.

With all these rules, it may seem difficult to create a password that meets all of the criteria for good passwords while avoiding the traits of a bad one. Fortunately, there are some steps you can take to generate an easily-remembered, secure password.

48.1.3.1.1. Secure Password Creation Methodology

There are many methods that people use to create secure passwords. One of the more popular methods involves acronyms. For example:

- Think of an easily-remembered phrase, such as:

  "over the river and through the woods, to grandmother's house we go."

- Next, turn it into an acronym (including the punctuation).

  `otrattw,tghwg.`

- Add complexity by substituting numbers and symbols for letters in the acronym. For example, substitute **`7`** for **`t`** and the at symbol (`@`) for **`a`**:

  `o7r@77w,7ghwg.`

- Add more complexity by capitalizing at least one letter, such as **`H`**.

  `o7r@77w,7gHwg.`

- *Finally, do not use the example password above for any systems, ever.*

While creating secure passwords is imperative, managing them properly is also important, especially for system administrators within larger organizations. The following section details good practices for creating and managing user passwords within an organization.

48.1.3.2. Creating User Passwords Within an Organization

If an organization has a large number of users, the system administrators have two basic options available to force the use of good passwords. They can create passwords for the user, or they can let users create their own passwords, while verifying the passwords are of acceptable quality.

Creating the passwords for the users ensures that the passwords are good, but it becomes a daunting task as the organization grows. It also increases the risk of users writing their passwords down.

For these reasons, most system administrators prefer to have the users create their own passwords, but actively verify that the passwords are good and, in some cases, force users to change their passwords periodically through password aging.

48.1.3.2.1. Forcing Strong Passwords

To protect the network from intrusion it is a good idea for system administrators to verify that the passwords used within an organization are strong ones. When users are asked to create or change passwords, they can use the command line application **`passwd`**, which is *Pluggable Authentication Manager* (*PAM*) aware and therefore checks to see if the password is too short or otherwise easy to crack. This check is performed using the **`pam_cracklib.so`** PAM module. Since PAM is customizable, it is possible to add more password integrity checkers, such as **`pam_passwdqc`** (available from http://www.openwall.com/passwdqc/) or to write a new module. For a list of available PAM modules, refer to http://www.kernel.org/pub/linux/libs/pam/modules.html. For more information about PAM, refer to Section 48.4, “Pluggable Authentication Modules (PAM)”.

The password check that is performed at the time of their creation does not discover bad passwords as effectively as running a password cracking program against the passwords.

Many password cracking programs are available that run under Red Hat Enterprise Linux, although none ship with the operating system. Below is a brief list of some of the more popular password cracking programs:

Note

None of these tools are supplied with Red Hat Enterprise Linux and are therefore not supported by Red Hat, Inc. in any way.

- ***John The Ripper*** — A fast and flexible password cracking program. It allows the use of multiple word lists and is capable of brute-force password cracking. It is available online at http://www.openwall.com/john/.
- ***Crack*** — Perhaps the most well known password cracking software, **Crack** is also very fast, though not as easy to use as **John The Ripper**. It can be found online at http://www.openwall.com/john/.
- ***Slurpie*** — **Slurpie** is similar to **John The Ripper** and **Crack**, but it is designed to run on multiple computers simultaneously, creating a distributed password cracking attack. It can be found along with a number of other distributed attack security evaluation tools online at http://www.ussrback.com/distributed.htm.

Warning

Always get authorization in writing before attempting to crack passwords within an organization.

48.1.3.2.2. Password Aging

Password aging is another technique used by system administrators to defend against bad passwords within an organization. Password aging means that after a specified period (usually 90 days), the user is prompted to create a new password. The theory behind this is that if a user is forced to change their password periodically, a cracked password is only useful to an intruder for a limited amount of time. The downside to password aging, however, is that users are more likely to write their passwords down.

There are two primary programs used to specify password aging under Red Hat Enterprise Linux: the **`chage`** command or the graphical **User Manager** (**`system-config-users`**) application.

The **`-M`** option of the **`chage`** command specifies the maximum number of days the password is valid. For example, to set a user's password to expire in 90 days, use the following command:

```
chage -M 90 <username>
```

In the above command, replace *<username>* with the name of the user. To disable password expiration, it is traditional to use a value of **`99999`** after the **`-M`** option (this equates to a little over 273 years).

You can also use the **`chage`** command in interactive mode to modify multiple password aging and account details. Use the following command to enter interactive mode:

```
chage <username>
```

The following is a sample interactive session using this command:

```
~]# chage davido
Changing the aging information for davido
Enter the new value, or press ENTER for the default

        Minimum Password Age [0]: 10
        Maximum Password Age [99999]: 90
        Last Password Change (YYYY-MM-DD) [2006-08-18]:
        Password Expiration Warning [7]:
        Password Inactive [-1]:
        Account Expiration Date (YYYY-MM-DD) [1969-12-31]:
~]#
```

Refer to the man page for chage for more information on the available options.

You can also use the graphical **User Manager** application to create password aging policies, as follows. Note: you need Administrator privileges to perform this procedure.

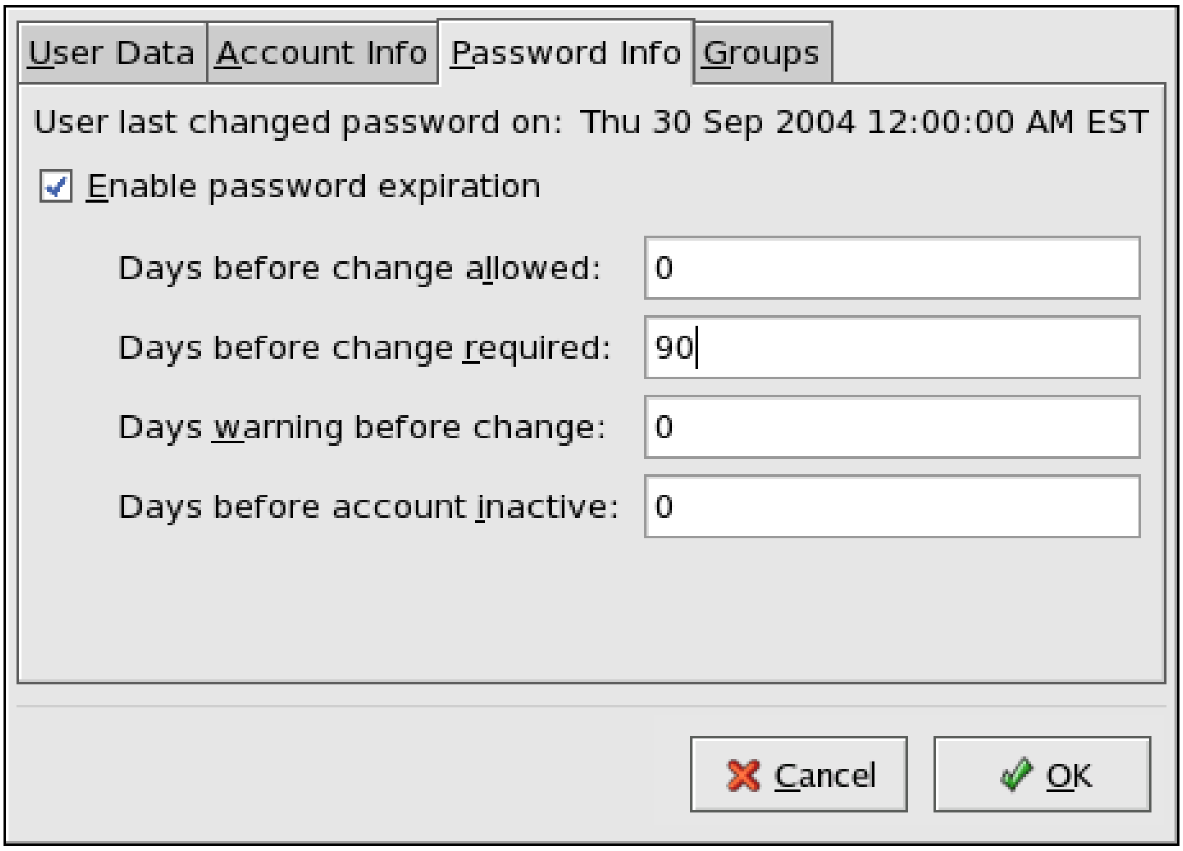
1. Click the **System** menu on the Panel, point to **Administration** and then click **Users and Groups** to display the User Manager. Alternatively, type the command **`system-config-users`** at a shell prompt.

2. Click the **`Users`** tab, and select the required user in the list of users.

3. Click **`Properties`** on the toolbar to display the User Properties dialog box (or choose **Properties** on the **File** menu).

4. Click the **`Password Info`** tab, and select the check box for **`Enable password expiration`**.

5. Enter the required value in the **`Days before change required`** field, and click **`OK`**.



User last changed password on: Thu 30 Sep 2004 12:00:00 AM EST

☑ Enable password expiration

Days before change allowed:	0
Days before change required:	90
Days warning before change:	0
Days before account inactive:	0

Figure 48.1. Specifying password aging options

For more information about user and group configuration (including instructions on forcing first time passwords), refer to Chapter 37, *Users and Groups*.

48.1.4. Administrative Controls

When administering a home machine, the user must perform some tasks as the root user or by acquiring effective root privileges via a *setuid* program, such as **sudo** or **su**. A setuid program is one that operates with the user ID (*UID*) of the program's owner rather than the user operating the program. Such programs are denoted by an **s** in the owner section of a long format listing, as in the following example:

```
-rwsr-xr-x    1 root     root        47324 May  1 08:09 /bin/su
```

Note

The **s** may be upper case or lower case. If it appears as upper case, it means that the underlying permission bit has not been set.

For the system administrators of an organization, however, choices must be made as to how much administrative access users within the organization should have to their machine. Through a PAM module called **pam_console.so**, some activities normally reserved only for the root user, such as rebooting and mounting removable media are allowed for the first user that logs in at the physical console (refer to Section 48.4, “Pluggable Authentication Modules (PAM)” for more information about the **pam_console.so** module.) However, other important system administration tasks, such as altering network settings, configuring a new mouse, or mounting network devices, are not possible without administrative privileges. As a result, system administrators must decide how much access the users on their network should receive.

48.1.4.1. Allowing Root Access

If the users within an organization are trusted and computer-literate, then allowing them root access may not be an issue. Allowing root access by users means that minor activities, like adding devices or configuring network interfaces, can be handled by the individual users, leaving system administrators free to deal with network security and other important issues.

On the other hand, giving root access to individual users can lead to the following issues:

- *Machine Misconfiguration* — Users with root access can misconfigure their machines and require assistance to resolve issues. Even worse, they might open up security holes without knowing it.
- *Running Insecure Services* — Users with root access might run insecure servers on their machine, such as FTP or Telnet, potentially putting usernames and passwords at risk. These services transmit this information over the network in plain text.
- *Running Email Attachments As Root* — Although rare, email viruses that affect Linux do exist. The only time they are a threat, however, is when they are run by the root user.

48.1.4.2. Disallowing Root Access

If an administrator is uncomfortable allowing users to log in as root for these or other reasons, the root password should be kept secret, and access to runlevel one or single user mode should be disallowed through boot loader password protection (refer to Section 48.1.2.2, “Boot Loader Passwords” for more information on this topic.)

The following are four different ways that an administrator can further ensure that root logins are disallowed:

Changing the root shell

To prevent users from logging in directly as root, the system administrator can set the root account's shell to **`/sbin/nologin`** in the **`/etc/passwd`** file.

Table 48.1. Disabling the Root Shell

Effects	Does Not Affect
Prevents access to the root shell and logs any such attempts. The following programs are prevented from accessing the root account: • **`login`** • **`gdm`** • **`kdm`** • **`xdm`** • **`su`** • **`ssh`** • **`scp`** • **`sftp`**	Programs that do not require a shell, such as FTP clients, mail clients, and many setuid programs. The following programs are *not* prevented from accessing the root account: • **sudo** • FTP clients • Email clients

Disabling root access via any console device (tty)

To further limit access to the root account, administrators can disable root logins at the console by editing the **`/etc/securetty`** file. This file lists all devices the root user is allowed to log into. If the file does not exist at all, the root user can log in through any communication device on the system, whether via the console or a raw network interface. This is dangerous, because a user can log in to their machine as root via Telnet, which transmits the password in plain text over the network.

By default, Red Hat Enterprise Linux's **/etc/securetty** file only allows the root user to log in at the console physically attached to the machine. To prevent the root user from logging in, remove the contents of this file by typing the following command at a shell prompt as root:

```
echo > /etc/securetty
```

To enable **securetty** support in the KDM, GDM, and XDM login managers, add the following line:

```
auth [user_unknown=ignore success=ok ignore=ignore default=bad]
pam_securetty.so
```

to the files listed below:

- **/etc/pam.d/gdm**
- **/etc/pam.d/gdm-autologin**
- **/etc/pam.d/gdm-fingerprint**
- **/etc/pam.d/gdm-password**
- **/etc/pam.d/gdm-smartcard**
- **/etc/pam.d/kdm**
- **/etc/pam.d/kdm-np**
- **/etc/pam.d/xdm**

Warning

A blank **/etc/securetty** file does *not* prevent the root user from logging in remotely using the OpenSSH suite of tools because the console is not opened until after authentication.

Table 48.2. Disabling Root Logins

Effects	Does Not Affect
Prevents access to the root account via the console or the network. The following programs are prevented from accessing the root account: - **login** - **gdm** - **kdm** - **xdm** - Other network services that open a tty	Programs that do not log in as root, but perform administrative tasks through setuid or other mechanisms. The following programs are *not* prevented from accessing the root account: - **su** - **sudo** - **ssh** - **scp** - **sftp**

Disabling root SSH logins

To prevent root logins via the SSH protocol, edit the SSH daemon's configuration file, **/etc/ssh/sshd_config**, and change the line that reads:

```
#PermitRootLogin yes
```

to read as follows:

```
PermitRootLogin no
```

Table 48.3. Disabling Root SSH Logins

Effects	Does Not Affect
Prevents root access via the OpenSSH suite of tools. The following programs are prevented from accessing the root account: • **`ssh`** • **`scp`** • **`sftp`**	Programs that are not part of the OpenSSH suite of tools.

Using PAM to limit root access to services

PAM, through the **`/lib/security/pam_listfile.so`** module, allows great flexibility in denying specific accounts. The administrator can use this module to reference a list of users who are not allowed to log in. To limit root access to a system service, edit the file for the target service in the **`/etc/pam.d/`** directory and make sure the **`pam_listfile.so`** module is required for authentication.

The following is an example of how the module is used for the **`vsftpd`** FTP server in the **`/etc/pam.d/vsftpd`** PAM configuration file (the `\` character at the end of the first line is *not* necessary if the directive is on a single line):

```
auth   required   /lib/security/pam_listfile.so   item=user \
 sense=deny file=/etc/vsftpd.ftpusers onerr=succeed
```

This instructs PAM to consult the **`/etc/vsftpd.ftpusers`** file and deny access to the service for any listed user. The administrator can change the name of this file, and can keep separate lists for each service or use one central list to deny access to multiple services.

If the administrator wants to deny access to multiple services, a similar line can be added to the PAM configuration files, such as **`/etc/pam.d/pop`** and **`/etc/pam.d/imap`** for mail clients, or **`/etc/pam.d/ssh`** for SSH clients.

For more information about PAM, refer to Section 48.4, “Pluggable Authentication Modules (PAM)”.

Table 48.4. Disabling Root Using PAM

Effects	Does Not Affect
Prevents root access to network services that are PAM aware. The following services are prevented from accessing the root account: • **login** • **gdm** • **kdm** • **xdm** • **ssh** • **scp** • **sftp** • FTP clients • Email clients • Any PAM aware services	Programs and services that are not PAM aware.

48.1.4.3. Limiting Root Access

Rather than completely denying access to the root user, the administrator may want to allow access only via setuid programs, such as **su** or **sudo**.

48.1.4.3.1. The su Command

When a user executes the **su** command, they are prompted for the root password and, after authentication, is given a root shell prompt.

Once logged in via the **su** command, the user *is* the root user and has absolute administrative access to the system [16]. In addition, once a user has become root, it is possible for them to use the **su** command to change to any other user on the system without being prompted for a password.

Because this program is so powerful, administrators within an organization may wish to limit who has access to the command.

One of the simplest ways to do this is to add users to the special administrative group called *wheel*. To do this, type the following command as root:

```
usermod -G wheel <username>
```

In the previous command, replace *<username>* with the username you want to add to the **wheel** group.

You can also use the **User Manager** to modify group memberships, as follows. Note: you need Administrator privileges to perform this procedure.

1. Click the **System** menu on the Panel, point to **Administration** and then click **Users and Groups** to display the User Manager. Alternatively, type the command **system-config-users** at a shell prompt.

2. Click the **Users** tab, and select the required user in the list of users.

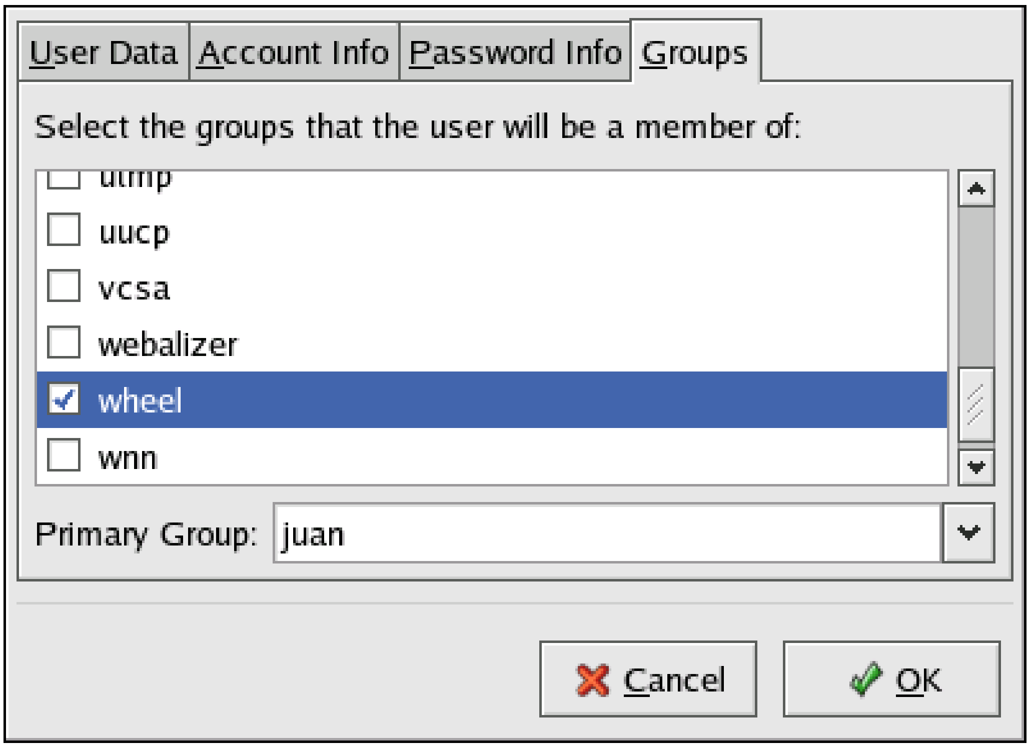
3. Click **Properties** on the toolbar to display the User Properties dialog box (or choose **Properties** on the **File** menu).

4. Click the **Groups** tab, select the check box for the wheel group, and then click **OK**. Refer to Figure 48.2, “Adding users to the "wheel" group.”.

5. Open the PAM configuration file for **su** (**/etc/pam.d/su**) in a text editor and remove the comment # from the following line:

   ```
   auth           required        pam_wheel.so use_uid
   ```

   This change means that only members of the administrative group **wheel** can switch to another user using the su command.



Select the groups that the user will be a member of:

- [ ] utmp
- [ ] uucp
- [ ] vcsa
- [ ] webalizer
- [✓] wheel
- [ ] wnn

Primary Group: juan

Figure 48.2. Adding users to the "wheel" group.

Note

The root user is part of the **wheel** group by default.

48.1.4.3.2. The sudo Command

The **sudo** command offers another approach to giving users administrative access. When trusted users precede an administrative command with **sudo**, they are prompted for *their own* password. Then, when they have been authenticated and assuming that the command is permitted, the administrative command is executed as if they were the root user.

The basic format of the **sudo** command is as follows:

```
sudo <command>
```

In the above example, <*command*> would be replaced by a command normally reserved for the root user, such as **mount**.

Important

Users of the **sudo** command should take extra care to log out before walking away from their machines since sudoers can use the command again without being asked for a password within a five minute period. This setting can be altered via the configuration file, **/etc/sudoers**.

The **sudo** command allows for a high degree of flexibility. For instance, only users listed in the **/etc/sudoers** configuration file are allowed to use the **sudo** command and the command is executed in *the user's* shell, not a root shell. This means the root shell can be completely disabled, as shown in Section 48.1.4.2, “Disallowing Root Access”.

The **sudo** command also provides a comprehensive audit trail. Each successful authentication is logged to the file **/var/log/messages** and the command issued along with the issuer's user name is logged to the file **/var/log/secure**.

Another advantage of the **sudo** command is that an administrator can allow different users access to specific commands based on their needs.

Administrators wanting to edit the **sudo** configuration file, **/etc/sudoers**, should use the **visudo** command.

To give someone full administrative privileges, type **visudo** and add a line similar to the following in the user privilege specification section:

```
juan ALL=(ALL) ALL
```

This example states that the user, **juan**, can use **sudo** from any host and execute any command.

The example below illustrates the granularity possible when configuring **sudo**:

```
%users  localhost=/sbin/shutdown -h now
```

This example states that any user can issue the command **/sbin/shutdown -h now** as long as it is issued from the console.

The man page for **sudoers** has a detailed listing of options for this file.

48.1.5. Available Network Services

While user access to administrative controls is an important issue for system administrators within an organization, monitoring which network services are active is of paramount importance to anyone who administers and operates a Linux system.

Many services under Red Hat Enterprise Linux behave as network servers. If a network service is running on a machine, then a server application (called a *daemon*), is listening for connections on one or more network ports. Each of these servers should be treated as a potential avenue of attack.

48.1.5.1. Risks To Services

Network services can pose many risks for Linux systems. Below is a list of some of the primary issues:

- *Denial of Service Attacks (DoS)* — By flooding a service with requests, a denial of service attack can render a system unusable as it tries to log and answer each request.

- *Script Vulnerability Attacks* — If a server is using scripts to execute server-side actions, as Web servers commonly do, a cracker can attack improperly written scripts. These script vulnerability attacks can lead to a buffer overflow condition or allow the attacker to alter files on the system.

- *Buffer Overflow Attacks* — Services that connect to ports numbered 0 through 1023 must run as an administrative user. If the application has an exploitable buffer overflow, an attacker could gain access to the system as the user running the daemon. Because exploitable buffer overflows exist, crackers use automated tools to identify systems with vulnerabilities, and once they have gained access, they use automated rootkits to maintain their access to the system.

Note

The threat of buffer overflow vulnerabilities is mitigated in Red Hat Enterprise Linux by *ExecShield*, an executable memory segmentation and protection technology supported by x86-compatible uni- and multi-processor kernels. ExecShield reduces the risk of buffer overflow by separating virtual memory into executable and non-executable segments. Any program code that tries to execute outside of the executable segment (such as malicious code injected from a buffer overflow exploit) triggers a segmentation fault and terminates.

Execshield also includes support for *No eXecute* (NX) technology on AMD64 platforms and *eXecute Disable* (XD) technology on Itanium and Intel® 64 systems. These technologies work in conjunction with ExecShield to prevent malicious code from running in the executable portion of virtual memory with a granularity of 4KB of executable code, lowering the risk of attack from stealthy buffer overflow exploits.

Note

To limit exposure to attacks over the network, all services that are unused should be turned off.

48.1.5.2. Identifying and Configuring Services

To enhance security, most network services installed with Red Hat Enterprise Linux are turned off by default. There are, however, some notable exceptions:

- **`cupsd`** — The default print server for Red Hat Enterprise Linux.

- **`lpd`** — An alternative print server.

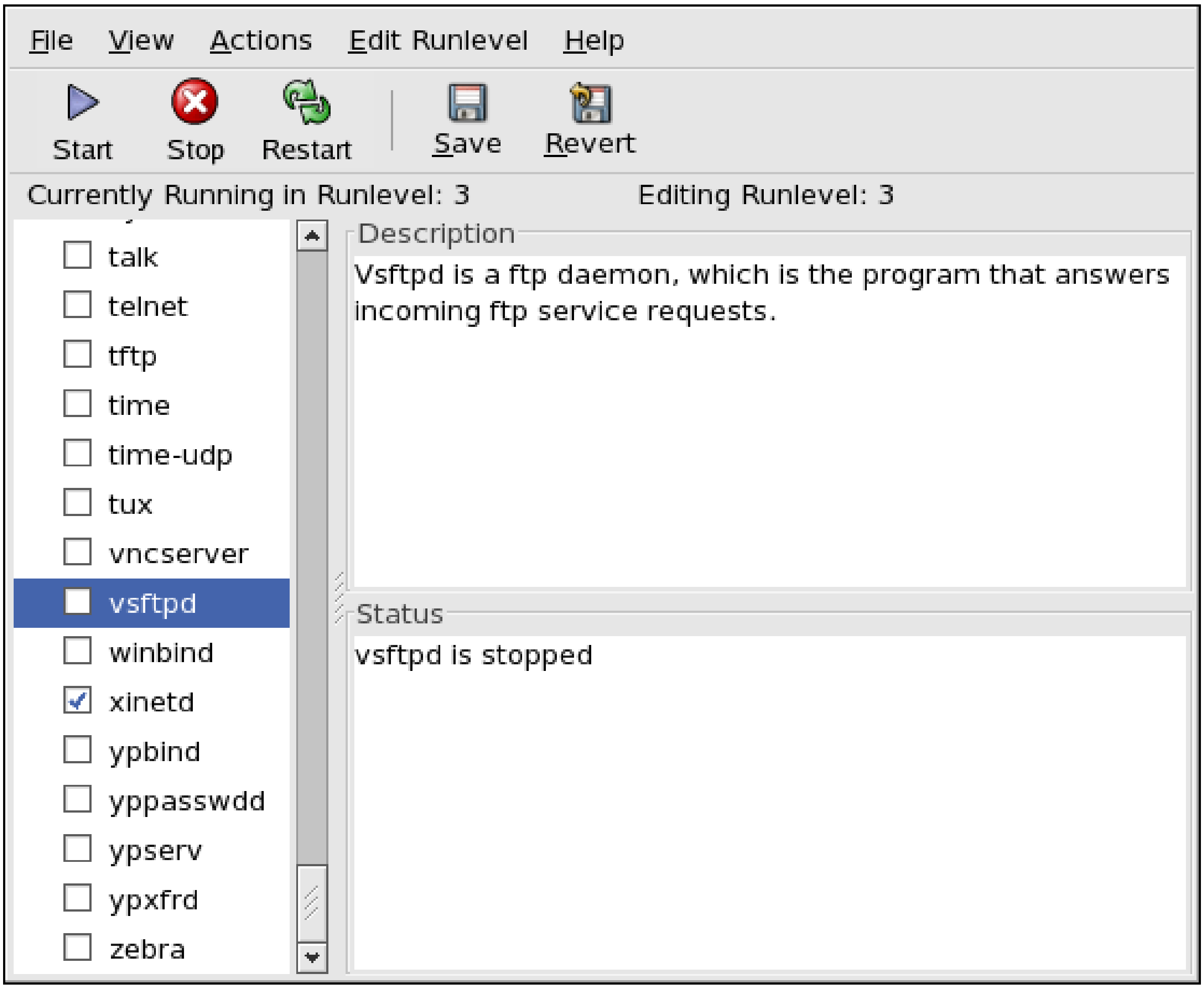
- **`xinetd`** — A super server that controls connections to a range of subordinate servers, such as **`gssftp`** and **`telnet`**.

- **`sendmail`** — The Sendmail *Mail Transport Agent* (MTA) is enabled by default, but only listens for connections from the localhost.

- **`sshd`** — The OpenSSH server, which is a secure replacement for Telnet.

When determining whether to leave these services running, it is best to use common sense and err on the side of caution. For example, if a printer is not available, do not leave **`cupsd`** running. The same is true for **`portmap`**. If you do not mount NFSv3 volumes or use NIS (the **`ypbind`** service), then **`portmap`** should be disabled.

Red Hat Enterprise Linux ships with three programs designed to switch services on or off. They are the **Services Configuration Tool** (**`system-config-services`**), **ntsysv**, and **`chkconfig`**. For information on using these tools, refer to Chapter 18, *Controlling Access to Services*.



Currently Running in Runlevel: 3 Editing Runlevel: 3

- [ ] talk
- [ ] telnet
- [ ] tftp
- [ ] time
- [ ] time-udp
- [ ] tux
- [ ] vncserver
- [ ] vsftpd
- [ ] winbind
- [x] xinetd
- [ ] ypbind
- [ ] yppasswdd
- [ ] ypserv
- [ ] ypxfrd
- [ ] zebra

Vsftpd is a ftp daemon, which is the program that answers incoming ftp service requests.

vsftpd is stopped

Figure 48.3. Services Configuration Tool

If unsure of the purpose for a particular service, the **Services Configuration Tool** has a description field, illustrated in Figure 48.3, "Services Configuration Tool", that provides additional information.

Checking which network services are available to start at boot time is only part of the story. You should also check which ports are open and listening. Refer to Section 48.2.8, "Verifying Which Ports Are Listening" for more information.

48.1.5.3. Insecure Services

Potentially, any network service is insecure. This is why turning off unused services is so important. Exploits for services are routinely revealed and patched, making it very important to regularly update packages associated with any network service. Refer to Section 47.5, "Security Updates" for more information.

Some network protocols are inherently more insecure than others. These include any services that:

- *Transmit Usernames and Passwords Over a Network Unencrypted* — Many older protocols, such as Telnet and FTP, do not encrypt the authentication session and should be avoided whenever possible.

- *Transmit Sensitive Data Over a Network Unencrypted* — Many protocols transmit data over the network unencrypted. These protocols include Telnet, FTP, HTTP, and SMTP. Many network file systems, such as NFS and SMB, also transmit information over the network unencrypted. It is the user's responsibility when using these protocols to limit what type of data is transmitted.

  Remote memory dump services, like **`netdump`**, transmit the contents of memory over the network unencrypted. Memory dumps can contain passwords or, even worse, database entries and other sensitive information.

  Other services like **`finger`** and **`rwhod`** reveal information about users of the system.

Examples of inherently insecure services include **`rlogin`**, **`rsh`**, **`telnet`**, and **`vsftpd`**.

All remote login and shell programs (**`rlogin`**, **`rsh`**, and **`telnet`**) should be avoided in favor of SSH. Refer to Section 48.1.7, “Security Enhanced Communication Tools” for more information about **`sshd`**.

FTP is not as inherently dangerous to the security of the system as remote shells, but FTP servers must be carefully configured and monitored to avoid problems. Refer to Section 48.2.6, “Securing FTP” for more information about securing FTP servers.

Services that should be carefully implemented and behind a firewall include:

- **`finger`**
- **`authd`** (this was called **`identd`** in previous Red Hat Enterprise Linux releases.)
- **`netdump`**
- **`netdump-server`**
- **`nfs`**
- **`rwhod`**
- **`sendmail`**
- **`smb`** (Samba)
- **`yppasswdd`**
- **`ypserv`**
- **`ypxfrd`**

More information on securing network services is available in Section 48.2, “Server Security”.

The next section discusses tools available to set up a simple firewall.

48.1.6. Personal Firewalls

After the *necessary* network services are configured, it is important to implement a firewall.

Important

You should configure the necessary services and implement a firewall *before* connecting to the Internet or any other network that you do not trust.

Firewalls prevent network packets from accessing the system's network interface. If a request is made to a port that is blocked by a firewall, the request is ignored. If a service is listening on one of these blocked ports, it does not receive the packets and is effectively disabled. For this reason, care should be taken when configuring a firewall to block access to ports not in use, while not blocking access to ports used by configured services.

For most users, the best tool for configuring a simple firewall is the graphical firewall configuration tool which ships with Red Hat Enterprise Linux: the **Security Level Configuration Tool** (`system-config-securitylevel`). This tool creates broad `iptables` rules for a general-purpose firewall using a control panel interface.

Refer to Section 48.8.2, “Basic Firewall Configuration” for more information about using this application and its available options.

For advanced users and server administrators, manually configuring a firewall with `iptables` is probably a better option. Refer to Section 48.8, “Firewalls” for more information. Refer to Section 48.9, “IPTables” for a comprehensive guide to the `iptables` command.

48.1.7. Security Enhanced Communication Tools

As the size and popularity of the Internet has grown, so has the threat of communication interception. Over the years, tools have been developed to encrypt communications as they are transferred over the network.

Red Hat Enterprise Linux ships with two basic tools that use high-level, public-key-cryptography-based encryption algorithms to protect information as it travels over the network.

- *OpenSSH* — A free implementation of the SSH protocol for encrypting network communication.
- *Gnu Privacy Guard (GPG)* — A free implementation of the PGP (Pretty Good Privacy) encryption application for encrypting data.

OpenSSH is a safer way to access a remote machine and replaces older, unencrypted services like `telnet` and `rsh`. OpenSSH includes a network service called `sshd` and three command line client applications:

- `ssh` — A secure remote console access client.
- `scp` — A secure remote copy command.
- `sftp` — A secure pseudo-ftp client that allows interactive file transfer sessions.

Important

Although the `sshd` service is inherently secure, the service *must* be kept up-to-date to prevent security threats. Refer to Section 47.5, “Security Updates” for more information.

GPG is one way to ensure private email communication. It can be used both to email sensitive data over public networks and to protect sensitive data on hard drives.

48.2. Server Security

When a system is used as a server on a public network, it becomes a target for attacks. Hardening the system and locking down services is therefore of paramount importance for the system administrator.

Before delving into specific issues, review the following general tips for enhancing server security:

- Keep all services current, to protect against the latest threats.
- Use secure protocols whenever possible.
- Serve only one type of network service per machine whenever possible.
- Monitor all servers carefully for suspicious activity.

48.2.1. Securing Services With TCP Wrappers and xinetd

TCP Wrappers provide access control to a variety of services. Most modern network services, such as SSH, Telnet, and FTP, make use of TCP Wrappers, which stand guard between an incoming request and the requested service.

The benefits offered by TCP Wrappers are enhanced when used in conjunction with **`xinetd`**, a super server that provides additional access, logging, binding, redirection, and resource utilization control.

Note

It is a good idea to use iptables firewall rules in conjunction with TCP Wrappers and **`xinetd`** to create redundancy within service access controls. Refer to Section 48.8, “Firewalls” for more information about implementing firewalls with iptables commands.

Refer to Section 18.2, “TCP Wrappers” for more information on configuring TCP Wrappers and **`xinetd`**.

The following subsections assume a basic knowledge of each topic and focus on specific security options.

48.2.1.1. Enhancing Security With TCP Wrappers

TCP Wrappers are capable of much more than denying access to services. This section illustrates how they can be used to send connection banners, warn of attacks from particular hosts, and enhance logging functionality. Refer to the **`hosts_options`** man page for information about the TCP Wrapper functionality and control language.

48.2.1.1.1. TCP Wrappers and Connection Banners

Displaying a suitable banner when users connect to a service is a good way to let potential attackers know that the system administrator is being vigilant. You can also control what information about the system is presented to users. To implement a TCP Wrappers banner for a service, use the **`banner`** option.

This example implements a banner for **`vsftpd`**. To begin, create a banner file. It can be anywhere on the system, but it must have same name as the daemon. For this example, the file is called **`/etc/banners/vsftpd`** and contains the following line:

```
220-Hello, %c
220-All activity on ftp.example.com is logged.
220-Inappropriate use will result in your access privileges being removed.
```

The **`%c`** token supplies a variety of client information, such as the username and hostname, or the username and IP address to make the connection even more intimidating.

For this banner to be displayed to incoming connections, add the following line to the **`/etc/hosts.allow`** file:

```
vsftpd : ALL : banners /etc/banners/
```

48.2.1.1.2. TCP Wrappers and Attack Warnings

If a particular host or network has been detected attacking the server, TCP Wrappers can be used to warn the administrator of subsequent attacks from that host or network using the **`spawn`** directive.

In this example, assume that a cracker from the 206.182.68.0/24 network has been detected attempting to attack the server. Place the following line in the **`/etc/hosts.deny`** file to deny any connection attempts from that network, and to log the attempts to a special file:

```
ALL : 206.182.68.0 : spawn /bin/ 'date' %c %d >> /var/log/intruder_alert
```

The **`%d`** token supplies the name of the service that the attacker was trying to access.

To allow the connection and log it, place the **`spawn`** directive in the **`/etc/hosts.allow`** file.

Note

Because the **`spawn`** directive executes any shell command, create a special script to notify the administrator or execute a chain of commands in the event that a particular client attempts to connect to the server.

48.2.1.1.3. TCP Wrappers and Enhanced Logging

If certain types of connections are of more concern than others, the log level can be elevated for that service using the **`severity`** option.

For this example, assume that anyone attempting to connect to port 23 (the Telnet port) on an FTP server is a cracker. To denote this, place an **`emerg`** flag in the log files instead of the default flag, **`info`**, and deny the connection.

To do this, place the following line in **`/etc/hosts.deny`**:

```
in.telnetd : ALL : severity emerg
```

This uses the default **`authpriv`** logging facility, but elevates the priority from the default value of **`info`** to **`emerg`**, which posts log messages directly to the console.

48.2.1.2. Enhancing Security With xinetd

This section focuses on using **`xinetd`** to set a trap service and using it to control resource levels available to any given **`xinetd`** service. Setting resource limits for services can help thwart *Denial of Service* (DoS) attacks. Refer to the man pages for **`xinetd`** and **`xinetd.conf`** for a list of available options.

48.2.1.2.1. Setting a Trap

One important feature of **`xinetd`** is its ability to add hosts to a global **`no_access`** list. Hosts on this list are denied subsequent connections to services managed by **`xinetd`** for a specified period or until **`xinetd`** is restarted. You can do this using the **`SENSOR`** attribute. This is an easy way to block hosts attempting to scan the ports on the server.

The first step in setting up a **SENSOR** is to choose a service you do not plan on using. For this example, Telnet is used.

Edit the file **/etc/xinetd.d/telnet** and change the **flags** line to read:

```
flags           = SENSOR
```

Add the following line:

```
deny_time       = 30
```

This denies any further connection attempts to that port by that host for 30 minutes. Other acceptable values for the **deny_time** attribute are FOREVER, which keeps the ban in effect until **xinetd** is restarted, and NEVER, which allows the connection and logs it.

Finally, the last line should read:

```
disable         = no
```

This enables the trap itself.

While using **SENSOR** is a good way to detect and stop connections from undesirable hosts, it has two drawbacks:

- It does not work against stealth scans.
- An attacker who knows that a **SENSOR** is running can mount a Denial of Service attack against particular hosts by forging their IP addresses and connecting to the forbidden port.

48.2.1.2.2. Controlling Server Resources

Another important feature of **xinetd** is its ability to set resource limits for services under its control.

It does this using the following directives:

- **cps = <number_of_connections> <wait_period>** — Limits the rate of incoming connections. This directive takes two arguments:
  - **<number_of_connections>** — The number of connections per second to handle. If the rate of incoming connections is higher than this, the service is temporarily disabled. The default value is fifty (50).
  - **<wait_period>** — The number of seconds to wait before re-enabling the service after it has been disabled. The default interval is ten (10) seconds.
- **instances = <number_of_connections>** — Specifies the total number of connections allowed to a service. This directive accepts either an integer value or **UNLIMITED**.
- **per_source = <number_of_connections>** — Specifies the number of connections allowed to a service by each host. This directive accepts either an integer value or **UNLIMITED**.
- **rlimit_as = <number[K|M]>** — Specifies the amount of memory address space the service can occupy in kilobytes or megabytes. This directive accepts either an integer value or **UNLIMITED**.
- **rlimit_cpu = <number_of_seconds>** — Specifies the amount of time in seconds that a service may occupy the CPU. This directive accepts either an integer value or **UNLIMITED**.

Using these directives can help prevent any single **xinetd** service from overwhelming the system, resulting in a denial of service.

48.2.2. Securing Portmap

The **portmap** service is a dynamic port assignment daemon for RPC services such as NIS and NFS. It has weak authentication mechanisms and has the ability to assign a wide range of ports for the services it controls. For these reasons, it is difficult to secure.

Note

Securing **portmap** only affects NFSv2 and NFSv3 implementations, since NFSv4 no longer requires it. If you plan to implement an NFSv2 or NFSv3 server, then **portmap** is required, and the following section applies.

If running RPC services, follow these basic rules.

48.2.2.1. Protect portmap With TCP Wrappers

It is important to use TCP Wrappers to limit which networks or hosts have access to the **portmap** service since it has no built-in form of authentication.

Further, use *only* IP addresses when limiting access to the service. Avoid using hostnames, as they can be forged by DNS poisoning and other methods.

48.2.2.2. Protect portmap With iptables

To further restrict access to the **portmap** service, it is a good idea to add iptables rules to the server and restrict access to specific networks.

Below are two example iptables commands. The first allows TCP connections to the port 111 (used by the **portmap** service) from the 192.168.0.0/24 network. The second allows TCP connections to the same port from the localhost. This is necessary for the **sgi_fam** service used by **Nautilus**. All other packets are dropped.

```
iptables -A INPUT -p tcp -s! 192.168.0.0/24 --dport 111 -j DROP
iptables -A INPUT -p tcp -s 127.0.0.1 --dport 111 -j ACCEPT
```

To similarly limit UDP traffic, use the following command.

```
iptables -A INPUT -p udp -s! 192.168.0.0/24 --dport 111 -j DROP
```

Note

Refer to Section 48.8, “Firewalls” for more information about implementing firewalls with iptables commands.

48.2.3. Securing NIS

The *Network Information Service* (NIS) is an RPC service, called **`ypserv`**,--> which is used in conjunction with **`portmap`** and other related services to distribute maps of usernames, passwords, and other sensitive information to any computer claiming to be within its domain.

An NIS server is comprised of several applications. They include the following:

- **`/usr/sbin/rpc.yppasswdd`** — Also called the **`yppasswdd`** service, this daemon allows users to change their NIS passwords.
- **`/usr/sbin/rpc.ypxfrd`** — Also called the **`ypxfrd`** service, this daemon is responsible for NIS map transfers over the network.
- **`/usr/sbin/yppush`** — This application propagates changed NIS databases to multiple NIS servers.
- **`/usr/sbin/ypserv`** — This is the NIS server daemon.

NIS is somewhat insecure by today's standards. It has no host authentication mechanisms and transmits all of its information over the network unencrypted, including password hashes. As a result, extreme care must be taken when setting up a network that uses NIS. This is further complicated by the fact that the default configuration of NIS is inherently insecure.

It is recommended that anyone planning to implement an NIS server first secure the **`portmap`** service as outlined in Section 48.2.2, “Securing Portmap”, then address the following issues, such as network planning.

48.2.3.1. Carefully Plan the Network

Because NIS transmits sensitive information unencrypted over the network, it is important the service be run behind a firewall and on a segmented and secure network. Whenever NIS information is transmitted over an insecure network, it risks being intercepted. Careful network design can help prevent severe security breaches.

48.2.3.2. Use a Password-like NIS Domain Name and Hostname

Any machine within an NIS domain can use commands to extract information from the server without authentication, as long as the user knows the NIS server's DNS hostname and NIS domain name.

For instance, if someone either connects a laptop computer into the network or breaks into the network from outside (and manages to spoof an internal IP address), the following command reveals the **`/etc/passwd`** map:

```
ypcat -d <NIS_domain> -h <DNS_hostname> passwd
```

If this attacker is a root user, they can obtain the **`/etc/shadow`** file by typing the following command:

```
ypcat -d <NIS_domain> -h <DNS_hostname> shadow
```

Note

If Kerberos is used, the **`/etc/shadow`** file is not stored within an NIS map.

To make access to NIS maps harder for an attacker, create a random string for the DNS hostname, such as **`o7hfawtgmhwg.domain.com`**. Similarly, create a *different* randomized NIS domain name. This makes it much more difficult for an attacker to access the NIS server.

48.2.3.3. Edit the `/var/yp/securenets` File

If the **`/var/yp/securenets`** file is blank or does not exist (as is the case after a default installation), NIS listens to all networks. One of the first things to do is to put netmask/network pairs in the file so that **`ypserv`** only responds to requests from the appropriate network.

Below is a sample entry from a **`/var/yp/securenets`** file:

```
255.255.255.0     192.168.0.0
```

Warning

Never start an NIS server for the first time without creating the **`/var/yp/securenets`** file.

This technique does not provide protection from an IP spoofing attack, but it does at least place limits on what networks the NIS server services.

48.2.3.4. Assign Static Ports and Use iptables Rules

All of the servers related to NIS can be assigned specific ports except for **`rpc.yppasswdd`** — the daemon that allows users to change their login passwords. Assigning ports to the other two NIS server daemons, **`rpc.ypxfrd`** and **`ypserv`**, allows for the creation of firewall rules to further protect the NIS server daemons from intruders.

To do this, add the following lines to **`/etc/sysconfig/network`**:

```
YPSERV_ARGS="-p 834" YPXFRD_ARGS="-p 835"
```

The following iptables rules can then be used to enforce which network the server listens to for these ports:

```
iptables -A INPUT -p tcp -s! 192.168.0.0/24 --dport 834 -j DROP
iptables -A INPUT -p tcp -s! 192.168.0.0/24 --dport 835 -j DROP
iptables -A INPUT -p udp -s! 192.168.0.0/24 --dport 834 -j DROP
iptables -A INPUT -p udp -s! 192.168.0.0/24 --dport 835 -j DROP
```

This means that the server only allows connections to ports 834 and 835 if the requests come from the 192.168.0.0/24 network.

Note

Refer to Section 48.8, “Firewalls” for more information about implementing firewalls with iptables commands.

48.2.3.5. Use Kerberos Authentication

One of the issues to consider when NIS is used for authentication is that whenever a user logs into a machine, a password hash from the **`/etc/shadow`** map is sent over the network. If an intruder gains access to an NIS domain and sniffs network traffic, they can collect usernames and password hashes. With enough time, a password cracking program can guess weak passwords, and an attacker can gain access to a valid account on the network.

Since Kerberos uses secret-key cryptography, no password hashes are ever sent over the network, making the system far more secure. Refer to Section 48.6, "Kerberos" for more information about Kerberos.

48.2.4. Securing NFS

The *Network File System* (NFS) is a service that provides network accessible file systems for client machines. Refer to Chapter 21, *Network File System (NFS)* for more information about NFS. The following subsections assume a basic knowledge of NFS.

Important

The version of NFS included in Red Hat Enterprise Linux, NFSv4, no longer requires the **`portmap`** service as outlined in Section 48.2.2, "Securing Portmap". NFS traffic now utilizes TCP in all versions, rather than UDP, and requires it when using NFSv4. NFSv4 now includes Kerberos user and group authentication, as part of the **`RPCSEC_GSS`** kernel module. Information on **`portmap`** is still included, since Red Hat Enterprise Linux supports NFSv2 and NFSv3, both of which utilize **`portmap`**.

48.2.4.1. Carefully Plan the Network

Now that NFSv4 has the ability to pass all information encrypted using Kerberos over a network, it is important that the service be configured correctly if it is behind a firewall or on a segmented network. NFSv2 and NFSv3 still pass data insecurely, and this should be taken into consideration. Careful network design in all of these regards can help prevent security breaches.

48.2.4.2. Beware of Syntax Errors

The NFS server determines which file systems to export and which hosts to export these directories to by consulting the **`/etc/exports`** file. Be careful not to add extraneous spaces when editing this file.

For instance, the following line in the **`/etc/exports`** file shares the directory **`/tmp/nfs/`** to the host **`bob.example.com`** with read/write permissions.

```
/tmp/nfs/     bob.example.com(rw)
```

The following line in the **`/etc/exports`** file, on the other hand, shares the same directory to the host **`bob.example.com`** with read-only permissions and shares it to the *world* with read/write permissions due to a single space character after the hostname.

```
/tmp/nfs/     bob.example.com (rw)
```

It is good practice to check any configured NFS shares by using the **`showmount`** command to verify what is being shared:

```
showmount -e <hostname>
```

48.2.4.3. Do Not Use the `no_root_squash` Option

By default, NFS shares change the root user to the **`nfsnobody`** user, an unprivileged user account. This changes the owner of all root-created files to **`nfsnobody`**, which prevents uploading of programs with the setuid bit set.

If **no_root_squash** is used, remote root users are able to change any file on the shared file system and leave applications infected by Trojans for other users to inadvertently execute.

48.2.5. Securing the Apache HTTP Server

The Apache HTTP Server is one of the most stable and secure services that ships with Red Hat Enterprise Linux. A large number of options and techniques are available to secure the Apache HTTP Server — too numerous to delve into deeply here.

When configuring the Apache HTTP Server, it is important to read the documentation available for the application. This includes Chapter 25, *Apache HTTP Server*, and the Stronghold manuals, available at http://www.redhat.com/docs/manuals/stronghold/.

System Administrators should be careful when using the following configuration options:

48.2.5.1. `FollowSymLinks`

This directive is enabled by default, so be sure to use caution when creating symbolic links to the document root of the Web server. For instance, it is a bad idea to provide a symbolic link to `/`.

48.2.5.2. The `Indexes` Directive

This directive is enabled by default, but may not be desirable. To prevent visitors from browsing files on the server, remove this directive.

48.2.5.3. The `UserDir` Directive

The **UserDir** directive is disabled by default because it can confirm the presence of a user account on the system. To enable user directory browsing on the server, use the following directives:

```
UserDir enabled
UserDir disabled root
```

These directives activate user directory browsing for all user directories other than **/root/**. To add users to the list of disabled accounts, add a space-delimited list of users on the **UserDir disabled** line.

48.2.5.4. Do Not Remove the `IncludesNoExec` Directive

By default, the *Server-Side Includes* (SSI) module cannot execute commands. It is recommended that you do not change this setting unless absolutely necessary, as it could potentially enable an attacker to execute commands on the system.

48.2.5.5. Restrict Permissions for Executable Directories

Ensure that only the root user has write permissions to any directory containing scripts or CGIs. To do this, type the following commands:

```
chown root <directory_name>
chmod 755 <directory_name>
```

Important

Always verify that any scripts running on the system work as intended *before* putting them into production.

48.2.6. Securing FTP

The *File Transfer Protocol* (FTP) is an older TCP protocol designed to transfer files over a network. Because all transactions with the server, including user authentication, are unencrypted, it is considered an insecure protocol and should be carefully configured.

Red Hat Enterprise Linux provides three FTP servers.

- **`gssftpd`** — A Kerberos-aware **`xinetd`**-based FTP daemon that does not transmit authentication information over the network.
- **Red Hat Content Accelerator** (**`tux`**) — A kernel-space Web server with FTP capabilities.
- **`vsftpd`** — A standalone, security oriented implementation of the FTP service.

The following security guidelines are for setting up the **`vsftpd`** FTP service.

48.2.6.1. FTP Greeting Banner

Before submitting a username and password, all users are presented with a greeting banner. By default, this banner includes version information useful to crackers trying to identify weaknesses in a system.

To change the greeting banner for **`vsftpd`**, add the following directive to the **`/etc/vsftpd/vsftpd.conf`** file:

```
ftpd_banner=<insert_greeting_here>
```

Replace <*insert_greeting_here*> in the above directive with the text of the greeting message.

For mutli-line banners, it is best to use a banner file. To simplify management of multiple banners, place all banners in a new directory called **`/etc/banners/`**. The banner file for FTP connections in this example is **`/etc/banners/ftp.msg`**. Below is an example of what such a file may look like:

```
######### # Hello, all activity on ftp.example.com is logged. #########
```

Note

It is not necessary to begin each line of the file with **`220`** as specified in Section 48.2.1.1.1, "TCP Wrappers and Connection Banners".

To reference this greeting banner file for **`vsftpd`**, add the following directive to the **`/etc/vsftpd/vsftpd.conf`** file:

```
banner_file=/etc/banners/ftp.msg
```

Important

Make sure that you specify the path to the banner file correctly in **/etc/vsftpd/vsftpd.conf**, or else every attempt to connect to **vsftpd** will result in the connection being closed immediately and a **500 OOPS: cannot open banner *<path_to_banner_file>*** error message.

Note that the **banner_file** directive in **/etc/vsftpd/vfsftpd.conf** takes precedence over any **ftpd_banner** directives in the configuration file: if **banner_file** is specified, then **ftpd_banner** is ignored.

It also is possible to send additional banners to incoming connections using TCP Wrappers as described in Section 48.2.1.1.1, “TCP Wrappers and Connection Banners”.

48.2.6.2. Anonymous Access

The presence of the **/var/ftp/** directory activates the anonymous account.

The easiest way to create this directory is to install the **vsftpd** package. This package establishes a directory tree for anonymous users and configures the permissions on directories to read-only for anonymous users.

By default the anonymous user cannot write to any directories.

Warning

If enabling anonymous access to an FTP server, be aware of where sensitive data is stored.

48.2.6.2.1. Anonymous Upload

To allow anonymous users to upload files, it is recommended that a write-only directory be created within **/var/ftp/pub/**.

To do this, type the following command:

```
mkdir /var/ftp/pub/upload
```

Next, change the permissions so that anonymous users cannot view the contents of the directory:

```
chmod 730 /var/ftp/pub/upload
```

A long format listing of the directory should look like this:

```
drwx-wx---    2 root     ftp          4096 Feb 13 20:05 upload
```

Warning

Administrators who allow anonymous users to read and write in directories often find that their servers become a repository of stolen software.

Additionally, under **vsftpd**, add the following line to the **/etc/vsftpd/vsftpd.conf** file:

```
anon_upload_enable=YES
```

48.2.6.3. User Accounts

Because FTP transmits unencrypted usernames and passwords over insecure networks for authentication, it is a good idea to deny system users access to the server from their user accounts.

To disable all user accounts in **vsftpd**, add the following directive to **/etc/vsftpd/vsftpd.conf**:

```
local_enable=NO
```

48.2.6.3.1. Restricting User Accounts

To disable FTP access for specific accounts or specific groups of accounts, such as the root user and those with **sudo** privileges, the easiest way is to use a PAM list file as described in Section 48.1.4.2, “Disallowing Root Access”. The PAM configuration file for **vsftpd** is **/etc/pam.d/vsftpd**.

It is also possible to disable user accounts within each service directly.

To disable specific user accounts in **vsftpd**, add the username to **/etc/vsftpd.ftpusers**

48.2.6.4. Use TCP Wrappers To Control Access

Use TCP Wrappers to control access to either FTP daemon as outlined in Section 48.2.1.1, “Enhancing Security With TCP Wrappers”.

48.2.7. Securing Sendmail

Sendmail is a Mail Transport Agent (MTA) that uses the Simple Mail Transport Protocol (SMTP) to deliver electronic messages between other MTAs and to email clients or delivery agents. Although many MTAs are capable of encrypting traffic between one another, most do not, so sending email over any public networks is considered an inherently insecure form of communication.

Refer to Chapter 27, *Email* for more information about how email works and an overview of common configuration settings. This section assumes a basic knowledge of how to generate a valid **/etc/mail/sendmail.cf** by editing the **/etc/mail/sendmail.mc** and using the **m4** command.

It is recommended that anyone planning to implement a Sendmail server address the following issues.

48.2.7.1. Limiting a Denial of Service Attack

Because of the nature of email, a determined attacker can flood the server with mail fairly easily and cause a denial of service. By setting limits to the following directives in **/etc/mail/sendmail.mc**, the effectiveness of such attacks is limited.

- **confCONNECTION_RATE_THROTTLE** — The number of connections the server can receive per second. By default, Sendmail does not limit the number of connections. If a limit is set and reached, further connections are delayed.

- **confMAX_DAEMON_CHILDREN** — The maximum number of child processes that can be spawned by the server. By default, Sendmail does not assign a limit to the number of child processes. If a limit is set and reached, further connections are delayed.

- **confMIN_FREE_BLOCKS** — The minimum number of free blocks which must be available for the server to accept mail. The default is 100 blocks.
- **confMAX_HEADERS_LENGTH** — The maximum acceptable size (in bytes) for a message header.
- **confMAX_MESSAGE_SIZE** — The maximum acceptable size (in bytes) for a single message.

48.2.7.2. NFS and Sendmail

Never put the mail spool directory, **/var/spool/mail/**, on an NFS shared volume.

Because NFSv2 and NFSv3 do not maintain control over user and group IDs, two or more users can have the same UID, and receive and read each other's mail.

Note

With NFSv4 using Kerberos, this is not the case, since the **SECRPC_GSS** kernel module does not utilize UID-based authentication. However, it is considered good practice *not* to put the mail spool directory on NFS shared volumes.

48.2.7.3. Mail-only Users

To help prevent local user exploits on the Sendmail server, it is best for mail users to only access the Sendmail server using an email program. Shell accounts on the mail server should not be allowed and all use shells in the **/etc/passwd** file should be set to **/sbin/nologin** (with the possible exception of the root user).

48.2.8. Verifying Which Ports Are Listening

After configuring network services, it is important to pay attention to which ports are actually listening on the system's network interfaces. Any open ports can be evidence of an intrusion.

There are two basic approaches for listing the ports that are listening on the network. The less reliable approach is to query the network stack using commands such as **netstat -an** or **lsof -i**. This method is less reliable since these programs do not connect to the machine from the network, but rather check to see what is running on the system. For this reason, these applications are frequent targets for replacement by attackers. Crackers attempt to cover their tracks if they open unauthorized network ports by replacing **netstat** and **lsof** with their own, modified versions.

A more reliable way to check which ports are listening on the network is to use a port scanner such as **nmap**.

The following command issued from the console determines which ports are listening for TCP connections from the network:

```
nmap -sT -O localhost
```

The output of this command appears as follows:

```
Starting nmap 3.55 ( http://www.insecure.org/nmap/ ) at 2004-09-24 13:49 EDT
Interesting ports on localhost.localdomain (127.0.0.1):
(The 1653 ports scanned but not shown below are in state: closed)
PORT      STATE SERVICE
22/tcp    open  ssh
```

```
25/tcp    open  smtp
111/tcp   open  rpcbind
113/tcp   open  auth
631/tcp   open  ipp
834/tcp   open  unknown
2601/tcp  open  zebra
32774/tcp open  sometimes-rpc11
Device type: general purpose
Running: Linux 2.4.X|2.5.X|2.6.X OS details: Linux 2.5.25 - 2.6.3 or Gentoo
1.2 Linux 2.4.19 rc1-rc7)
Uptime 12.857 days (since Sat Sep 11 17:16:20 2004)
Nmap run completed -- 1 IP address (1 host up) scanned in 5.190 seconds
```

This output shows the system is running **`portmap`** due to the presence of the **`sunrpc`** service. However, there is also a mystery service on port 834. To check if the port is associated with the official list of known services, type:

```
cat /etc/services | grep 834
```

This command returns no output. This indicates that while the port is in the reserved range (meaning 0 through 1023) and requires root access to open, it is not associated with a known service.

Next, check for information about the port using **`netstat`** or **`lsof`**. To check for port 834 using **`netstat`**, use the following command:

```
netstat -anp | grep 834
```

The command returns the following output:

```
tcp   0    0 0.0.0.0:834    0.0.0.0:*   LISTEN   653/ypbind
```

The presence of the open port in **`netstat`** is reassuring because a cracker opening a port surreptitiously on a hacked system is not likely to allow it to be revealed through this command. Also, the **`[p]`** option reveals the process ID (PID) of the service that opened the port. In this case, the open port belongs to **`ypbind`** (NIS), which is an RPC service handled in conjunction with the **`portmap`** service.

The **`lsof`** command reveals similar information to **`netstat`** since it is also capable of linking open ports to services:

```
lsof -i | grep 834
```

The relevant portion of the output from this command follows:

```
ypbind      653        0    7u  IPv4       1319                 TCP *:834
(LISTEN)
ypbind      655        0    7u  IPv4       1319                 TCP *:834
(LISTEN)
ypbind      656        0    7u  IPv4       1319                 TCP *:834
(LISTEN)
ypbind      657        0    7u  IPv4       1319                 TCP *:834
(LISTEN)
```

These tools reveal a great deal about the status of the services running on a machine. These tools are flexible and can provide a wealth of information about network services and configuration. Refer to the man pages for **`lsof`**, **`netstat`**, **`nmap`**, and **`services`** for more information.

48.3. Single Sign-on (SSO)

48.3.1. Introduction

The Red Hat Enterprise Linux SSO functionality reduces the number of times Red Hat Enterprise Linux desktop users have to enter their passwords. Several major applications leverage the same underlying authentication and authorization mechanisms so that users can log in to Red Hat Enterprise Linux from the log-in screen, and then not need to re-enter their passwords. These applications are detailed below.

In addition, users can log in to their machines even when there is no network (*offline mode*) or where network connectivity is unreliable, for example, wireless access. In the latter case, services will degrade gracefully.

48.3.1.1. Supported Applications

The following applications are currently supported by the unified log-in scheme in Red Hat Enterprise Linux:

- Login
- Screensaver
- Firefox and Thunderbird

48.3.1.2. Supported Authentication Mechanisms

Red Hat Enterprise Linux currently supports the following authentication mechanisms:

- Kerberos name/password login
- Smart card/PIN login

48.3.1.3. Supported Smart Cards

Red Hat Enterprise Linux has been tested with the Cyberflex e-gate card and reader, but any card that complies with both Java card 2.1.1 and Global Platform 2.0.1 specifications should operate correctly, as should any reader that is supported by PCSC-lite.

Red Hat Enterprise Linux has also been tested with Common Access Cards (CAC). The supported reader for CAC is the SCM SCR 331 USB Reader.

As of Red Hat Enterprise Linux 5.2, Gemalto smart cards (Cyberflex Access 64k v2, standard with DER SHA1 value configured as in PKCSI v2.1) are now supported. These smart cards now use readers compliant with Chip/Smart Card Interface Devices (CCID).

48.3.1.4. Advantages of Red Hat Enterprise Linux Single Sign-on

Numerous security mechanisms currently exist that utilize a large number of protocols and credential stores. Examples include SSL, SSH, IPsec, and Kerberos. Red Hat Enterprise Linux SSO aims to unify these schemes to support the requirements listed above. This does not mean replacing Kerberos with X.509v3 certificates, but rather uniting them to reduce the burden on both system users and the administrators who manage them.

To achieve this goal, Red Hat Enterprise Linux:

- Provides a single, shared instance of the NSS crypto libraries on each operating system.
- Ships the Certificate System's Enterprise Security Client (ESC) with the base operating system. The ESC application monitors smart card insertion events. If it detects that the user has inserted a smart card that was designed to be used with the Red Hat Enterprise Linux Certificate System server product, it displays a user interface instructing the user how to enroll that smart card.
- Unifies Kerberos and NSS so that users who log in to the operating system using a smart card also obtain a Kerberos credential (which allows them to log in to file servers, etc.)

48.3.2. Getting Started with your new Smart Card

Before you can use your smart card to log in to your system and take advantage of the increased security options this technology provides, you need to perform some basic installation and configuration steps. These are described below.

Note

This section provides a high-level view of getting started with your smart card. More detailed information is available in the Red Hat Certificate System Enterprise Security Client Guide.

1. Log in with your Kerberos name and password
2. Make sure you have the **nss-tools** package loaded.
3. Download and install your corporate-specific root certificates. Use the following command to install the root CA certificate:

   ```
   certutil -A -d /etc/pki/nssdb -n "root ca cert" -t "CT,C,C" \
    -i ./ca_cert_in_base64_format.crt
   ```

4. Verify that you have the following RPMs installed on your system: esc, pam_pkcs11, coolkey, ifd-egate, ccid, gdm, authconfig, and authconfig-gtk.
5. Enable Smart Card Login Support

   a. On the Gnome Title Bar, select System->Administration->Authentication.

   b. Type your machine's root password if necessary.

   c. In the Authentication Configuration dialog, click the **Authentication** tab.

   d. Select the **Enable Smart Card Support** check box.

   e. Click the **Configure Smart Card...** button to display the Smartcard Settings dialog, and specify the required settings:

      - **Require smart card for login** — Clear this check box. After you have successfully logged in with the smart card you can select this option to prevent users from logging in without a smart card.
      - **Card Removal Action** — This controls what happens when you remove the smart card after you have logged in. The available options are:

- **`Lock`** — Removing the smart card locks the X screen.
- **`Ignore`** — Removing the smart card has no effect.

6. If you need to enable the Online Certificate Status Protocol (OCSP), open the **`/etc/pam_pkcs11/pam_pkcs11.conf`** file, and locate the following line:

   `enable_ocsp = false;`

   Change this value to true, as follows:

   `enable_ocsp = true;`

7. Enroll your smart card

8. If you are using a CAC card, you also need to perform the following steps:

   a. Change to the root account and create a file called **`/etc/pam_pkcs11/cn_map`**.

   b. Add the following entry to the **`cn_map`** file:

      MY.CAC_CN.123454 -> *myloginid*

      where *MY.CAC_CN.123454* is the Common Name on your CAC and *myloginid* is your UNIX login ID.

9. Logout

48.3.2.1. Troubleshooting

If you have trouble getting your smart card to work, try using the following command to locate the source of the problem:

```
pklogin_finder debug
```

If you run the **`pklogin_finder`** tool in debug mode while an enrolled smart card is plugged in, it attempts to output information about the validity of certificates, and if it is successful in attempting to map a login ID from the certificates that are on the card.

48.3.3. How Smart Card Enrollment Works

Smart cards are said to be *enrolled* when they have received an appropriate certificate signed by a valid Certificate Authority (CA). This involves several steps, described below:

1. The user inserts their smart card into the smart card reader on their workstation. This event is recognized by the Enterprise Security Client (ESC).

2. The enrollment page is displayed on the user's desktop. The user completes the required details and the user's system then connects to the Token Processing System (TPS) and the CA.

3. The TPS enrolls the smart card using a certificate signed by theCA.

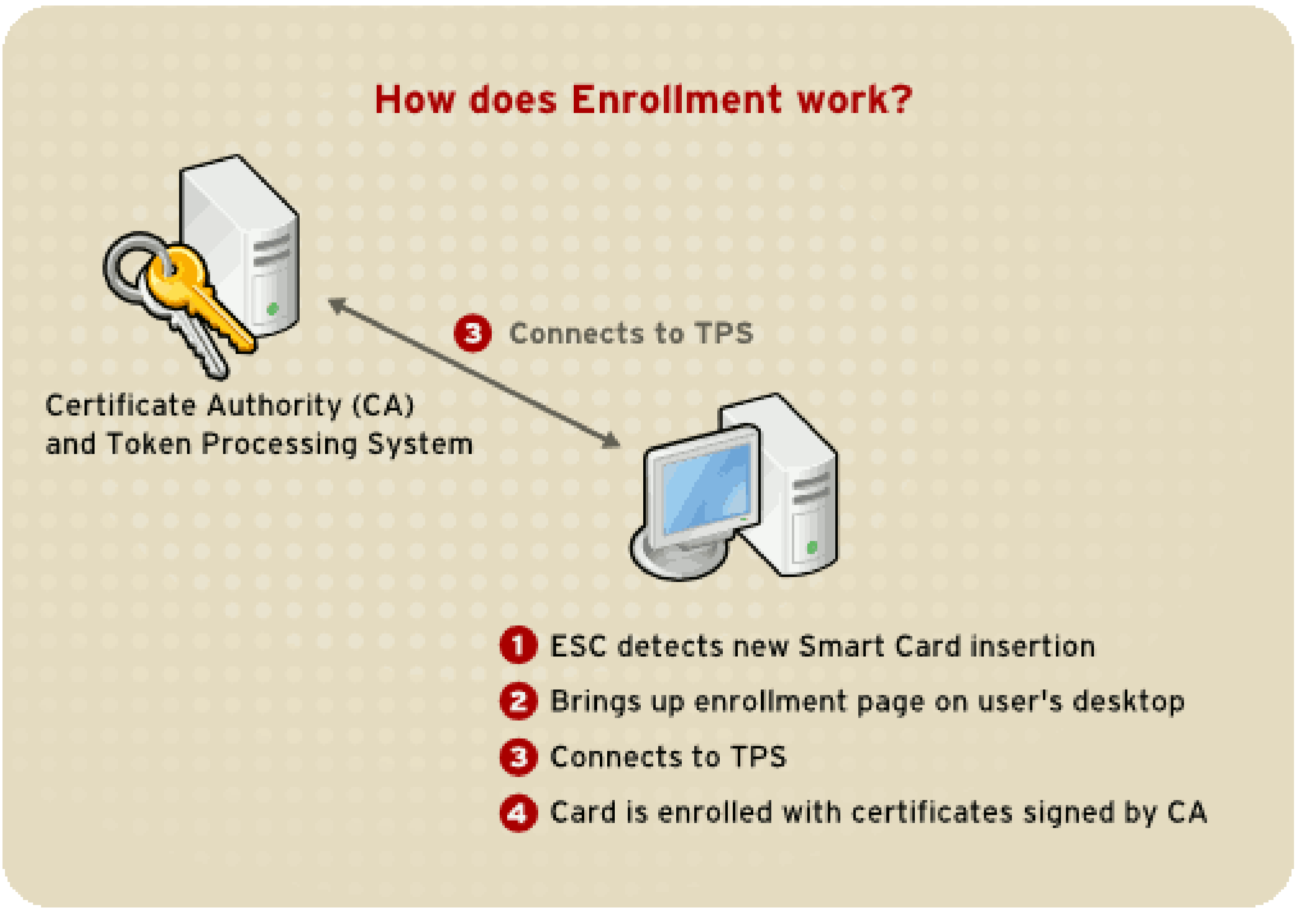

How does Enrollment work?
Certificate Authority (CA) and Token Processing System
3 Connects to TPS
1 ESC detects new Smart Card insertion
2 Brings up enrollment page on user's desktop
3 Connects to TPS
4 Card is enrolled with certificates signed by CA

Figure 48.4. How Smart Card Enrollment Works

48.3.4. How Smart Card Login Works

This section provides a brief overview of the process of logging in using a smart card.

1. When the user inserts their smart card into the smart card reader, this event is recognized by the PAM facility, which prompts for the user's PIN.

2. The system then looks up the user's current certificates and verifies their validity. The certificate is then mapped to the user's UID.

3. This is validated against the KDC and login granted.

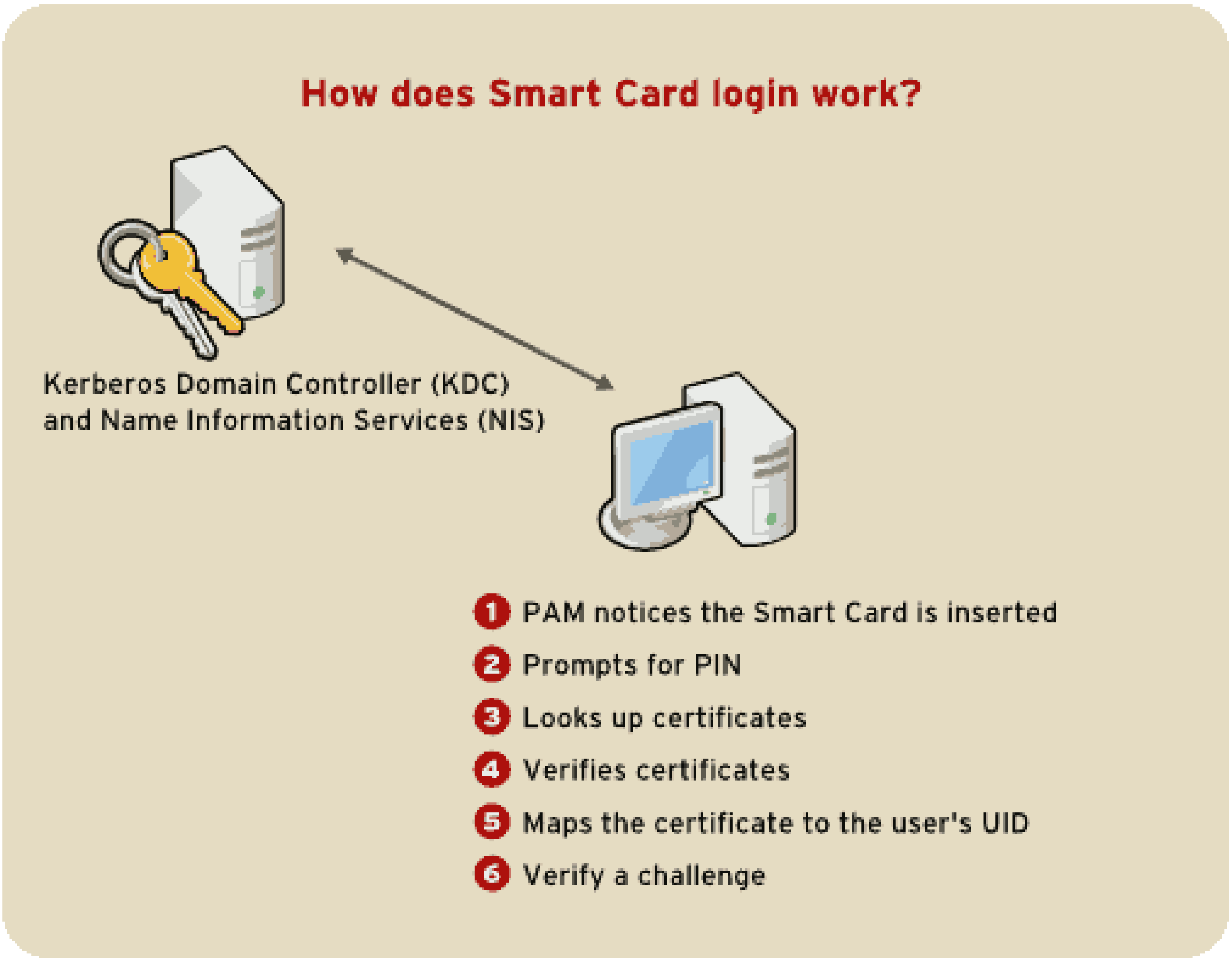

How does Smart Card login work?
Kerberos Domain Controller (KDC) and Name Information Services (NIS)
1 PAM notices the Smart Card is inserted
2 Prompts for PIN
3 Looks up certificates
4 Verifies certificates
5 Maps the certificate to the user's UID
6 Verify a challenge

Figure 48.5. How Smart Card Login Works

Note

You cannot log in with a card that has not been enrolled, even if it has been formatted. You need to log in with a formatted, enrolled card, or not using a smart card, before you can enroll a new card.

Refer to Section 48.6, “Kerberos” and Section 48.4, “Pluggable Authentication Modules (PAM)” for more information on Kerberos and PAM.

48.3.5. Configuring Firefox to use Kerberos for SSO

You can configure Firefox to use Kerberos for Single Sign-on. In order for this functionality to work correctly, you need to configure your web browser to send your Kerberos credentials to the appropriate KDC.The following section describes the configuration changes and other requirements to achieve this.

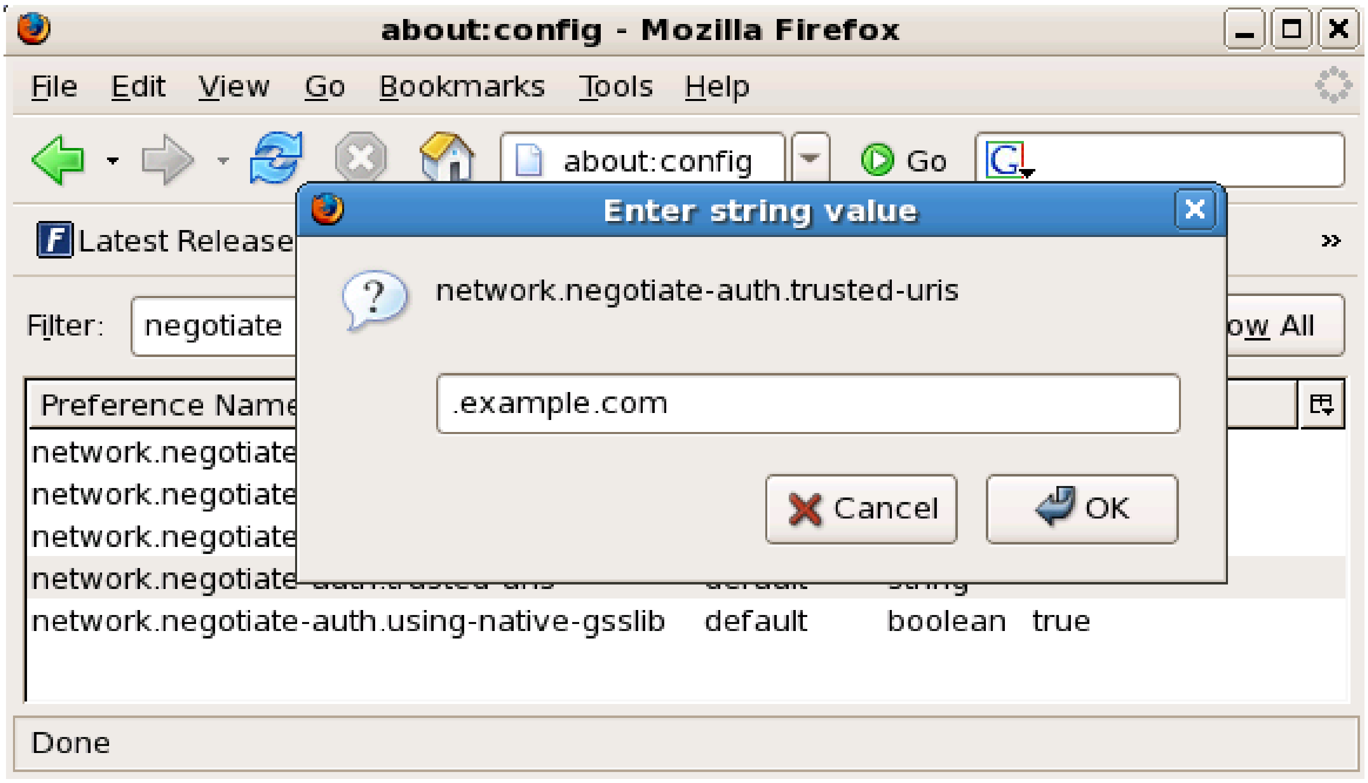
1. In the address bar of Firefox, type **`about:config`** to display the list of current configuration options.
2. In the **`Filter`** field, type **`negotiate`** to restrict the list of options.
3. Double-click the *network.negotiate-auth.trusted-uris* entry to display the *Enter string value* dialog box.
4. Enter the name of the domain against which you want to authenticate, for example, *.example.com*.

5. Repeat the above procedure for the *network.negotiate-auth.delegation-uris* entry, using the same domain.

Note

You can leave this value blank, as it allows Kerberos ticket passing, which is not required.

If you do not see these two configuration options listed, your version of Firefox may be too old to support Negotiate authentication, and you should consider upgrading.



network.negotiate-auth.trusted-uris

.example.com

network.negotiate-auth.using-native-gsslib default boolean true

Figure 48.6. Configuring Firefox for SSO with Kerberos

You now need to ensure that you have Kerberos tickets. In a command shell, type **kinit** to retrieve Kerberos tickets. To display the list of available tickets, type **klist**. The following shows an example output from these commands:

```
~]$ kinit
Password for user@EXAMPLE.COM:

~]$ klist
Ticket cache: FILE:/tmp/krb5cc_10920
Default principal: user@EXAMPLE.COM

Valid starting     Expires            Service principal
10/26/06 23:47:54  10/27/06 09:47:54  krbtgt/USER.COM@USER.COM
        renew until 10/26/06 23:47:54

Kerberos 4 ticket cache: /tmp/tkt10920
klist: You have no tickets cached
```

48.3.5.1. Troubleshooting

If you have followed the configuration steps above and Negotiate authentication is not working, you can turn on verbose logging of the authentication process. This could help you find the cause of the problem. To enable verbose logging, use the following procedure:

1. Close all instances of Firefox.

2. Open a command shell, and enter the following commands:

   ```
   export NSPR_LOG_MODULES=negotiateauth:5
   export NSPR_LOG_FILE=/tmp/moz.log
   ```

3. Restart Firefox *from that shell*, and visit the website you were unable to authenticate to earlier. Information will be logged to **`/tmp/moz.log`**, and may give a clue to the problem. For example:

   ```
   -1208550944[90039d0]: entering nsNegotiateAuth::GetNextToken()
   -1208550944[90039d0]: gss_init_sec_context() failed: Miscellaneous
   failure
   No credentials cache found
   ```

   This indicates that you do not have Kerberos tickets, and need to run **`kinit`**.

If you are able to run **`kinit`** successfully from your machine but you are unable to authenticate, you might see something like this in the log file:

```
-1208994096[8d683d8]: entering nsAuthGSSAPI::GetNextToken()
-1208994096[8d683d8]: gss_init_sec_context() failed: Miscellaneous failure
Server not found in Kerberos database
```

This generally indicates a Kerberos configuration problem. Make sure that you have the correct entries in the [domain_realm] section of the **`/etc/krb5.conf`** file. For example:

```
.example.com = EXAMPLE.COM
example.com = EXAMPLE.COM
```

If nothing appears in the log it is possible that you are behind a proxy, and that proxy is stripping off the HTTP headers required for Negotiate authentication. As a workaround, you can try to connect to the server using HTTPS instead, which allows the request to pass through unmodified. Then proceed to debug using the log file, as described above.

48.4. Pluggable Authentication Modules (PAM)

Programs that grant users access to a system use *authentication* to verify each other's identity (that is, to establish that a user is who they say they are).

Historically, each program had its own way of authenticating users. In Red Hat Enterprise Linux, many programs are configured to use a centralized authentication mechanism called *Pluggable Authentication Modules* (PAM).

PAM uses a pluggable, modular architecture, which affords the system administrator a great deal of flexibility in setting authentication policies for the system.

In most situations, the default PAM configuration file for a PAM-aware application is sufficient. Sometimes, however, it is necessary to edit a PAM configuration file. Because misconfiguration of PAM can compromise system security, it is important to understand the structure of these files before making any modifications. Refer to Section 48.4.3, “PAM Configuration File Format” for more information.

48.4.1. Advantages of PAM

PAM offers the following advantages:

- a common authentication scheme that can be used with a wide variety of applications.
- significant flexibility and control over authentication for both system administrators and application developers.
- a single, fully-documented library which allows developers to write programs without having to create their own authentication schemes.

48.4.2. PAM Configuration Files

The **`/etc/pam.d/`** directory contains the PAM configuration files for each PAM-aware application. In earlier versions of PAM, the **`/etc/pam.conf`** file was used, but this file is now deprecated and is only used if the **`/etc/pam.d/`** directory does not exist.

48.4.2.1. PAM Service Files

Each PAM-aware application or *service* has a file in the **`/etc/pam.d/`** directory. Each file in this directory has the same name as the service to which it controls access.

The PAM-aware program is responsible for defining its service name and installing its own PAM configuration file in the **`/etc/pam.d/`** directory. For example, the **`login`** program defines its service name as **`login`** and installs the **`/etc/pam.d/login`** PAM configuration file.

48.4.3. PAM Configuration File Format

Each PAM configuration file contains a group of directives formatted as follows:

```
<module interface>  <control flag>   <module name>   <module arguments>
```

Each of these elements is explained in the following sections.

48.4.3.1. Module Interface

Four types of PAM module interface are currently available. Each of these corresponds to a different aspect of the authorization process:

- **`auth`** — This module interface authenticates use. For example, it requests and verifies the validity of a password. Modules with this interface can also set credentials, such as group memberships or Kerberos tickets.
- **`account`** — This module interface verifies that access is allowed. For example, it may check if a user account has expired or if a user is allowed to log in at a particular time of day.
- **`password`** — This module interface is used for changing user passwords.
- **`session`** — This module interface configures and manages user sessions. Modules with this interface can also perform additional tasks that are needed to allow access, like mounting a user's home directory and making the user's mailbox available.

Note

An individual module can provide any or all module interfaces. For instance, **`pam_unix.so`** provides all four module interfaces.

In a PAM configuration file, the module interface is the first field defined. For example, a typical line in a configuration may look like this:

```
auth required pam_unix.so
```

This instructs PAM to use the **`pam_unix.so`** module's **`auth`** interface.

48.4.3.1.1. Stacking Module Interfaces

Module interface directives can be *stacked*, or placed upon one another, so that multiple modules are used together for one purpose. If a module's control flag uses the "sufficient" or "requisite" value (refer to Section 48.4.3.2, “Control Flag” for more information on these flags), then the order in which the modules are listed is important to the authentication process.

Stacking makes it easy for an administrator to require specific conditions to exist before allowing the user to authenticate. For example, the **`reboot`** command normally uses several stacked modules, as seen in its PAM configuration file:

```
~]# cat /etc/pam.d/reboot
#%PAM-1.0
auth sufficient pam_rootok.so
auth required pam_console.so
#auth include system-auth
account required pam_permit.so
```

- The first line is a comment and is not processed.
- **`auth sufficient pam_rootok.so`** — This line uses the **`pam_rootok.so`** module to check whether the current user is root, by verifying that their UID is 0. If this test succeeds, no other modules are consulted and the command is executed. If this test fails, the next module is consulted.
- **`auth required pam_console.so`** — This line uses the **`pam_console.so`** module to attempt to authenticate the user. If this user is already logged in at the console, **`pam_console.so`** checks whether there is a file in the **`/etc/security/console.apps/`** directory with the same name as the service name (reboot). If such a file exists, authentication succeeds and control is passed to the next module.
- **`#auth include system-auth`** — This line is commented and is not processed.
- **`account required pam_permit.so`** — This line uses the **`pam_permit.so`** module to allow the root user or anyone logged in at the console to reboot the system.

48.4.3.2. Control Flag

All PAM modules generate a success or failure result when called. Control flags tell PAM what do with the result. Modules can be stacked in a particular order, and the control flags determine how important the success or failure of a particular module is to the overall goal of authenticating the user to the service.

There are four predefined control flags:

- **`required`** — The module result must be successful for authentication to continue. If the test fails at this point, the user is not notified until the results of all module tests that reference that interface are complete.
- **`requisite`** — The module result must be successful for authentication to continue. However, if a test fails at this point, the user is notified immediately with a message reflecting the first failed **`required`** *or* **`requisite`** module test.
- **`sufficient`** — The module result is ignored if it fails. However, if the result of a module flagged **`sufficient`** is successful *and* no previous modules flagged **`required`** have failed, then no other results are required and the user is authenticated to the service.
- **`optional`** — The module result is ignored. A module flagged as **`optional`** only becomes necessary for successful authentication when no other modules reference the interface.

Important

The order in which **`required`** modules are called is not critical. Only the **`sufficient`** and **`requisite`** control flags cause order to become important.

A newer control flag syntax that allows for more precise control is now available for PAM.

The **`pam.d`** man page, and the PAM documentation, located in the **`/usr/share/doc/pam-<`*`version-number`*`>/`** directory, where <*version-number*> is the version number for PAM on your system, describe this newer syntax in detail.

48.4.3.3. Module Name

The module name provides PAM with the name of the pluggable module containing the specified module interface. In older versions of Red Hat Enterprise Linux, the full path to the module was provided in the PAM configuration file. However, since the advent of *multilib* systems, which store 64-bit PAM modules in the **`/lib64/security/`** directory, the directory name is omitted because the application is linked to the appropriate version of **`libpam`**, which can locate the correct version of the module.

48.4.3.4. Module Arguments

PAM uses *arguments* to pass information to a pluggable module during authentication for some modules.

For example, the **`pam_userdb.so`** module uses information stored in a Berkeley DB file to authenticate the user. Berkeley DB is an open source database system embedded in many applications. The module takes a **db** argument so that Berkeley DB knows which database to use for the requested service.

The following is a typical **`pam_userdb.so`** line in a PAM configuration. The <*path-to-file*> is the full path to the Berkeley DB database file:

```
auth required pam_userdb.so db=<path-to-file>
```

Invalid arguments are *generally* ignored and do not otherwise affect the success or failure of the PAM module. Some modules, however, may fail on invalid arguments. Most modules report errors to the **`/var/log/secure`** file.

48.4.4. Sample PAM Configuration Files

The following is a sample PAM application configuration file:

```
#%PAM-1.0
auth required  pam_securetty.so
auth required  pam_unix.so nullok
auth required  pam_nologin.so
account required  pam_unix.so
password required  pam_cracklib.so retry=3
password required  pam_unix.so shadow nullok use_authtok
session required  pam_unix.so
```

- The first line is a comment, indicated by the hash mark (#) at the beginning of the line.
- Lines two through four stack three modules for login authentication.

  `auth required pam_securetty.so` — This module ensures that *if* the user is trying to log in as root, the tty on which the user is logging in is listed in the **`/etc/securetty`** file, *if* that file exists.

  If the tty is not listed in the file, any attempt to log in as root fails with a **`Login incorrect`** message.

  `auth required pam_unix.so nullok` — This module prompts the user for a password and then checks the password using the information stored in **`/etc/passwd`** and, if it exists, **`/etc/shadow`**.

  In the authentication phase, the **`pam_unix.so`** module automatically detects whether the user's passworc is in the **`passwd`** file or the **`shadow`** file. Refer to Section 37.6, “Shadow Passwords” for more information.

  - The argument **`nullok`** instructs the **`pam_unix.so`** module to allow a blank password.

- **`auth required pam_nologin.so`** — This is the final authentication step. It checks whether the **`/etc/nologin`** file exists. If it exists and the user is not root, authentication fails.

Note

In this example, all three **`auth`** modules are checked, even if the first **`auth`** module fails. This prevents the user from knowing at what stage their authentication failed. Such knowledge in the hands of an attacker could allow them to more easily deduce how to crack the system.

- **`account required pam_unix.so`** — This module performs any necessary account verification. For example, if shadow passwords have been enabled, the account interface of the **`pam_unix.so`** module checks to see if the account has expired or if the user has not changed the password within the allowed grace period.
- **`password required pam_cracklib.so retry=3`** — If a password has expired, the password component of the **`pam_cracklib.so`** module prompts for a new password. It then tests the newly created password to see whether it can easily be determined by a dictionary-based password cracking program.

  - The argument **`retry=3`** specifies that if the test fails the first time, the user has two more chances to create a strong password.

- **`password required pam_unix.so shadow nullok use_authtok`** — This line specifies that if the program changes the user's password, it should use the **`password`** interface of the **`pam_unix.so`** module to do so.

  - The argument **`shadow`** instructs the module to create shadow passwords when updating a user's password.

- The argument **`nullok`** instructs the module to allow the user to change their password *from* a blank password, otherwise a null password is treated as an account lock.

- The final argument on this line, **`use_authtok`**, provides a good example of the importance of order when stacking PAM modules. This argument instructs the module not to prompt the user for a new password. Instead, it accepts any password that was recorded by a previous password module. In this way, all new passwords must pass the **`pam_cracklib.so`** test for secure passwords before being accepted.

- **`session required pam_unix.so`** — The final line instructs the session interface of the **`pam_unix.so`** module to manage the session. This module logs the user name and the service type to **`/var/log/secure`** at the beginning and end of each session. This module can be supplemented by stacking it with other session modules for additional functionality.

48.4.5. Creating PAM Modules

You can create or add new PAM modules at any time for use by PAM-aware applications.

For example, a developer might create a one-time-password creation method and write a PAM module to support it. PAM-aware programs can immediately use the new module and password method without being recompiled or otherwise modified.

This allows developers and system administrators to mix-and-match, as well as test, authentication methods for different programs without recompiling them.

Documentation on writing modules is included in the **`/usr/share/doc/pam-<`*`version-number`*`>/`** directory, where <*version-number*> is the version number for PAM on your system.

48.4.6. PAM and Administrative Credential Caching

A number of graphical administrative tools in Red Hat Enterprise Linux provide users with elevated privileges for up to five minutes using the **`pam_timestamp.so`** module. It is important to understand how this mechanism works, because a user who walks away from a terminal while **`pam_timestamp.so`** is in effect leaves the machine open to manipulation by anyone with physical access to the console.

In the PAM timestamp scheme, the graphical administrative application prompts the user for the root password when it is launched. When the user has been authenticated, the **`pam_timestamp.so`** module creates a timestamp file. By default, this is created in the **`/var/run/sudo/`** directory. If the timestamp file already exists, graphical administrative programs do not prompt for a password. Instead, the **`pam_timestamp.so`** module freshens the timestamp file, reserving an extra five minutes of unchallenged administrative access for the user.

You can verify the actual state of the timestamp file by inspecting the **`/var/run/sudo/<user>`** file. For the desktop, the relevant file is **`unknown:root`**. If it is present and its timestamp is less than five minutes old, the credentials are valid.

The existence of the timestamp file is indicated by an authentication icon, which appears in the notification area of the panel.

Figure 48.7. The Authentication Icon

48.4.6.1. Removing the Timestamp File

Before abandoning a console where a PAM timestamp is active, it is recommended that the timestamp file be destroyed. To do this from a graphical environment, click the authentication icon on the panel. This causes a dialog box to appear. Click the **`Forget Authorization`** button to destroy the active timestamp file.

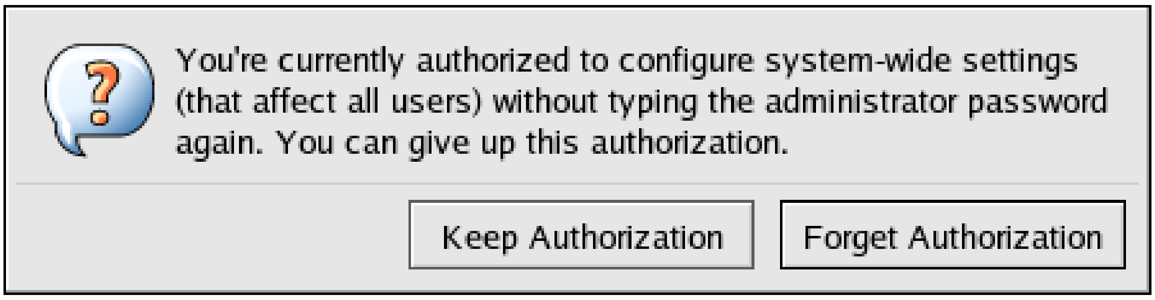

You're currently authorized to configure system-wide settings (that affect all users) without typing the administrator password again. You can give up this authorization.

Figure 48.8. Dismiss Authentication Dialog

You should be aware of the following with respect to the PAM timestamp file:

- If logged in to the system remotely using **`ssh`**, use the **`/sbin/pam_timestamp_check -k root`** command to destroy the timestamp file.
- You need to run the **`/sbin/pam_timestamp_check -k root`** command from the same terminal window from which you launched the privileged application.
- You must be logged in as the user who originally invoked the **`pam_timestamp.so`** module in order to use the **`/sbin/pam_timestamp_check -k`** command. Do not log in as root to use this command.
- If you want to kill the credentials on the desktop (without using the **`Forget Authorization`** action on the icon), use the following command:

  ```
  pam_timestamp_check -k root </dev/null >/dev/null 2>/dev/null
  ```

  Failure to use this command will only remove the credentials (if any) from the pty where you run the command.

Refer to the **`pam_timestamp_check`** man page for more information about destroying the timestamp file using **`pam_timestamp_check`**.

48.4.6.2. Common pam_timestamp Directives

The **`pam_timestamp.so`** module accepts several directives. The following are the two most commonly used options:

- **`timestamp_timeout`** — Specifies the period (in seconds) for which the timestamp file is valid. The default value is 300 (five minutes).
- **`timestampdir`** — Specifies the directory in which the timestamp file is stored. The default value is **`/var/run/sudo/`**.

Refer to Section 48.4.8.1, “Installed Documentation” for more information about controlling the **`pam_timestamp.so`** module.

48.4.7. PAM and Device Ownership

In Red Hat Enterprise Linux, the first user who logs in at the physical console of the machine can manipulate certain devices and perform certain tasks normally reserved for the root user. This is controlled by a PAM module called **`pam_console.so`**.

48.4.7.1. Device Ownership

When a user logs in to a Red Hat Enterprise Linux system, the **`pam_console.so`** module is called by **`login`** or the graphical login programs, **gdm**, **kdm**, and **xdm**. If this user is the first user to log in at the physical console — referred to as the *console user* — the module grants the user ownership of a variety of devices normally owned by root. The console user owns these devices until the last local session for that user ends. After this user has logged out, ownership of the devices reverts back to the root user.

The devices affected include, but are not limited to, sound cards, diskette drives, and CD-ROM drives.

This facility allows a local user to manipulate these devices without obtaining root access, thus simplifying common tasks for the console user.

You can modify the list of devices controlled by **`pam_console.so`** by editing the following files:

- **`/etc/security/console.perms`**
- **`/etc/security/console.perms.d/50-default.perms`**

You can change the permissions of different devices than those listed in the above files, or override the specified defaults. Rather than modify the **`50-default.perms`** file, you should create a new file (for example, ***`xx`*`-name.perms`**) and enter the required modifications. The name of the new default file must begin with a number higher than 50 (for example, **`51-default.perms`**). This will override the defaults in the **`50-default.perms`** file.

Warning

If the **gdm**, **kdm**, or **xdm** display manager configuration file has been altered to allow remote users to log in *and* the host is configured to run at runlevel 5, it is advisable to change the**`<console>`** and **`<xconsole>`** directives in the **`/etc/security/console.perms`** to the following values:

```
<console>=tty[0-9][0-9]* vc/[0-9][0-9]* :0\.[0-9] :0
<xconsole>=:0\.[0-9] :0
```

This prevents remote users from gaining access to devices and restricted applications on the machine.

If the **gdm**, **kdm**, or **xdm** display manager configuration file has been altered to allow remote users to log in *and* the host is configured to run at any multiple user runlevel other than 5, it is advisable to remove the **`<xconsole>`** directive entirely and change the **`<console>`** directive to the following value:

```
<console>=tty[0-9][0-9]* vc/[0-9][0-9]*
```

48.4.7.2. Application Access

The console user also has access to certain programs configured for use in the **`/etc/security/console.apps/`** directory.

This directory contains configuration files which enable the console user to run certain applications in **`/sbin`** and **`/usr/sbin`**.

These configuration files have the same name as the applications that they set up.

One notable group of applications that the console user has access to are three programs that shut down or reboot the system:

- **`/sbin/halt`**
- **`/sbin/reboot`**
- **`/sbin/poweroff`**

Because these are PAM-aware applications, they call the **`pam_console.so`** module as a requirement for use.

Refer to Section 48.4.8.1, "Installed Documentation" for more information.

48.4.8. Additional Resources

The following resources further explain methods to use and configure PAM. In addition to these resources, read the PAM configuration files on the system to better understand how they are structured.

48.4.8.1. Installed Documentation

- PAM-related man pages — Several man pages exist for the various applications and configuration files involved with PAM. The following is a list of some of the more important man pages.

  Configuration Files

  - **`pam`** — Good introductory information on PAM, including the structure and purpose of the PAM configuration files.

    Note that this man page discusses both **`/etc/pam.conf`** and individual configuration files in the **`/etc/pam.d/`** directory. By default, Red Hat Enterprise Linux uses the individual configuration files in the **`/etc/pam.d/`** directory, ignoring **`/etc/pam.conf`** even if it exists.

  - **`pam_console`** — Describes the purpose of the **`pam_console.so`** module. It also describes the appropriate syntax for an entry within a PAM configuration file.

  - **`console.apps`** — Describes the format and options available in the **`/etc/security/console.apps`** configuration file, which defines which applications are accessible by the console user assigned by PAM.

  - **`console.perms`** — Describes the format and options available in the **`/etc/security/console.perms`** configuration file, which specifies the console user permissions assigned by PAM.

  - **`pam_timestamp`** — Describes the **`pam_timestamp.so`** module.

- **`/usr/share/doc/pam-<version-number>`** — Contains a *System Administrators' Guide*, a *Module Writers' Manual*, and the *Application Developers' Manual*, as well as a copy of the PAM standard, DCE-RFC 86.0, where <*version-number*> is the version number of PAM.

- **`/usr/share/doc/pam-<version-number>/txts/README.pam_timestamp`** — Contains information about the **`pam_timestamp.so`** PAM module, where <*version-number*> is the version number of PAM.

48.4.8.2. Useful Websites

- http://www.kernel.org/pub/linux/libs/pam/ — The primary distribution website for the Linux-PAM project, containing information on various PAM modules, a FAQ, and additional PAM documentation.

Note

The documentation in the above website is for the last released upstream version of PAM and might not be 100% accurate for the PAM version included in Red Hat Enterprise Linux.

48.5. TCP Wrappers and xinetd

Controlling access to network services is one of the most important security tasks facing a server administrator. Red Hat Enterprise Linux provides several tools for this purpose. For example, an **`iptables`**-based firewall filters out unwelcome network packets within the kernel's network stack. For network services that utilize it, *TCP Wrappers* add an additional layer of protection by defining which hosts are or are not allowed to connect to "*wrapped*" network services. One such wrapped network service is the **`xinetd`** *super server*. This service is called a super server because it controls connections to a subset of network services and further refines access control.

Figure 48.9, "Access Control to Network Services" is a basic illustration of how these tools work together to protect network services.

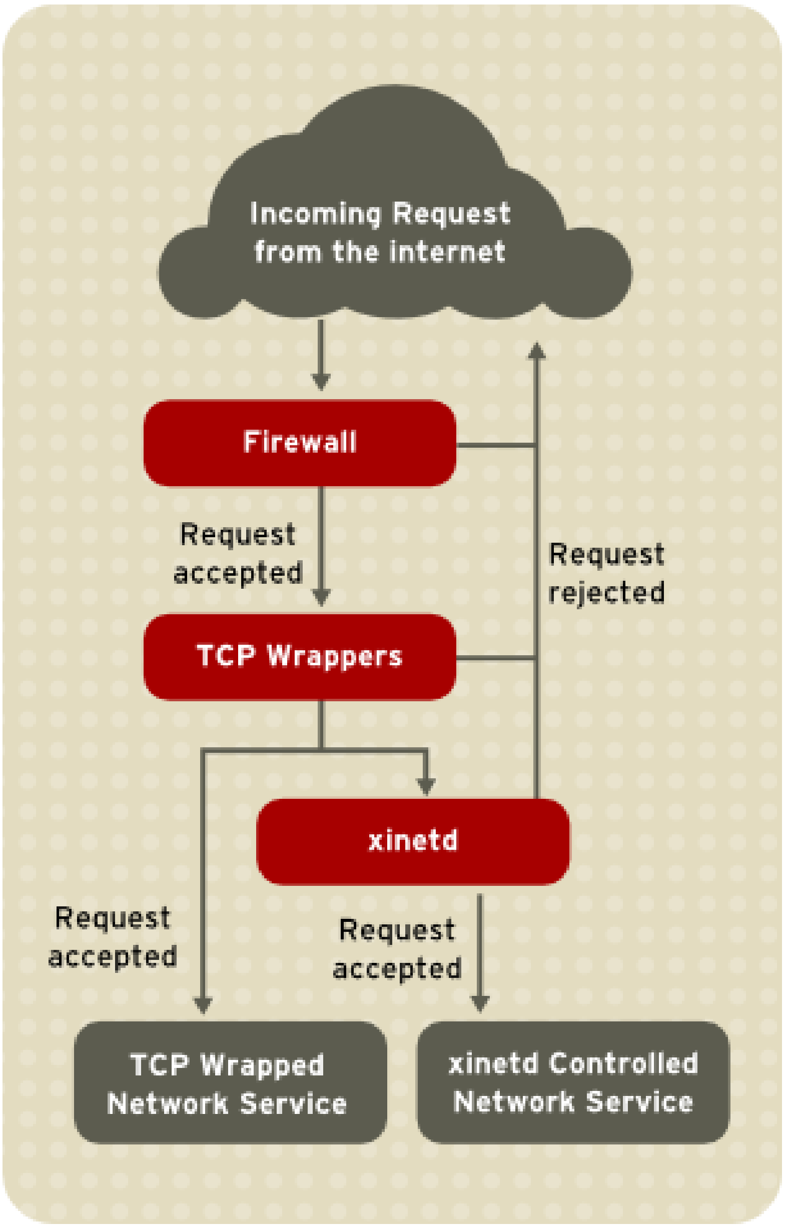

Incoming Request from the internet
Firewall
Request accepted
Request rejected
TCP Wrappers
xinetd
Request accepted
Request accepted
TCP Wrapped Network Service
xinetd Controlled Network Service

Figure 48.9. Access Control to Network Services

This chapter focuses on the role of TCP Wrappers and **`xinetd`** in controlling access to network services and reviews how these tools can be used to enhance both logging and utilization management. Refer to Section 48.9, "IPTables" for information about using firewalls with **`iptables`**.

48.5.1. TCP Wrappers

The TCP Wrappers package (**`tcp_wrappers`**) is installed by default and provides host-based access control to network services. The most important component within the package is the **`/usr/lib/libwrap.a`** library. In general terms, a TCP-wrapped service is one that has been compiled against the **`libwrap.a`** library.

When a connection attempt is made to a TCP-wrapped service, the service first references the host's access files (**`/etc/hosts.allow`** and **`/etc/hosts.deny`**) to determine whether or not the client is allowed to connect. In most cases, it then uses the syslog daemon (**`syslogd`**) to write the name of the requesting client and the requested service to **`/var/log/secure`** or **`/var/log/messages`**.

If a client is allowed to connect, TCP Wrappers release control of the connection to the requested service and take no further part in the communication between the client and the server.

In addition to access control and logging, TCP Wrappers can execute commands to interact with the client before denying or releasing control of the connection to the requested network service.

Because TCP Wrappers are a valuable addition to any server administrator's arsenal of security tools, most network services within Red Hat Enterprise Linux are linked to the **`libwrap.a`** library. Some such applications include **`/usr/sbin/sshd`**, **`/usr/sbin/sendmail`**, and **`/usr/sbin/xinetd`**.

Note

To determine if a network service binary is linked to **`libwrap.a`**, type the following command as the root user:

```
ldd <binary-name> | grep libwrap
```

Replace <*binary-name*> with the name of the network service binary.

If the command returns straight to the prompt with no output, then the network service is *not* linked to **`libwrap.a`**.

The following example indicates that **`/usr/sbin/sshd`** is linked to **`libwrap.a`**:

```
~]# ldd /usr/sbin/sshd | grep libwrap
        libwrap.so.0 => /usr/lib/libwrap.so.0 (0x00655000)
~]#
```

48.5.1.1. Advantages of TCP Wrappers

TCP Wrappers provide the following advantages over other network service control techniques:

- *Transparency to both the client and the wrapped network service*— Both the connecting client and the wrapped network service are unaware that TCP Wrappers are in use. Legitimate users are logged and connected to the requested service while connections from banned clients fail.
- *Centralized management of multiple protocols* — TCP Wrappers operate separately from the network services they protect, allowing many server applications to share a common set of access control configuration files, making for simpler management.

48.5.2. TCP Wrappers Configuration Files

To determine if a client is allowed to connect to a service, TCP Wrappers reference the following two files, which are commonly referred to as *hosts access* files:

- **`/etc/hosts.allow`**
- **`/etc/hosts.deny`**

When a TCP-wrapped service receives a client request, it performs the following steps:

1. *It references **`/etc/hosts.allow`**.* — The TCP-wrapped service sequentially parses the **`/etc/hosts.allow`** file and applies the first rule specified for that service. If it finds a matching rule, it allows the connection. If not, it moves on to the next step.

2. *It references **/etc/hosts.deny**.* — The TCP-wrapped service sequentially parses the **/etc/hosts.deny** file. If it finds a matching rule, it denies the connection. If not, it grants access to the service.

The following are important points to consider when using TCP Wrappers to protect network services:

- Because access rules in **hosts.allow** are applied first, they take precedence over rules specified in **hosts.deny**. Therefore, if access to a service is allowed in **hosts.allow**, a rule denying access to that same service in **hosts.deny** is ignored.
- The rules in each file are read from the top down and the first matching rule for a given service is the only one applied. The order of the rules is extremely important.
- If no rules for the service are found in either file, or if neither file exists, access to the service is granted.
- TCP-wrapped services do not cache the rules from the hosts access files, so any changes to **hosts.allow** or **hosts.deny** take effect immediately, without restarting network services.

Warning

If the last line of a hosts access file is not a newline character (created by pressing the **Enter** key), the last rule in the file fails and an error is logged to either **/var/log/messages** or **/var/log/secure**. This is also the case for a rule that spans multiple lines without using the backslash character. The following example illustrates the relevant portion of a log message for a rule failure due to either of these circumstances:

```
warning: /etc/hosts.allow, line 20: missing newline or line too long
```

48.5.2.1. Formatting Access Rules

The format for both **/etc/hosts.allow** and **/etc/hosts.deny** is identical. Each rule must be on its own line. Blank lines or lines that start with a hash (#) are ignored.

Each rule uses the following basic format to control access to network services:

```
<daemon list>: <client list> [: <option>: <option>: ...]
```

- *<daemon list>* — A comma-separated list of process names (*not* service names) or the **ALL** wildcard. The daemon list also accepts operators (refer to Section 48.5.2.1.4, “Operators”) to allow greater flexibility.
- *<client list>* — A comma-separated list of hostnames, host IP addresses, special patterns, or wildcards which identify the hosts affected by the rule. The client list also accepts operators listed in Section 48.5.2.1.4, “Operators” to allow greater flexibility.
- *<option>* — An optional action or colon-separated list of actions performed when the rule is triggered. Option fields support expansions, launch shell commands, allow or deny access, and alter logging behavior.

Note

More information on the specialist terms above can be found elsewhere in this Guide:

- Section 48.5.2.1.1, “Wildcards”
- Section 48.5.2.1.2, “Patterns”
- Section 48.5.2.2.4, “Expansions”
- Section 48.5.2.2, “Option Fields”

The following is a basic sample hosts access rule:

```
vsftpd : .example.com
```

This rule instructs TCP Wrappers to watch for connections to the FTP daemon (**`vsftpd`**) from any host in the **`example.com`** domain. If this rule appears in **`hosts.allow`**, the connection is accepted. If this rule appears in **`hosts.deny`**, the connection is rejected.

The next sample hosts access rule is more complex and uses two option fields:

```
sshd : .example.com  \ : spawn /bin/echo `/bin/date` access
denied>>/var/log/sshd.log \ : deny
```

Note that each option field is preceded by the backslash (\). Use of the backslash prevents failure of the rule due to length.

This sample rule states that if a connection to the SSH daemon (**`sshd`**) is attempted from a host in the **`example.com`** domain, execute the **`echo`** command to append the attempt to a special log file, and deny the connection. Because the optional **`deny`** directive is used, this line denies access even if it appears in the **`hosts.allow`** file. Refer to Section 48.5.2.2, “Option Fields” for a more detailed look at available options.

48.5.2.1.1. Wildcards

Wildcards allow TCP Wrappers to more easily match groups of daemons or hosts. They are used most frequently in the client list field of access rules.

The following wildcards are available:

- **`ALL`** — Matches everything. It can be used for both the daemon list and the client list.
- **`LOCAL`** — Matches any host that does not contain a period (.), such as localhost.
- **`KNOWN`** — Matches any host where the hostname and host address are known or where the user is known.
- **`UNKNOWN`** — Matches any host where the hostname or host address are unknown or where the user is unknown.
- **`PARANOID`** — Matches any host where the hostname does not match the host address.

Warning

The **`KNOWN`**, **`UNKNOWN`**, and **`PARANOID`** wildcards should be used with care, because they rely on functioning DNS server for correct operation. Any disruption to name resolution may prevent legitimate users from gaining access to a service.

48.5.2.1.2. Patterns

Patterns can be used in the client field of access rules to more precisely specify groups of client hosts.

The following is a list of common patterns for entries in the client field:

- *Hostname beginning with a period (.)* — Placing a period at the beginning of a hostname matches all hosts sharing the listed components of the name. The following example applies to any host within the **`example.com`** domain:

  ```
  ALL : .example.com
  ```

- *IP address ending with a period (.)* — Placing a period at the end of an IP address matches all hosts sharing the initial numeric groups of an IP address. The following example applies to any host within the **`192.168.x.x`** network:

  ```
  ALL : 192.168.
  ```

- *IP address/netmask pair* — Netmask expressions can also be used as a pattern to control access to a particular group of IP addresses. The following example applies to any host with an address range of **`192.168.0.0`** through **`192.168.1.255`**:

  ```
  ALL : 192.168.0.0/255.255.254.0
  ```

  Important

  When working in the IPv4 address space, the address/prefix length (*prefixlen*) pair declarations (CIDR notation) are not supported. Only IPv6 rules can use this format.

- *[IPv6 address]/prefixlen pair* — [net]/prefixlen pairs can also be used as a pattern to control access to a particular group of IPv6 addresses. The following example would apply to any host with an address range of **`3ffe:505:2:1::`** through **`3ffe:505:2:1:ffff:ffff:ffff:ffff`**:

  ```
  ALL : [3ffe:505:2:1::]/64
  ```

- *The asterisk (*)* — Asterisks can be used to match entire groups of hostnames or IP addresses, as long as they are not mixed in a client list containing other types of patterns. The following example would apply to any host within the **`example.com`** domain:

  ```
  ALL : *.example.com
  ```

- *The slash (/)* — If a client list begins with a slash, it is treated as a file name. This is useful if rules specifying large numbers of hosts are necessary. The following example refers TCP Wrappers to the **`/etc/telnet.hosts`** file for all Telnet connections:

```
in.telnetd : /etc/telnet.hosts
```

Other, lesser used, patterns are also accepted by TCP Wrappers. Refer to the **hosts_access** man 5 page for more information.

Warning

Be very careful when using hostnames and domain names. Attackers can use a variety of tricks to circumvent accurate name resolution. In addition, disruption to DNS service prevents even authorized users from using network services. It is, therefore, best to use IP addresses whenever possible.

48.5.2.1.3. Portmap and TCP Wrappers

Portmap's implementation of TCP Wrappers does not support host look-ups, which means **portmap** can not use hostnames to identify hosts. Consequently, access control rules for portmap in **hosts.allow** or **hosts.deny** must use IP addresses, or the keyword **ALL**, for specifying hosts.

Changes to **portmap** access control rules may not take effect immediately. You may need to restart the **portmap** service.

Widely used services, such as NIS and NFS, depend on **portmap** to operate, so be aware of these limitations.

48.5.2.1.4. Operators

At present, access control rules accept one operator, **EXCEPT**. It can be used in both the daemon list and the client list of a rule.

The **EXCEPT** operator allows specific exceptions to broader matches within the same rule.

In the following example from a **hosts.allow** file, all **example.com** hosts are allowed to connect to all services except **cracker.example.com**:

```
ALL: .example.com EXCEPT cracker.example.com
```

In another example from a **hosts.allow** file, clients from the **192.168.0.*x*** network can use all services except for FTP:

```
ALL EXCEPT vsftpd: 192.168.0.
```

Note

Organizationally, it is often easier to avoid using **EXCEPT** operators. This allows other administrators to quickly scan the appropriate files to see what hosts are allowed or denied access to services, without having to sort through **EXCEPT** operators.

48.5.2.2. Option Fields

In addition to basic rules that allow and deny access, the Red Hat Enterprise Linux implementation of TCP Wrappers supports extensions to the access control language through *option fields*. By using option fields in hosts access rules, administrators can accomplish a variety of tasks such as altering log behavior, consolidating access control, and launching shell commands.

48.5.2.2.1. Logging

Option fields let administrators easily change the log facility and priority level for a rule by using the **severity** directive.

In the following example, connections to the SSH daemon from any host in the **example.com** domain are logged to the default **authpriv** **syslog** facility (because no facility value is specified) with a priority of **emerg**:

```
sshd : .example.com : severity emerg
```

It is also possible to specify a facility using the **severity** option. The following example logs any SSH connection attempts by hosts from the **example.com** domain to the **local0** facility with a priority of **alert**:

```
sshd : .example.com : severity local0.alert
```

Note

In practice, this example does not work until the syslog daemon (**syslogd**) is configured to log to the **local0** facility. Refer to the **syslog.conf** man page for information about configuring custom log facilities.

48.5.2.2.2. Access Control

Option fields also allow administrators to explicitly allow or deny hosts in a single rule by adding the **allow** or **deny** directive as the final option.

For example, the following two rules allow SSH connections from **client-1.example.com**, but deny connections from **client-2.example.com**:

```
sshd : client-1.example.com : allow
sshd : client-2.example.com : deny
```

By allowing access control on a per-rule basis, the option field allows administrators to consolidate all access rules into a single file: either **hosts.allow** or **hosts.deny**. Some administrators consider this an easier way of organizing access rules.

48.5.2.2.3. Shell Commands

Option fields allow access rules to launch shell commands through the following two directives:

- **spawn** — Launches a shell command as a child process. This directive can perform tasks like using **/usr/sbin/safe_finger** to get more information about the requesting client or create special log files using the **echo** command.

  In the following example, clients attempting to access Telnet services from the **example.com** domain are quietly logged to a special file:

```
in.telnetd : .example.com \
 : spawn /bin/echo `/bin/date` from %h>>/var/log/telnet.log \
 : allow
```

- **`twist`** — Replaces the requested service with the specified command. This directive is often used to set up traps for intruders (also called "honey pots"). It can also be used to send messages to connecting clients. The **`twist`** directive must occur at the end of the rule line.

  In the following example, clients attempting to access FTP services from the **`example.com`** domain are sent a message using the **echo** command:

  ```
  vsftpd : .example.com \
   : twist /bin/echo "421 This domain has been black-listed. Access denied!"
  ```

For more information about shell command options, refer to the **`hosts_options`** man page.

48.5.2.2.4. Expansions

Expansions, when used in conjunction with the **`spawn`** and **`twist`** directives, provide information about the client, server, and processes involved.

The following is a list of supported expansions:

- **%a** — Returns the client's IP address.
- **%A** — Returns the server's IP address.
- **%c** — Returns a variety of client information, such as the username and hostname, or the username and IP address.
- **%d** — Returns the daemon process name.
- **%h** — Returns the client's hostname (or IP address, if the hostname is unavailable).
- **%H** — Returns the server's hostname (or IP address, if the hostname is unavailable).
- **%n** — Returns the client's hostname. If unavailable, **unknown** is printed. If the client's hostname and host address do not match, **`paranoid`** is printed.
- **%N** — Returns the server's hostname. If unavailable, **unknown** is printed. If the server's hostname and host address do not match, **`paranoid`** is printed.
- **%p** — Returns the daemon's process ID.
- **%s** —Returns various types of server information, such as the daemon process and the host or IP address of the server.
- **%u** — Returns the client's username. If unavailable, **unknown** is printed.

The following sample rule uses an expansion in conjunction with the **spawn** command to identify the client host in a customized log file.

When connections to the SSH daemon (**sshd**) are attempted from a host in the **`example.com`** domain, execute the **echo** command to log the attempt, including the client hostname (by using the**%h** expansion), to a special file:

```
sshd : .example.com  \
 : spawn /bin/echo `/bin/date` access denied to %h>>/var/log/sshd.log \
 : deny
```

Similarly, expansions can be used to personalize messages back to the client. In the following example, clients attempting to access FTP services from the **`example.com`** domain are informed that they have been banned from the server:

```
vsftpd : .example.com \
: twist /bin/echo "421 %h has been banned from this server!"
```

For a full explanation of available expansions, as well as additional access control options, refer to section 5 of the man pages for **`hosts_access`** (**`man 5 hosts_access`**) and the man page for **`hosts_options`**.

Refer to Section 48.5.5, “Additional Resources” for more information about TCP Wrappers.

48.5.3. xinetd

The **`xinetd`** daemon is a TCP-wrapped *super service* which controls access to a subset of popular network services, including FTP, IMAP, and Telnet. It also provides service-specific configuration options for access control, enhanced logging, binding, redirection, and resource utilization control.

When a client attempts to connect to a network service controlled by **`xinetd`**, the super service receives the request and checks for any TCP Wrappers access control rules.

If access is allowed, **`xinetd`** verifies that the connection is allowed under its own access rules for that service. It also checks that the service can have more resources allotted to it and that it is not in breach of any defined rules.

If all these conditions are met (that is, access is allowed to the service; the service has not reached its resource limit; and the service is not in breach of any defined rule), **`xinetd`** then starts an instance of the requested service and passes control of the connection to it. After the connection has been established, **`xinetd`** takes no further part in the communication between the client and the server.

48.5.4. xinetd Configuration Files

The configuration files for **`xinetd`** are as follows:

- **`/etc/xinetd.conf`** — The global **`xinetd`** configuration file.
- **`/etc/xinetd.d/`** — The directory containing all service-specific files.

48.5.4.1. The /etc/xinetd.conf File

The **`/etc/xinetd.conf`** file contains general configuration settings which affect every service under **`xinetd`**'s control. It is read when the **`xinetd`** service is first started, so for configuration changes to take effect, you need to restart the **`xinetd`** service. The following is a sample **`/etc/xinetd.conf`** file:

```
defaults
{
         instances                 = 60
  log_type                  = SYSLOG authpriv
  log_on_success            = HOST PID
  log_on_failure          = HOST
  cps                     = 25 30
}
includedir /etc/xinetd.d
```

These lines control the following aspects of **`xinetd`**:

- **`instances`** — Specifies the maximum number of simultaneous requests that **`xinetd`** can process.
- **`log_type`** — Configures **`xinetd`** to use the **`authpriv`** log facility, which writes log entries to the **`/var/log/secure`** file. Adding a directive such as **`FILE /var/log/xinetdlog`** would create a custom log file called **`xinetdlog`** in the **`/var/log/`** directory.
- **`log_on_success`** — Configures **`xinetd`** to log successful connection attempts. By default, the remote host's IP address and the process ID of the server processing the request are recorded.
- **`log_on_failure`** — Configures **`xinetd`** to log failed connection attempts or if the connection was denied.
- **`cps`** — Configures **`xinetd`** to allow no more than 25 connections per second to any given service. If this limit is exceeded, the service is retired for 30 seconds.
- **`includedir /etc/xinetd.d/`** — Includes options declared in the service-specific configuration files located in the **`/etc/xinetd.d/`** directory. Refer to Section 48.5.4.2, "The /etc/xinetd.d/ Directory" for more information.

Note

Often, both the **`log_on_success`** and **`log_on_failure`** settings in **`/etc/xinetd.conf`** are further modified in the service-specific configuration files. More information may therefore appear in a given service's log file than the **`/etc/xinetd.conf`** file may indicate. Refer to Section 48.5.4.3.1, "Logging Options" for further information.

48.5.4.2. The /etc/xinetd.d/ Directory

The **`/etc/xinetd.d/`** directory contains the configuration files for each service managed by **`xinetd`** and the names of the files correlate to the service. As with **`xinetd.conf`**, this directory is read only when the **`xinetd`** service is started. For any changes to take effect, the administrator must restart the **`xinetd`** service.

The format of files in the **`/etc/xinetd.d/`** directory use the same conventions as **`/etc/xinetd.conf`**. The primary reason the configuration for each service is stored in a separate file is to make customization easier and less likely to affect other services.

To gain an understanding of how these files are structured, consider the **`/etc/xinetd.d/krb5-telnet`** file:

```
service telnet
{
         flags            = REUSE
  socket_type     = stream
  wait            = no
  user            = root
  server          = /usr/kerberos/sbin/telnetd
  log_on_failure  += USERID
  disable         = yes
}
```

These lines control various aspects of the **telnet** service:

- **service** — Specifies the service name, usually one of those listed in the **/etc/services** file.
- **flags** — Sets any of a number of attributes for the connection. **REUSE** instructs **xinetd** to reuse the socket for a Telnet connection.

> **Note**
>
> The **REUSE** flag is deprecated. All services now implicitly use the **REUSE** flag.

- **socket_type** — Sets the network socket type to **stream**.
- **wait** — Specifies whether the service is single-threaded (**yes**) or multi-threaded (**no**).
- **user** — Specifies which user ID the process runs under.
- **server** — Specifies which binary executable to launch.
- **log_on_failure** — Specifies logging parameters for **log_on_failure** in addition to those already defined in **xinetd.conf**.
- **disable** — Specifies whether the service is disabled (**yes**) or enabled (**no**).

Refer to the **xinetd.conf** man page for more information about these options and their usage.

48.5.4.3. Altering xinetd Configuration Files

A range of directives is available for services protected by **xinetd**. This section highlights some of the more commonly used options.

48.5.4.3.1. Logging Options

The following logging options are available for both **/etc/xinetd.conf** and the service-specific configuration files within the **/etc/xinetd.d/** directory.

The following is a list of some of the more commonly used logging options:

- **ATTEMPT** — Logs the fact that a failed attempt was made (**log_on_failure**).
- **DURATION** — Logs the length of time the service is used by a remote system (**log_on_success**).
- **EXIT** — Logs the exit status or termination signal of the service (**log_on_success**).
- **HOST** — Logs the remote host's IP address (**log_on_failure** and **log_on_success**).
- **PID** — Logs the process ID of the server receiving the request (**log_on_success**).
- **USERID** — Logs the remote user using the method defined in RFC 1413 for all multi-threaded stream services (**log_on_failure** and**log_on_success**).

For a complete list of logging options, refer to the **xinetd.conf** man page.

48.5.4.3.2. Access Control Options

Users of **xinetd** services can choose to use the TCP Wrappers hosts access rules, provide access control via the **xinetd** configuration files, or a mixture of both. Refer to Section 48.5.2, "TCP Wrappers Configuration Files" for more information about TCP Wrappers hosts access control files.

This section discusses using **xinetd** to control access to services.

Note

Unlike TCP Wrappers, changes to access control only take effect if the **xinetd** administrator restarts the **xinetd** service.

Also, unlike TCP Wrappers, access control through **xinetd** only affects services controlled by **xinetd**.

The **xinetd** hosts access control differs from the method used by TCP Wrappers. While TCP Wrappers places all of the access configuration within two files, **/etc/hosts.allow** and **/etc/hosts.deny**, **xinetd**'s access control is found in each service's configuration file in the **/etc/xinetd.d/** directory.

The following hosts access options are supported by **xinetd**:

- **only_from** — Allows only the specified hosts to use the service.
- **no_access** — Blocks listed hosts from using the service.
- **access_times** — Specifies the time range when a particular service may be used. The time range must be stated in 24-hour format notation, HH:MM-HH:MM.

The **only_from** and **no_access** options can use a list of IP addresses or host names, or can specify an entire network. Like TCP Wrappers, combining **xinetd** access control with the enhanced logging configuration can increase security by blocking requests from banned hosts while verbosely recording each connection attempt.

For example, the following **/etc/xinetd.d/telnet** file can be used to block Telnet access from a particular network group and restrict the overall time range that even allowed users can log in:

```
service telnet
{
         disable         = no
  flags            = REUSE
  socket_type      = stream
  wait             = no
  user             = root
  server           = /usr/kerberos/sbin/telnetd
  log_on_failure  += USERID
  no_access        = 172.16.45.0/24
  log_on_success  += PID HOST EXIT
  access_times     = 09:45-16:15
}
```

In this example, when a client system from the **10.0.1.0/24** network, such as **10.0.1.2**, tries to access the Telnet service, it receives the following message:

```
Connection closed by foreign host.
```

In addition, their login attempts are logged in **/var/log/messages** as follows:

```
Sep  7 14:58:33 localhost xinetd[5285]: FAIL: telnet address
from=172.16.45.107
Sep  7 14:58:33 localhost xinetd[5283]: START: telnet pid=5285
from=172.16.45.107
Sep  7 14:58:33 localhost xinetd[5283]: EXIT: telnet status=0 pid=5285
duration=0(sec)
```

When using TCP Wrappers in conjunction with **xinetd** access controls, it is important to understand the relationship between the two access control mechanisms.

The following is the sequence of events followed by **xinetd** when a client requests a connection:

1. The **xinetd** daemon accesses the TCP Wrappers hosts access rules using a **libwrap.a** library call. If a deny rule matches the client, the connection is dropped. If an allow rule matches the client, the connection is passed to **xinetd**.

2. The **xinetd** daemon checks its own access control rules both for the **xinetd** service and the requested service. If a deny rule matches the client, the connection is dropped. Otherwise, **xinetd** starts an instance of the requested service and passes control of the connection to that service.

Important

Care should be taken when using TCP Wrappers access controls in conjunction with **xinetd** access controls. Misconfiguration can cause undesirable effects.

48.5.4.3.3. Binding and Redirection Options

The service configuration files for **xinetd** support binding the service to an IP address and redirecting incoming requests for that service to another IP address, hostname, or port.

Binding is controlled with the **bind** option in the service-specific configuration files and links the service to one IP address on the system. When this is configured, the **bind** option only allows requests to the correct IP address to access the service. You can use this method to bind different services to different network interfaces based on requirements.

This is particularly useful for systems with multiple network adapters or with multiple IP addresses. On such a system, insecure services (for example, Telnet), can be configured to listen only on the interface connected to a private network and not to the interface connected to the Internet.

The **redirect** option accepts an IP address or hostname followed by a port number. It configures the service to redirect any requests for this service to the specified host and port number. This feature can be used to point to another port number on the same system, redirect the request to a different IP address on the same machine, shift the request to a totally different system and port number, or any combination of these options. A user connecting to a certain service on a system may therefore be rerouted to another system without disruption.

The **xinetd** daemon is able to accomplish this redirection by spawning a process that stays alive for the duration of the connection between the requesting client machine and the host actually providing the service, transferring data between the two systems.

The advantages of the **bind** and **redirect** options are most clearly evident when they are used together. By binding a service to a particular IP address on a system and then redirecting requests for this service to a second machine that only the first machine can see, an internal system can be used to provide services for a totally different network. Alternatively, these options can be used to limit the exposure of a particular service on a multi-homed machine to a known IP address, as well as redirect any requests for that service to another machine especially configured for that purpose.

For example, consider a system that is used as a firewall with this setting for its Telnet service:

```
service telnet
{
         socket_type  = stream
  wait   = no
  server   = /usr/kerberos/sbin/telnetd
  log_on_success  += DURATION USERID
  log_on_failure  += USERID
  bind                    = 123.123.123.123
  redirect                = 10.0.1.13 23
}
```

The **bind** and **redirect** options in this file ensure that the Telnet service on the machine is bound to the external IP address (**123.123.123.123**), the one facing the Internet. In addition, any requests for Telnet service sent to **123.123.123.123** are redirected via a second network adapter to an internal IP address (**10.0.1.13**) that only the firewall and internal systems can access. The firewall then sends the communication between the two systems, and the connecting system thinks it is connected to **123.123.123.123** when it is actually connected to a different machine.

This feature is particularly useful for users with broadband connections and only one fixed IP address. When using Network Address Translation (NAT), the systems behind the gateway machine, which are using internal-only IP addresses, are not available from outside the gateway system. However, when certain services controlled by **xinetd** are configured with the **bind** and **redirect** options, the gateway machine can act as a proxy between outside systems and a particular internal machine configured to provide the service. In addition, the various **xinetd** access control and logging options are also available for additional protection.

48.5.4.3.4. Resource Management Options

The **xinetd** daemon can add a basic level of protection from Denial of Service (DoS) attacks. The following is a list of directives which can aid in limiting the effectiveness of such attacks:

- **per_source** — Defines the maximum number of instances for a service per source IP address. It accepts only integers as an argument and can be used in both **xinetd.conf** and in the service-specific configuration files in the **xinetd.d/** directory.

- **cps** — Defines the maximum number of connections per second. This directive takes two integer arguments separated by white space. The first argument is the maximum number of connections allowed to the service per second. The second argument is the number of seconds that **xinetd** must wait before re-enabling the service. It accepts only integers as arguments and can be used in either the **xinetd.conf** file or the service-specific configuration files in the **xinetd.d/** directory.

- **max_load** — Defines the CPU usage or load average threshold for a service. It accepts a floating point number argument.

The load average is a rough measure of how many processes are active at a given time. See the **`uptime`**, **`who`**, and **`procinfo`** commands for more information about load average.

There are more resource management options available for **`xinetd`**. Refer to the **`xinetd.conf`** man page for more information.

48.5.5. Additional Resources

More information about TCP Wrappers and **`xinetd`** is available from system documentation and on the Internet.

48.5.5.1. Installed Documentation

The documentation on your system is a good place to start looking for additional configuration options for TCP Wrappers, **`xinetd`**, and access control.

- **`/usr/share/doc/tcp_wrappers-<`*`version`*`>/`** — This directory contains a **`README`** file that discusses how TCP Wrappers work and the various hostname and host address spoofing risks that exist.
- **`/usr/share/doc/xinetd-<`*`version`*`>/`** — This directory contains a **`README`** file that discusses aspects of access control and a **`sample.conf`** file with various ideas for modifying service-specific configuration files in the **`/etc/xinetd.d/`** directory.
- TCP Wrappers and **`xinetd`**-related man pages — A number of man pages exist for the various applications and configuration files involved with TCP Wrappers and **`xinetd`**. The following are some of the more important man pages:

  Server Applications

  - **`man xinetd`** — The man page for **`xinetd`**.

  Configuration Files

  - **`man 5 hosts_access`** — The man page for the TCP Wrappers hosts access control files.
  - **`man hosts_options`** — The man page for the TCP Wrappers options fields.
  - **`man xinetd.conf`** — The man page listing **`xinetd`** configuration options.

48.5.5.2. Useful Websites

- http://www.xinetd.org/ — The home of **`xinetd`**, containing sample configuration files, a full listing of features, and an informative FAQ.
- http://www.macsecurity.org/resources/xinetd/tutorial.shtml — A thorough tutorial that discusses many different ways to optimize default **`xinetd`** configuration files to meet specific security goals.

48.5.5.3. Related Books

- *Hacking Linux Exposed* by Brian Hatch, James Lee, and George Kurtz; Osbourne/McGraw-Hill — An excellent security resource with information about TCP Wrappers and **`xinetd`**.

48.6. Kerberos

System security and integrity within a network can be unwieldy. It can occupy the time of several administrators just to keep track of what services are being run on a network and the manner in which these services are used.

Further, authenticating users to network services can prove dangerous when the method used by the protocol is inherently insecure, as evidenced by the transfer of unencrypted passwords over a network using the traditional FTP and Telnet protocols.

Kerberos is a way to eliminate the need for protocols that allow unsafe methods of authentication, thereby enhancing overall network security.

48.6.1. What is Kerberos?

Kerberos is a network authentication protocol created by MIT, and uses symmetric-key cryptography [17] to authenticate users to network services, which means passwords are never actually sent over the network.

Consequently, when users authenticate to network services using Kerberos, unauthorized users attempting to gather passwords by monitoring network traffic are effectively thwarted.

48.6.1.1. Advantages of Kerberos

Most conventional network services use password-based authentication schemes. Such schemes require a user to authenticate to a given network server by supplying their username and password. Unfortunately, the transmission of authentication information for many services is unencrypted. For such a scheme to be secure, the network has to be inaccessible to outsiders, and all computers and users on the network must be trusted and trustworthy.

Even if this is the case, a network that is connected to the Internet can no longer be assumed to be secure. Any attacker who gains access to the network can use a simple packet analyzer, also known as a packet sniffer, to intercept usernames and passwords, compromising user accounts and the integrity of the entire security infrastructure.

The primary design goal of Kerberos is to eliminate the transmission of unencrypted passwords across the network. If used properly, Kerberos effectively eliminates the threat that packet sniffers would otherwise pose on a network.

48.6.1.2. Disadvantages of Kerberos

Although Kerberos removes a common and severe security threat, it may be difficult to implement for a variety of reasons:

- Migrating user passwords from a standard UNIX password database, such as **`/etc/passwd`** or **`/etc/shadow`**, to a Kerberos password database can be tedious, as there is no automated mechanism to perform this task. Refer to Question 2.23 in the online Kerberos FAQ:

  http://www.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html

- Kerberos has only partial compatibility with the Pluggable Authentication Modules (PAM) system used by most Red Hat Enterprise Linux servers. Refer to Section 48.6.4, “Kerberos and PAM” for more information about this issue.

- Kerberos assumes that each user is trusted but is using an untrusted host on an untrusted network. Its primary goal is to prevent unencrypted passwords from being transmitted across that network. However, if anyone other than the proper user has access to the one host that issues tickets used for authentication — called the *key distribution center* (*KDC*) — the entire Kerberos authentication system is at risk.

- For an application to use Kerberos, its source must be modified to make the appropriate calls into the

Kerberos libraries. Applications modified in this way are considered to be *Kerberos-aware*, or *kerberized*. For some applications, this can be quite problematic due to the size of the application or its design. For other incompatible applications, changes must be made to the way in which the server and client communicate. Again, this may require extensive programming. Closed-source applications that do not have Kerberos support by default are often the most problematic.

- Kerberos is an all-or-nothing solution. If Kerberos is used on the network, any unencrypted passwords transferred to a non-Kerberos aware service is at risk. Thus, the network gains no benefit from the use of Kerberos. To secure a network with Kerberos, one must either use Kerberos-aware versions of *all* client/server applications that transmit passwords unencrypted, or not use *any* such client/server applications at all.

48.6.2. Kerberos Terminology

Kerberos has its own terminology to define various aspects of the service. Before learning how Kerberos works, it is important to learn the following terms.

authentication server (AS)

A server that issues tickets for a desired service which are in turn given to users for access to the service. The AS responds to requests from clients who do not have or do not send credentials with a request. It is usually used to gain access to the ticket-granting server (TGS) service by issuing a ticket-granting ticket (TGT). The AS usually runs on the same host as the key distribution center (KDC).

ciphertext

Encrypted data.

client

An entity on the network (a user, a host, or an application) that can receive a ticket from Kerberos.

credentials

A temporary set of electronic credentials that verify the identity of a client for a particular service. Also called a ticket.

credential cache or ticket file

A file which contains the keys for encrypting communications between a user and various network services. Kerberos 5 supports a framework for using other cache types, such as shared memory, but files are more thoroughly supported.

crypt hash

A one-way hash used to authenticate users. These are more secure than using unencrypted data, but they are still relatively easy to decrypt for an experienced cracker.

GSS-API

The Generic Security Service Application Program Interface (defined in RFC-2743 published by The Internet Engineering Task Force) is a set of functions which provide security services. This API is used by clients and services to authenticate to each other without either program having specific knowledge of the underlying mechanism. If a network service (such as cyrus-IMAP) uses GSS-API, it can authenticate using Kerberos.

hash

Also known as a *hash value*. A value generated by passing a string through a *hash function*. These values are typically used to ensure that transmitted data has not been tampered with.

hash function

A way of generating a digital "fingerprint" from input data. These functions rearrange, transpose or otherwise alter data to produce a *hash value*.

key

Data used when encrypting or decrypting other data. Encrypted data cannot be decrypted without the proper key or extremely good fortune on the part of the cracker.

key distribution center (KDC)

A service that issues Kerberos tickets, and which usually run on the same host as the ticket-granting server (TGS).

keytab (or key table)

A file that includes an unencrypted list of principals and their keys. Servers retrieve the keys they need from keytab files instead of using **`kinit`**. The default keytab file is **`/etc/krb5.keytab`**. The KDC administration server, **`/usr/kerberos/sbin/kadmind`**, is the only service that uses any other file (it uses **`/var/kerberos/krb5kdc/kadm5.keytab`**).

kinit

The **`kinit`** command allows a principal who has already logged in to obtain and cache the initial ticket-granting ticket (TGT). Refer to the **`kinit`** man page for more information.

principal (or principal name)

The principal is the unique name of a user or service allowed to authenticate using Kerberos. A principal follows the form **`root[/instance]@REALM`**. For a typical user, the root is the same as their login ID. The **`instance`** is optional. If the principal has an instance, it is separated from the root with a forward slash ("/"). An empty string ("") is considered a valid instance (which differs from the default **`NULL`** instance), but using it can be confusing. All principals in a realm have their own key, which for users is derived from a password or is randomly set for services.

realm

A network that uses Kerberos, composed of one or more servers called KDCs and a potentially large number of clients.

service

A program accessed over the network.

ticket

A temporary set of electronic credentials that verify the identity of a client for a particular service. Also called credentials.

ticket-granting server (TGS)

A server that issues tickets for a desired service which are in turn given to users for access to the service. The TGS usually runs on the same host as the KDC.

ticket-granting ticket (TGT)

A special ticket that allows the client to obtain additional tickets without applying for them from the KDC.

unencrypted password

A plain text, human-readable password.

48.6.3. How Kerberos Works

Kerberos differs from username/password authentication methods. Instead of authenticating each user to each network service, Kerberos uses symmetric encryption and a trusted third party (a KDC), to authenticate users to a suite of network services. When a user authenticates to the KDC, the KDC sends a ticket specific to that session back to the user's machine, and any Kerberos-aware services look for the ticket on the user's machine rather than requiring the user to authenticate using a password.

When a user on a Kerberos-aware network logs in to their workstation, their principal is sent to the KDC as part of a request for a TGT from the Authentication Server. This request can be sent by the log-in program so that it is transparent to the user, or can be sent by the **`kinit`** program after the user logs in.

The KDC then checks for the principal in its database. If the principal is found, the KDC creates a TGT, which is encrypted using the user's key and returned to that user.

The login or **`kinit`** program on the client then decrypts the TGT using the user's key, which it computes from the user's password. The user's key is used only on the client machine and is *not* transmitted over the network.

The TGT is set to expire after a certain period of time (usually ten to twenty-four hours) and is stored in the client machine's credentials cache. An expiration time is set so that a compromised TGT is of use to an attacker for only a short period of time. After the TGT has been issued, the user does not have to re-enter their password until the TGT expires or until they log out and log in again.

Whenever the user needs access to a network service, the client software uses the TGT to request a new ticket for that specific service from the TGS. The service ticket is then used to authenticate the user to that service transparently.

Warning

The Kerberos system can be compromised if a user on the network authenticates against a non-Kerberos aware service by transmitting a password in plain text. The use of non-Kerberos aware services is highly discouraged. Such services include Telnet and FTP. The use of other encrypted protocols, such as SSH or SSL-secured services, however, is preferred, although not ideal.

This is only a broad overview of how Kerberos authentication works. Refer to Section 48.6.10, “Additional Resources” for links to more in-depth information.

Note

Kerberos depends on the following network services to function correctly.

- Approximate clock synchronization between the machines on the network.

  A clock synchronization program should be set up for the network, such as **`ntpd`**. Refer to **`/usr/share/doc/ntp-<`*`version-number`*`>/index.html`** for details on setting up Network Time Protocol servers (where <*version-number*> is the version number of the **`ntp`** package installed on your system).

- Domain Name Service (DNS).

  You should ensure that the DNS entries and hosts on the network are all properly configured. Refer to the *Kerberos V5 System Administrator's Guide* in **`/usr/share/doc/krb5-server-<`*`version-number`*`>`** for more information (where <*version-number*> is the version number of the **`krb5-server`** package installed on your system).

48.6.4. Kerberos and PAM

Kerberos-aware services do not currently make use of Pluggable Authentication Modules (PAM) — these services bypass PAM completely. However, applications that use PAM can make use of Kerberos for authentication if the **`pam_krb5`** module (provided in the **`pam_krb5`** package) is installed. The **`pam_krb5`** package contains sample configuration files that allow services such as **`login`** and **`gdm`** to authenticate users as well as obtain initial credentials using their passwords. If access to network servers is always performed using Kerberos-aware services or services that use GSS-API, such as IMAP, the network can be considered reasonably safe.

Note

Administrators should be careful not to allow users to authenticate to most network services using Kerberos passwords. Many protocols used by these services do not encrypt the password before sending it over the network, destroying the benefits of the Kerberos system. For example, users should not be allowed to authenticate to Telnet services with the same password they use for Kerberos authentication.

48.6.5. Configuring a Kerberos 5 Server

When setting up Kerberos, install the KDC first. If it is necessary to set up slave servers, install the master first.

To configure the first Kerberos KDC, follow these steps:

1. Ensure that time synchronization and DNS are functioning correctly on all client and server machines before configuring Kerberos. Pay particular attention to time synchronization between the Kerberos server and its clients. If the time difference between the server and client is greater than five minutes (this is configurable in Kerberos 5), Kerberos clients can not authenticate to the server. This time synchronization is necessary to prevent an attacker from using an old Kerberos ticket to masquerade as a valid user.

   It is advisable to set up a Network Time Protocol (NTP) compatible client/server network even if

Kerberos is not being used. Red Hat Enterprise Linux includes the **ntp** package for this purpose. Refer to **/usr/share/doc/ntp-*<version-number>*/index.html** (where *<version-number>* is the version number of the **ntp** package installed on your system) for details about how to set up Network Time Protocol servers, and http://www.ntp.org for more information about NTP.

2. Install the **krb5-libs**, **krb5-server**, and **krb5-workstation** packages on the dedicated machine which runs the KDC. This machine needs to be very secure — if possible, it should not run any services other than the KDC.

3. Edit the **/etc/krb5.conf** and **/var/kerberos/krb5kdc/kdc.conf** configuration files to reflect the realm name and domain-to-realm mappings. A simple realm can be constructed by replacing instances of *EXAMPLE.COM* and *example.com* with the correct domain name — being certain to keep uppercase and lowercase names in the correct format — and by changing the KDC from *kerberos.example.com* to the name of the Kerberos server. By convention, all realm names are uppercase and all DNS hostnames and domain names are lowercase. For full details about the formats of these configuration files, refer to their respective man pages.

4. Create the database using the **kdb5_util** utility from a shell prompt:

```
/usr/kerberos/sbin/kdb5_util create -s
```

The **create** command creates the database that stores keys for the Kerberos realm. The **-s** switch forces creation of a *stash* file in which the master server key is stored. If no stash file is present from which to read the key, the Kerberos server (**krb5kdc**) prompts the user for the master server password (which can be used to regenerate the key) every time it starts.

5. Edit the **/var/kerberos/krb5kdc/kadm5.acl** file. This file is used by **kadmind** to determine which principals have administrative access to the Kerberos database and their level of access. Most organizations can get by with a single line:

```
*/admin@EXAMPLE.COM  *
```

Most users are represented in the database by a single principal (with a *NULL*, or empty, instance, such as *joe@EXAMPLE.COM*). In this configuration, users with a second principal with an instance of *admin* (for example, *joe/admin@EXAMPLE.COM*) are able to wield full power over the realm's Kerberos database.

After **kadmind** has been started on the server, any user can access its services by running **kadmin** on any of the clients or servers in the realm. However, only users listed in the **kadm5.acl** file can modify the database in any way, except for changing their own passwords.

Note

The **kadmin** utility communicates with the **kadmind** server over the network, and uses Kerberos to handle authentication. Consequently, the first principal must already exist before connecting to the server over the network to administer it. Create the first principal with the **kadmin.local** command, which is specifically designed to be used on the same host as the KDC and does not use Kerberos for authentication.

Type the following **kadmin.local** command at the KDC terminal to create the first principal:

```
/usr/kerberos/sbin/kadmin.local -q "addprinc username/admin"
```

6. Start Kerberos using the following commands:

   ```
   service krb5kdc start
   service kadmin start
   service krb524 start
   ```

7. Add principals for the users using the **`addprinc`** command within **`kadmin`**. **`kadmin`** and **`kadmin.local`** are command line interfaces to the KDC. As such, many commands — such as **`addprinc`** — are available after launching the **`kadmin`** program. Refer to the **`kadmin`** man page for more information.

8. Verify that the KDC is issuing tickets. First, run **`kinit`** to obtain a ticket and store it in a credential cache file. Next, use **`klist`** to view the list of credentials in the cache and use **`kdestroy`** to destroy the cache and the credentials it contains.

   Note

   By default, **`kinit`** attempts to authenticate using the same system login username (not the Kerberos server). If that username does not correspond to a principal in the Kerberos database, **`kinit`** issues an error message. If that happens, supply **`kinit`** with the name of the correct principal as an argument on the command line (**`kinit`** ***`<principal>`***).

Once these steps are completed, the Kerberos server should be up and running.

48.6.6. Configuring a Kerberos 5 Client

Setting up a Kerberos 5 client is less involved than setting up a server. At a minimum, install the client packages and provide each client with a valid **`krb5.conf`** configuration file. While **`ssh`** and **`slogin`** are the preferred method of remotely logging in to client systems, Kerberized versions of **`rsh`** and **`rlogin`** are still available, though deploying them requires that a few more configuration changes be made.

1. Be sure that time synchronization is in place between the Kerberos client and the KDC. Refer to Section 48.6.5, “Configuring a Kerberos 5 Server” for more information. In addition, verify that DNS is working properly on the Kerberos client before configuring the Kerberos client programs.

2. Install the **`krb5-libs`** and **`krb5-workstation`** packages on all of the client machines. Supply a valid **`/etc/krb5.conf`** file for each client (usually this can be the same **`krb5.conf`** file used by the KDC).

3. Before a workstation in the realm can use Kerberos to authenticate users who connect using **`ssh`** or Kerberized **`rsh`** or **`rlogin`**, it must have its own host principal in the Kerberos database. The **`sshd`**, **`kshd`**, and **`klogind`** server programs all need access to the keys for the *host* service's principal. Additionally, in order to use the kerberized **`rsh`** and **`rlogin`** services, that workstation must have the **`xinetd`** package installed.

   Using **`kadmin`**, add a host principal for the workstation on the KDC. The instance in this case is the hostname of the workstation. Use the **`-randkey`** option for the **`kadmin`**'s **`addprinc`** command to create the principal and assign it a random key:

   ```
   addprinc -randkey host/blah.example.com
   ```

   Now that the principal has been created, keys can be extracted for the workstation by running **`kadmin`** *on the workstation itself*, and using the **`ktadd`** command within **`kadmin`**:

```
ktadd -k /etc/krb5.keytab host/blah.example.com
```

4. To use other kerberized network services, they must first be started. Below is a list of some common kerberized services and instructions about enabling them:

   - **ssh** — OpenSSH uses GSS-API to authenticate users to servers if the client's and server's configuration both have **GSSAPIAuthentication** enabled. If the client also has **GSSAPIDelegateCredentials** enabled, the user's credentials are made available on the remote system.
   - **rsh** and **rlogin** — To use the kerberized versions of **rsh** and **rlogin**, enable **klogin**, **eklogin**, and **kshell**.
   - Telnet — To use kerberized Telnet, **krb5-telnet** must be enabled.
   - FTP — To provide FTP access, create and extract a key for the principal with a root of **ftp**. Be certain to set the instance to the fully qualified hostname of the FTP server, then enable **gssftp**.
   - IMAP — To use a kerberized IMAP server, the **cyrus-imap** package uses Kerberos 5 if it also has the **cyrus-sasl-gssapi** package installed. The **cyrus-sasl-gssapi** package contains the Cyrus SASL plugins which support GSS-API authentication. Cyrus IMAP should function properly with Kerberos as long as the **cyrus** user is able to find the proper key in **/etc/krb5.keytab**, and the root for the principal is set to **imap** (created with **kadmin**).

     An alternative to **cyrus-imap** can be found in the **dovecot** package, which is also included in Red Hat Enterprise Linux. This package contains an IMAP server but does not, to date, support GSS-API and Kerberos.
   - CVS — To use a kerberized CVS server, **gserver** uses a principal with a root of **cvs** and is otherwise identical to the CVS **pserver**.

   Refer to Chapter 18, *Controlling Access to Services* for details about how to enable services.

48.6.7. Domain-to-Realm Mapping

When a client attempts to access a service running on a particular server, it knows the name of the service (*host*) and the name of the server (*foo.example.com*), but because more than one realm may be deployed on your network, it must guess at the name of the realm in which the service resides.

By default, the name of the realm is taken to be the DNS domain name of the server, upper-cased.

foo.example.org → EXAMPLE.ORG
foo.example.com → EXAMPLE.COM
foo.hq.example.com → HQ.EXAMPLE.COM

In some configurations, this will be sufficient, but in others, the realm name which is derived will be the name of a non-existent realm. In these cases, the mapping from the server's DNS domain name to the name of its realm must be specified in the *domain_realm* section of the client system's **krb5.conf**. For example:

```
[domain_realm]
.example.com = EXAMPLE.COM
example.com = EXAMPLE.COM
```

The above configuration specifies two mappings. The first mapping specifies that any system in the "example.com" DNS domain belongs to the *EXAMPLE.COM* realm. The second specifies that a system with the exact name "example.com" is also in the realm. (The distinction between a domain and a specific host is marked by the presence or lack of an initial ".".) The mapping can also be stored directly in DNS.

48.6.8. Setting Up Secondary KDCs

For a number of reasons, you may choose to run multiple KDCs for a given realm. In this scenario, one KDC (the *master KDC*) keeps a writable copy of the realm database and runs**kadmind** (it is also your realm's *admin server*), and one or more KDCs (*slave KDCs*) keep read-only copies of the database and run **kpropd**.

The master-slave propagation procedure entails the master KDC dumping its database to a temporary dump file and then transmitting that file to each of its slaves, which then overwrite their previously-received read-only copies of the database with the contents of the dump file.

To set up a slave KDC, first ensure that the master KDC's **krb5.conf** and **kdc.conf** files are copied to the slave KDC.

Start **kadmin.local** from a root shell on the master KDC and use its **add_principal** command to create a new entry for the master KDC's *host* service, and then use its **ktadd** command to simultaneously set a random key for the service and store the random key in the master's default keytab file. This key will be used by the **kprop** command to authenticate to the slave servers. You will only need to do this once, regardless of how many slave servers you install.

```
~]# kadmin.local -r EXAMPLE.COM
Authenticating as principal root/admin@EXAMPLE.COM with password.
kadmin: add_principal -randkey host/masterkdc.example.com
Principal "host/host/masterkdc.example.com@EXAMPLE.COM" created.
kadmin: ktadd host/masterkdc.example.com
Entry for principal host/masterkdc.example.com with kvno 3, encryption type
Triple DES cbc mode with \
 HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/masterkdc.example.com with kvno 3, encryption type
ArcFour with HMAC/md5 \
 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/masterkdc.example.com with kvno 3, encryption type
DES with HMAC/sha1 added \
 to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/masterkdc.example.com with kvno 3, encryption type
DES cbc mode with RSA-MD5 \
 added to keytab WRFILE:/etc/krb5.keytab.
kadmin: quit
```

Start **kadmin** from a root shell on the slave KDC and use its **add_principal** command to create a new entry for the slave KDC's *host* service, and then use **kadmin**'s **ktadd** command to simultaneously set a random key for the service and store the random key in the slave's default keytab file. This key is used by the **kpropd** service when authenticating clients.

```
~]# kadmin -p jimbo/admin@EXAMPLE.COM -r EXAMPLE.COM
Authenticating as principal jimbo/admin@EXAMPLE.COM with password.
Password for jimbo/admin@EXAMPLE.COM:
kadmin: add_principal -randkey host/slavekdc.example.com
Principal "host/slavekdc.example.com@EXAMPLE.COM" created.
kadmin: ktadd host/slavekdc.example.com@EXAMPLE.COM
Entry for principal host/slavekdc.example.com with kvno 3, encryption type
```

```
Triple DES cbc mode with \
 HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/slavekdc.example.com with kvno 3, encryption type
ArcFour with HMAC/md5 added \
 to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/slavekdc.example.com with kvno 3, encryption type
DES with HMAC/sha1 added \
 to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/slavekdc.example.com with kvno 3, encryption type
DES cbc mode with RSA-MD5 added \
 to keytab WRFILE:/etc/krb5.keytab.
kadmin: quit
```

With its service key, the slave KDC could authenticate any client which would connect to it. Obviously, not all of them should be allowed to provide the slave's **kprop** service with a new realm database. To restrict access, the **kprop** service on the slave KDC will only accept updates from clients whose principal names are listed in **/var/kerberos/krb5kdc/kpropd.acl**. Add the master KDC's host service's name to that file.

```
~]# echo host/masterkdc.example.com@EXAMPLE.COM >
/var/kerberos/krb5kdc/kpropd.acl
```

Once the slave KDC has obtained a copy of the database, it will also need the master key which was used to encrypt it. If your KDC database's master key is stored in a *stash* file on the master KDC (typically named **/var/kerberos/krb5kdc/.k5.REALM**, either copy it to the slave KDC using any available secure method, or create a dummy database and identical stash file on the slave KDC by running **kdb5_util create -s** (the dummy database will be overwritten by the first successful database propagation) and supplying the same password.

Ensure that the slave KDC's firewall allows the master KDC to contact it using TCP on port 754 (*krb5_prop*), and start the **kprop** service. Then, double-check that the **kadmin** service is *disabled*.

Now perform a manual database propagation test by dumping the realm database, on the master KDC, to the default data file which the **kprop** command will read (**/var/kerberos/krb5kdc/slave_datatrans**), and then use the **kprop** command to transmit its contents to the slave KDC.

```
~]# /usr/kerberos/sbin/kdb5_util dump /var/kerberos/krb5kdc/slave_datatrans
~]# kprop slavekdc.example.com
```

Using **kinit**, verify that a client system whose **krb5.conf** lists only the slave KDC in its list of KDCs for your realm is now correctly able to obtain initial credentials from the slave KDC.

That done, simply create a script which dumps the realm database and runs the **kprop** command to transmit the database to each slave KDC in turn, and configure the **cron** service to run the script periodically.

48.6.9. Setting Up Cross Realm Authentication

Cross-realm authentication is the term which is used to describe situations in which clients (typically users) of one realm use Kerberos to authenticate to services (typically server processes running on a particular server system) which belong to a realm other than their own.

For the simplest case, in order for a client of a realm named **A.EXAMPLE.COM** to access a service in the **B.EXAMPLE.COM** realm, both realms must share a key for a principal named **krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM**, and both keys must have the same key version number associated with them.

To accomplish this, select a very strong password or passphrase, and create an entry for the principal in both realms using kadmin.

```
~]# kadmin -r A.EXAMPLE.COM
kadmin: add_principal krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM
Enter password for principal "krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM":
Re-enter password for principal "krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM":
Principal "krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM" created.
kadmin: quit
~]# kadmin -r B.EXAMPLE.COM
kadmin: add_principal krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM
Enter password for principal "krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM":
Re-enter password for principal "krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM":
Principal "krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM" created.
kadmin: quit
```

Use the **`get_principal`** command to verify that both entries have matching key version numbers (**`kvno`** values) and encryption types.

Warning

Security-conscious administrators may attempt to use the **`add_principal`** command's **`-randkey`** option to assign a random key instead of a password, dump the new entry from the database of the first realm, and import it into the second. This will not work unless the master keys for the realm databases are identical, as the keys contained in a database dump are themselves encrypted using the master key.

Clients in the **`A.EXAMPLE.COM`** realm are now able to authenticate to services in the **`B.EXAMPLE.COM`** realm. Put another way, the **`B.EXAMPLE.COM`** realm now *trusts* the **`A.EXAMPLE.COM`** realm, or phrased even more simply, **`B.EXAMPLE.COM`** now *trusts* **`A.EXAMPLE.COM`**.

This brings us to an important point: cross-realm trust is unidirectional by default. The KDC for the **`B.EXAMPLE.COM`** realm may trust clients from the **`A.EXAMPLE.COM`** to authenticate to services in the **`B.EXAMPLE.COM`** realm, but the fact that it does has no effect on whether or not clients in the **`B.EXAMPLE.COM`** realm are trusted to authenticate to services in the **`A.EXAMPLE.COM`** realm. To establish trust in the other direction, both realms would need to share keys for the **`krbtgt/A.EXAMPLE.COM@B.EXAMPLE.COM`** service (take note of the reversed in order of the two realms compared to the example above).

If direct trust relationships were the only method for providing trust between realms, networks which contain multiple realms would be very difficult to set up. Luckily, cross-realm trust is transitive. If clients from **`A.EXAMPLE.COM`** can authenticate to services in **`B.EXAMPLE.COM`**, and clients from **`B.EXAMPLE.COM`** can authenticate to services in **`C.EXAMPLE.COM`**, then clients in **`A.EXAMPLE.COM`** can also authenticate to services in **`C.EXAMPLE.COM`**, *even if **`C.EXAMPLE.COM`** doesn't directly trust **`A.EXAMPLE.COM`***. This means that, on a network with multiple realms which all need to trust each other, making good choices about which trust relationships to set up can greatly reduce the amount of effort required.

Now you face the more conventional problems: the client's system must be configured so that it can properly deduce the realm to which a particular service belongs, and it must be able to determine how to obtain credentials for services in that realm.

First things first: the principal name for a service provided from a specific server system in a given realm typically looks like this:

```
service/server.example.com@EXAMPLE.COM
```

In this example, *service* is typically either the name of the protocol in use (other common values include *ldap*, *imap*, *cvs*, and *HTTP*) or *host*, *server.example.com* is the fully-qualified domain name of the system which runs the service, and **`EXAMPLE.COM`** is the name of the realm.

To deduce the realm to which the service belongs, clients will most often consult DNS or the **`domain_realm`** section of **`/etc/krb5.conf`** to map either a hostname (*server.example.com*) or a DNS domain name (*.example.com*) to the name of a realm (*EXAMPLE.COM*).

Having determined which to which realm a service belongs, a client then has to determine the set of realms which it needs to contact, and in which order it must contact them, to obtain credentials for use in authenticating to the service.

This can be done in one of two ways.

The default method, which requires no explicit configuration, is to give the realms names within a shared hierarchy. For an example, assume realms named **`A.EXAMPLE.COM`**, **`B.EXAMPLE.COM`**, and **`EXAMPLE.COM`**. When a client in the **`A.EXAMPLE.COM`** realm attempts to authenticate to a service in **`B.EXAMPLE.COM`**, it will, by default, first attempt to get credentials for the **`EXAMPLE.COM`** realm, and then to use those credentials to obtain credentials for use in the **`B.EXAMPLE.COM`** realm.

The client in this scenario treats the realm name as one might treat a DNS name. It repeatedly strips off the components of its own realm's name to generate the names of realms which are "above" it in the hierarchy until it reaches a point which is also "above" the service's realm. At that point it begins prepending components of the service's realm name until it reaches the service's realm. Each realm which is involved in the process is another "hop".

For example, using credentials in **`A.EXAMPLE.COM`**, authenticating to a service in **`B.EXAMPLE.COM`**:

A.EXAMPLE.COM → EXAMPLE.COM → B.EXAMPLE.COM

- **`A.EXAMPLE.COM`** and **`EXAMPLE.COM`** share a key for **`krbtgt/EXAMPLE.COM@A.EXAMPLE.COM`**
- **`EXAMPLE.COM`** and **`B.EXAMPLE.COM`** share a key for **`krbtgt/B.EXAMPLE.COM@EXAMPLE.COM`**

Another example, using credentials in **`SITE1.SALES.EXAMPLE.COM`**, authenticating to a service in **`EVERYWHERE.EXAMPLE.COM`**:

SITE1.SALES.EXAMPLE.COM → SALES.EXAMPLE.COM → EXAMPLE.COM → EVERYWHERE.EXAMPLI

- **`SITE1.SALES.EXAMPLE.COM`** and **`SALES.EXAMPLE.COM`** share a key for **`krbtgt/SALES.EXAMPLE.COM@SITE1.SALES.EXAMPLE.COM`**
- **`SALES.EXAMPLE.COM`** and **`EXAMPLE.COM`** share a key for **`krbtgt/EXAMPLE.COM@SALES.EXAMPLE.COM`**
- **`EXAMPLE.COM`** and **`EVERYWHERE.EXAMPLE.COM`** share a key for **`krbtgt/EVERYWHERE.EXAMPLE.COM@EXAMPLE.COM`**

Another example, this time using realm names whose names share no common suffix (**`DEVEL.EXAMPLE.COM`** and **`PROD.EXAMPLE.ORG`**):

DEVEL.EXAMPLE.COM → EXAMPLE.COM → COM → ORG → EXAMPLE.ORG → PROD.EXAMPLE.ORG

- **DEVEL.EXAMPLE.COM** and **EXAMPLE.COM** share a key for **krbtgt/EXAMPLE.COM@DEVEL.EXAMPLE.COM**
- **EXAMPLE.COM** and **COM** share a key for **krbtgt/COM@EXAMPLE.COM**
- **COM** and **ORG** share a key for **krbtgt/ORG@COM**
- **ORG** and **EXAMPLE.ORG** share a key for **krbtgt/EXAMPLE.ORG@ORG**
- **EXAMPLE.ORG** and **PROD.EXAMPLE.ORG** share a key for **krbtgt/PROD.EXAMPLE.ORG@EXAMPLE.ORG**

The more complicated, but also more flexible, method involves configuring the **capaths** section of **/etc/krb5.conf**, so that clients which have credentials for one realm will be able to look up which realm is next in the chain which will eventually lead to the being able to authenticate to servers.

The format of the **capaths** section is relatively straightforward: each entry in the section is named after a realm in which a client might exist. Inside of that subsection, the set of intermediate realms from which the client must obtain credentials is listed as values of the key which corresponds to the realm in which a service might reside. If there are no intermediate realms, the value "." is used.

Here's an example:

```
[capaths]
A.EXAMPLE.COM = {
 B.EXAMPLE.COM = .
 C.EXAMPLE.COM = B.EXAMPLE.COM
 D.EXAMPLE.COM = B.EXAMPLE.COM
 D.EXAMPLE.COM = C.EXAMPLE.COM
}
```

In this example, clients in the **A.EXAMPLE.COM** realm can obtain cross-realm credentials for **B.EXAMPLE.COM** directly from the **A.EXAMPLE.COM** KDC.

If those clients wish to contact a service in the**C.EXAMPLE.COM** realm, they will first need to obtain necessary credentials from the **B.EXAMPLE.COM** realm (this requires that **krbtgt/B.EXAMPLE.COM@A.EXAMPLE.COM** exist), and then use **those** credentials to obtain credentials for use in the **C.EXAMPLE.COM** realm (using **krbtgt/C.EXAMPLE.COM@B.EXAMPLE.COM**).

If those clients wish to contact a service in the **D.EXAMPLE.COM** realm, they will first need to obtain necessary credentials from the **B.EXAMPLE.COM** realm, and then credentials from the **C.EXAMPLE.COM** realm, before finally obtaining credentials for use with the **D.EXAMPLE.COM** realm.

Note

Without a capath entry indicating otherwise, Kerberos assumes that cross-realm trust relationships form a hierarchy.

Clients in the **A.EXAMPLE.COM** realm can obtain cross-realm credentials from **B.EXAMPLE.COM** realm directly. Without the "." indicating this, the client would instead attempt to use a hierarchical path, in this case:

A.EXAMPLE.COM → EXAMPLE.COM → B.EXAMPLE.COM

48.6.10. Additional Resources

48.6.10. Additional Resources

For more information about Kerberos, refer to the following resources.

48.6.10.1. Installed Documentation

- The *Kerberos V5 Installation Guide* and the *Kerberos V5 System Administrator's Guide* in PostScript and HTML formats. These can be found in the **`/usr/share/doc/krb5-server-<version-number>/`** directory (where *<version-number>* is the version number of the **`krb5-server`** package installed on your system).
- The *Kerberos V5 UNIX User's Guide* in PostScript and HTML formats. These can be found in the **`/usr/share/doc/krb5-workstation-<version-number>/`** directory (where *<version-number>* is the version number of the **`krb5-workstation`** package installed on your system).
- Kerberos man pages — There are a number of man pages for the various applications and configuration files involved with a Kerberos implementation. The following is a list of some of the more important man pages.

  Client Applications

  - **`man kerberos`** — An introduction to the Kerberos system which describes how credentials work and provides recommendations for obtaining and destroying Kerberos tickets. The bottom of the man page references a number of related man pages.
  - **`man kinit`** — Describes how to use this command to obtain and cache a ticket-granting ticket.
  - **`man kdestroy`** — Describes how to use this command to destroy Kerberos credentials.
  - **`man klist`** — Describes how to use this command to list cached Kerberos credentials.

  Administrative Applications

  - **`man kadmin`** — Describes how to use this command to administer the Kerberos V5 database.
  - **`man kdb5_util`** — Describes how to use this command to create and perform low-level administrative functions on the Kerberos V5 database.

  Server Applications

  - **`man krb5kdc`** — Describes available command line options for the Kerberos V5 KDC.
  - **`man kadmind`** — Describes available command line options for the Kerberos V5 administration server.

  Configuration Files

  - **`man krb5.conf`** — Describes the format and options available within the configuration file for the Kerberos V5 library.
  - **`man kdc.conf`** — Describes the format and options available within the configuration file for the Kerberos V5 AS and KDC.

48.6.10.2. Useful Websites

- http://web.mit.edu/kerberos/www/ — *Kerberos: The Network Authentication Protocol* webpage from MIT.

- http://www.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html — The Kerberos Frequently Asked Questions (FAQ).
- ftp://athena-dist.mit.edu/pub/kerberos/doc/usenix.PS — The PostScript version of *Kerberos: An Authentication Service for Open Network Systems* by Jennifer G. Steiner, Clifford Neuman, and Jeffrey I. Schiller. This document is the original paper describing Kerberos.
- http://web.mit.edu/kerberos/www/dialogue.html — *Designing an Authentication System: a Dialogue in Four Scenes* originally by Bill Bryant in 1988, modified by Theodore Ts'o in 1997. This document is a conversation between two developers who are thinking through the creation of a Kerberos-style authentication system. The conversational style of the discussion make this a good starting place for people who are completely unfamiliar with Kerberos.
- http://www.ornl.gov/~jar/HowToKerb.html — *How to Kerberize your site* is a good reference for kerberizing a network.
- http://www.networkcomputing.com/netdesign/kerb1.html — *Kerberos Network Design Manual* is a thorough overview of the Kerberos system.

48.7. Virtual Private Networks (VPNs)

Organizations with several satellite offices often connect to each other with dedicated lines for efficiency and protection of sensitive data in transit. For example, many businesses use frame relay or *Asynchronous Transfer Mode* (ATM) lines as an end-to-end networking solution to link one office with others. This can be an expensive proposition, especially for small to medium sized businesses (SMBs) that want to expand without paying the high costs associated with enterprise-level, dedicated digital circuits.

To address this need, *Virtual Private Networks* (VPNs) were developed. Following the same functional principles as dedicated circuits, VPNs allow for secured digital communication between two parties (or networks), creating a *Wide Area Network* (WAN) from existing *Local Area Networks* (LANs). Where it differs from frame relay or ATM is in its transport medium. VPNs transmit over IP using datagrams as the transport layer, making it a secure conduit through the Internet to an intended destination. Most free software VPN implementations incorporate open standard encryption methods to further mask data in transit.

Some organizations employ hardware VPN solutions to augment security, while others use software or protocol-based implementations. Several vendors provide hardware VPN solutions, such as Cisco, Nortel, IBM, and Checkpoint. There is a free software-based VPN solution for Linux called FreeS/Wan that utilizes a standardized *Internet Protocol Security* (IPsec) implementation. These VPN solutions, irrespective of whether they are hardware or software based, act as specialized routers that exist between the IP connection from one office to another.

48.7.1. How Does a VPN Work?

When a packet is transmitted from a client, it sends it through the VPN router or gateway, which adds an *Authentication Header* (AH) for routing and authentication. The data is then encrypted and, finally, enclosed with an *Encapsulating Security Payload* (ESP). This latter constitutes the decryption and handling instructions.

The receiving VPN router strips the header information, decrypts the data, and routes it to its intended destination (either a workstation or other node on a network). Using a network-to-network connection, the receiving node on the local network receives the packets already decrypted and ready for processing. The encryption/decryption process in a network-to-network VPN connection is transparent to a local node.

With such a heightened level of security, an attacker must not only intercept a packet, but decrypt the packet as well. Intruders who employ a man-in-the-middle attack between a server and client must also have access to at least one of the private keys for authenticating sessions. Because they employ several layers of authentication and encryption, VPNs are a secure and effective means of connecting multiple remote nodes to act as a unified intranet.

48.7.2. VPNs and Red Hat Enterprise Linux

Red Hat Enterprise Linux provides various options in terms of implementing a software solution to securely connect to a WAN. *Internet Protocol Security* (IPsec) is the supported VPN implementation for Red Hat Enterprise Linux, and sufficiently addresses the usability needs of organizations with branch offices or remote users.

48.7.3. IPsec

Red Hat Enterprise Linux supports IPsec for connecting remote hosts and networks to each other using a secure tunnel on a common carrier network such as the Internet. IPsec can be implemented using a host-to-host (one computer workstation to another) or network-to-network (one LAN/WAN to another) configuration.

The IPsec implementation in Red Hat Enterprise Linux uses *Internet Key Exchange* (*IKE*), a protocol implemented by the Internet Engineering Task Force (IETF), used for mutual authentication and secure associations between connecting systems.

48.7.4. Creating an IPsec Connection

An IPsec connection is split into two logical phases. In phase 1, an IPsec node initializes the connection with the remote node or network. The remote node or network checks the requesting node's credentials and both parties negotiate the authentication method for the connection.

On Red Hat Enterprise Linux systems, an IPsec connection uses the *pre-shared key* method of IPsec node authentication. In a pre-shared key IPsec connection, both hosts must use the same key in order to move to Phase 2 of the IPsec connection.

Phase 2 of the IPsec connection is where the *Security Association* (SA) is created between IPsec nodes. This phase establishes an SA database with configuration information, such as the encryption method, secret session key exchange parameters, and more. This phase manages the actual IPsec connection between remote nodes and networks.

The Red Hat Enterprise Linux implementation of IPsec uses IKE for sharing keys between hosts across the Internet. The **`racoon`** keying daemon handles the IKE key distribution and exchange. Refer to the **`racoon`** man page for more information about this daemon.

48.7.5. IPsec Installation

Implementing IPsec requires that the **`ipsec-tools`** RPM package be installed on all IPsec hosts (if using a host-to-host configuration) or routers (if using a network-to-network configuration). The RPM package contains essential libraries, daemons, and configuration files for setting up the IPsec connection, including:

- **`/sbin/setkey`** — manipulates the key management and security attributes of IPsec in the kernel. This executable is controlled by the **`racoon`** key management daemon. Refer to the **`setkey`**(8) man page for more information.
- **`/usr/sbin/racoon`** — the IKE key management daemon, used to manage and control security associations and key sharing between IPsec-connected systems.
- **`/etc/racoon/racoon.conf`** — the **`racoon`** daemon configuration file used to configure various aspects of the IPsec connection, including authentication methods and encryption algorithms used in the connection. Refer to the **`racoon.conf`**(5) man page for a complete listing of available directives.

To configure IPsec on Red Hat Enterprise Linux, you can use the **Network Administration Tool**, or manually edit the networking and IPsec configuration files.

- To connect two network-connected hosts via IPsec, refer to Section 48.7.6, “IPsec Host-to-Host Configuration”.
- To connect one LAN/WAN to another via IPsec, refer to Section 48.7.7, “IPsec Network-to-Network Configuration”.

48.7.6. IPsec Host-to-Host Configuration

IPsec can be configured to connect one desktop or workstation (host) to another using a host-to-host connection. This type of connection uses the network to which each host is connected to create a secure tunnel between each host. The requirements of a host-to-host connection are minimal, as is the configuration of IPsec on each host. The hosts need only a dedicated connection to a carrier network (such as the Internet) and Red Hat Enterprise Linux to create the IPsec connection.

48.7.6.1. Host-to-Host Connection

A host-to-host IPsec connection is an encrypted connection between two systems, both running IPsec with the same authentication key. With the IPsec connection active, any network traffic between the two hosts is encrypted.

To configure a host-to-host IPsec connection, use the following steps for each host:

Note

You should perform the following procedures on the actual machine that you are configuring. Avoid attempting to configure and establish IPsec connections remotely.

1. In a command shell, type **`system-config-network`** to start the **Network Administration Tool**.
2. On the **`IPsec`** tab, click **`New`** to start the IPsec configuration wizard.
3. Click **`Forward`** to start configuring a host-to-host IPsec connection.
4. Enter a unique name for the connection, for example, **`ipsec0`**. If required, select the check box to automatically activate the connection when the computer starts. Click **`Forward`** to continue.
5. Select **`Host to Host encryption`** as the connection type, and then click **`Forward`**.
6. Select the type of encryption to use: manual or automatic.

   If you select manual encryption, an encryption key must be provided later in the process. If you select automatic encryption, the **`racoon`** daemon manages the encryption key. The **`ipsec-tools`** package must be installed if you want to use automatic encryption.

   Click **`Forward`** to continue.
7. Enter the IP address of the remote host.

   To determine the IP address of the remote host, use the following command *on the remote host*:

```
ifconfig <device>
```

where *<device>* is the Ethernet device that you want to use for the VPN connection.

If only one Ethernet card exists in the system, the device name is typically eth0. The following example shows the relevant information from this command (note that this is an example output only):

```
eth0      Link encap:Ethernet  HWaddr 00:0C:6E:E8:98:1D
          inet addr:172.16.44.192  Bcast:172.16.45.255
Mask:255.255.254.0
```

The IP address is the number following the **`inet addr:`** label.

Note

For host-to-host connections, both hosts should have a public, routable address. Alternatively, both hosts can have a private, non-routable address (for example, from the 10.x.x.x or 192.168.x.x ranges) as long as they are on the sam LAN.

If the hosts are on different LANs, or one has a public address while the other has a private address, refer to Section 48.7.7, "IPsec Network-to-Network Configuration".

Click **`Forward`** to continue.

8. If manual encryption was selected in step 6, specify the encryption key to use, or click **`Generate`** to create one.

   a. Specify an authentication key or click **`Generate`** to generate one. It can be any combination of numbers and letters.

   b. Click **`Forward`** to continue.

9. Verify the information on the **`IPsec — Summary`** page, and then click **`Apply`**.

10. Click **File** > **Save** to save the configuration.

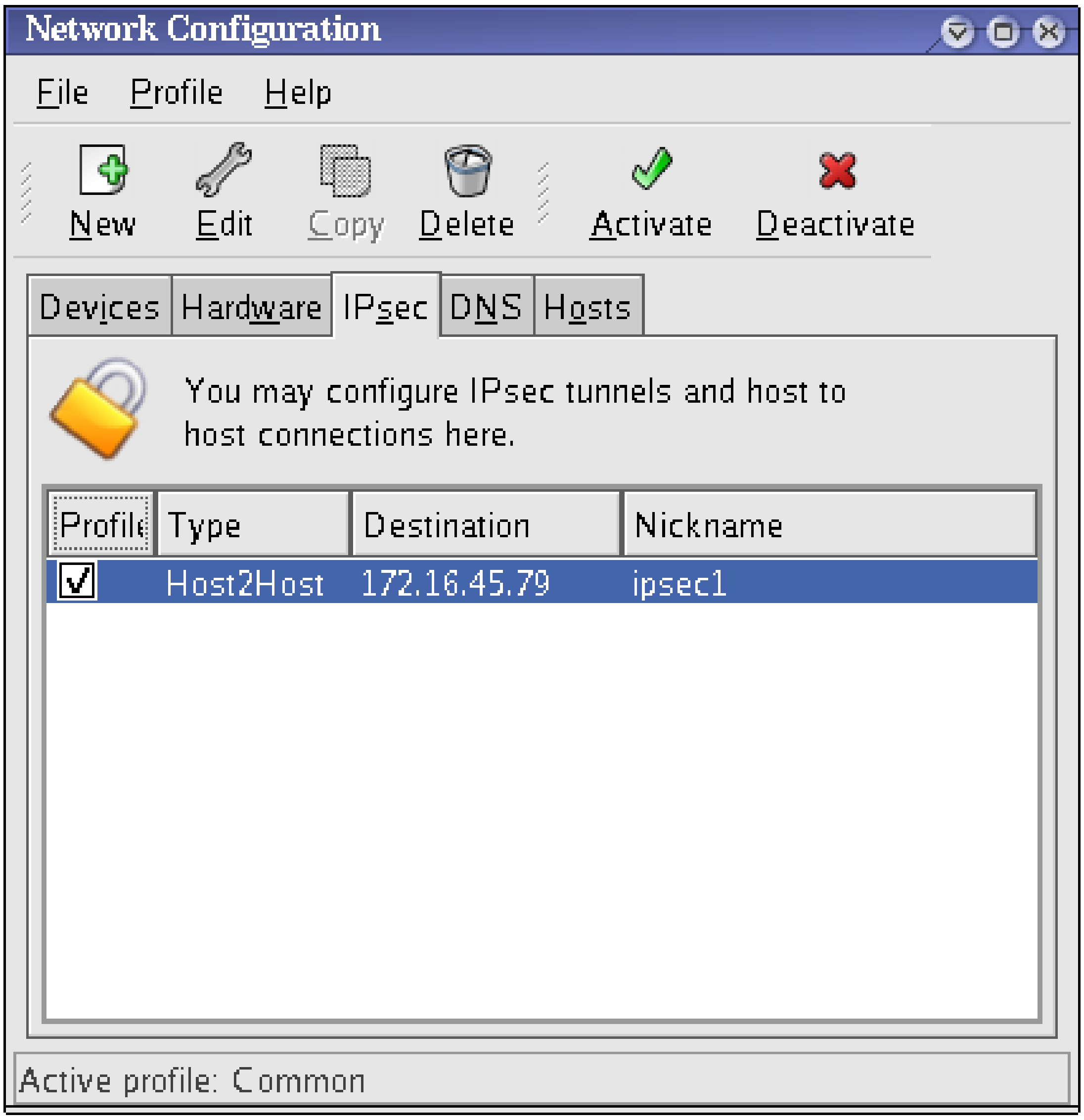
    You may need to restart the network for the changes to take effect. To restart the network, use the following command:

    ```
    service network restart
    ```

11. Select the IPsec connection from the list and click the **`Activate`** button.

12. Repeat the entire procedure for the other host. It is essential that the same keys from step 8 be used on the other hosts. Otherwise, IPsec will not work.

After configuring the IPsec connection, it appears in the IPsec list as shown in Figure 48.10, "IPsec Connection".


Network Configuration
File Profile Help
New Edit Copy Delete Activate Deactivate
Devices Hardware IPsec DNS Hosts

You may configure IPsec tunnels and host to host connections here.

Profile	Type	Destination	Nickname
✓	Host2Host	172.16.45.79	ipsec1

Active profile: Common

Figure 48.10. IPsec Connection

The following files are created when the IPsec connection is configured:

- **`/etc/sysconfig/network-scripts/ifcfg-<nickname>`**
- **`/etc/sysconfig/network-scripts/keys-<nickname>`**
- **`/etc/racoon/<remote-ip>.conf`**
- **`/etc/racoon/psk.txt`**

If automatic encryption is selected, **`/etc/racoon/racoon.conf`** is also created.

When the interface is up, **`/etc/racoon/racoon.conf`** is modified to include **`<remote-ip>.conf`**.

48.7.6.2. Manual IPsec Host-to-Host Configuration

The first step in creating a connection is to gather system and network information from each workstation. For a host-to-host connection, you need the following:

- The IP address of each host

- A unique name, for example, **`ipsec1`**. This is used to identify the IPsec connection and to distinguish it from other devices or connections.
- A fixed encryption key or one automatically generated by **`racoon`**.
- A pre-shared authentication key that is used during the initial stage of the connection and to exchange encryption keys during the session.

For example, suppose Workstation A and Workstation B want to connect to each other through an IPsec tunnel. They want to connect using a pre-shared key with the value of **`Key_Value01`**, and the users agree to let **`racoon`** automatically generate and share an authentication key between each host. Both host users decide to name their connections **`ipsec1`**.

Note

You should choose a PSK that uses a mixture of upper- and lower-case characters, numbers and punctuation. An easily-guessable PSK constitutes a security risk.

It is not necessary to use the same connection name for each host. You should choose a name that is convenient and meaningful for your installation.

The following is the IPsec configuration file for Workstation A for a host-to-host IPsec connection with Workstation B. The unique name to identify the connection in this example is *ipsec1*, so the resulting file is called **`/etc/sysconfig/network-scripts/ifcfg-ipsec1`**.

```
DST=X.X.X.X
TYPE=IPSEC
ONBOOT=no
IKE_METHOD=PSK
```

For Workstation A, *X.X.X.X* is the IP address of Workstation B. For Workstation B,*X.X.X.X* is the IP address of Workstation A. This connection is not set to initiate on boot-up (**`ONBOOT=no`**) and it uses the pre-shared key method of authentication (**`IKE_METHOD=PSK`**).

The following is the content of the pre-shared key file (called **`/etc/sysconfig/network-scripts/keys-ipsec1`**) that both workstations need to authenticate each other. The contents of this file should be identical on both workstations, and only the root user should be able to read or write this file.

```
IKE_PSK=Key_Value01
```

Important

To change the **`keys-ipsec1`** file so that only the root user can read or edit the file, use the following command after creating the file:

```
chmod 600 /etc/sysconfig/network-scripts/keys-ipsec1
```

To change the authentication key at any time, edit the **`keys-ipsec1`** file on both workstations. *Both authentication keys must be identical for proper connectivity*.

The next example shows the specific configuration for the phase 1 connection to the remote host. The file is called ***X.X.X.X*.conf**, where *X.X.X.X* is the IP address of the remote IPsec host. Note that this file is automatically generated when the IPsec tunnel is activated and should not be edited directly.

```
remote X.X.X.X
{
         exchange_mode aggressive, main;
  my_identifier address;
  proposal {
   encryption_algorithm 3des;
  hash_algorithm sha1;
  authentication_method pre_shared_key;
  dh_group 2 ;
 }
}
```

The default phase 1 configuration file that is created when an IPsec connection is initialized contains the following statements used by the Red Hat Enterprise Linux implementation of IPsec:

remote *X.X.X.X*

Specifies that the subsequent stanzas of this configuration file apply only to the remote node identified by the *X.X.X.X* IP address.

exchange_mode aggressive

The default configuration for IPsec on Red Hat Enterprise Linux uses an aggressive authentication mode, which lowers the connection overhead while allowing configuration of several IPsec connections with multiple hosts.

my_identifier address

Specifies the identification method to use when authenticating nodes. Red Hat Enterprise Linux uses IP addresses to identify nodes.

encryption_algorithm 3des

Specifies the encryption cipher used during authentication. By default, *Triple Data Encryption Standard* (3DES) is used.

hash_algorithm sha1;

Specifies the hash algorithm used during phase 1 negotiation between nodes. By default, Secure Hash Algorithm version 1 is used.

authentication_method pre_shared_key

Specifies the authentication method used during node negotiation. By default, Red Hat Enterprise Linux uses pre-shared keys for authentication.

dh_group 2

Specifies the Diffie-Hellman group number for establishing dynamically-generated session keys. By default, modp1024 (group 2) is used.

48.7.6.2.1. The Racoon Configuration File

The **/etc/racoon/racoon.conf** files should be identical on all IPsec nodes *except* for the **include "/etc/racoon/*X.X.X.X*.conf"** statement. This statement (and the file it references) is generated when the IPsec tunnel is activated. For Workstation A, the *X.X.X.X* in the **`include`** statement is Workstation B's IP address. The opposite is true of Workstation B. The following shows a typical **`racoon.conf`** file when the IPsec connection is activated.

```
# Racoon IKE daemon configuration file.
# See 'man racoon.conf' for a description of the format and entries.

path include "/etc/racoon";
path pre_shared_key "/etc/racoon/psk.txt";
path certificate "/etc/racoon/certs";

sainfo anonymous
{
        pfs_group 2;
        lifetime time 1 hour ;
        encryption_algorithm 3des, blowfish 448, rijndael ;
        authentication_algorithm hmac_sha1, hmac_md5 ;
        compression_algorithm deflate ;
}
include "/etc/racoon/X.X.X.X.conf";
```

This default **`racoon.conf`** file includes defined paths for IPsec configuration, pre-shared key files, and certificates. The fields in **`sainfo anonymous`** describe the phase 2 SA between the IPsec nodes — the nature of the IPsec connection (including the supported encryption algorithms used) and the method of exchanging keys. The following list defines the fields of phase 2:

sainfo anonymous

Denotes that SA can anonymously initialize with any peer provided that the IPsec credentials match.

pfs_group 2

Defines the Diffie-Hellman key exchange protocol, which determines the method by which the IPsec nodes establish a mutual temporary session key for the second phase ofIPsec connectivity. By default, the Red Hat Enterprise Linux implementation of IPsec uses group 2 (or **`modp1024`**) of the Diffie-Hellman cryptographic key exchange groups. Group 2 uses a 1024-bit modular exponentiation that prevents attackers from decrypting previous IPsec transmissions even if a private key is compromised.

lifetime time 1 hour

This parameter specifies the lifetime of an SA and can be quantified either by time or by bytes of data. The default Red Hat Enterprise Linux implementation of IPsec specifies a one hour lifetime.

encryption_algorithm 3des, blowfish 448, rijndael

Specifies the supported encryption ciphers for phase 2. Red Hat Enterprise Linux supports 3DES, 448-bit Blowfish, and Rijndael (the cipher used in the *Advanced Encryption Standard*, or AES).

authentication_algorithm hmac_sha1, hmac_md5

Lists the supported hash algorithms for authentication. Supported modes are sha1 and md5 hashed message authentication codes (HMAC).

compression_algorithm deflate

Defines the Deflate compression algorithm for IP Payload Compression (IPCOMP) support, which allows for potentially faster transmission of IP datagrams over slow connections.

To start the connection, use the following command on each host:

```
ifup <nickname>
```

where <nickname> is the name you specified for the IPsec connection.

To test the IPsec connection, run the **tcpdump** utility to view the network packets being transferred between the hosts and verify that they are encrypted via IPsec. The packet should include an AH header and should be shown as ESP packets. ESP means it is encrypted. For example:

```
~]# tcpdump -n -i eth0 host <targetSystem>

IP 172.16.45.107 > 172.16.44.192: AH(spi=0x0954ccb6,seq=0xbb):
ESP(spi=0x0c9f2164,seq=0xbb)
```

48.7.7. IPsec Network-to-Network Configuration

IPsec can also be configured to connect an entire network (such as a LAN or WAN) to a remote network using a network-to-network connection. A network-to-network connection requires the setup of IPsec routers on each side of the connecting networks to transparently process and route information from one node on a LAN to a node on a remote LAN. Figure 48.11, “A network-to-network IPsec tunneled connection” shows a network-to-network IPsec tunneled connection.

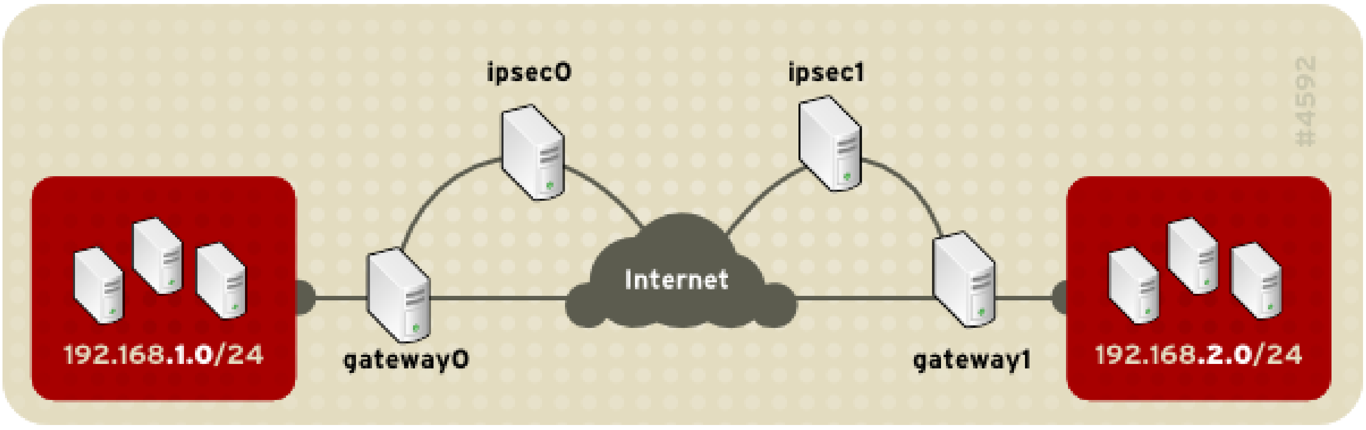


Figure 48.11. A network-to-network IPsec tunneled connection

This diagram shows two separate LANs separated by the Internet. These LANs use IPsec routers to authenticate and initiate a connection using a secure tunnel through the Internet. Packets that are intercepted in transit would require brute-force decryption in order to crack the cipher protecting the packets between these LANs. The process of communicating from one node in the 192.168.1.0/24 IP range to another in the 192.168.2.0/24 range is completely transparent to the nodes as the processing, encryption/decryption, and routing of the IPsec packets are completely handled by the IPsec router.

The information needed for a network-to-network connection include:

- The externally-accessible IP addresses of the dedicated IPsec routers
- The network address ranges of the LAN/WAN served by the IPsec routers (such as 192.168.1.0/24 or 10.0.1.0/24)
- The IP addresses of the gateway devices that route the data from the network nodes to the Internet

- A unique name, for example, **`ipsec1`**. This is used to identify the IPsec connection and to distinguish it from other devices or connections.
- A fixed encryption key or one automatically generated by **`racoon`**
- A pre-shared authentication key that is used during the initial stage of the connection and to exchange encryption keys during the session.

48.7.7.1. Network-to-Network (VPN) Connection

A network-to-network IPsec connection uses two IPsec routers, one for each network, through which the network traffic for the private subnets is routed.

For example, as shown in Figure 48.12, "Network-to-Network IPsec", if the 192.168.1.0/24 private network sends network traffic to the 192.168.2.0/24 private network, the packets go through gateway0, to ipsec0, through the Internet, to ipsec1, to gateway1, and to the 192.168.2.0/24 subnet.

IPsec routers require publicly addressable IP addresses and a second Ethernet device connected to their respective private networks. Traffic only travels through an IPsec router if it is intended for another IPsec router with which it has an encrypted connection.

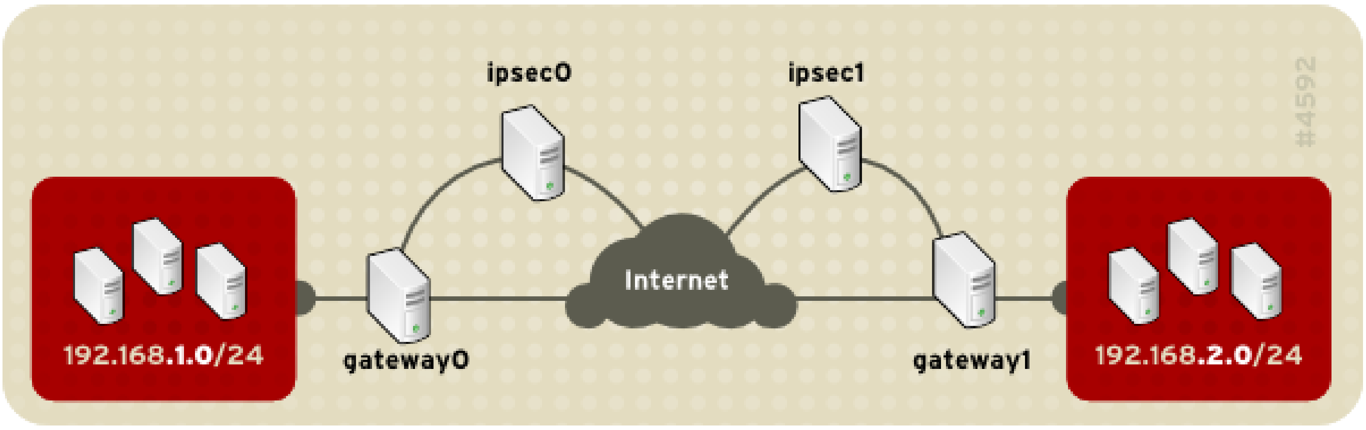


Figure 48.12. Network-to-Network IPsec

Alternate network configuration options include a firewall between each IP router and the Internet, and an intranet firewall between each IPsec router and subnet gateway. The IPsec router and the gateway for the subnet can be one system with two Ethernet devices: one with a public IP address that acts as the IPsec router; and one with a private IP address that acts as the gateway for the private subnet. Each IPsec router can use the gateway for its private network or a public gateway to send the packets to the other IPsec router.

Use the following procedure to configure a network-to-network IPsec connection:

1. In a command shell, type **`system-config-network`** to start the **Network Administration Tool**.
2. On the **`IPsec`** tab, click **`New`** to start the IPsec configuration wizard.
3. Click **`Forward`** to start configuring a network-to-network IPsec connection.
4. Enter a unique nickname for the connection, for example, **`ipsec0`**. If required, select the check box to automatically activate the connection when the computer starts. Click **`Forward`** to continue.
5. Select **`Network to Network encryption (VPN)`** as the connection type, and then click **`Forward`**.
6. Select the type of encryption to use: manual or automatic.

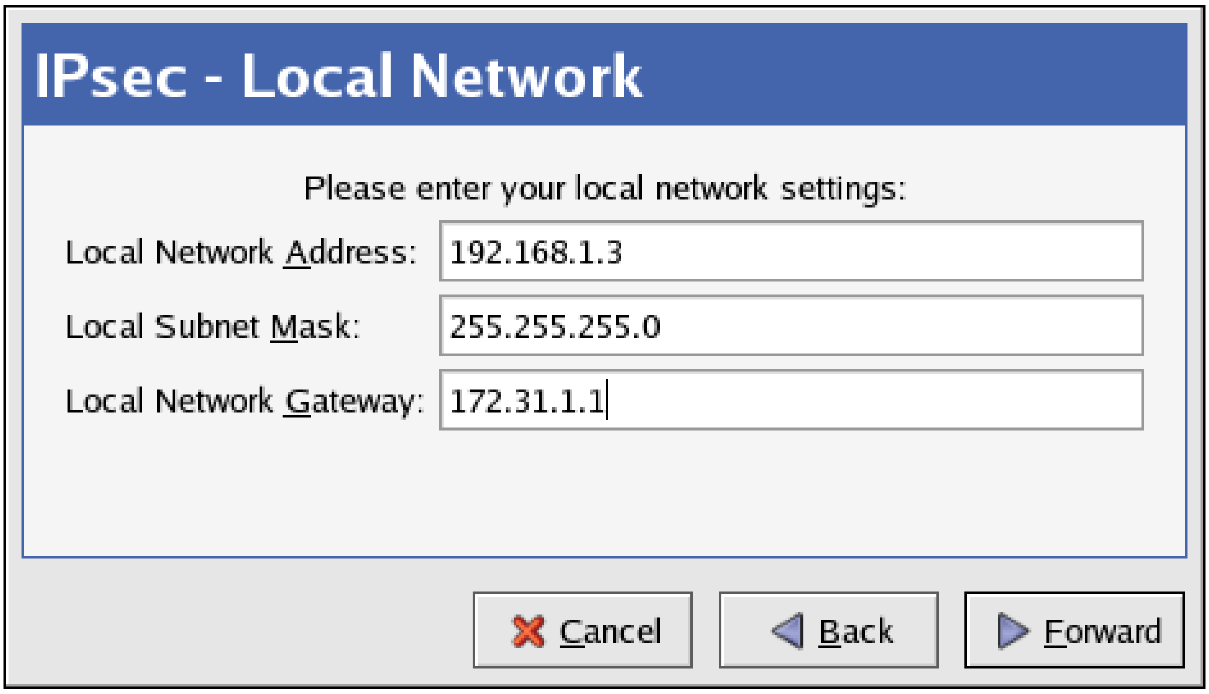
If you select manual encryption, an encryption key must be provided later in the process. If you select automatic encryption, the **`racoon`** daemon manages the encryption key. The **`ipsec-tools`** package must be installed if you want to use automatic encryption.

Click **`Forward`** to continue.

7. On the **`Local Network`** page, enter the following information:

   - **`Local Network Address`** — The IP address of the device on the IPsec router connected to the private network.
   - **`Local Subnet Mask`** — The subnet mask of the local network IP address.
   - **`Local Network Gateway`** — The gateway for the private subnet.

   Click **`Forward`** to continue.



IPsec - Local Network

Please enter your local network settings:

Local Network Address: 192.168.1.3

Local Subnet Mask: 255.255.255.0

Local Network Gateway: 172.31.1.1

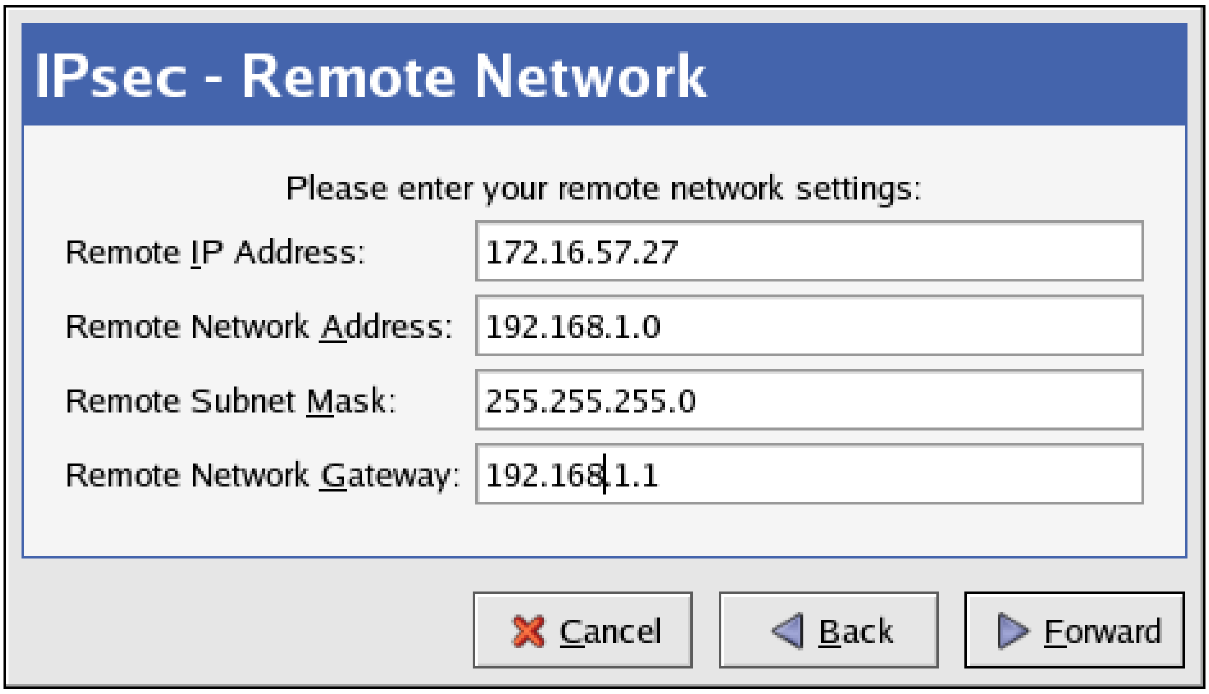
Figure 48.13. Local Network Information

8. On the **`Remote Network`** page, enter the following information:

   - **`Remote IP Address`** — The publicly addressable IP address of the IPsec router for the *other* private network. In our example, for ipsec0, enter the publicly addressable IP address of ipsec1, and vice versa.
   - **`Remote Network Address`** — The network address of the private subnet behind the *other* IPsec router. In our example, enter **`192.168.1.0`** if configuring ipsec1, and enter **`192.168.2.0`** if configuring ipsec0.
   - **`Remote Subnet Mask`** — The subnet mask of the remote IP address.
   - **`Remote Network Gateway`** — The IP address of the gateway for the remote network address.
   - If manual encryption was selected in step 6, specify the encryption key to use or click **`Generate`** to create one.

Specify an authentication key or click **Generate** to generate one. This key can be any combination of numbers and letters.

Click **Forward** to continue.


IPsec - Remote Network

Please enter your remote network settings:

Remote IP Address: 172.16.57.27

Remote Network Address: 192.168.1.0

Remote Subnet Mask: 255.255.255.0

Remote Network Gateway: 192.168.1.1

Figure 48.14. Remote Network Information

9. Verify the information on the **IPsec — Summary** page, and then click **Apply**.
10. Select **File** > **Save** to save the configuration.
11. Select the IPsec connection from the list, and then click **Activate** to activate the connection.
12. Enable IP forwarding:
    a. Edit **/etc/sysctl.conf** and set **net.ipv4.ip_forward** to **1**.
    b. Use the following command to enable the change:

```
sysctl -p /etc/sysctl.conf
```

The network script to activate the IPsec connection automatically creates network routes to send packets through the IPsec router if necessary.

48.7.7.2. Manual IPsec Network-to-Network Configuration

Suppose LAN A (lana.example.com) and LAN B (lanb.example.com) want to connect to each other through an IPsec tunnel. The network address for LAN A is in the 192.168.1.0/24 range, while LAN B uses the 192.168.2.0/24 range. The gateway IP address is 192.168.1.254 for LAN A and 192.168.2.254 for LAN B. The IPsec routers are separate from each LAN gateway and use two network devices: eth0 is assigned to an externally-accessible static IP address which accesses the Internet, while eth1 acts as a routing point to process and transmit LAN packets from one network node to the remote network nodes.

The IPsec connection between each network uses a pre-shared key with the value of **`r3dh4tl1nux`**, and the administrators of A and B agree to let **`racoon`** automatically generate and share an authentication key between each IPsec router. The administrator of LAN A decides to name the IPsec connection **`ipsec0`**, while the administrator of LAN B names the IPsec connection **`ipsec1`**.

The following example shows the contents of the **`ifcfg`** file for a network-to-network IPsec connection for LAN A. The unique name to identify the connection in this example is *ipsec0*, so the resulting file is called **`/etc/sysconfig/network-scripts/ifcfg-ipsec0`**.

```
TYPE=IPSEC
ONBOOT=yes
IKE_METHOD=PSK
SRCGW=192.168.1.254
DSTGW=192.168.2.254
SRCNET=192.168.1.0/24
DSTNET=192.168.2.0/24
DST=X.X.X.X
```

The following list describes the contents of this file:

TYPE=IPSEC

Specifies the type of connection.

ONBOOT=yes

Specifies that the connection should initiate on boot-up.

IKE_METHOD=PSK

Specifies that the connection uses the pre-shared key method of authentication.

SRCGW=192.168.1.254

The IP address of the source gateway. For LAN A, this is the LAN A gateway, and for LAN B, the LAN B gateway.

DSTGW=192.168.2.254

The IP address of the destination gateway. For LAN A, this is the LAN B gateway, and for LAN B, the LAN A gateway.

SRCNET=192.168.1.0/24

Specifies the source network for the IPsec connection, which in this example is the network range for LAN A.

DSTNET=192.168.2.0/24

Specifies the destination network for the IPsec connection, which in this example is the network range for LAN B.

DST=X.X.X.X

The externally-accessible IP address of LAN B.

The following example is the content of the pre-shared key file called **`/etc/sysconfig/network-scripts/keys-ipsecX`** (where *X* is 0 for LAN A and 1 for LAN B) that both networks use to authenticate each other. The contents of this file should be identical and only the root user should be able to read or write this file.

```
IKE_PSK=r3dh4tl1nux
```

Important

To change the **`keys-ipsecX`** file so that only the root user can read or edit the file, use the following command after creating the file:

```
chmod 600 /etc/sysconfig/network-scripts/keys-ipsec1
```

To change the authentication key at any time, edit the **`keys-ipsecX`** file on both IPsec routers. *Both keys must be identical for proper connectivity*.

The following example is the contents of the **`/etc/racoon/racoon.conf`** configuration file for the IPsec connection. Note that the **`include`** line at the bottom of the file is automatically generated and only appears i the IPsec tunnel is running.

```
# Racoon IKE daemon configuration file.
# See 'man racoon.conf' for a description of the format and entries.
path include "/etc/racoon";
path pre_shared_key "/etc/racoon/psk.txt";
path certificate "/etc/racoon/certs";

sainfo anonymous
{
 pfs_group 2;
 lifetime time 1 hour ;
 encryption_algorithm 3des, blowfish 448, rijndael ;
 authentication_algorithm hmac_sha1, hmac_md5 ;
 compression_algorithm deflate ;
}
include "/etc/racoon/X.X.X.X.conf"
```

The following is the specific configuration for the connection to the remote network. The file is called **`X.X.X.X.conf`** (where *X.X.X.X* is the IP address of the remote IPsec router). Note that this file is automatically generated when the IPsec tunnel is activated and should not be edited directly.

```
remote X.X.X.X
{
        exchange_mode aggressive, main;
 my_identifier address;
 proposal {
  encryption_algorithm 3des;
  hash_algorithm sha1;
```

```
    authentication_method pre_shared_key;
    dh_group 2 ;
  }
}
```

Prior to starting the IPsec connection, IP forwarding should be enabled in the kernel. To enable IP forwarding:

1. Edit **`/etc/sysctl.conf`** and set **`net.ipv4.ip_forward`** to **`1`**.

2. Use the following command to enable the change:

   ```
   sysctl -p /etc/sysctl.conf
   ```

To start the IPsec connection, use the following command on each router:

```
ifup ipsec0
```

The connections are activated, and both LAN A and LAN B are able to communicate with each other. The routes are created automatically via the initialization script called by running **`ifup`** on the IPsec connection. To show a list of routes for the network, use the following command:

```
ip route list
```

To test the IPsec connection, run the **`tcpdump`** utility on the externally-routable device (eth0 in this example) to view the network packets being transferred between the hosts (or networks), and verify that they are encrypted via IPsec. For example, to check the IPsec connectivity of LAN A, use the following command:

```
tcpdump -n -i eth0 host lana.example.com
```

The packet should include an AH header and should be shown as ESP packets. ESP means it is encrypted. For example (back slashes denote a continuation of one line):

```
12:24:26.155529 lanb.example.com > lana.example.com:
AH(spi=0x021c9834,seq=0x358): \
 lanb.example.com > lana.example.com: ESP(spi=0x00c887ad,seq=0x358) (DF) \
 (ipip-proto-4)
```

48.7.8. Starting and Stopping an IPsec Connection

If the IPsec connection was not configured to activate on boot, you can control it from the command line.

To start the connection, use the following command on each host for host-to-host IPsec, or each IPsec router for network-to-network IPsec:

```
ifup <nickname>
```

where <*nickname*> is the nickname configured earlier, such as **`ipsec0`**.

To stop the connection, use the following command:

```
ifdown <nickname>
```

48.8. Firewalls

48.8. Firewalls

Information security is commonly thought of as a process and not a product. However, standard security implementations usually employ some form of dedicated mechanism to control access privileges and restrict network resources to users who are authorized, identifiable, and traceable. Red Hat Enterprise Linux includes several tools to assist administrators and security engineers with network-level access control issues.

Firewalls are one of the core components of a network security implementation. Several vendors market firewall solutions catering to all levels of the marketplace: from home users protecting one PC to data center solutions safeguarding vital enterprise information. Firewalls can be stand-alone hardware solutions, such as firewall appliances by Cisco, Nokia, and Sonicwall. Vendors such as Checkpoint, McAfee, and Symantec have also developed proprietary software firewall solutions for home and business markets.

Apart from the differences between hardware and software firewalls, there are also differences in the way firewalls function that separate one solution from another. Table 48.5, “Firewall Types” details three common types of firewalls and how they function:

Table 48.5. Firewall Types

Method	Description	Advantages	Disadvantages
NAT	*Network Address Translation* (NAT) places private IP subnetworks behind one or a small pool of public IP addresses, masquerading all requests to one source rather than several. The Linux kernel has built-in NAT functionality through the Netfilter kernel subsystem.	· Can be configured transparently to machines on a LAN · Protection of many machines and services behind one or more external IP addresses simplifies administration duties · Restriction of user access to and from the LAN can be configured by opening and closing ports on the NAT firewall/gateway	· Cannot prevent malicious activity once users connect to a service outside of the firewall
Packet Filter	A packet filtering firewall reads each data packet that passes through a LAN. It can read and process packets by header information and filters the packet based on sets of programmable rules implemented by the firewall administrator. The Linux kernel has built-in packet filtering functionality through the Netfilter kernel subsystem.	· Customizable through the `iptables` front-end utility · Does not require any customization on the client side, as all network activity is filtered at the router level rather than the application level · Since packets are not transmitted through a proxy, network performance is faster due to direct connection from client to remote host	· Cannot filter packets for content like proxy firewalls · Processes packets at the protocol layer, but cannot filter packets at an application layer · Complex network architectures can make establishing packet filtering rules difficult, especially if coupled with *IP masquerading* or local subnets and DMZ networks

Method	Description	Advantages	Disadvantages
Proxy	Proxy firewalls filter all requests of a certain protocol or type from LAN clients to a proxy machine, which then makes those requests to the Internet on behalf of the local client. A proxy machine acts as a buffer between malicious remote users and the internal network client machines.	· Gives administrators control over what applications and protocols function outside of the LAN · Some proxy servers can cache frequently-accessed data locally rather than having to use the Internet connection to request it. This helps to reduce bandwidth consumption · Proxy services can be logged and monitored closely, allowing tighter control over resource utilization on the network	· Proxies are often application-specific (HTTP, Telnet, etc.), or protocol-restricted (most proxies work with TCP-connected services only) · Application services cannot run behind a proxy, so your application servers must use a separate form of network security · Proxies can become a network bottleneck, as all requests and transmissions are passed through one source rather than directly from a client to a remote service

48.8.1. Netfilter and IPTables

The Linux kernel features a powerful networking subsystem called *Netfilter*. The Netfilter subsystem provides stateful or stateless packet filtering as well as NAT and IP masquerading services. Netfilter also has the ability to *mangle* IP header information for advanced routing and connection state management. Netfilter is controlled using the **`iptables`** tool.

48.8.1.1. IPTables Overview

The power and flexibility of Netfilter is implemented using the **`iptables`** administration tool, a command line tool similar in syntax to its predecessor, **`ipchains`**.

A similar syntax does not mean similar implementation, however. **`ipchains`** requires intricate rule sets for: filtering source paths; filtering destination paths; and filtering both source and destination connection ports.

By contrast, **`iptables`** uses the Netfilter subsystem to enhance network connection, inspection, and processing. **`iptables`** features advanced logging, pre- and post-routing actions, network address translation, and port forwarding, all in one command line interface.

This section provides an overview of **`iptables`**. For more detailed information, refer to Section 48.9, “IPTables”.

48.8.2. Basic Firewall Configuration

Just as a firewall in a building attempts to prevent a fire from spreading, a computer firewall attempts to prevent malicious software from spreading to your computer. It also helps to prevent unauthorized users from accessing your computer.

In a default Red Hat Enterprise Linux installation, a firewall exists between your computer or network and any untrusted networks, for example the Internet. It determines which services on your computer remote users can access. A properly configured firewall can greatly increase the security of your system. It is recommended that you configure a firewall for any Red Hat Enterprise Linux system with an Internet connection.

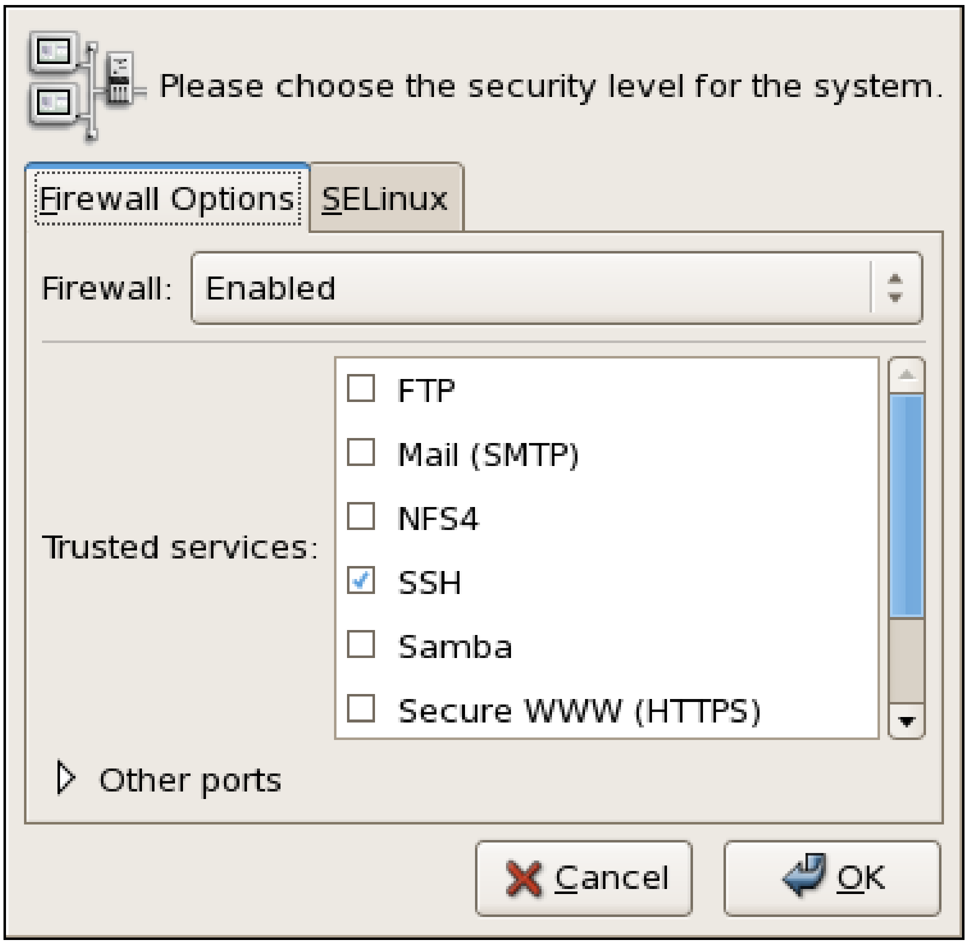
48.8.2.1. Security Level Configuration Tool

During the **`Firewall Configuration`** screen of the Red Hat Enterprise Linux installation, you were given the option to enable a basic firewall as well as to allow specific devices, incoming services, and ports.

After installation, you can change this preference by using the **Security Level Configuration Tool**.

To start this application, use the following command:

```
system-config-securitylevel
```


Please choose the security level for the system.

Firewall: Enabled

Trusted services:
- [ ] FTP
- [ ] Mail (SMTP)
- [ ] NFS4
- [x] SSH
- [ ] Samba
- [ ] Secure WWW (HTTPS)

Figure 48.15. Security Level Configuration Tool

Note

The **Security Level Configuration Tool** only configures a basic firewall. If the system needs more complex rules, refer to Section 48.9, "IPTables" for details on configuring specific **`iptables`** rules.

48.8.2.2. Enabling and Disabling the Firewall

Select one of the following options for the firewall:

- **`Disabled`** — Disabling the firewall provides complete access to your system and does no security checking. This should only be selected if you are running on a trusted network (not the Internet) or need to configure a custom firewall using the iptables command line tool.

> **Warning**
>
> Firewall configurations and any customized firewall rules are stored in the **`/etc/sysconfig/iptables`** file. If you choose **`Disabled`** and click **`OK`**, these configurations and firewall rules will be lost.

- **`Enabled`** — This option configures the system to reject incoming connections that are not in response to outbound requests, such as DNS replies or DHCP requests. If access to services running on this machine is needed, you can choose to allow specific services through the firewall.

  If you are connecting your system to the Internet, but do not plan to run a server, this is the safest choice.

48.8.2.3. Trusted Services

Enabling options in the **`Trusted services`** list allows the specified service to pass through the firewall.

`WWW (HTTP)`

The HTTP protocol is used by Apache (and by other Web servers) to serve web pages. If you plan on making your Web server publicly available, select this check box. This option is not required for viewing pages locally or for developing web pages. This service requires that the **`httpd`** package be installed.

Enabling **`WWW (HTTP)`** will not open a port for HTTPS, the SSL version of HTTP. If this service is required, select the **`Secure WWW (HTTPS)`** check box.

`FTP`

The FTP protocol is used to transfer files between machines on a network. If you plan on making your FTP server publicly available, select this check box. This service requires that the **`vsftpd`** package be installed.

`SSH`

Secure Shell (SSH) is a suite of tools for logging into and executing commands on a remote machine. To allow remote access to the machine via ssh, select this check box. This service requires that the **`openssh-server`** package be installed.

`Telnet`

Telnet is a protocol for logging into remote machines. Telnet communications are unencrypted and provide no security from network snooping. Allowing incoming Telnet access is not recommended. To allow remote access to the machine via telnet, select this check box. This service requires that the **`telnet-server`** package be installed.

`Mail (SMTP)`

SMTP is a protocol that allows remote hosts to connect directly to your machine to deliver mail. You do not need to enable this service if you collect your mail from your ISP's server using POP3 or IMAP, or if you use a tool such as **`fetchmail`**. To allow delivery of mail to your machine, select this check box. Note that an improperly configured SMTP server can allow remote machines to use your server to send spam.

NFS4

The Network File System (NFS) is a file sharing protocol commonly used on *NIX systems. Version 4 of this protocol is more secure than its predecessors. If you want to share files or directories on your system with other network users, select this check box.

Samba

Samba is an implementation of Microsoft's proprietary SMB networking protocol. If you need to share files, directories, or locally-connected printers with Microsoft Windows machines, select this check box.

48.8.2.4. Other Ports

The **Security Level Configuration Tool** includes an **`Other ports`** section for specifying custom IP ports as being trusted by **`iptables`**. For example, to allow IRC and Internet printing protocol (IPP) to pass through the firewall, add the following to the **`Other ports`** section:

`194:tcp,631:tcp`

48.8.2.5. Saving the Settings

Click **`OK`** to save the changes and enable or disable the firewall. If **`Enable firewall`** was selected, the options selected are translated to **`iptables`** commands and written to the **`/etc/sysconfig/iptables`** file. The **`iptables`** service is also started so that the firewall is activated immediately after saving the selected options. If **`Disable firewall`** was selected, the **`/etc/sysconfig/iptables`** file is removed and the **`iptables`** service is stopped immediately.

The selected options are also written to the **`/etc/sysconfig/system-config-securitylevel`** file so that the settings can be restored the next time the application is started. Do not edit this file by hand.

Even though the firewall is activated immediately, the **`iptables`** service is not configured to start automatically at boot time. Refer to Section 48.8.2.6, “Activating the IPTables Service” for more information.

48.8.2.6. Activating the IPTables Service

The firewall rules are only active if the **`iptables`** service is running. To manually start the service, use the following command:

```
service iptables restart
```

To ensure that **`iptables`** starts when the system is booted, use the following command:

```
chkconfig --level 345 iptables on
```

The **`ipchains`** service is not included in Red Hat Enterprise Linux. However, if **`ipchains`** is installed (for example, an upgrade was performed and the system had **`ipchains`** previously installed), the **`ipchains`** and **`iptables`** services should not be activated simultaneously. To make sure the **`ipchains`** service is disabled and configured not to start at boot time, use the following two commands:

```
service ipchains stop
chkconfig --level 345 ipchains off
```

48.8.3. Using IPTables

The first step in using **`iptables`** is to start the **`iptables`** service. Use the following command to start the **`iptables`** service:

```
service iptables start
```

Note

The **`ip6tables`** service can be turned off if you intend to use the **`iptables`** service only. If you deactivate the **`ip6tables`** service, remember to deactivate the IPv6 network also. Never leave a network device active without the matching firewall.

To force **`iptables`** to start by default when the system is booted, use the following command:

```
chkconfig --level 345 iptables on
```

This forces **`iptables`** to start whenever the system is booted into runlevel 3, 4, or 5.

48.8.3.1. IPTables Command Syntax

The following sample **`iptables`** command illustrates the basic command syntax:

```
iptables -A <chain> -j <target>
```

The **`-A`** option specifies that the rule be appended to <*chain*>. Each chain is comprised of one or more *rules*, and is therefore also known as a *ruleset*.

The three built-in chains are INPUT, OUTPUT, and FORWARD. These chains are permanent and cannot be deleted. The chain specifies the point at which a packet is manipulated.

The **`-j <target>`** option specifies the target of the rule; i.e., what to do if the packet matches the rule. Examples of built-in targets are ACCEPT, DROP, and REJECT.

Refer to the **`iptables`** man page for more information on the available chains, options, and targets.

48.8.3.2. Basic Firewall Policies

Establishing basic firewall policies creates a foundation for building more detailed, user-defined rules.

Each **`iptables`** chain is comprised of a default policy, and zero or more rules which work in concert with the default policy to define the overall ruleset for the firewall.

The default policy for a chain can be either DROP or ACCEPT. Security-minded administrators typically implement a default policy of DROP, and only allow specific packets on a case-by-case basis. For example, the following policies block all incoming and outgoing packets on a network gateway:

```
iptables -P INPUT DROP
iptables -P OUTPUT DROP
```

It is also recommended that any *forwarded packets* — network traffic that is to be routed from the firewall to its destination node — be denied as well, to restrict internal clients from inadvertent exposure to the Internet. To do this, use the following rule:

```
iptables -P FORWARD DROP
```

When you have established the default policies for each chain, you can create and save further rules for your particular network and security requirements.

The following sections describe how to save iptables rules and outline some of the rules you might implement in the course of building your iptables firewall.

48.8.3.3. Saving and Restoring IPTables Rules

Changes to **`iptables`** are transitory; if the system is rebooted or if the **`iptables`** service is restarted, the rules are automatically flushed and reset. To save the rules so that they are loaded when the **`iptables`** service is started, use the following command:

```
service iptables save
```

The rules are stored in the file **`/etc/sysconfig/iptables`** and are applied whenever the service is started or the machine is rebooted.

48.8.4. Common IPTables Filtering

Preventing remote attackers from accessing a LAN is one of the most important aspects of network security. The integrity of a LAN should be protected from malicious remote users through the use of stringent firewall rules.

However, with a default policy set to block all incoming, outgoing, and forwarded packets, it is impossible for the firewall/gateway and internal LAN users to communicate with each other or with external resources.

To allow users to perform network-related functions and to use networking applications, administrators must open certain ports for communication.

For example, to allow access to port 80 *on the firewall*, append the following rule:

```
iptables -A INPUT -p tcp -m tcp --dport 80 -j ACCEPT
```

This allows users to browse websites that communicate using the standard port 80. To allow access to secure websites (for example, https://www.example.com/), you also need to provide access to port 443, as follows:

```
iptables -A INPUT -p tcp -m tcp --dport 443 -j ACCEPT
```

Important

When creating an **`iptables`** ruleset, order is important.

If a rule specifies that any packets from the 192.168.100.0/24 subnet be dropped, and this is followed by a rule that allows packets from 192.168.100.13 (which is within the dropped subnet), then the second rule is ignored.

The rule to allow packets from 192.168.100.13 must precede the rule that drops the remainder of the subnet.

To insert a rule in a specific location in an existing chain, use the **`-I`** option. For example:

```
iptables -I INPUT 1 -i lo -p all -j ACCEPT
```

This rule is inserted as the first rule in the INPUT chain to allow local loopback device traffic.

There may be times when you require remote access to the LAN. Secure services, for example SSH, can be used for encrypted remote connection to LAN services.

Administrators with PPP-based resources (such as modem banks or bulk ISP accounts), dial-up access can be used to securely circumvent firewall barriers. Because they are direct connections, modem connections are typically behind a firewall/gateway.

For remote users with broadband connections, however, special cases can be made. You can configure **`iptables`** to accept connections from remote SSH clients. For example, the following rules allow remote SSH access:

```
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
iptables -A OUTPUT -p tcp --sport 22 -j ACCEPT
```

These rules allow incoming and outbound access for an individual system, such as a single PC directly connected to the Internet or a firewall/gateway. However, they do not allow nodes behind the firewall/gateway to access these services. To allow LAN access to these services, you can use *Network Address Translation* (NAT) with **`iptables`** filtering rules.

48.8.5. `FORWARD` and NAT Rules

Most ISPs provide only a limited number of publicly routable IP addresses to the organizations they serve.

Administrators must, therefore, find alternative ways to share access to Internet services without giving public IP addresses to every node on the LAN. Using private IP addresses is the most common way of allowing all nodes on a LAN to properly access internal and external network services.

Edge routers (such as firewalls) can receive incoming transmissions from the Internet and route the packets to the intended LAN node. At the same time, firewalls/gateways can also route outgoing requests from a LAN node to the remote Internet service.

This forwarding of network traffic can become dangerous at times, especially with the availability of modern cracking tools that can spoof *internal* IP addresses and make the remote attacker's machine act as a node on your LAN.

To prevent this, **`iptables`** provides routing and forwarding policies that can be implemented to prevent abnormal usage of network resources.

The **FORWARD** chain allows an administrator to control where packets can be routed within a LAN. For example, to allow forwarding for the entire LAN (assuming the firewall/gateway is assigned an internal IP address on eth1), use the following rules:

```
iptables -A FORWARD -i eth1 -j ACCEPT
iptables -A FORWARD -o eth1 -j ACCEPT
```

This rule gives systems behind the firewall/gateway access to the internal network. The gateway routes packets from one LAN node to its intended destination node, passing all packets through its **eth1** device.

Note

By default, the IPv4 policy in Red Hat Enterprise Linux kernels disables support for IP forwarding. This prevents machines that run Red Hat Enterprise Linux from functioning as dedicated edge routers. To enable IP forwarding, use the following command:

```
sysctl -w net.ipv4.ip_forward=1
```

This configuration change is only valid for the current session; it does not persist beyond a reboot or network service restart. To permanently set IP forwarding, edit the **/etc/sysctl.conf** file as follows:

Locate the following line:

```
net.ipv4.ip_forward = 0
```

Edit it to read as follows:

```
net.ipv4.ip_forward = 1
```

Use the following command to enable the change to the **sysctl.conf** file:

```
sysctl -p /etc/sysctl.conf
```

48.8.5.1. Postrouting and IP Masquerading

Accepting forwarded packets via the firewall's internal IP device allows LAN nodes to communicate with each other; however they still cannot communicate externally to the Internet.

To allow LAN nodes with private IP addresses to communicate with external public networks, configure the firewall for *IP masquerading*, which masks requests from LAN nodes with the IP address of the firewall's external device (in this case, eth0):

```
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

This rule uses the NAT packet matching table (**-t nat**) and specifies the built-in POSTROUTING chain for NAT (**-A POSTROUTING**) on the firewall's external networking device (**-o eth0**).

POSTROUTING allows packets to be altered as they are leaving the firewall's external device.

The **-j MASQUERADE** target is specified to mask the private IP address of a node with the external IP address of the firewall/gateway.

48.8.5.2. Prerouting

If you have a server on your internal network that you want make available externally, you can use the **-j DNAT** target of the PREROUTING chain in NAT to specify a destination IP address and port where incoming packets requesting a connection to your internal service can be forwarded.

For example, if you want to forward incoming HTTP requests to your dedicated Apache HTTP Server at 172.31.0.23, use the following command:

```
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j DNAT --to
172.31.0.23:80
```

This rule specifies that the nat table use the built-in PREROUTING chain to forward incoming HTTP requests exclusively to the listed destination IP address of 172.31.0.23.

Note

If you have a default policy of DROP in your FORWARD chain, you must append a rule to forward all incoming HTTP requests so that destination NAT routing is possible. To do this, use the following command:

```
iptables -A FORWARD -i eth0 -p tcp --dport 80 -d 172.31.0.23 -j ACCEPT
```

This rule forwards all incoming HTTP requests from the firewall to the intended destination; the Apache HTTP Server behind the firewall.

48.8.5.3. DMZs and IPTables

You can create **iptables** rules to route traffic to certain machines, such as a dedicated HTTP or FTP server, in a *demilitarized zone* (DMZ). A DMZ is a special local subnetwork dedicated to providing services on a public carrier, such as the Internet.

For example, to set a rule for routing incoming HTTP requests to a dedicated HTTP server at 10.0.4.2 (outside of the 192.168.1.0/24 range of the LAN), NAT uses the **PREROUTING** table to forward the packets to the appropriate destination:

```
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j DNAT --to-
destination 10.0.4.2:80
```

With this command, all HTTP connections to port 80 from outside of the LAN are routed to the HTTP server on a network separate from the rest of the internal network. This form of network segmentation can prove safer than allowing HTTP connections to a machine on the network.

If the HTTP server is configured to accept secure connections, then port 443 must be forwarded as well.

48.8.6. Malicious Software and Spoofed IP Addresses

More elaborate rules can be created that control access to specific subnets, or even specific nodes, within a LAN. You can also restrict certain dubious applications or programs such as Trojans, worms, and other client/server viruses from contacting their server.

For example, some Trojans scan networks for services on ports from 31337 to 31340 (called the *elite* ports in cracking terminology).

Since there are no legitimate services that communicate via these non-standard ports, blocking them can effectively diminish the chances that potentially infected nodes on your network independently communicate with their remote master servers.

The following rules drop all TCP traffic that attempts to use port 31337:

```
iptables -A OUTPUT -o eth0 -p tcp --dport 31337 --sport 31337 -j DROP
iptables -A FORWARD -o eth0 -p tcp --dport 31337 --sport 31337 -j DROP
```

You can also block outside connections that attempt to spoof private IP address ranges to infiltrate your LAN.

For example, if your LAN uses the 192.168.1.0/24 range, you can design a rule that instructs the Internet-facing network device (for example, eth0) to drop any packets to that device with an address in your LAN IP range.

Because it is recommended to reject forwarded packets as a default policy, any other spoofed IP address to the external-facing device (eth0) is rejected automatically.

```
iptables -A FORWARD -s 192.168.1.0/24 -i eth0 -j DROP
```

Note

There is a distinction between the **`DROP`** and **`REJECT`** targets when dealing with *appended* rules.

The **`REJECT`** target denies access and returns a **`connection refused`** error to users who attempt to connect to the service. The **`DROP`** target, as the name implies, drops the packet without any warning.

Administrators can use their own discretion when using these targets. However, to avoid user confusion and attempts to continue connecting, the **`REJECT`** target is recommended.

48.8.7. IPTables and Connection Tracking

You can inspect and restrict connections to services based on their *connection state.* A module within **`iptables`** uses a method called *connection tracking* to store information about incoming connections. You can allow or deny access based on the following connection states:

- **`NEW`** — A packet requesting a new connection, such as an HTTP request.
- **`ESTABLISHED`** — A packet that is part of an existing connection.
- **`RELATED`** — A packet that is requesting a new connection but is part of an existing connection. For example, FTP uses port 21 to establish a connection, but data is transferred on a different port (typically port 20).
- **`INVALID`** — A packet that is not part of any connections in the connection tracking table.

You can use the stateful functionality of **`iptables`** connection tracking with any network protocol, even if the protocol itself is stateless (such as UDP). The following example shows a rule that uses connection tracking to forward only the packets that are associated with an established connection:

```
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
```

48.8.8. IPv6

The introduction of the next-generation Internet Protocol, called IPv6, expands beyond the 32-bit address limit of IPv4 (or IP). IPv6 supports 128-bit addresses, and carrier networks that are IPv6 aware are therefore able to address a larger number of routable addresses than IPv4.

Red Hat Enterprise Linux supports IPv6 firewall rules using the Netfilter 6 subsystem and the **`ip6tables`** command. In Red Hat Enterprise Linux 5, both IPv4 and IPv6 services are enabled by default.

The **`ip6tables`** command syntax is identical to **`iptables`** in every aspect except that it supports 128-bit addresses. For example, use the following command to enable SSH connections on an IPv6-aware network server:

```
ip6tables -A INPUT -i eth0 -p tcp -s 3ffe:ffff:100::1/128 --dport 22 -j
ACCEPT
```

For more information about IPv6 networking, refer to the IPv6 Information Page at http://www.ipv6.org/.

48.8.9. Additional Resources

There are several aspects to firewalls and the Linux Netfilter subsystem that could not be covered in this chapter. For more information, refer to the following resources.

48.8.9.1. Installed Documentation

- Refer to Section 48.9, “IPTables” for more detailed information on the **`iptables`** command, including definitions for many command options.
- The **`iptables`** man page contains a brief summary of the various options.

48.8.9.2. Useful Websites

- http://www.netfilter.org/ — The official homepage of the Netfilter and **`iptables`** project.
- http://www.tldp.org/ — The Linux Documentation Project contains several useful guides relating to firewall creation and administration.
- http://www.iana.org/assignments/port-numbers — The official list of registered and common service ports as assigned by the Internet Assigned Numbers Authority.

48.8.9.3. Related Documentation

- *Red Hat Linux Firewalls*, by Bill McCarty; Red Hat Press — a comprehensive reference to building network and server firewalls using open source packet filtering technology such as Netfilter and **`iptables`**. It includes topics that cover analyzing firewall logs, developing firewall rules, and customizing your firewall using various graphical tools.
- *Linux Firewalls*, by Robert Ziegler; New Riders Press — contains a wealth of information on building firewalls using both 2.2 kernel **`ipchains`** as well as Netfilter and **`iptables`**. Additional security topics

such as remote access issues and intrusion detection systems are also covered.

48.9. IPTables

Included with Red Hat Enterprise Linux are advanced tools for network *packet filtering* — the process of controlling network packets as they enter, move through, and exit the network stack within the kernel. Kernel versions prior to 2.4 relied on **`ipchains`** for packet filtering and used lists of rules applied to packets at each step of the filtering process. The 2.4 kernel introduced **`iptables`** (also called *netfilter*), which is similar to **`ipchains`** but greatly expands the scope and control available for filtering network packets.

This chapter focuses on packet filtering basics, defines the differences between **`ipchains`** and **`iptables`**, explains various options available with **`iptables`** commands, and explains how filtering rules can be preserved between system reboots.

Refer to Section 48.9.7, "Additional Resources" for instructions on how to construct **`iptables`** rules and setting up a firewall based on these rules.

Warning

The default firewall mechanism in the 2.4 and later kernels is **`iptables`**, but **`iptables`** cannot be used if **`ipchains`** is already running. If **`ipchains`** is present at boot time, the kernel issues an error and fails to start **`iptables`**.

The functionality of **`ipchains`** is not affected by these errors.

48.9.1. Packet Filtering

The Linux kernel uses the **Netfilter** facility to filter packets, allowing some of them to be received by or pass through the system while stopping others. This facility is built in to the Linux kernel, and has three built-in *tables* or *rules lists*, as follows:

- **`filter`** — The default table for handling network packets.
- **`nat`** — Used to alter packets that create a new connection and used for *Network Address Translation* (*NAT*).
- **`mangle`** — Used for specific types of packet alteration.

Each table has a group of built-in *chains*, which correspond to the actions performed on the packet by **`netfilter`**.

The built-in chains for the **`filter`** table are as follows:

- *INPUT* — Applies to network packets that are targeted for the host.
- *OUTPUT* — Applies to locally-generated network packets.
- *FORWARD* — Applies to network packets routed through the host.

The built-in chains for the **`nat`** table are as follows:

- *PREROUTING* — Alters network packets when they arrive.
- *OUTPUT* — Alters locally-generated network packets before they are sent out.

- *POSTROUTING* — Alters network packets before they are sent out.

The built-in chains for the **`mangle`** table are as follows:

- *INPUT* — Alters network packets targeted for the host.
- *OUTPUT* — Alters locally-generated network packets before they are sent out.
- *FORWARD* — Alters network packets routed through the host.
- *PREROUTING* — Alters incoming network packets before they are routed.
- *POSTROUTING* — Alters network packets before they are sent out.

Every network packet received by or sent from a Linux system is subject to at least one table. However, a packet may be subjected to multiple rules within each table before emerging at the end of the chain. The structure and purpose of these rules may vary, but they usually seek to identify a packet coming from or going to a particular IP address, or set of addresses, when using a particular protocol and network service.

Note

By default, firewall rules are saved in the **`/etc/sysconfig/iptables`** or **`/etc/sysconfig/ip6tables`** files.

The **`iptables`** service starts before any DNS-related services when a Linux system is booted. This means that firewall rules can only reference numeric IP addresses (for example, 192.168.0.1). Domain names (for example, host.example.com) in such rules produce errors.

Regardless of their destination, when packets match a particular rule in one of the tables, a *target* or action is applied to them. If the rule specifies an **`ACCEPT`** target for a matching packet, the packet skips the rest of the rule checks and is allowed to continue to its destination. If a rule specifies a **`DROP`** target, that packet is refused access to the system and nothing is sent back to the host that sent the packet. If a rule specifies a **`QUEUE`** target, the packet is passed to user-space. If a rule specifies the optional**`REJECT`** target, the packet is dropped, but an error packet is sent to the packet's originator.

Every chain has a default policy to **`ACCEPT`**, **`DROP`**, **`REJECT`**, or **`QUEUE`**. If none of the rules in the chain apply to the packet, then the packet is dealt with in accordance with the default policy.

The **`iptables`** command configures these tables, as well as sets up new tables if necessary.

48.9.2. Differences Between IPTables and IPChains

Both **`ipchains`** and **`iptables`** use chains of rules that operate within the Linux kernel to filter packets based on matches with specified rules or rule sets. However, **`iptables`** offers a more extensible way of filtering packets, giving the administrator greater control without building undue complexity into the system.

You should be aware of the following significant differences between **`ipchains`** and **`iptables`**:

Using `iptables`, each filtered packet is processed using rules from only one chain rather than multiple chains.

For example, a FORWARD packet coming into a system using **`ipchains`** would have to go through the INPUT, FORWARD, and OUTPUT chains to continue to its destination. However, **`iptables`** only sends packets to the INPUT chain if they are destined for the local system, and only sends them to the OUTPUT chain if the local system generated the packets. It is therefore important to place the rule designed to catch a particular packet within the chain that actually handles the packet.

The DENY target has been changed to DROP.

In **ipchains**, packets that matched a rule in a chain could be directed to the DENY target. This target must be changed to DROP in **iptables**.

Order matters when placing options in a rule.

In **ipchains**, the order of the rule options does not matter.

The **iptables** command has a stricter syntax. The **iptables** command requires that the protocol (ICMP, TCP, or UDP) be specified before the source or destination ports.

Network interfaces must be associated with the correct chains in firewall rules.

For example, incoming interfaces (**-i** option) can only be used in INPUT or FORWARD chains. Similarly, outgoing interfaces (**-o** option) can only be used in FORWARD or OUTPUT chains.

In other words, INPUT chains and incoming interfaces work together; OUTPUT chains and outgoing interfaces work together. FORWARD chains work with both incoming and outgoing interfaces.

OUTPUT chains are no longer used by incoming interfaces, and INPUT chains are not seen by packets moving through outgoing interfaces.

This is not a comprehensive list of the changes. Refer to Section 48.9.7, “Additional Resources” for more specific information.

48.9.3. Command Options for IPTables

Rules for filtering packets are created using the **iptables** command. The following aspects of the packet are most often used as criteria:

- *Packet Type* — Specifies the type of packets the command filters.
- *Packet Source/Destination* — Specifies which packets the command filters based on the source or destination of the packet.
- *Target* — Specifies what action is taken on packets matching the above criteria.

Refer to Section 48.9.3.4, “IPTables Match Options” and Section 48.9.3.5, “Target Options” for more information about specific options that address these aspects of a packet.

The options used with specific **iptables** rules must be grouped logically, based on the purpose and conditions of the overall rule, for the rule to be valid. The remainder of this section explains commonly-used options for the **iptables** command.

48.9.3.1. Structure of IPTables Command Options

Many **iptables** commands have the following structure:

```
iptables [-t <table-name>] <command> <chain-name> \
   <parameter-1> <option-1> \
   <parameter-n> <option-n>
```

<table-name> — Specifies which table the rule applies to. If omitted, the **filter** table is used.

<command> — Specifies the action to perform, such as appending or deleting a rule.

<chain-name> — Specifies the chain to edit, create, or delete.

<parameter>-<option> pairs — Parameters and associated options that specify how to process a packet that matches the rule.

The length and complexity of an **`iptables`** command can change significantly, based on its purpose.

For example, a command to remove a rule from a chain can be very short:

```
iptables -D <chain-name> <line-number>
```

In contrast, a command that adds a rule which filters packets from a particular subnet using a variety of specific parameters and options can be rather long. When constructing **`iptables`** commands, it is important to remember that some parameters and options require further parameters and options to construct a valid rule. This can produce a cascading effect, with the further parameters requiring yet more parameters. Until every parameter and option that requires another set of options is satisfied, the rule is not valid.

Type **`iptables -h`** to view a comprehensive list of **`iptables`** command structures.

48.9.3.2. Command Options

Command options instruct **`iptables`** to perform a specific action. Only one command option is allowed per **`iptables`** command. With the exception of the help command, all commands are written in upper-case characters.

The **`iptables`** commands are as follows:

- **`-A`** — Appends the rule to the end of the specified chain. Unlike the **`-I`** option described below, it does not take an integer argument. It always appends the rule to the end of the specified chain.
- **`-C`** — Checks a particular rule before adding it to the user-specified chain. This command can help you construct complex **`iptables`** rules by prompting you for additional parameters and options.
- **`-D <integer> | <rule>`** — Deletes a rule in a particular chain by number (such as **`5`** for the fifth rule in a chain), or by rule specification. The rule specification must exactly match an existing rule.
- **`-E`** — Renames a user-defined chain. A user-defined chain is any chain other than the default, pre-existing chains. (Refer to the **`-N`** option, below, for information on creating user-defined chains.) This is a cosmetic change and does not affect the structure of the table.

  Note

  If you attempt to rename one of the default chains, the system reports a **`Match not found`** error. You cannot rename the default chains.

- **`-F`** — Flushes the selected chain, which effectively deletes every rule in the chain. If no chain is specified, this command flushes every rule from every chain.
- **`-h`** — Provides a list of command structures, as well as a quick summary of command parameters and options.
- **`-I [<integer>]`** — Inserts the rule in the specified chain at a point specified by a user-defined integer argument. If no argument is specified, the rule is inserted at the top of the chain.

Warning

As noted above, the order of rules in a chain determines which rules apply to which packets. This is important to remember when adding rules using either the **`-A`** or **`-I`** option.

This is especially important when adding rules using the **`-I`** with an integer argument. If you specify an existing number when adding a rule to a chain, **`iptables`** adds the new rule *before* (or above) the existing rule.

- **`-L`** — Lists all of the rules in the chain specified after the command. To list all rules in all chains in the default **`filter`** table, do not specify a chain or table. Otherwise, the following syntax should be used to list the rules in a specific chain in a particular table:

  ```
  iptables -L <chain-name> -t <table-name>
  ```

  Additional options for the **`-L`** command option, which provide rule numbers and allow more verbose rule descriptions, are described in Section 48.9.3.6, “Listing Options”.

- **`-N`** — Creates a new chain with a user-specified name. The chain name must be unique, otherwise an error message is displayed.

- **`-P`** — Sets the default policy for the specified chain, so that when packets traverse an entire chain without matching a rule, they are sent to the specified target, such as ACCEPT or DROP.

- **`-R`** — Replaces a rule in the specified chain. The rule's number must be specified after the chain's name. The first rule in a chain corresponds to rule number one.

- **`-X`** — Deletes a user-specified chain. You cannot delete a built-in chain.

- **`-Z`** — Sets the byte and packet counters in all chains for a table to zero.

48.9.3.3. IPTables Parameter Options

Certain **`iptables`** commands, including those used to add, append, delete, insert, or replace rules within a particular chain, require various parameters to construct a packet filtering rule.

- **`-c`** — Resets the counters for a particular rule. This parameter accepts the **`PKTS`** and **`BYTES`** options to specify which counter to reset.

- **`-d`** — Sets the destination hostname, IP address, or network of a packet that matches the rule. When matching a network, the following IP address/netmask formats are supported:

  - ***`N.N.N.N/M.M.M.M`*** — Where *N.N.N.N* is the IP address range and *M.M.M.M* is the netmask.

  - ***`N.N.N.N/M`*** — Where *N.N.N.N* is the IP address range and *M* is the bitmask.

- **`-f`** — Applies this rule only to fragmented packets.

  You can use the exclamation point character (**`!`**) option after this parameter to specify that only unfragmented packets are matched.

> **Note**
>
> Distinguishing between fragmented and unfragmented packets is desirable, despite fragmented packets being a standard part of the IP protocol.
>
> Originally designed to allow IP packets to travel over networks with differing frame sizes, these days fragmentation is more commonly used to generate DoS attacks using mal-formed packets. It's also worth noting that IPv6 disallows fragmentation entirely.

- **`-i`** — Sets the incoming network interface, such as **`eth0`** or **`ppp0`**. With **`iptables`**, this optional parameter may only be used with the INPUT and FORWARD chains when used with the **`filter`** table and the PREROUTING chain with the **`nat`** and **`mangle`** tables.

  This parameter also supports the following special options:

  - Exclamation point character (**`!`**) — Reverses the directive, meaning any specified interfaces are excluded from this rule.
  - Plus character (**`+`**) — A wildcard character used to match all interfaces that match the specified string. For example, the parameter **`-i eth+`** would apply this rule to any Ethernet interfaces but exclude any other interfaces, such as **`ppp0`**.

  If the **`-i`** parameter is used but no interface is specified, then every interface is affected by the rule.

- **`-j`** — Jumps to the specified target when a packet matches a particular rule.

  The standard targets are **`ACCEPT`**, **`DROP`**, **`QUEUE`**, and **`RETURN`**.

  Extended options are also available through modules loaded by default with the Red Hat Enterprise Linux **`iptables`** RPM package. Valid targets in these modules include **`LOG`**, **`MARK`**, and **`REJECT`**, among others. Refer to the **`iptables`** man page for more information about these and other targets.

  This option can also be used to direct a packet matching a particular rule to a user-defined chain outside of the current chain so that other rules can be applied to the packet.

  If no target is specified, the packet moves past the rule with no action taken. The counter for this rule, however, increases by one.

- **`-o`** — Sets the outgoing network interface for a rule. This option is only valid for the OUTPUT and FORWARD chains in the **`filter`** table, and the POSTROUTING chain in the **`nat`** and **`mangle`** tables. This parameter accepts the same options as the incoming network interface parameter (**`-i`**).

- **`-p <protocol>`** — Sets the IP protocol affected by the rule. This can be either **`icmp`**, **`tcp`**, **`udp`**, or **`all`**, or it can be a numeric value, representing one of these or a different protocol. You can also use any protocols listed in the **`/etc/protocols`** file.

  The "**`all`**" protocol means the rule applies to every supported protocol. If no protocol is listed with this rule, it defaults to "**`all`**".

- **`-s`** — Sets the source for a particular packet using the same syntax as the destination (**`-d`**) parameter.

48.9.3.4. IPTables Match Options

Different network protocols provide specialized matching options which can be configured to match a particular packet using that protocol. However, the protocol must first be specified in the **`iptables`** command. For example, **-p *<protocol-name>*** enables options for the specified protocol. Note that you can also use the protocol ID, instead of the protocol name. Refer to the following examples, each of which have the same effect:

```
iptables -A INPUT -p icmp --icmp-type any -j ACCEPT
iptables -A INPUT -p 5813 --icmp-type any -j ACCEPT
```

Service definitions are provided in the **/etc/services** file. For readability, it is recommended that you use the service names rather than the port numbers.

Important

Secure the **/etc/services** file to prevent unauthorized editing. If this file is editable, crackers can use it to enable ports on your machine you have otherwise closed. To secure this file, type the following commands as root:

```
chown root.root /etc/services
chmod 0644 /etc/services
chattr +i /etc/services
```

This prevents the file from being renamed, deleted or having links made to it.

48.9.3.4.1. TCP Protocol

These match options are available for the TCP protocol (**-p tcp**):

- **--dport** — Sets the destination port for the packet.

  To configure this option, use a network service name (such as www or smtp); a port number; or a range of port numbers.

  To specify a range of port numbers, separate the two numbers with a colon (**:**). For example: **-p tcp --dport 3000:3200**. The largest acceptable valid range is **0:65535**.

  Use an exclamation point character (**!**) after the **--dport** option to match all packets that *do not* use that network service or port.

  To browse the names and aliases of network services and the port numbers they use, view the **/etc/services** file.

  The **--destination-port** match option is synonymous with **--dport**.

- **--sport** — Sets the source port of the packet using the same options as **--dport**. The **--source-port** match option is synonymous with **--sport**.

- **--syn** — Applies to all TCP packets designed to initiate communication, commonly called *SYN packets*. Any packets that carry a data payload are not touched.

  Use an exclamation point character (**!**) after the **--syn** option to match all non-SYN packets.

- **--tcp-flags <tested flag list> <set flag list>** — Allows TCP packets that have specific bits (flags) set, to match a rule.

The **`--tcp-flags`** match option accepts two parameters. The first parameter is the mask; a comma-separated list of flags to be examined in the packet. The second parameter is a comma-separated list of flags that must be set for the rule to match.

The possible flags are:

- **`ACK`**
- **`FIN`**
- **`PSH`**
- **`RST`**
- **`SYN`**
- **`URG`**
- **`ALL`**
- **`NONE`**

For example, an **`iptables`** rule that contains the following specification only matches TCP packets that have the SYN flag set and the ACK and FIN flags not set:

`--tcp-flags ACK,FIN,SYN SYN`

Use the exclamation point character (**`!`**) after the **`--tcp-flags`** to reverse the effect of the match option.

- **`--tcp-option`** — Attempts to match with TCP-specific options that can be set within a particular packet. This match option can also be reversed with the exclamation point character (**`!`**).

48.9.3.4.2. UDP Protocol

These match options are available for the UDP protocol (**`-p udp`**):

- **`--dport`** — Specifies the destination port of the UDP packet, using the service name, port number, or range of port numbers. The **`--destination-port`** match option is synonymous with **`--dport`**.
- **`--sport`** — Specifies the source port of the UDP packet, using the service name, port number, or range of port numbers. The **`--source-port`** match option is synonymous with **`--sport`**.

For the **`--dport`** and **`--sport`** options, to specify a range of port numbers, separate the two numbers with a colon (:). For example: **`-p tcp --dport 3000:3200`**. The largest acceptable valid range is 0:65535.

48.9.3.4.3. ICMP Protocol

The following match options are available for the Internet Control Message Protocol (ICMP) (**`-p icmp`**):

- **`--icmp-type`** — Sets the name or number of the ICMP type to match with the rule. A list of valid ICMP names can be retrieved by typing the **`iptables -p icmp -h`** command.

48.9.3.4.4. Additional Match Option Modules

Additional match options are available through modules loaded by the **`iptables`** command.

To use a match option module, load the module by name using the **`-m <module-name>`**, where *<module-name>* is the name of the module.

Many modules are available by default. You can also create modules to provide additional functionality.

The following is a partial list of the most commonly used modules:

- **`limit`** module — Places limits on how many packets are matched to a particular rule.

  When used in conjunction with the **`LOG`** target, the **`limit`** module can prevent a flood of matching packets from filling up the system log with repetitive messages or using up system resources.

  Refer to Section 48.9.3.5, “Target Options” for more information about the **`LOG`** target.

  The **`limit`** module enables the following options:

  - **`--limit`** — Sets the maximum number of matches for a particular time period, specified as a ***`<value>/<period>`*** pair. For example, using **`--limit 5/hour`** allows five rule matches per hour.

    Periods can be specified in seconds, minutes, hours, or days.

    If a number and time modifier are not used, the default value of **`3/hour`** is assumed.

  - **`--limit-burst`** — Sets a limit on the number of packets able to match a rule at one time.

    This option is specified as an integer and should be used in conjunction with the **`--limit`** option.

    If no value is specified, the default value of five (5) is assumed.

- **`state`** module — Enables state matching.

  The **`state`** module enables the following options:

  - **`--state`** — match a packet with the following connection states:

    - **`ESTABLISHED`** — The matching packet is associated with other packets in an established connection. You need to accept this state if you want to maintain a connection between a client and a server.

    - **`INVALID`** — The matching packet cannot be tied to a known connection.

    - **`NEW`** — The matching packet is either creating a new connection or is part of a two-way connection not previously seen. You need to accept this state if you want to allow new connections to a service.

    - **`RELATED`** — The matching packet is starting a new connection related in some way to an existing connection. An example of this is FTP, which uses one connection for control traffic (port 21), and a separate connection for data transfer (port 20).

    These connection states can be used in combination with one another by separating them with commas, such as **`-m state --state INVALID,NEW`**.

- **`mac`** module — Enables hardware MAC address matching.

  The **`mac`** module enables the following option:

  - **`--mac-source`** — Matches a MAC address of the network interface card that sent the packet. To exclude a MAC address from a rule, place an exclamation point character (**`!`**) after the **`--mac-source`** match option.

Refer to the **`iptables`** man page for more match options available through modules.

48.9.3.5. Target Options

48.9.3.5. Target Options

When a packet has matched a particular rule, the rule can direct the packet to a number of different targets which determine the appropriate action. Each chain has a default target, which is used if none of the rules on that chain match a packet or if none of the rules which match the packet specify a target.

The following are the standard targets:

- ***`<user-defined-chain>`*** — A user-defined chain within the table. User-defined chain names must be unique. This target passes the packet to the specified chain.
- **`ACCEPT`** — Allows the packet through to its destination or to another chain.
- **`DROP`** — Drops the packet without responding to the requester. The system that sent the packet is not notified of the failure.
- **`QUEUE`** — The packet is queued for handling by a user-space application.
- **`RETURN`** — Stops checking the packet against rules in the current chain. If the packet with a**`RETURN`** target matches a rule in a chain called from another chain, the packet is returned to the first chain to resume rule checking where it left off. If the **`RETURN`** rule is used on a built-in chain and the packet cannot move up to its previous chain, the default target for the current chain is used.

In addition, extensions are available which allow other targets to be specified. These extensions are called target modules or match option modules and most only apply to specific tables and situations. Refer to Section 48.9.3.4.4, “Additional Match Option Modules” for more information about match option modules.

Many extended target modules exist, most of which only apply to specific tables or situations. Some of the most popular target modules included by default in Red Hat Enterprise Linux are:

- **`LOG`** — Logs all packets that match this rule. Because the packets are logged by the kernel, the **`/etc/syslog.conf`** file determines where these log entries are written. By default, they are placed in the **`/var/log/messages`** file.

  Additional options can be used after the **`LOG`** target to specify the way in which logging occurs:

  - **`--log-level`** — Sets the priority level of a logging event. Refer to the **`syslog.conf`** man page for a list of priority levels.
  - **`--log-ip-options`** — Logs any options set in the header of an IP packet.
  - **`--log-prefix`** — Places a string of up to 29 characters before the log line when it is written. This is useful for writing syslog filters for use in conjunction with packet logging.

    Note

    Due to an issue with this option, you should add a trailing space to the *log-prefix* value.

  - **`--log-tcp-options`** — Logs any options set in the header of a TCP packet.
  - **`--log-tcp-sequence`** — Writes the TCP sequence number for the packet in the log.
- **`REJECT`** — Sends an error packet back to the remote system and drops the packet.

The **REJECT** target accepts **--reject-with *<type>*** (where *<type>* is the rejection type) allowing more detailed information to be returned with the error packet. The message **port-unreachable** is the default error type given if no other option is used. Refer to the **iptables** man page for a full list of ***<type>*** options.

Other target extensions, including several that are useful for IP masquerading using the **nat** table, or with packet alteration using the **mangle** table, can be found in the **iptables** man page.

48.9.3.6. Listing Options

The default list command, **iptables -L [<chain-name>]**, provides a very basic overview of the default filter table's current chains. Additional options provide more information:

- **-v** — Displays verbose output, such as the number of packets and bytes each chain has processed, the number of packets and bytes each rule has matched, and which interfaces apply to a particular rule.
- **-x** — Expands numbers into their exact values. On a busy system, the number of packets and bytes processed by a particular chain or rule may be abbreviated to **Kilobytes**, **Megabytes** (Megabytes) or **Gigabytes**. This option forces the full number to be displayed.
- **-n** — Displays IP addresses and port numbers in numeric format, rather than the default hostname and network service format.
- **--line-numbers** — Lists rules in each chain next to their numeric order in the chain. This option is useful when attempting to delete the specific rule in a chain or to locate where to insert a rule within a chain.
- **-t <table-name>** — Specifies a table name. If omitted, defaults to the filter table.

The following examples illustrate the use of several of these options. Note the difference in the byte display by including the **-x** option.

```
~]# iptables -L OUTPUT -v -n -x
Chain OUTPUT (policy ACCEPT 64005 packets, 6445791 bytes)
    pkts      bytes target     prot opt in     out     source
destination
    1593   133812 ACCEPT     icmp --  *      *       0.0.0.0/0
0.0.0.0/0

~]# iptables -L OUTPUT -v -n
Chain OUTPUT (policy ACCEPT 64783 packets, 6492K bytes)
    pkts bytes target     prot opt in     out     source
destination
    1819  153K ACCEPT     icmp --  *      *       0.0.0.0/0
0.0.0.0/0
~]#
```

48.9.4. Saving IPTables Rules

Rules created with the **iptables** command are stored in memory. If the system is restarted before saving the **iptables** rule set, all rules are lost. For netfilter rules to persist through a system reboot, they need to be saved. To save netfilter rules, type the following command as root:

```
service iptables save
```

This executes the **iptables** init script, which runs the **/sbin/iptables-save** program and writes the current **iptables** configuration to **/etc/sysconfig/iptables**. The existing **/etc/sysconfig/iptables** file is saved as **/etc/sysconfig/iptables.save**.

The next time the system boots, the **iptables** init script reapplies the rules saved in **/etc/sysconfig/iptables** by using the **/sbin/iptables-restore** command.

While it is always a good idea to test a new **iptables** rule before committing it to the **/etc/sysconfig/iptables** file, it is possible to copy **iptables** rules into this file from another system's version of this file. This provides a quick way to distribute sets of **iptables** rules to multiple machines.

You can also save the iptables rules to a separate file for distribution, backup or other purposes. To save your iptables rules, type the following command as root:

```
iptables-save > <filename>
```

where <*filename*> is a user-defined name for your ruleset.

Important

If distributing the **/etc/sysconfig/iptables** file to other machines, type **/sbin/service iptables restart** for the new rules to take effect.

Note

Note the difference between the **iptables** *command* (**/sbin/iptables**), which is used to manipulate the tables and chains that constitute the **iptables** functionality, and the **iptables** *service* (**/sbin/iptables service**), which is used to enable and disable the **iptables** service itself.

48.9.5. IPTables Control Scripts

There are two basic methods for controlling **iptables** in Red Hat Enterprise Linux:

- **Security Level Configuration Tool** (**system-config-securitylevel**) — A graphical interface for creating, activating, and saving basic firewall rules. Refer to Section 48.8.2, “Basic Firewall Configuration” for more information.
- **/sbin/service iptables <*option*>** — Used to manipulate various functions of **iptables** using its initscript. The following options are available:
  - **start** — If a firewall is configured (that is, **/etc/sysconfig/iptables** exists), all running **iptables** are stopped completely and then started using the **/sbin/iptables-restore** command. This option only works if the **ipchains** kernel module is not loaded. To check if this module is loaded, type the following command as root:

    ```
    lsmod | grep ipchains
    ```

    If this command returns no output, it means the module is not loaded. If necessary, use the **/sbin/rmmod** command to remove the module.

- **`stop`** — If a firewall is running, the firewall rules in memory are flushed, and all iptables modules and helpers are unloaded.

  If the **`IPTABLES_SAVE_ON_STOP`** directive in the **`/etc/sysconfig/iptables-config`** configuration file is changed from its default value to **`yes`**, current rules are saved to **`/etc/sysconfig/iptables`** and any existing rules are moved to the file **`/etc/sysconfig/iptables.save`**.

  Refer to Section 48.9.5.1, "IPTables Control Scripts Configuration File" for more information about the **`iptables-config`** file.

- **`restart`** — If a firewall is running, the firewall rules in memory are flushed, and the firewall is started again if it is configured in **`/etc/sysconfig/iptables`**. This option only works if the **`ipchains`** kernel module is not loaded.

  If the **`IPTABLES_SAVE_ON_RESTART`** directive in the **`/etc/sysconfig/iptables-config`** configuration file is changed from its default value to **`yes`**, current rules are saved to **`/etc/sysconfig/iptables`** and any existing rules are moved to the file **`/etc/sysconfig/iptables.save`**.

  Refer to Section 48.9.5.1, "IPTables Control Scripts Configuration File" for more information about the **`iptables-config`** file.

- **`status`** — Displays the status of the firewall and lists all active rules.

  The default configuration for this option displays IP addresses in each rule. To display domain and hostname information, edit the **`/etc/sysconfig/iptables-config`** file and change the value of **`IPTABLES_STATUS_NUMERIC`** to **no**. Refer to Section 48.9.5.1, "IPTables Control Scripts Configuration File" for more information about the **`iptables-config`** file.

- **`panic`** — Flushes all firewall rules. The policy of all configured tables is set to **`DROP`**.

  This option could be useful if a server is known to be compromised. Rather than physically disconnecting from the network or shutting down the system, you can use this option to stop all further network traffic but leave the machine in a state ready for analysis or other forensics.

- **`save`** — Saves firewall rules to **`/etc/sysconfig/iptables`** using **`iptables-save`**. Refer to Section 48.9.4, "Saving IPTables Rules" for more information.

Note

To use the same initscript commands to control netfilter for IPv6, substitute **`ip6tables`** for **`iptables`** in the **`/sbin/service`** commands listed in this section. For more information about IPv6 and netfilter, refer to Section 48.9.6, "IPTables and IPv6".

48.9.5.1. IPTables Control Scripts Configuration File

The behavior of the **`iptables`** initscripts is controlled by the **`/etc/sysconfig/iptables-config`** configuration file. The following is a list of directives contained in this file:

- **`IPTABLES_MODULES`** — Specifies a space-separated list of additional **`iptables`** modules to load when a firewall is activated. These can include connection tracking and NAT helpers.

- **`IPTABLES_MODULES_UNLOAD`** — Unloads modules on restart and stop. This directive accepts the following values:

  - **yes** — The default value. This option must be set to achieve a correct state for a firewall restart or stop.
  - **no** — This option should only be set if there are problems unloading the netfilter modules.

- **IPTABLES_SAVE_ON_STOP** — Saves current firewall rules to **/etc/sysconfig/iptables** when the firewall is stopped. This directive accepts the following values:
  - **yes** — Saves existing rules to **/etc/sysconfig/iptables** when the firewall is stopped, moving the previous version to the **/etc/sysconfig/iptables.save** file.
  - **no** — The default value. Does not save existing rules when the firewall is stopped.
- **IPTABLES_SAVE_ON_RESTART** — Saves current firewall rules when the firewall is restarted. This directive accepts the following values:
  - **yes** — Saves existing rules to **/etc/sysconfig/iptables** when the firewall is restarted, moving the previous version to the **/etc/sysconfig/iptables.save** file.
  - **no** — The default value. Does not save existing rules when the firewall is restarted.
- **IPTABLES_SAVE_COUNTER** — Saves and restores all packet and byte counters in all chains and rules. This directive accepts the following values:
  - **yes** — Saves the counter values.
  - **no** — The default value. Does not save the counter values.
- **IPTABLES_STATUS_NUMERIC** — Outputs IP addresses in numeric form instead of domain or hostnames. This directive accepts the following values:
  - **yes** — The default value. Returns only IP addresses within a status output.
  - **no** — Returns domain or hostnames within a status output.

48.9.6. IPTables and IPv6

If the **iptables-ipv6** package is installed, netfilter in Red Hat Enterprise Linux can filter the next-generation IPv6 Internet protocol. The command used to manipulate the IPv6 netfilter is **ip6tables**.

Most directives for this command are identical to those used for **iptables**, except the **nat** table is not yet supported. This means that it is not yet possible to perform IPv6 network address translation tasks, such as masquerading and port forwarding.

Rules for **ip6tables** are saved in the **/etc/sysconfig/ip6tables** file. Previous rules saved by the **ip6tables** initscripts are saved in the **/etc/sysconfig/ip6tables.save** file.

Configuration options for the **ip6tables** init script are stored in **/etc/sysconfig/ip6tables-config**, and the names for each directive vary slightly from their **iptables** counterparts.

For example, the **iptables-config** directive **IPTABLES_MODULES**:the equivalent in the **ip6tables-config** file is **IP6TABLES_MODULES**.

48.9.7. Additional Resources

Refer to the following sources for additional information on packet filtering with **iptables**.

- Section 48.8, “Firewalls” — Contains a chapter about the role of firewalls within an overall security strategy as well as strategies for constructing firewall rules.

48.9.7.1. Installed Documentation

- **`man iptables`** — Contains a description of **`iptables`** as well as a comprehensive list of targets, options, and match extensions.

48.9.7.2. Useful Websites

- http://www.netfilter.org/ — The home of the netfilter/iptables project. Contains assorted information about **`iptables`**, including a FAQ addressing specific problems and various helpful guides by Rusty Russell, the Linux IP firewall maintainer. The HOWTO documents on the site cover subjects such as basic networking concepts, kernel packet filtering, and NAT configurations.

- http://www.linuxnewbie.org/nhf/Security/IPtables_Basics.html — An introduction to the way packets move through the Linux kernel, plus an introduction to constructing basic **`iptables`** commands.

---

[14] Since system BIOSes differ between manufacturers, some may not support password protection of either type, while others may support one type but not the other.

[15] GRUB also accepts unencrypted passwords, but it is recommended that an MD5 hash be used for added security.

[16] This access is still subject to the restrictions imposed by SELinux, if it is enabled.

[17] A system where both the client and the server share a common key that is used to encrypt and decrypt network communication.

Chapter 49. Security and SELinux

49.1. Access Control Mechanisms (ACMs)

This section provides a basic introduction to *Access Control Mechanisms* (ACMs). ACMs provide a means for system administrators to control which users and processes can access different files, devices, interfaces, etc., in a computer system. This is a primary consideration when securing a computer system or network of any size.

49.1.1. Discretionary Access Control (DAC)

Discretionary Access Control (DAC) defines the basic access controls for objects in a filesystem. This is the typical access control provided by file permissions, sharing, etc. Such access is generally at the discretion of the owner of the object (file, directory, device, etc.).

DAC provides a means of restricting access to objects based on the identity of the users or groups (subjects) that try to access those objects. Depending on a subject's access permissions, they may also be able to pass permissions to other subjects.

49.1.2. Access Control Lists (ACLs)

Access Control Lists (ACLs) provide further control over which objects a subject can access. For more information, refer to Chapter 10, *Access Control Lists*.

49.1.3. Mandatory Access Control (MAC)

Mandatory Access Control (MAC) is a security mechanism that restricts the level of control that users (subjects) have over the objects that they create. Unlike in a DAC implementation, where users have full control over their own files, directories, etc., MAC adds additional labels, or categories, to all file system objects. Users and processes must have the appropriate access to these categories before they can interact with these objects.

In Red Hat Enterprise Linux, MAC is enforced by SELinux. For more information, refer to Section 49.2, “Introduction to SELinux”.

49.1.4. Role-based Access Control (RBAC)

Role-based Access Control (RBAC) is an alternative method of controlling user access to file system objects. Instead of access being controlled by user permissions, the system administrator establishes *Roles* based on business functional requirements or similar criteria. These Roles have different types and levels of access to objects.

In contrast to DAC or MAC systems, where users have access to objects based on their own and the object's permissions, users in an RBAC system must be members of the appropriate group, or Role, before they can interact with files, directories, devices, etc.

From an administrative point of view, this makes it easier to control who has access to various parts of the file system, just by controlling their group memberships.

49.1.5. Multi-Level Security (MLS)

Multi-Level Security (MLS) is a specific Mandatory Access Control (MAC) security scheme. Under this scheme, processes are called Subjects. Files, sockets and other passive operating system entities are called Objects. For more information, refer to Section 49.6, “Multi-Level Security (MLS)”.

49.1.6. Multi-Category Security (MCS)

Multi-Category Security (MCS) is an enhancement to SELinux, and allows users to label files with categories. MCS is an adaptation of MLS and re-uses much of the MLS framework in SELinux. For more information, refer to Section 49.4.1, “Introduction”

49.2. Introduction to SELinux

Security-Enhanced Linux (SELinux) is a security architecture integrated into the 2.6.x kernel using the *Linux Security Modules* (LSM). It is a project of the United States National Security Agency (NSA) and the SELinux community. SELinux integration into Red Hat Enterprise Linux was a joint effort between the NSA and Red Hat.

49.2.1. SELinux Overview

SELinux provides a flexible *Mandatory Access Control* (MAC) system built into the Linux kernel. Under standard Linux *Discretionary Access Control* (DAC), an application or process running as a user (UID or SUID) has the user's permissions to objects such as files, sockets, and other processes. Running a MAC kernel protects the system from malicious or flawed applications that can damage or destroy the system.

SELinux defines the access and transition rights of every user, application, process, and file on the system. SELinux then governs the interactions of these entities using a security policy that specifies how strict or lenient a given Red Hat Enterprise Linux installation should be.

On a day-to-day basis, system users will be largely unaware of SELinux. Only system administrators need to consider how strict a policy to implement for their server environment. The policy can be as strict or as lenient as needed, and is very finely detailed. This detail gives the SELinux kernel complete, granular control over the entire system.

The SELinux Decision Making Process

When a subject, (for example, an application), attempts to access an object (for example, a file), the policy enforcement server in the kernel checks an *access vector cache* (AVC), where subject and object permissions are cached. If a decision cannot be made based on data in the AVC, the request continues to the security server, which looks up the *security context* of the application and the file in a matrix. Permission is then granted or denied, with an **`avc: denied`** message detailed in **`/var/log/messages`** if permission is denied. The security context of subjects and objects is applied from the installed policy, which also provides the information to populate the security server's matrix.

Refer to the following diagram:

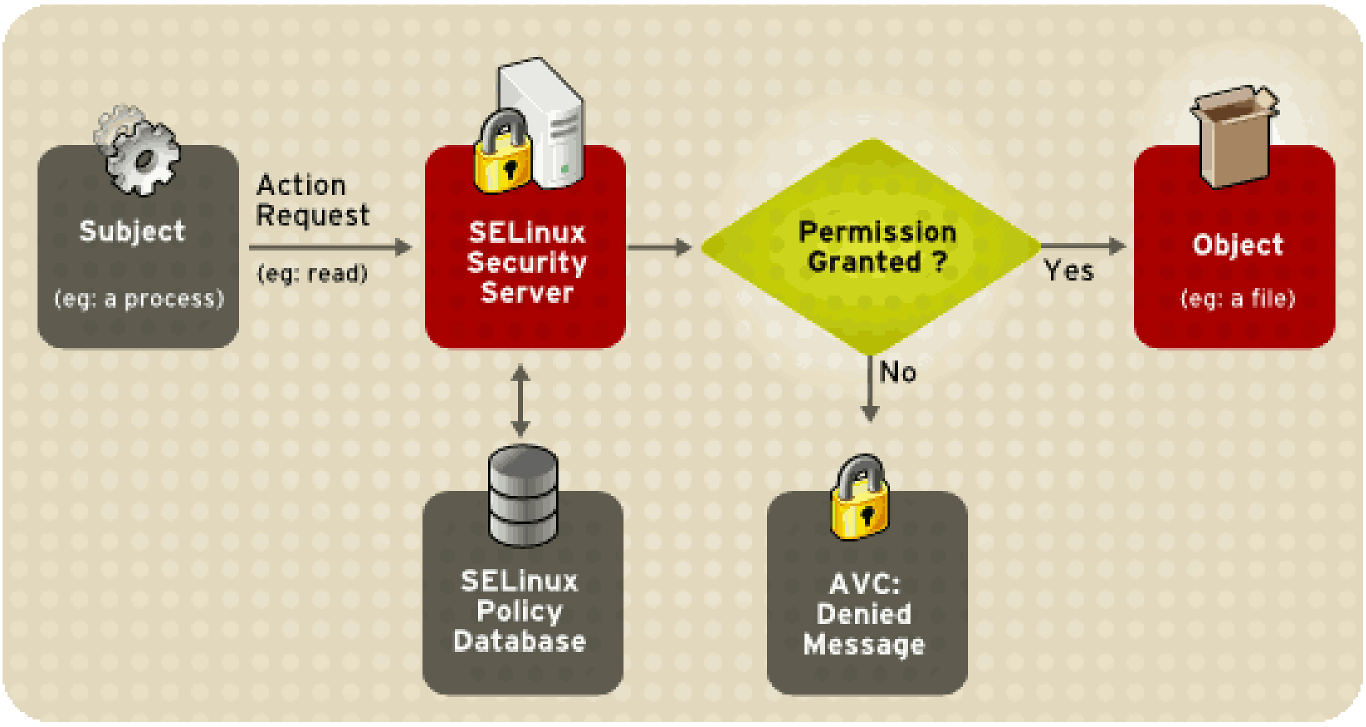


Figure 49.1. SELinux Decision Process

SELinux Operating Modes

Instead of running in *enforcing* mode, SELinux can run in *permissive* mode, where the AVC is checked and denials are logged, but SELinux does not enforce the policy. This can be useful for troubleshooting and for developing or fine-tuning SELinux policy.

For more information about how SELinux works, refer to Section 49.2.3, “Additional Resources”.

49.2.2. Files Related to SELinux

The following sections describe SELinux configuration files and related file systems.

49.2.2.1. The SELinux Pseudo-File System

The **`/selinux/`** pseudo-file system contains commands that are most commonly used by the kernel subsystem. This type of file system is similar to the **`/proc/`** pseudo-file system.

Administrators and users do not normally need to manipulate this component.

The following example shows sample contents of the **`/selinux/`** directory:

```
-rw-rw-rw-  1 root root 0 Sep 22 13:14 access
dr-xr-xr-x  1 root root 0 Sep 22 13:14 booleans
--w-------  1 root root 0 Sep 22 13:14 commit_pending_bools
-rw-rw-rw-  1 root root 0 Sep 22 13:14 context
-rw-rw-rw-  1 root root 0 Sep 22 13:14 create
--w-------  1 root root 0 Sep 22 13:14 disable
-rw-r--r--  1 root root 0 Sep 22 13:14 enforce
-rw-------  1 root root 0 Sep 22 13:14 load
```

```
-r--r--r--  1 root root 0 Sep 22 13:14 mls
-r--r--r--  1 root root 0 Sep 22 13:14 policyvers
-rw-rw-rw-  1 root root 0 Sep 22 13:14 relabel
-rw-rw-rw-  1 root root 0 Sep 22 13:14 user
```

For example, running the **`cat`** command on the **`enforce`** file reveals either a **`1`** for enforcing mode or **`0`** for permissive mode.

49.2.2.2. SELinux Configuration Files

The following sections describe SELinux configuration and policy files, and related file systems located in the **`/etc/`** directory.

49.2.2.2.1. The `/etc/sysconfig/selinux` Configuration File

There are two ways to configure SELinux under Red Hat Enterprise Linux: using the **SELinux Administration Tool** (**`system-config-selinux`**), or manually editing the configuration file (**`/etc/sysconfig/selinux`**).

The **`/etc/sysconfig/selinux`** file is the primary configuration file for enabling or disabling SELinux, as well as for setting which policy to enforce on the system and how to enforce it.

Note

The **`/etc/sysconfig/selinux`** contains a symbolic link to the actual configuration file, **`/etc/selinux/config`**.

The following explains the full subset of options available for configuration:

- **`SELINUX=`*`enforcing|permissive|disabled`*** — Defines the top-level state of SELinux on a system.

  - **`enforcing`** — The SELinux security policy is enforced.

  - **`permissive`** — The SELinux system prints warnings but does not enforce policy.

    This is useful for debugging and troubleshooting purposes. In permissive mode, more denials are logged because subjects can continue with actions that would otherwise be denied in enforcing mode. For example, traversing a directory tree in permissive mode produces **`avc: denied`** messages for every directory level read. In enforcing mode, SELinux would have stopped the initial traversal and kept further denial messages from occurring.

  - **`disabled`** — SELinux is fully disabled. SELinux hooks are disengaged from the kernel and the pseudo-file system is unregistered.

Note

Actions made while SELinux is disabled may result in the file system no longer having the correct security context. That is, the security context defined by the policy. The best way to relabel the file system is to create the flag file **`/.autorelabel`** and reboot the machine. This causes the relabel to occur very early in the boot process, before any processes are running on the system. Using this procedure means that processes can not accidentally create files in the wrong context or start up in the wrong context.

It is possible to use the **`fixfiles relabel`** command prior to enabling SELinux to relabel the file system. This method is not recommended, however, because after it is complete, it is still possible to have processes potentially running on the system in the wrong context. These processes could create files that would also be in the wrong context.

Note

Additional white space at the end of a configuration line or as extra lines at the end of the file may cause unexpected behavior. To be safe, remove unnecessary white space.

- **`SELINUXTYPE=targeted|strict`** — Specifies which policy SELinux should enforce.
  - **`targeted`** — Only targeted network daemons are protected.

Important

The following daemons are protected in the default targeted policy: **`dhcpd, httpd (apache.te), named, nscd, ntpd, portmap, snmpd, squid`**, and **`syslogd`**. The rest of the system runs in the unconfined_t domain. This domain allows subjects and objects with that security context to operate using standard Linux security.

The policy files for these daemons are located in **`/etc/selinux/targeted/src/policy/domains/program`**. These files are subject to change as newer versions of Red Hat Enterprise Linux are released.

Policy enforcement for these daemons can be turned on or off, using Boolean values controlled by the **SELinux Administration Tool** (**`system-config-selinux`**).

Setting a Boolean value for a targeted daemon to **`1`** disables SELinux protection for the daemon. For example, you can set **`dhcpd_disable_trans`** to **`1`** to prevent **`init`**, which executes apps labeled **`dhcpd_exec_t`**, from transitioning to the **`dhcpd_t`** domain.

Use the **`getsebool -a`** command to list all SELinux booleans. The following is an example of using the **`setsebool`** command to set an SELinux boolean. The **`-P`** option makes the change permanent. Without this option, the boolean would be reset to **`1`** at reboot.

```
setsebool -P dhcpd_disable_trans=0
```

  - **`strict`** — Full SELinux protection, for all daemons. Security contexts are defined for all subjects and objects, and every action is processed by the policy enforcement server.

- **`SETLOCALDEFS=0|1`** — Controls how local definitions (users and booleans) are set. Set this value to 1 to have these definitions controlled by **`load_policy`** from files in **`/etc/selinux/<`*`policyname`*`>`**. or set it to 0 to have them controlled by **`semanage`**.

> **Warning**
>
> You should not change this value from the default (0) unless you are fully aware of the impact of such a change.

49.2.2.2.2. The `/etc/selinux/` Directory

The **`/etc/selinux/`** directory is the primary location for all policy files as well as the main configuration file.

The following example shows sample contents of the **`/etc/selinux/`** directory:

```
-rw-r--r--  1 root root  448 Sep 22 17:34 config
drwxr-xr-x  5 root root 4096 Sep 22 17:27 strict
drwxr-xr-x  5 root root 4096 Sep 22 17:28 targeted
```

The two subdirectories, **`strict/`** and **`targeted/`**, are the specific directories where the policy files of the same name (that is, **`strict`** and **`targeted`**) are contained.

49.2.2.3. SELinux Utilities

The following are some of the commonly used SELinux utilities:

- **`/usr/sbin/setenforce`** — Modifies in real-time the mode in which SELinux runs.

  For example:

  `setenforce 1` — SELinux runs in enforcing mode.

  `setenforce 0` — SELinux runs in permissive mode.

  To actually disable SELinux, you need to either specify the appropriate **`setenforce`** parameter in **`/etc/sysconfig/selinux`** or pass the parameter **`selinux=0`** to the kernel, either in **`/etc/grub.conf`** or at boot time.

- **`/usr/sbin/sestatus -v`** — Displays the detailed status of a system running SELinux. The following example shows an excerpt of **`sestatus -v`** output:

```
SELinux status:                 enabled
SELinuxfs mount:                /selinux
Current mode:                   enforcing
Mode from config file:          enforcing
Policy version:                 21
Policy from config file:        targeted
```

```
Process contexts:
Current context:                user_u:system_r:unconfined_t:s0
Init context:                   system_u:system_r:init_t:s0
/sbin/mingetty                  system_u:system_r:getty_t:s0
```

- **`/usr/bin/newrole`** — Runs a new shell in a new context, or role. Policy must allow the transition to the new role.

> **Note**
>
> This command is only available if you have the **`policycoreutils-newrole`** package installed, which is required for the strict and MLS policies.

- **`/sbin/restorecon`** — Sets the security context of one or more files by marking the extended attributes with the appropriate file or security context.
- **`/sbin/fixfiles`** — Checks or corrects the security context database on the file system.

Refer to the man page associated with these utilities for more information.

Refer to the **`setools`** or **`policycoreutils`** package contents for more information on all available binary utilities. To view the contents of a package, use the following command:

`rpm -ql <`*`package-name`*`>`

49.2.3. Additional Resources

Refer to the following resources for more detailed information on SELinux.

49.2.3.1. Installed Documentation

- **`/usr/share/doc/setools-<`*`version-number`*`>/`** All documentation for utilities contained in the **`setools`** package. This includes all helper scripts, sample configuration files, and documentation.

49.2.3.2. Useful Websites

- http://www.nsa.gov/research/selinux/index.shtml Homepage for the NSA SELinux development team. Many resources are available in HTML and PDF formats. Although many of these links are not SELinux specific, some concepts may apply.
- http://docs.fedoraproject.org/ Homepage for the Fedora documentation project, which contains Fedora Core specific materials that may be more timely, since the release cycle is much shorter.
- http://selinux.sourceforge.net Homepage for the SELinux community.

49.3. Brief Background and History of SELinux

SELinux was originally a development project from the National Security Agency (*NSA*) [18] and others. It is an implementation of the *Flask* operating system security architecture. [19] The NSA integrated SELinux into the Linux kernel using the *Linux Security Modules* (*LSM*) framework. SELinux motivated the creation of LSM, at the suggestion of Linus Torvalds, who wanted a modular approach to security instead of just accepting SELinux into the kernel.

Originally, the SELinux implementation used *persistent security IDs* (PSIDs) stored in an unused field of the ext2 inode. These numerical representations (i.e., non-human-readable) were mapped by SELinux to a security context label. Unfortunately, this required modifying each file system type to support PSIDs, so was not a scalable solution or one that would be supported upstream in the Linux kernel.

The next evolution of SELinux was as a loadable kernel module for the 2.4.<*x*> series of Linux kernels. This module stored PSIDs in a normal file, and SELinux was able to support more file systems. This solution was not optimal for performance, and was inconsistent across platforms. Finally, the SELinux code was integrated upstream to the 2.6.*x* kernel, which has full support for LSM and has *extended attributes* (*xattrs*) in the ext3 file system. SELinux was moved to using xattrs to store security context information. The xattr namespace provides useful separation for multiple security modules existing on the same system.

Much of the work to get the kernel ready for upstream, as well as subsequent SELinux development, has been a joint effort between the NSA, Red Hat, and the community of SELinux developers.

For more information about the history of SELinux, the definitive website is http://www.nsa.gov/research/selinux/index.shtml.

49.4. Multi-Category Security (MCS)

49.4.1. Introduction

Multi-Category Security (MCS) is an enhancement to SELinux, and allows users to label files with categories. These categories are used to further constrain *Discretionary Access Control* (DAC) and *Type Enforcement* (TE) logic. They may also be used when displaying or printing files. An example of a category is "Company_Confidential". Only users with access to this category can access files labeled with the category, assuming the existing DAC and TE rules also permit access.

The term *categories* refers to the same non-hierarchical categories used by *Multi-Level Security* (MLS). Under MLS, objects and subjects are labeled with *Security Levels*. These Security Levels consist of a hierarchical sensitivity value (such as "Top Secret") and zero or more non-hierarchical categories (such as "Crypto"). Categories provide compartments within sensitivity levels and enforce the need-to-know security principle. Refer to Section 49.6, “Multi-Level Security (MLS)” for more information about Multi-Level Security.

49.4.1.1. What is Multi-Category Security?

MCS is an adaptation of MLS. From a technical point of view, MCS is a policy change, combined with a few userland modifications to hide some of the unneeded MLS technology. Some kernel changes were also made, but only relating to making it easy to upgrade to MCS (or MLS) without invoking a full file system relabel.

The hope is to improve the quality of the system as a whole, reduce costs, leverage the open source process, increase transparency, and make the technology base useful to more than a small group of extremely special-case users.

49.4.2. Applications for Multi-Category Security

Beyond access control, MCS could be used to display the MCS categories at the top and bottom of printed pages. This may also include a cover sheet to indicate document handling procedures. It should also be possible to integrate MCS with future developments in SELinux, such as Security Enhanced X. Integration with a directory server will also make MCS support for email easier. This could involve users manually labeling outgoing emails or by attaching suitably labeled files. The email client can then determine whether the recipients are known to be cleared to access the categories associated with the emails.

49.4.3. SELinux Security Contexts

SELinux stores security contexts as an extended attribute of a file. The **"security."** namespace is used for security modules, and the **security.selinux** name is used to persistently store SELinux security labels on files. The contents of this attribute will vary depending on the file or directory you inspect and the policy the machine is enforcing.

> **Note**
>
> This is expected to change in the 2.6.15 kernel (and already has in the latest -mm kernels), so that **getxattr(2)** always returns the kernel's canonicalized version of the label.

You can use the **ls -Z** command to view the category label of a file:

```
~]# ls -Z gravityControl.txt
-rw-r--r--  user     user     user_u:object_r:tmp_t:Moonbase_Plans
gravityControl.txt
```

You can use the **gefattr(1)** command to view the internal category value (c10):

```
~]# getfattr -n security.selinux gravityControl.txt
# file: gravityControl.txt
security.selinux="user_u:object_r:tmp_t:s0:c10\000"
```

Refer to Section 49.5, “Getting Started with Multi-Category Security (MCS)” for details on creating categories and assigning them to files.

49.5. Getting Started with Multi-Category Security (MCS)

This section provides an introduction to using MCS labels to extend the Mandatory Access Control (MAC) capabilities of SELinux. It discusses MCS categories, SELinux user identities, and how they apply to Linux user accounts and files. It builds on the conceptual information provided in Section 49.4, “Multi-Category Security (MCS)”, and introduces some basic examples of usage.

49.5.1. Introduction

MCS labeling from a user and system administrator standpoint is straightforward. It consists of configuring a set of categories, which are simply text labels, such as "Company_Confidential" or "Medical_Records", and then assigning users to those categories. The system administrator first configures the categories, then assigns users to them as required. The users can then use the labels as they see fit.

The names of the categories and their meanings are set by the system administrator, and can be set to whatever is required for the specific deployment. A system in a home environment may have only one category of "Private", and be configured so that only trusted local users are assigned to this category.

In a corporate environment, categories could be used to identify documents confidential to specific departments. Categories could be established for "Finance", "Payroll", "Marketing", and "Personnel". Only users assigned to those categories can access resources labeled with the same category.

After users have been assigned to categories, they can label any of their own files with any of the categories to which they have been assigned. For example, a home user in the system described above could label all of their personal files as "Private", and no service such as Apache or vsftp would ever be able to access those files, because they don't have access to the "Private" category.

MCS works on a simple principle: to access a file, a user needs to be assigned to all of the categories with which the file is labeled. The MCS check is applied after normal Linux Discretionary Access Control (DAC) and Type Enforcement (TE) rules, so it can only further restrict security.

49.5.2. Comparing SELinux and Standard Linux User Identities

SELinux maintains its own user identity for processes, separately from Linux user identities. In the targeted policy (the default for Red Hat Enterprise Linux), only a minimal number of SELinux user identities exist:

- system_u — System processes
- root — System administrator
- user_u — All login users

Use the **semanage user -l** command to list SELinux users:

```
~]# semanage user -l

                Labeling   MLS/       MLS/
SELinux User    Prefix     MCS Level  MCS Range                 SELinux Roles

root            user       s0         s0-s0:c0.c1023            system_r
sysadm_r user_r
system_u        user       s0         s0-s0:c0.c1023            system_r
user_u          user       s0         s0-s0:c0.c1023            system_r
sysadm_r user_r
```

Refer to Section 49.8.3, “Understanding the Users and Roles in the Targeted Policy” for more information about SELinux users and roles.

SELinux Logins

One of the properties of targeted policy is that login users all run in the same security context. From a TE point of view, in targeted policy, they are security-equivalent. To effectively use MCS, however, we need to be able to assign different sets of categories to different Linux users, even though they are all the same SELinux user (**user_u**). This is solved by introducing the concept of an SELinux login. This is used during the login process to assign MCS categories to Linux users when their shell is launched.

Use the **semanage login -a** command to assign Linux users to SELinux user identities:

```
~]# semanage login -a james
~]# semanage login -a daniel
~]# semanage login -a olga
```

Now when you list the SELinux users, you can see the Linux users assigned to a specific SELinux user identity:

```
~]# semanage login -l

Login Name                SELinux User              MLS/MCS Range

__default__               user_u                    s0
```

```
james                     user_u                    s0
daniel                    user_u                    s0
root                      root                      s0-s0:c0.c1023
olga                      user_u                    s0
```

Notice that at this stage only the root account is assigned to any categories. By default, the root account is configured with access to all categories.

Red Hat Enterprise Linux and SELinux are preconfigured with several default categories, but to make effective use of MCS, the system administrator typically modifies these or creates further categories to suit local requirements.

49.5.3. Configuring Categories

SELinux maintains a mapping between internal sensitivity and category levels and their human-readable representations in the **`setrans.conf`** file. The system administrator edits this file to manage and maintain the required categories.

Use the **`chcat -L`** command to list the current categories:

```
~]# chcat -L
s0
s0-s0:c0.c1023                 SystemLow-SystemHigh
s0:c0.c1023                    SystemHigh
```

To modify the categories or to start creating your own, modify the **`/etc/selinux/`*`<selinuxtype>`*`/setrans.conf`** file. For the example introduced above, add the Marketing, Finance, Payroll, and Personnel categories as follows (this example uses the targeted policy, and irrelevant sections of the file have been omitted):

```
~]# vi /etc/selinux/targeted/setrans.conf
s0:c0=Marketing
s0:c1=Finance
s0:c2=Payroll
s0:c3=Personnel
```

Use the **`chcat -L`** command to check the newly-added categories:

```
~]# chcat -L
s0:c0                          Marketing
s0:c1                          Finance
s0:c2                          Payroll
s0:c3                          Personnel
s0
s0-s0:c0.c1023                 SystemLow-SystemHigh
s0:c0.c1023                    SystemHigh
```

Note

After you make any changes to the **setrans.conf** file, you need to restart the MCS translation service before those changes take effect. Use the following command to restart the service:

```
~]# service mcstrans restart
```

49.5.4. Assigning Categories to Users

Now that the required categories have been added to the system, you can start assigning them to SELinux users and files. To further develop the example above, assume that James is in the Marketing department, Daniel is in the Finance and Payroll departments, and Olga is in the Personnel department. Each of these users has already been assigned an SELinux login.

Use the **chcat** command to assign MCS categories to SELinux logins:

```
~]# chcat -l -- +Marketing james
~]# chcat -l -- +Finance,+Payroll daniel
~]# chcat -l -- +Personnel olga
```

You can also use the **chcat** command with additional command-line arguments to list the categories that are assigned to users:

```
~]# chcat -L -l daniel james olga
daniel: Finance,Payroll
james: Marketing
olga: Personnel
```

You can add further Linux users, assign them to SELinux user identities and then assign categories to them as required. For example, if the company director also requires a user account with access to all categories, follow the same procedure as above:

```
# Create a user account for the company director (Karl)
~]# useradd karl
~]# passwd karl
Changing password for user karl.
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.

# Assign the user account to an SELinux login
~]# semanage login -a karl

# Assign all the MCS categories to the new login
~]# chcat -l -- +Marketing,+Finance,+Payroll,+Personnel karl
```

Use the **chcat** command to verify the addition of the new user:

```
~]# chcat -L -l daniel james olga karl
daniel: Finance,Payroll
james: Marketing
```

```
olga: Personnel
karl: Marketing,Finance,Payroll,Personnel
```

Note

MCS category access is assigned during login. Consequently, a user does not have access to newly-assigned categories until they log in again. Similarly, if access to a category is revoked, this is only apparent to the user after the next login.

49.5.5. Assigning Categories to Files

At this point we have a system that has several user accounts, each of which is mapped to an SELinux user identity. We have also established a number of categories that are suitable for the particular deployment, and assigned those categories to different users.

All of the files on the system, however, still fall under the same category, and are therefore accessible by everyone (but still according to the standard Linux DAC and TE constraints). We now need to assign categories to the various files on the system so that only the appropriate users can access them.

For this example, we create a file in Daniel's home directory:

```
[daniel@dhcp-133 ~]$ echo "Financial Records 2006" > financeRecords.txt
```

Use the **`ls -Z`** command to check the initial security context of the file:

```
[daniel@dhcp-133 ~]$ ls -Z financeRecords.txt
-rw-r--r--  daniel daniel user_u:object_r:user_home_t
financeRecords.txt
```

Notice that at this stage the file has the default context for a file created in the user's home directory (**`user_home_t`**) and has no categories assigned to it. We can add the required category using the **`chcat`** command. Now when you check the security context of the file, you can see the category has been applied.

```
[daniel@dhcp-133 ~]$ chcat -- +Finance financeRecords.txt
[daniel@dhcp-133 ~]$ ls -Z financeRecords.txt
-rw-r--r--  daniel daniel root:object_r:user_home_t:Finance
financeRecords.txt
```

In many cases, you need to assign more than one category to a file. For example, some files may need to be accessible to users from both the Finance and Payroll departments.

```
[daniel@dhcp-133 ~]$ chcat -- +Payroll financeRecords.txt
[daniel@dhcp-133 ~]$ ls -Z financeRecords.txt
-rw-r--r--  daniel daniel root:object_r:user_home_t:Finance,Payroll
financeRecords.txt
```

Each of the categories that have been assigned to the file are displayed in the security context. You can add and delete categories to files as required. Only users assigned to those categories can access that file, assuming that Linux DAC and TE permissions would already allow the access.

If a user who is assigned to a different category tries to access the file, they receive an error message:

```
[olga@dhcp-133 ~]$ cat financeRecords.txt
cat: financeRecords.txt: Permission Denied
```

Note

Refer to the man pages for **`semanage`** and **`chcat`** for more information on the available options for these commands.

49.6. Multi-Level Security (MLS)

Protecting sensitive or confidential data is paramount in many businesses. In the event such information is made public, businesses may face legal or financial ramifications. At the very least, they will suffer a loss of customer trust. In most cases, however, they can recover from these financial and other losses with appropriate investment or compensation.

The same cannot be said of the defense and related communities, which includes military services, intelligence organizations and some areas of police service. These organizations cannot easily recover should sensitive information be leaked, and may not recover at all. These communities require higher levels of security than those employed by businesses and other organizations.

Having information of different security levels on the same computer systems poses a real threat. It is not a straight-forward matter to isolate different information security levels, even though different users log in using different accounts, with different permissions and different access controls.

Some organizations go as far as to purchase dedicated systems for each security level. This is often prohibitively expensive, however. A mechanism is required to enable users at different security levels to access systems simultaneously, without fear of information contamination.

49.6.1. Why Multi-Level?

The term multi-level arises from the defense community's security classifications: Confidential, Secret, and Top Secret.

Individuals must be granted appropriate clearances before they can see classified information. Those with Confidential clearance are only authorized to view Confidential documents; they are not trusted to look at Secret or Top Secret information. The rules that apply to data flow operate from lower levels to higher levels, and never the reverse. This is illustrated below.

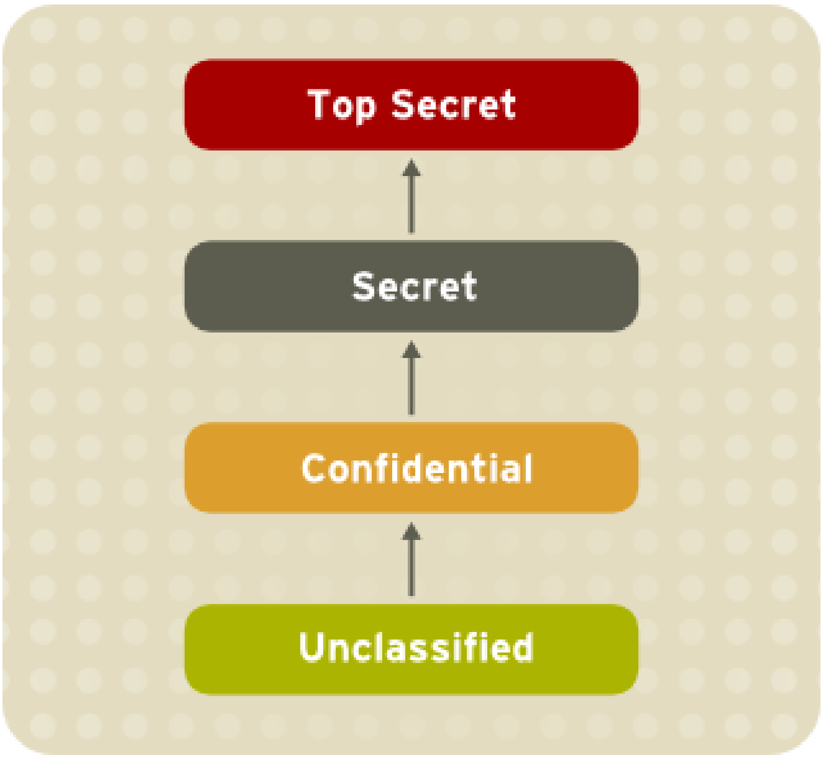


Figure 49.2. Information Security Levels

49.6.1.1. The Bell-La Padula Model (BLP)

SELinux, like most other systems that protect multi-level data, uses the BLP model. This model specifies how information can flow within the system based on labels attached to each subject and object. Refer to the following diagram:

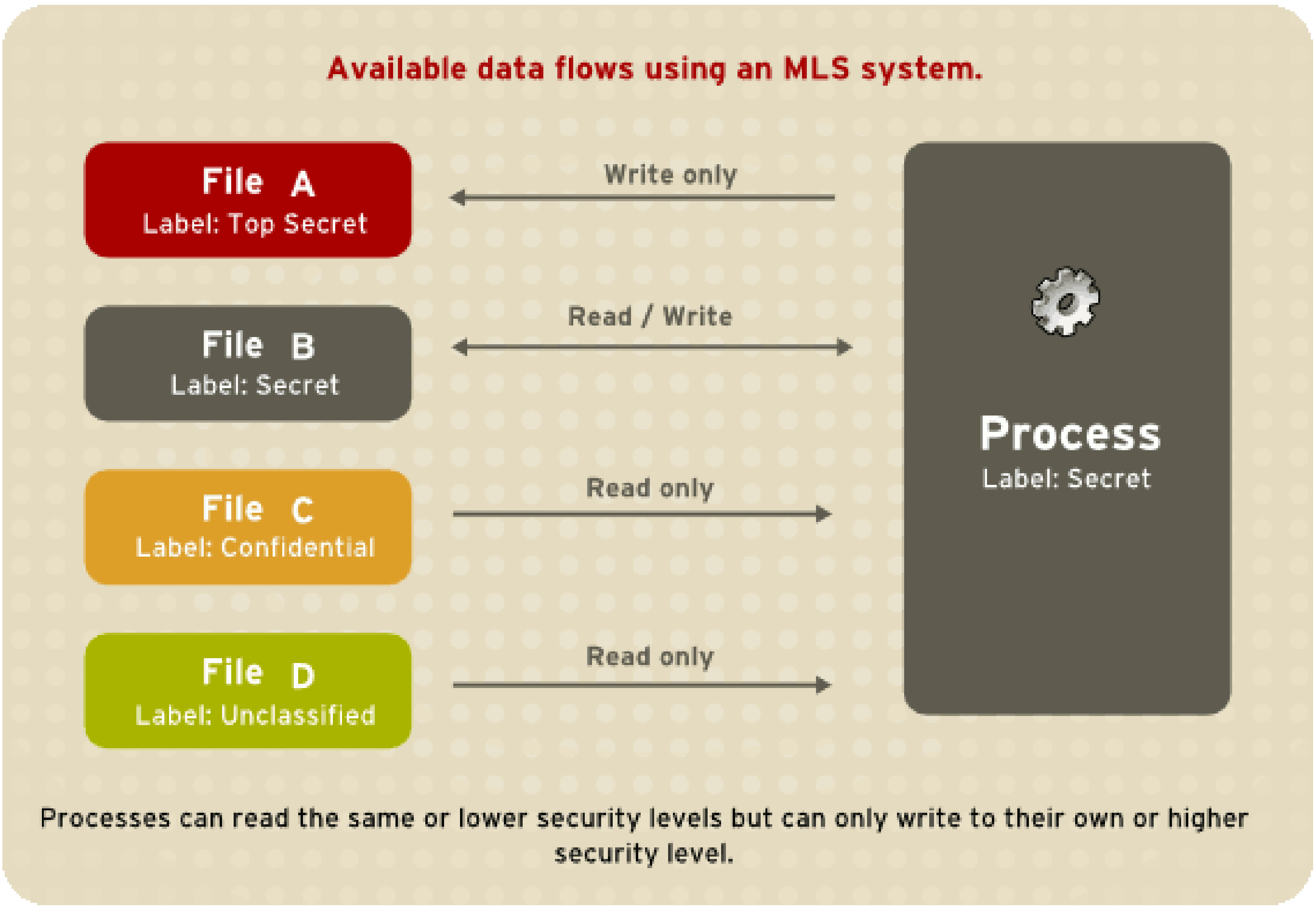


Figure 49.3. Available data flows using an MLS system

Under such a system, users, computers, and networks use labels to indicate security levels. Data can flow between like levels, for example between "Secret" and "Secret", or from a lower level to a higher level. This means that users at level "Secret" can share data with one another, and can also retrieve information from Confidential-level (i.e., lower-level), users. However, data cannot flow from a higher level to a lower level. This prevents processes at the "Secret" level from viewing information classified as "Top Secret". It also prevents processes at a higher level from accidentally writing information to a lower level. This is referred to as the "no read up, no write down" model.

49.6.1.2. MLS and System Privileges

MLS access rules are always combined with conventional access permissions (file permissions). For example, if a user with a security level of "Secret" uses Discretionary Access Control (DAC) to block access to a file by other users, this also blocks access by users with a security level of "Top Secret". A higher security clearance does not automatically give permission to arbitrarily browse a file system.

Users with top-level clearances do not automatically acquire administrative rights on multi-level systems. While they may have access to all information on the computer, this is different from having administrative rights.

49.6.2. Security Levels, Objects and Subjects

As discussed above, subjects and objects are labeled with *Security Levels* (SLs), which are composed of two types of entities:

1. **`Sensitivity`**: — A hierarchical attribute such as "Secret" or "Top Secret".

2. **`Categories`**: — A set of non-hierarchical attributes such as "US Only" or "UFO".

An SL must have one sensitivity, and may have zero or more categories.

Examples of SLs are: { Secret / UFO, Crypto }, { Top Secret / UFO, Crypto, Stargate } and { Unclassified }

Note the hierarchical sensitivity followed by zero or more categories. The reason for having categories as well as sensitivities is so that sensitivities can be further compartmentalized on a need-to-know basis. For example, while a process may be cleared to the "Secret" sensitivity level, it may not need any type of access to the project "Warp Drive" (which could be the name of a category).

Note

1. Security Levels on objects are called *Classifications*.
2. Security Levels on subjects are called *Clearances*.

Thus, objects are labeled with a Classification, while subjects operate with a specific Clearance. Security Levels can have also *Ranges*, but these are beyond the scope of this introduction.

49.6.3. MLS Policy

SELinux uses the *Bell-La Padula* BLP model, with Type Enforcement (TE) for integrity. In simple terms, MLS policy ensures that a Subject has an appropriate clearance to access an Object of a particular classification.

For example, under MLS, the system needs to know how to process a request such as: Can a process running with a clearance of { Top Secret / UFO, Rail gun } write to a file classified as { Top Secret / UFO } ?

The MLS model and the policy implemented for it will determine the answer. (Consider, for example, the problem of information leaking out of the Rail gun category into the file).

MLS meets a very narrow (yet critical) set of security requirements based around the way information and personnel are managed in rigidly controlled environments such as the military. MLS is typically difficult to work with and does not map well to general-case scenarios.

Type Enforcement (TE) under SELinux is a more flexible and expressive security scheme, which is in many cases more suitable than MLS.

There are, however, several scenarios where traditional MLS is still required. For example, a file server where the stored data may be of mixed classification and where clients connect at different clearances. This results in a large number of Security Levels and a need for strong isolation all on a single system.

This type of scenario is the reason that SELinux includes MLS as a security model, as an adjunct to TE.

49.6.4. Enabling MLS in SELinux

Note

It is not recommended to use the MLS policy on a system that is running the X Window System.

Follow these steps to enable the SELinux MLS policy on your system.

1. Install the *selinux-policy-mls* package:

```
~]# yum install selinux-policy-mls
```

2. Before the MLS policy is enabled, each file on the file system must be relabeled with an MLS label. When the file system is relabeled, confined domains may be denied access, which may prevent your system from booting correctly. To prevent this from happening, configure **`SELINUX=permissive`** in the **`/etc/selinux/config`** file. Also, enable the MLS policy by configuring **`SELINUXTYPE=mls`**. Your configuration file should look like this:

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#      enforcing - SELinux security policy is enforced.
#      permissive - SELinux prints warnings instead of enforcing.
#      disabled - No SELinux policy is loaded.
SELINUX=permissive
# SELINUXTYPE= can take one of these two values:
#      targeted - Targeted processes are protected,
#      minimum - Modification of targeted policy. Only selected
processes are protected.
#      mls - Multi Level Security protection.
SELINUXTYPE=mls
```

3. Make sure SELinux is running in the permissive mode:

```
~]# setenforce 0
~]# getenforce
Permissive
```

4. Create the **.autorelabel** file in root's home directory to ensure that files are relabeled upon next reboot:

```
~]# touch /.autorelabel
```

5. Reboot your system. During the next boot, all file systems will be relabeled according to the MLS policy. The label process labels all files with an appropriate SELinux context:

```
*** Warning -- SELinux mls policy relabel is required.
*** Relabeling could take a very long time, depending on file
*** system size and speed of hard drives.
***********
```

   Each * (asterisk) character on the bottom line represents 1000 files that have been labeled. In the above example, eleven * characters represent 11000 files which have been labeled. The time it takes to label all files depends upon the number of files on the system, and the speed of the hard disk drives. On modern systems, this process can take as little as 10 minutes. Once the labeling process finishes, the system will automatically reboot.

6. Once the file system is relabeled, execute the following commands to assure that the **/root** directory and all other home directories are properly labeled:

```
~]# genhomedircon
~]# restorecon -R -v /root /home <other_home_directories>
```

7. In permissive mode, SELinux policy is not enforced, but denials are still logged for actions that would have been denied if running in enforcing mode. Before changing to enforcing mode, as the Linux root user, run the **grep "SELinux is preventing" /var/log/messages** command to confirm that SELinux did not deny actions during the last boot. If SELinux did not deny actions during the last boot, this command does not return any output.

8. If there were no denial messages in **/var/log/messages**, or you have resolved all existing denials, configure **SELINUX=enforcing** in the **/etc/selinux/config** file:

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of these two values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected
processes are protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=mls
```

9. Reboot your system and make sure SELinux is running in permissive mode:

```
~]$ getenforce
Enforcing
```

and the MLS policy is enabled:

```
~]# sestatus |grep mls
Policy from config file:        mls
```

49.6.5. LSPP Certification

Efforts are being made to have Linux certified as an MLS operating system. The certification is equivalent to the old B1 rating, which has been reworked into the Labeled Security Protection Profile under the Common Criteria scheme.

49.7. SELinux Policy Overview

This chapter is an overview of SELinux policy, some of its internals, and how it works. It discusses the policy in general terms, while Section 49.8, “Targeted Policy Overview” focuses on the details of the targeted policy as it ships in Red Hat Enterprise Linux. This chapter starts with a brief overview of what policy is and where it resides.

Following on from this, the role of SELinux during the boot process is discussed. This is followed by discussions on file security contexts, object classes and permissions, attributes, types, access vectors, macros, users and roles, constraints, and a brief discussion summarizing special kernel interfaces.

49.7.1. What is the SELinux Policy?

The SELinux Policy is the set of rules that guide the SELinux security engine. It defines *types* for file objects and *domains* for processes. It uses roles to limit the domains that can be entered, and has user identities to specify the roles that can be attained. In essence, types and domains are equivalent, the difference being that types apply to objects while domains apply to processes.

49.7.1.1. SELinux Types

A type is a way of grouping items based on their similarity from a security perspective. This is not necessarily related to the unique purpose of an application or the content of a document. For example, a file can have any type of content and be for any purpose, but if it belongs to a user and exists in that user's home directory, it is considered to be of a specific security type, **`user_home_t`**.

These object types are considered alike because they are accessible in the same way by the same set of subjects. Similarly, processes tend to be of the same type if they have the same permissions as other subjects. In the targeted policy, programs that run in the **`unconfined_t`** domain have an executable file with a type such as **`sbin_t`**. From an SELinux perspective, this means they are all equivalent in terms of what they can and cannot do on the system.

For example, the binary executable file object at **`/usr/bin/postgres`** has the type postgresql_exec_t. All of the targeted daemons have their own **`*_exec_t`** type for their executable applications. In fact, the entire set of **PostgreSQL** executables such as **`createlang`**, **`pg_dump`**, and **`pg_restore`** have the same type, **`postgresql_exec_t`**, and they transition to the same domain, **`postgresql_t`**, upon execution.

49.7.1.1.1. Using Policy Rules to Define Type Access

The SELinux policy defines various rules which determine how each domain may access each type. Only what is specifically allowed by the rules is permitted. By default, every operation is denied and audited, meaning it is logged in the **`$AUDIT_LOG`** file. In Red Hat Enterprise Linux, this is set to **`/var/log/messages`**. The policy is compiled into binary format for loading into the kernel security server, and each time the security server makes a decision, it is cached in the AVC to optimize performance.

The policy can be defined either by modifying the existing files or by adding local *Type Enforcement (TE)* and *File Context (FC)* files to the policy tree. These new policies can be loaded into the kernel in real time. Otherwise, the policy is loaded during the boot process by **`init`**, as explained in Section 49.7.3, “The Role of Policy in the Boot Process”. Ultimately, every system operation is determined by the policy and the type-labeling of the files.

Important

After loading a new policy, it is recommended that you restart any services that may have new or changed labeling. Generally speaking, this is only the targeted daemons, as listed in Section 49.8.1, “What is the Targeted Policy?”.

49.7.1.2. SELinux and Mandatory Access Control

SELinux is an implementation of *Mandatory Access Control (MAC)*. Depending on the security policy type, SELinux implements either *Type Enforcement (TE)*, *Roles Based Access Control (RBAC)* or *Bell-La Padula Model Multi-Level Security (MLS)*.

The policy specifies the rules in the implemented environment. It is written in a language created specifically for writing security policy. Policy writers use **`m4`** macros to capture common sets of low-level rules. A number of **`m4`** macros are defined in the existing policy, which facilitate the writing of new policy. These rules are preprocessed into many additional rules as part of building the **`policy.conf`** file, which is compiled into the binary policy.

Access rights are divided differently among domains, and no domain is required to act as a master for all other domains. Moving between domains is controlled by the policy, through login programs, userspace programs such as **`newrole`**, or by requiring a new process execution in the new domain. This movement between domains is referred to as a *transition* .

49.7.2. Where is the Policy?

There are two components to the policy: the binary tree and the source tree. The binary tree is provided by the **`selinux-policy-<policyname>`** package and supplies the binary policy file.

Alternatively, the binary policy can be built from source when the **`selinux-policy-devel`** package is installed.

Note

Information on how to edit, write and compile policy is currently outside the scope of this document.

49.7.2.1. Binary Tree Files

- **`/etc/selinux/targeted/`** — this is the root directory for the targeted policy, and contains the binary tree.

- **`/etc/selinux/targeted/policy/`** — this is the location of the binary policy file **`policy.<xx>`**. In this guide, the variable **`SELINUX_POLICY`** is used for this directory.
- **`/etc/selinux/targeted/contexts/`** — this is the location of the security context information and configuration files, which are used during runtime by various applications.
- **`/etc/selinux/targeted/contexts/files/`** — contains the default contexts for the entire file system. This is referenced by **`restorecon`** when performing relabeling operations.
- **`/etc/selinux/targeted/contexts/users/`** — in the targeted policy, only the **`root`** file is in this directory. These files are used for determining context when a user logs in. For example, for the root user, the context is user_u:system_r:unconfined_t.
- **`/etc/selinux/targeted/modules/active/booleans*`** — this is where the runtime Booleans are configured.

Note

These files should never be manually changed. You should use the **`getsebool`**, **`setsebool`** and **`semanage`** tools to manipulate runtime Booleans.

49.7.2.2. Source Tree Files

For developing policy modules, the **`selinux-policy-devel`** package includes all of the interface files used to build policy. It is recommended that people who build policy use these files to build the policy modules.

This package installs the policy interface files under **`/usr/share/selinux/devel/include`** and has **`make`** files installed in **`/usr/share/selinux/devel/Makefile`**.

To help applications that need the various SELinux paths, **`libselinux`** provides a number of functions that return the paths to the different configuration files and directories. This negates the need for applications to hard-code the paths, especially since the active policy location is dependent on the SELINUXTYPE setting in **`/etc/selinux/config`**.

For example, if SELINUXTYPE is set to strict, the active policy location is under **`/etc/selinux/strict`**.

To view the list of available functions, use the following command:

```
man 3 selinux_binary_policy_path
```

Note

This man page is available only if you have the **`libselinux-devel`** RPM installed.

The use of **`libselinux`** and related functions is outside the scope of this document.

49.7.3. The Role of Policy in the Boot Process

SELinux plays an important role during the early stages of system start-up. Because all processes must be labeled with their correct domain, **`init`** performs some essential operations early in the boot process to maintain synchronization between labeling and policy enforcement.

1. After the kernel has been loaded during the boot process, the initial process is assigned the predefined *initial SELinux ID (initial SID)* kernel. Initial SIDs are used for bootstrapping before the policy is loaded.

2. **`/sbin/init`** mounts **`/proc/`**, and then searches for the **`selinuxfs`** file system type. If it is present, that means SELinux is enabled in the kernel.

3. If **`init`** does not find SELinux in the kernel, or if it is disabled via the ***`selinux=0`*** boot parameter, or if **`/etc/selinux/config`** specifies that **`SELINUX=disabled`**, the boot process proceeds with a non-SELinux system.

   At the same time, **`init`** sets the enforcing status if it is different from the setting in **`/etc/selinux/config`**. This happens when a parameter is passed during the boot process, such as ***`enforcing=0`*** or ***`enforcing=1`***. The kernel does not enforce any policy until the initial policy is loaded.

4. If SELinux is present, **`/selinux/`** is mounted.

5. **`init`** checks **`/selinux/policyvers`** for the supported policy version. The version number in **`/selinux/policyvers`** is the latest policy version your kernel supports. **`init`** inspects **`/etc/selinux/config`** to determine which policy is active, such as the targeted policy, and loads the associated file at **`$SELINUX_POLICY/policy.<`*`version`*`>`**.

   If the binary policy is *not* the version supported by the kernel, **`init`** attempts to load the policy file if it is a previous version. This provides backward compatibility with older policy versions.

   If the local settings in **`/etc/selinux/targeted/booleans`** are different from those compiled in the policy, **`init`** modifies the policy in memory based on the local settings prior to loading the policy into the kernel.

6. By this stage of the process, the policy is fully loaded into the kernel. The initial SIDs are then mapped to security contexts in the policy. In the case of the targeted policy, the new domain is user_u:system_r:unconfined_t. The kernel can now begin to retrieve security contexts dynamically from the in-kernel security server.

7. **`init`** then re-executes itself so that it can transition to a different domain, if the policy defines it. For the targeted policy, there is no transition defined and **`init`** remains in the **`unconfined_t`** domain.

8. At this point, **`init`** continues with its normal boot process.

The reason that **`init`** re-executes itself is to accommodate stricter SELinux policy controls. The objective of re-execution is to transition to a new domain with its own granular rules. The only way that a process can enter a domain is during execution, which means that such processes are the only *entry points* into the domains.

For example, if the policy has a specific domain for **`init`**, such as **`init_t`**, a method is required to change from the initial SID, such as kernel, to the correct runtime domain for **`init`**. Because this transition may need to occur, **`init`** is coded to re-execute itself after loading the policy.

This **`init`** transition occurs if the **`domain_auto_trans(kernel_t, init_exec_t, <`*`target_domain_t`*`>)`** rule is present in the policy. This rule states that an automatic transition occurs on anything executing in the **`kernel_t`** domain that executes a file of type init_exec_t. When this execution occurs, the new process is assigned the domain **`<`*`target_domain_t`*`>`**, using an actual target domain such as **init_t**.

49.7.4. Object Classes and Permissions

SELinux defines a number of classes for objects, making it easier to group certain permissions by specific classes. For example:

- File-related classes include **filesystem** for file systems, **file** for files, and **dir** for directories. Each class has its own associated set of permissions.

  The **filesystem** class can mount, unmount, get attributes, set quotas, relabel, and so forth. The **file** class has common file permissions such as read, write, get and set attributes, lock, relabel, link, rename, append, etc.

- Network related classes include **tcp_socket** for TCP sockets, **netif** for network interfaces, and **node** for network nodes.

  The **netif** class, for example, can send and receive on TCP, UDP and raw sockets (**tcp_recv**, **tcp_send**, **udp_send**, **udp_recv**, **rawip_recv**, and **rawip_send**.)

The object classes have matching declarations in the kernel, meaning that it is not trivial to add or change object class details. The same is true for permissions. Development work is ongoing to make it possible to dynamically register and unregister classes and permissions.

Permissions are the actions that a subject can perform on an object, if the policy allows it. These permissions are the access requests that SELinux actively allows or denies.

49.8. Targeted Policy Overview

This chapter is an overview and examination of the SELinux targeted policy, the current supported policy for Red Hat Enterprise Linux.

Much of the content in this chapter is applicable to all types of SELinux policy, in terms of file locations and the type of content in those files. The difference lies in which files exist in the key locations and their contents.

49.8.1. What is the Targeted Policy?

The SELinux policy is highly configurable. For Red Hat Enterprise Linux 5, Red Hat supports a single policy, the *targeted policy* . Under the targeted policy, every subject and object runs in the **unconfined_t** domain *except* for the specific targeted daemons. Objects that are in the **unconfined_t** domain have no restrictions and fall back to using standard Linux security, that is, DAC. The daemons that are part of the targeted policy run in their own domains and are restricted in every operation they perform on the system. This way daemons that are exploited or compromised in any way are contained and can only cause limited damage.

For example, the **http** and **ntp** daemons are both protected in the default targeted policy, and run in the **httpd_t** and **ntpd_t** domains, respectively. The **ssh** daemon, however, is not protected in this policy, and consequently runs in the **unconfined_t** domain.

Refer to the following sample output, which illustrates the various domains for the daemons mentioned above

```
user_u:system_r:httpd_t         25129 ?        00:00:00 httpd
user_u:system_r:ntpd_t          25176 ?        00:00:00 ntpd
system_u:system_r:unconfined_t         25245 ? 00:00:00 sshd
```

The Strict Policy

The opposite of the targeted policy is the *strict policy* . In the strict policy, every subject and object exists in a specific security domain, and all interactions and transitions are individually considered within the policy rules.

The strict policy is a much more complex environment, and does not ship with Red Hat Enterprise Linux. This guide focuses on the targeted policy that ships with Red Hat Enterprise Linux, and the components of SELinux used by the targeted daemons.

The targeted daemons are as follows: **`dhcpd`**; **`httpd`**; **`mysqld`**; **`named`**; **`nscd`**; **`ntpd`**; **`portmap`**; **`postgres`**; **`snmpd`**; **`squid`**; **`syslogd`**; and **`winbind`**.

Note

Depending on your installation, only some of these daemons may be present.

49.8.2. Files and Directories of the Targeted Policy

Refer to Section 49.7.2, “Where is the Policy?” for a list of the common files and directories used by SELinux.

49.8.3. Understanding the Users and Roles in the Targeted Policy

This section covers the specific roles enabled for the targeted policy. The **`unconfined_t`** type exists in every role, which significantly reduces the usefulness of roles in the targeted policy. More extensive use of roles requires a change to the strict policy paradigm, where every process runs in an individually considered domain.

Effectively, there are only two roles in the targeted policy: **`system_r`** and **`object_r`**. The initial role is **`system_r`**, and everything else inherits that role. The remaining roles are defined for compatibility purposes between the targeted policy and the strict policy. [20]

Three of the four roles are defined by the policy. The fourth role, **`object_r`**, is an implied role and is not found in policy source. Because roles are created and populated by types using one or more declarations in the policy, there is no single file that declares all roles. (Remember that the policy itself is generated from a number of separate files.)

`system_r`

This role is for all system processes except user processes:

```
system_r (28 types)
    dhcpd_t
    httpd_helper_t
    httpd_php_t
    httpd_suexec_t
    httpd_sys_script_t
    httpd_t
    httpd_unconfined_script_t
    initrc_t
    ldconfig_t
    mailman_cgi_t
    mailman_mail_t
    mailman_queue_t
    mysqld_t
```

```
     named_t
     ndc_t
     nscd_t
     ntpd_t
     pegasus_t
     portmap_t
     postgresql_t
     snmpd_t
     squid_t
     syslogd_t
     system_mail_t
     unconfined_t
     winbind_helper_t
     winbind_t
     ypbind_t
```

`user_r`

This is the default user role for regular Linux users. In a strict policy, individual users might be used, allowing for the users to have special roles to perform privileged operations. In the targeted policy, all users run in the **`unconfined_t`** domain.

`object_r`

In SELinux, roles are not utilized for objects when RBAC is being used. Roles are strictly for subjects. This is because roles are task-oriented and they group together entities which perform actions (for example, processes). All such entities are collectively referred to as subjects. For this reason, all objects have the role **`object_r`**, and the role is only used as a placeholder in the label.

`sysadm_r`

This is the system administrator role in a strict policy. If you log in directly as the root user, the default role may actually be **`staff_r`**. If this is true, use the **`newrole -r sysadm_r`** command to change to the SELinux system administrator role to perform system administration tasks. In the targeted policy, the following retain **`sysadm_r`** for compatibility:

```
sysadm_r (6 types)
    httpd_helper_t
    httpd_sys_script_t
    initrc_t
    ldconfig_t
    ndc_t
    unconfined_t
```

There is effectively only one user identity in the targeted policy. The **`user_u`** identity was chosen because **`libselinux`** falls back to **`user_u`** as the default SELinux user identity. This occurs when there is no matching SELinux user for the Linux user who is logging in. Using **`user_u`** as the single user in the targeted policy makes it easier to change to the strict policy. The remaining users exist for compatibility with the strict policy. [21]

The one exception is the SELinux user **`root`**. You may notice **`root`** as the user identity in a process's context. This occurs when the SELinux user **`root`** starts daemons from the command line, or restarts a daemon originally started by **`init`**.

[18] The NSA is the cryptologic agency of the United States of America's Federal government, charged with information assurance and signals intelligence. You can read more about the NSA at their website, http://www.nsa.gov/about/.

[19] Flask grew out of a project that integrated the *Distributed Trusted Operating System* (*DTOS*) into the Fluke research operating system. Flask was the name of the architecture and the implementation in the Fluke operating system.

[20] Any role could have been chosen for the targeted policy, but **`system_r`** already had existing authorization for the daemon domains, simplifying the process. This was done because no mechanism currently exists to alias roles.

[21] A user aliasing mechanism would also work here, to alias all identities from the strict policy to a single user identity in the targeted policy.

Chapter 50. Working With SELinux

SELinux presents both a new security paradigm and a new set of practices and tools for administrators and some end-users. The tools and techniques discussed in this chapter focus on standard operations performed by end-users, administrators, and analysts.

50.1. End User Control of SELinux

In general, end users have little interaction with SELinux when Red Hat Enterprise Linux is running the targeted policy. This is because users are running in the domain of **`unconfined_t`** along with the rest of the system *except* the targeted daemons.

In most situations, standard DAC controls prevent you from performing tasks for which you do not have the required access or permissions before SELinux is consulted. Consequently, it is likely that you will never generate an **`avc: denied`** message.

The following sections cover the general tasks and practices that an end user might need to perform on a Rec Hat Enterprise Linux system. These tasks apply to users of all privilege levels, not only to end users.

50.1.1. Moving and Copying Files

In file system operations, security context must now be considered in terms of the label of the file, the process accessing it, and the directories where the operation is happening. Because of this, moving and copying files with **`mv`** and **`cp`** may have unexpected results.

Copying Files: SELinux Options for cp

Unless you specify otherwise, **`cp`** follows the default behavior of creating a new file based on the domain of the creating process and the type of the target directory. Unless there is a specific rule to set the label, the file inherits the type from the target directory.

Use the **`-Z`** ***`user:role:type`*** option to specify the required label for the new file.

The **`-p`** (or **`--preserve=mode,ownership,timestamps`**) option preserves the specified attributes and, if possible, additional attributes such as links.

```
touch bar foo
ls -Z bar foo
-rw-rw-r--  auser   auser   user_u:object_r:user_home_t   bar
-rw-rw-r--  auser   auser   user_u:object_r:user_home_t   foo
```

If you use the **`cp`** command without any additional command-line arguments, a copy of the file is created in the new location using the default type of the creating process and the target directory. In this case, because there is no specific rule that applies to **`cp`** and **`/tmp`**, the new file has the type of the parent directory:

```
cp bar /tmp
ls -Z /tmp/bar
-rw-rw-r--  auser   auser   user_u:object_r:tmp_t   /tmp/bar
```

The type **`tmp_t`** is the default type for temporary files.

Use the **`-Z`** option to specify the label for the new file:

```
cp -Z user_u:object_r:user_home_t foo /tmp
ls -Z /tmp/foo
-rw-rw-r--  auser   auser   user_u:object_r:user_home_t   /tmp/foo
```

Moving Files: SELinux Options for mv

Moving files with **mv** retains the original type associated with the file. Care should be taken using this command as it can cause problems. For example, if you move files with the type **user_home_t** into **~/public_html**, then the **httpd** daemon is not able to serve those files until you relabel them. Refer to Section 50.1.3, "Relabeling a File or Directory" for more information about file labeling.

Table 50.1. Behavior of mv and cp Commands

Command	Behavior
mv	The file retains its original label. This may cause problems, confusion, or minor insecurity. For example, the **tmpwatch** program running in the **sbin_t** domain might not be allowed to delete an aged file in the **/tmp** directory because of the file's type.
cp	Makes a copy of the file using the default behavior based on the domain of the creating process (**cp**) and the type of the target directory.
cp -p	Makes a copy of the file, preserving the specified attributes and security contexts, if possible. The default attributes are mode, ownership, and timestamps. Additional attributes are links and all.
cp -Z <*user:role:type*>	Makes a copy of the file with the specified labels. The **-Z** option is synonymous with **--context**.

50.1.2. Checking the Security Context of a Process, User, or File Object

Checking a Process ID

In Red Hat Enterprise Linux, the **-Z** option is equivalent to **--context**, and can be used with the **ps**, **id**, **ls**, and **cp** commands. The behavior of the **cp** command with respect to SELinux is explained in Table 50.1, "Behavior of mv and cp Commands".

The following example shows a small sample of the output of the **ps** command. Most of the processes are running in the **unconfined_t** domain, with a few exceptions.

```
[user@localhost ~]$ ps auxZ
LABEL                           USER       PID %CPU %MEM    VSZ   RSS TTY
STAT START   TIME COMMAND
system_u:system_r:init_t        root         1  0.0  0.1   2032   620 ?
Ss   15:09   0:00 init [5]
system_u:system_r:kernel_t      root         2  0.0  0.0      0     0 ?
S    15:09   0:00 [migration/0]
system_u:system_r:kernel_t      root         3  0.0  0.0      0     0 ?
SN   15:09   0:00 [ksoftirqd/0]

user_u:system_r:unconfined_t    user      3122  0.0  0.6   6908  3232 ?
S    16:47   0:01 /usr/libexec/gconfd-2 5
user_u:system_r:unconfined_t    user      3125  0.0  0.1   2540   588 ?
S    16:47   0:00 /usr/bin/gnome-keyring-daemon
user_u:system_r:unconfined_t    user      3127  0.0  1.4  33612  6988 ?
```

```
Sl   16:47   0:00 /usr/libexec/gnome-settings-daemon
user_u:system_r:unconfined_t    user     3144  0.1  1.4  16528  7360 ?
Ss   16:47   0:01 metacity --sm-client-id=default1
user_u:system_r:unconfined_t    user     3148  0.2  2.9  79544 14808 ?
Ss   16:47   0:03 gnome-panel --sm-client-id default2
```

Checking a User ID

You can use the **-Z** option with the **id** command to determine a user's security context. Note that with this command you cannot combine **-Z** with other options.

```
[root@localhost ~]# id -Z
user_u:system_r:unconfined_t
```

Note that you cannot use the **-Z** option with the **id** command to inspect the security context of a different user. That is, you can only display the security context of the currently logged-in user:

```
[user@localhost ~]$ id
uid=501(user) gid=501(user) groups=501(user)
context=user_u:system_r:unconfined_t
[user@localhost ~]$ id root
uid=0(root) gid=0(root)
groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel)
[user@localhost ~]$ id -Z root
id: cannot display context when selinux not enabled or when displaying the
id
of a different user
```

Check a File ID

You can use the **-Z** option with the **ls** command to group common long-format information. You can display mode, user, group, security context, and filename information.

```
cd /etc
ls -Z h* -d
drwxr-xr-x  root root  system_u:object_r:etc_t        hal
-rw-r--r--  root root  system_u:object_r:etc_t        host.conf
-rw-r--r--  root root  user_u:object_r:etc_t          hosts
-rw-r--r--  root root  system_u:object_r:etc_t        hosts.allow
-rw-r--r--  root root  system_u:object_r:etc_t        hosts.canna
-rw-r--r--  root root  system_u:object_r:etc_t        hosts.deny
drwxr-xr-x  root root  system_u:object_r:hotplug_etc_t  hotplug
drwxr-xr-x  root root  system_u:object_r:etc_t        hotplug.d
drwxr-xr-x  root root  system_u:object_r:httpd_sys_content_t htdig
drwxr-xr-x  root root  system_u:object_r:httpd_config_t httpd
```

50.1.3. Relabeling a File or Directory

You may need to relabel a file when moving or copying into special directories related to the targeted daemons, such as **~/public_html** directories, or when writing scripts that work in directories outside of **/home**.

There are two general types of relabeling operations:

- Deliberately changing the type of a file
- Restoring files to the default state according to policy

There are also relabeling operations that an administrator performs. These are covered in Section 50.2.2, “Relabeling a File System”.

Note

The majority of SELinux permission control in the targeted policy is Type Enforcement (TE). Consequently, you can generally ignore the user and role information in a security label and focus on just changing the type. You do not normally need to consider the role and user settings on files.

Note

If relabeling affects the label on a daemon's executable, you should restart the daemon to be sure it is running in the correct domain. For example, if **`/usr/sbin/mysqld`** has the wrong security label, and you address this by using a relabeling operation such as **`restorecon`**, you must restart **`mysqld`** after the relabeling operation. Setting the executable file to have the correct type (**`mysqld_exec_t`**) ensures that it transitions to the proper domain when started.

Use the **`chcon`** command to change a file to the correct type. You need to know the correct type that you want to apply to use this command. The directories and files in the following example are labeled with the default type defined for file system objects created in **`/home`**:

```
cd ~
ls -Zd public_html/
drwxrwxr-x  auser  auser  user_u:object_r:user_home_t public_html/

ls -Z web_files/
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t   1.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t   2.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t   3.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t   4.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t   5.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t   index.html
```

If you move these files into the **`public_html`** directory, they retain the original type:

```
mv web_files/* public_html/
ls -Z public_html/
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t   1.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t   2.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t   3.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t   4.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t   5.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t   index.html
```

To make these files viewable from a special user public HTML folder, they need to have a type that **`httpd`** has permissions to read, presuming the Apache HTTP Server is configured for UserDir and the Boolean value **`httpd_enable_homedirs`** is enabled.

```
chcon -R -t httpd_user_content_t public_html/
ls -Z public_html
-rw-rw-r--  auser  auser  user_u:object_r:httpd_user_content_t   1.html
-rw-rw-r--  auser  auser  user_u:object_r:httpd_user_content_t   2.html
-rw-rw-r--  auser  auser  user_u:object_r:httpd_user_content_t   3.html
-rw-rw-r--  auser  auser  user_u:object_r:httpd_user_content_t   4.html
-rw-rw-r--  auser  auser  user_u:object_r:httpd_user_content_t   5.html
-rw-rw-r--  auser  auser  user_u:object_r:httpd_user_content_t   index.html

ls -Z public_html/ -d
drwxrwxr-x  auser  auser  user_u:object_r:httpd_user_content_t  public_html/
```

Note

If the file has no label, such as a file created while SELinux was disabled in the kernel, you need to give it a full label with **`chcon system_u:object_r:shlib_t foo.so`**. Otherwise, you will receive an error about applying a partial context to an unlabeled file.

Use the **`restorecon`** command to restore files to the default values according to the policy. There are two other methods for performing this operation that work on the entire file system: **`fixfiles`** or a policy relabeling operation. Each of these methods requires superuser privileges. Cautions against both of these methods appear in Section 50.2.2, “Relabeling a File System”.

The following example demonstrates restoring the default user home directory context to a set of files that have different types. The first two sets of files have different types, and are being moved into a directory for archiving. Their contexts are different from each other, and are incorrect for a standard user's home directory:

```
ls -Z /tmp/
-rw-rw-r--  auser  auser  user_u:object_r:tmp_t            /tmp/file1
-rw-rw-r--  auser  auser  user_u:object_r:tmp_t            /tmp/file2
-rw-rw-r--  auser  auser  user_u:object_r:tmp_t            /tmp/file3

mv /tmp/{1,2,3} archives/
mv public_html/* archives/
ls -Z archives/
-rw-rw-r--  auser  auser  user_u:object_r:tmp_t            file1
-rw-rw-r--  auser  auser  user_u:object_r:httpd_user_content_t    file1.html
-rw-rw-r--  auser  auser  user_u:object_r:tmp_t            file2
-rw-rw-r--  auser  auser  user_u:object_r:httpd_user_content_t    file2.html
-rw-rw-r--  auser  auser  user_u:object_r:tmp_t            file3
-rw-rw-r--  auser  auser  user_u:object_r:httpd_user_content_t    file3.html
-rw-rw-r--  auser  auser  user_u:object_r:httpd_user_content_t    file4.html
-rw-rw-r--  auser  auser  user_u:object_r:httpd_user_content_t    file5.html
-rw-rw-r--  auser  auser  user_u:object_r:httpd_user_content_t  index.html
```

The **`archives/`** directory already has the default type because it was created in the user's home directory:

```
ls -Zd archives/
drwxrwxr-x  auser  auser  user_u:object_r:user_home_t  archives/
```

Using the **`restorecon`** command to relabel the files uses the default file contexts set by the policy, so these files are labeled with the default label for their current directory.

```
/sbin/restorecon -R archives/
ls -Z archives/
-rw-rw-r--  auser  auser  system_u:object_r:user_home_t    file1
-rw-rw-r--  auser  auser  system_u:object_r:user_home_t    file1.html
-rw-rw-r--  auser  auser  system_u:object_r:user_home_t    file2
-rw-rw-r--  auser  auser  system_u:object_r:user_home_t    file2.html
-rw-rw-r--  auser  auser  system_u:object_r:user_home_t    file3
-rw-rw-r--  auser  auser  system_u:object_r:user_home_t    file3.html
-rw-rw-r--  auser  auser  system_u:object_r:user_home_t    file4.html
-rw-rw-r--  auser  auser  system_u:object_r:user_home_t    file5.html
-rw-rw-r--  auser  auser  system_u:object_r:user_home_t    index.html
```

50.1.4. Creating Archives That Retain Security Contexts

You can use either the **`tar`** or **`star`** utilities to create archives that retain SELinux security contexts. The following example uses **`star`** to demonstrate how to create such an archive. You need to use the appropriate **`-xattr`** and **`-H=exustar`** options to ensure that the extra attributes are captured and that the header for the **`*.star`** file is of a type that fully supports xattrs. Refer to the man page for more information about these and other options.

The following example illustrates the creation and extraction of a set of html files and directories. Note that the two directories have different labels. Unimportant parts of the file context have been omitted for printing purposes (indicated by ellipses '...'):

```
ls -Z public_html/ web_files/

public_html/:
-rw-rw-r--  auser  auser  ...httpd_user_content_t 1.html
-rw-rw-r--  auser  auser  ...httpd_user_content_t 2.html
-rw-rw-r--  auser  auser  ...httpd_user_content_t 3.html
-rw-rw-r--  auser  auser  ...httpd_user_content_t 4.html
-rw-rw-r--  auser  auser  ...httpd_user_content_t 5.html
-rw-rw-r--  auser  auser  ...httpd_user_content_t index.html
web_files/:
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t  1.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t  2.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t  3.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t  4.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t  5.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t  index.html
```

The following command creates the archive, retaining all of the SELinux security contexts:

```
star -xattr -H=exustar -c -f all_web.star public_html/ web_files/
star: 11 blocks + 0 bytes (total of 112640 bytes = 110.00k).
```

Use the **`ls`** command with the **`-Z`** option to validate the security context:

```
ls -Z all_web.star
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t \  all_web.star
```

You can now copy the archive to a different directory. In this example, the archive is copied to **`/tmp`**. If there is no specific policy to make a derivative temporary type, the default behavior is to acquire the **`tmp_t`** type.

```
cp all_web.star /tmp/ cd /tmp/

ls -Z all_web.star
-rw-rw-r--  auser  auser  user_u:object_r:tmp_t  all_web.star
```

Now you can expand the archives using **star** and it restores the extended attributes:

```
star -xattr -x -f all_web.star
star: 11 blocks + 0 bytes (total of 112640 bytes = 110.00k).

ls -Z /tmp/public_html/ /tmp/web_files/
/tmp/public_html/:
-rw-rw-r--  auser  auser  ...httpd_sys_content_t 1.html
-rw-rw-r--  auser  auser  ...httpd_sys_content_t 2.html
-rw-rw-r--  auser  auser  ...httpd_sys_content_t 3.html
-rw-rw-r--  auser  auser  ...httpd_sys_content_t 4.html
-rw-rw-r--  auser  auser  ...httpd_sys_content_t 5.html
-rw-rw-r--  auser  auser  ...httpd_sys_content_t index.html
/tmp/web_files/:
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t  1.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t  2.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t  3.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t  4.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t  5.html
-rw-rw-r--  auser  auser  user_u:object_r:user_home_t  \ index.html
```

Warning

If you use an absolute path when you create an archive using **star**, the archive expands on that same path. For example, an archive made with this command restores the files to **/var/log/httpd/**:

```
star -xattr -H=exustar -c -f httpd_logs.star /var/log/httpd/
```

If you attempt to expand this archive, **star** issues a warning if the files in the path are newer than the ones in the archive.

50.2. Administrator Control of SELinux

In addition to the tasks often performed by users in Section 50.1, “End User Control of SELinux”, SELinux administrators could be expected to perform a number of additional tasks. These tasks typically require root access to the system. Such tasks are significantly easier under the targeted policy. For example, there is no need to consider adding, editing, or deleting Linux users from the SELinux users, nor do you need to consider roles.

This section covers the types of tasks required of an administrator who maintains Red Hat Enterprise Linux running SELinux.

50.2.1. Viewing the Status of SELinux

The **sestatus** command provides a configurable view into the status of SELinux. The simplest form of this command shows the following information:

```
~]# sestatus
SELinux status:                 enabled
SELinuxfs mount:                /selinux
Current mode:                   enforcing
Mode from config file:          enforcing
Policy version:                 21
Policy from config file:        targeted
```

The **-v** option includes information about the security contexts of a series of files that are specified in **/etc/sestatus.conf**:

```
~]# sestatus -v
SELinux status:                 enabled
SELinuxfs mount:                /selinux
Current mode:                   enforcing
Mode from config file:          enforcing
Policy version:                 21
Policy from config file:        targeted

Process contexts:
Current context:                user_u:system_r:unconfined_t
Init context:                   system_u:system_r:init_t
/sbin/mingetty                  system_u:system_r:getty_t
/usr/sbin/sshd                  system_u:system_r:unconfined_t:s0-
s0:c0.c1023

File contexts:
Controlling term:               user_u:object_r:devpts_t
/etc/passwd                     system_u:object_r:etc_t
/etc/shadow                     system_u:object_r:shadow_t
/bin/bash                       system_u:object_r:shell_exec_t
/bin/login                      system_u:object_r:login_exec_t
/bin/sh                         system_u:object_r:bin_t ->
system_u:object_r:shell_exec_t
/sbin/agetty                    system_u:object_r:getty_exec_t
/sbin/init                      system_u:object_r:init_exec_t
/sbin/mingetty                  system_u:object_r:getty_exec_t
/usr/sbin/sshd                  system_u:object_r:sshd_exec_t
/lib/libc.so.6                  system_u:object_r:lib_t ->
system_u:object_r:lib_t
/lib/ld-linux.so.2              system_u:object_r:lib_t ->
system_u:object_r:ld_so_t
```

The **-b** displays the current state of booleans. You can use this in combination with grep or other tools to determine the status of particular booleans:

```
~]# sestatus -b | grep httpd | grep on$
httpd_builtin_scripting           on
httpd_disable_trans               on
httpd_enable_cgi                  on
httpd_enable_homedirs             on
httpd_unified                     on
```

50.2.2. Relabeling a File System

You may never need to relabel an entire file system. This usually occurs only when labeling a file system for SELinux for the first time, or when switching between different types of policy, such as changing from the targeted to the strict policy.

Relabeling a File System Using init

The recommended method for relabeling a file system is to reboot the machine. This allows the **`init`** process to perform the relabeling, ensuring that applications have the correct labels when they are started and that they are started in the right order. If you relabel a file system without rebooting, some processes may continue running with an incorrect context. Manually ensuring that all the daemons are restarted and running in the correct context can be difficult.

Use the following procedure to relabel a file system using this method.

```
touch /.autorelabel
reboot
```

At boot time, **`init.rc`** checks for the existence of **`/.autorelabel`**. If this file exists, SELinux performs a complete file system relabel (using the **`/sbin/fixfiles -f -F relabel`** command), and then deletes **`/.autorelabel`**.

Relabeling a File System Using fixfiles

It is possible to relabel a file system using the **`fixfiles`** command, or to relabel based on the RPM database:

Use the following command to relabel a file system only using the **`fixfiles`** command:

```
fixfiles relabel
```

Use the following command to relabel a file system based on the RPM database:

```
fixfiles -R <packagename> restore
```

Using **`fixfiles`** to restore contexts from packages is safer and quicker.

Warning

Running **`fixfiles`** on the entire file system without rebooting may make the system unstable.

If the relabeling operation applies a new policy that is different from the policy that was in place when the system booted, existing processes may be running in incorrect and insecure domains. For example, a process could be in a domain that is not an allowed transition for that process in the new policy, granting unexpected permissions to that process alone.

In addition, one of the options to **`fixfiles relabel`** prompts for approval to empty **`/tmp/`** because it is not possible to reliably relabel **`/tmp/`**. Since **`fixfiles`** is run as root, temporary files that applications are relying upon are erased. This could make the system unstable or behave unexpectedly.

50.2.3. Managing NFS Home Directories

In Red Hat Enterprise Linux 5, most targeted daemons do not interact with user data and are not affected by NFS-mounted home directories. One exception is the Apache HTTP Server. For example, CGI scripts that are on the mounted file system have the **`nfs_t`** type, which is not a type that **`httpd_t`** is allowed to execute.

If you are having problems with the default type of **`nfs_t`**, try mounting the home directories with a different context:

```
mount -t nfs -o context=user_u:object_r:user_home_dir_t \
 fileserver.example.com:/shared/homes/ /home
```

Warning

Section 50.2.9, "Specifying the Security Context of Entire File Systems" explains how to mount a directory so that **`httpd`** can execute scripts. If you do this for user home directories, it gives the Apache HTTP Server increased access to those directories. Remember that a mountpoint label applies to the entire mounted file system.

Future versions of the SELinux policy address the functionality of NFS.

50.2.4. Granting Access to a Directory or a Tree

Similar to standard Linux DAC permissions, a targeted daemon must have SELinux permissions to be able to descend the directory tree. This does not mean that a directory and its contents need to have the same type. There are many types, such as **`root_t`**, **`tmp_t`**, and **`usr_t`** that grant read access for a directory. These types are suitable for directories that do not contain any confidential information, and that you want to be widely readable. They could also be used for a parent directory of more secured directories with different contexts.

If you are working with an **`avc: denied`** message, there are some common problems that arise with directory traversal. For example, many programs run a command equivalent to **`ls -l /`** that is not necessary to their operation but generates a denial message in the logs. For this you need to create a **`dontaudit`** rule in your **`local.te`** file.

When trying to interpret AVC denial messages, do not be misled by the **`path=/`** component. This path is not related to the label for the root file system, **`/`**. It is actually relative to the root of the file system on the device node. For example, if your **`/var/`** directory is located on an LVM (*Logical Volume Management* [22]) device, **`/dev/dm-0`**, the device node is identified in the message as **`dev=dm-0`**. When you see **`path=/`** in this example, that is the top level of the LVM device **`dm-0`**, not necessarily the same as the root file system designation **`/`**.

50.2.5. Backing Up and Restoring the System

Refer to the explanation in Section 50.1.4, "Creating Archives That Retain Security Contexts".

50.2.6. Enabling or Disabling Enforcement

You can enable and disable SELinux enforcement at runtime or configure it to start in the correct mode at boot time, using the command line or GUI. SELinux can operate in one of three modes: *disabled* , meaning not enabled in the kernel; *permissive* , meaning SELinux is running and logging but not controlling permissions; or *enforcing* , meaning SELinux is running and enforcing policy.

Use the **setenforce** command to change between permissive and enforcing modes at runtime. Use **setenforce 0** to enter permissive mode; use **setenforce 1** to enter enforcing mode.

The **sestatus** command displays the current mode and the mode from the configuration file referenced during boot:

```
~]# sestatus | grep -i mode
Current mode:            permissive
Mode from config file:   permissive
```

Note that changing the runtime enforcement does not affect the boot time configuration:

```
~]# setenforce 1
~]# sestatus | grep -i mode
Current mode:            enforcing
Mode from config file:   permissive
```

You can also disable enforcing mode for a single daemon. For example, if you are trying to troubleshoot the **named** daemon and SELinux, you can turn off enforcing for just that daemon.

Use the **getsebool** command to get the current status of the boolean:

```
~]# getsebool named_disable_trans
named_disable_trans --> off
```

Use the following command to disable enforcing mode for this daemon:

```
~]# setsebool named_disable_trans 1
~]# getsebool named_disable_trans
named_disable_trans --> on
```

Note

This sets the runtime value only. Use the **-P** option to make the change persistent across reboots.

Any *_disable_trans booleans that are set to "on" invoke the conditional that prevents the process from transitioning to the domain on execution.

Use the following command to find which of these booleans are set:

```
~]# getsebool -a | grep disable.*on
httpd_disable_trans=1
mysqld_disable_trans=1
ntpd_disable_trans=1
```

You can set any number of boolean values using the **setsebool** command:

```
setsebool -P httpd_disable_trans=1 mysqld_disable_trans=1
ntpd_disable_trans=1
```

You can also use **togglesebool *<boolean_name>*** to change the value of a specific boolean:

```
~]# getsebool httpd_disable_trans
httpd_disable_trans --> off
~]# togglesebool httpd_disable_trans
httpd_disable_trans: active
```

You can configure all of these settings using **system-config-selinux**. The same configuration files are used, so changes appear bidirectionally.

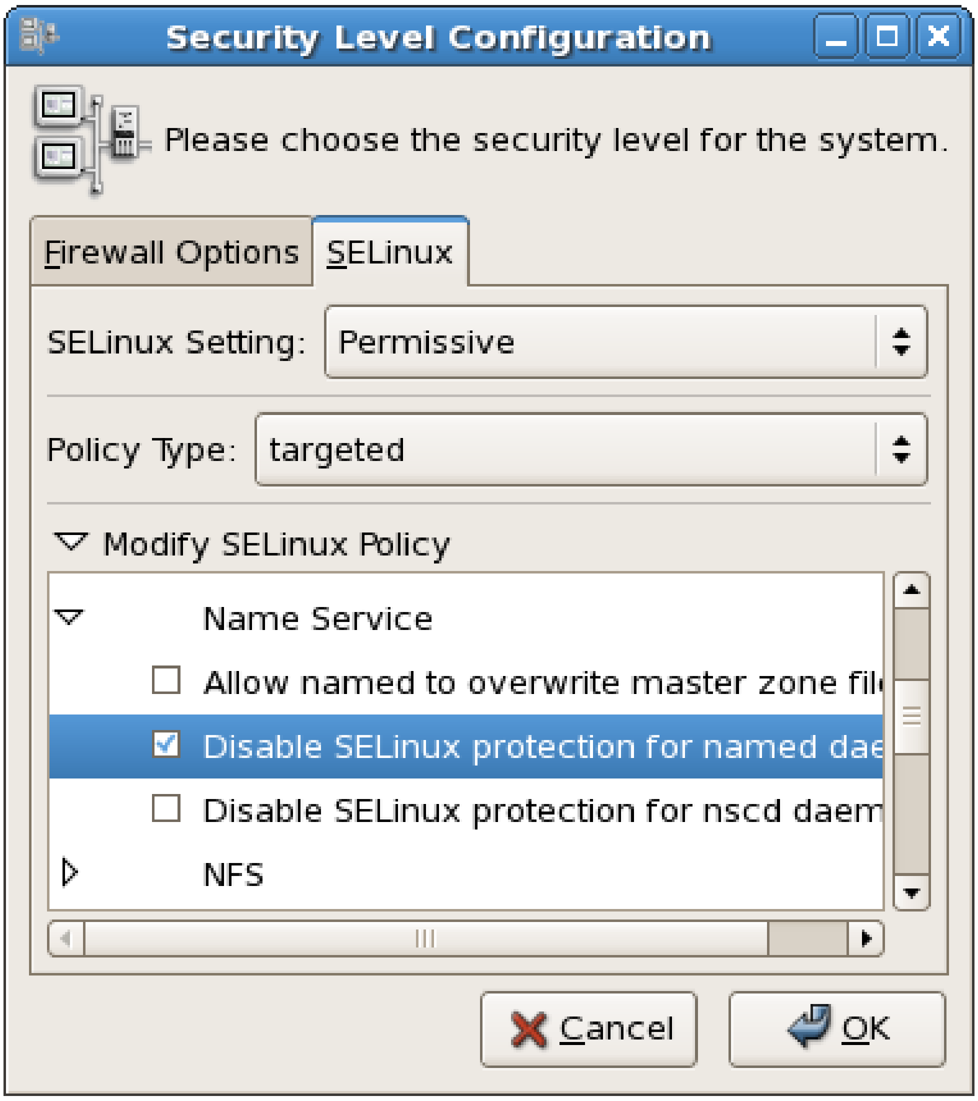
Changing a Runtime Boolean

Use the following procedure to change a runtime boolean using the GUI.

Note

Administrator privileges are required to perform this procedure.

1. On the **System** menu, point to **Administration** and then click **Security Level and Firewall** to display the Security Level Configuration dialog box.
2. Click the **`SELinux`** tab, and then click **`Modify SELinux Policy`**.
3. In the selection list, click the arrow next to the **`Name Service`** entry, and select the **`Disable SELinux protection for named daemon`** check box.
4. Click **`OK`** to apply the change. Note that it may take a short time for the policy to be reloaded.


Security Level Configuration
Please choose the security level for the system.
Firewall Options
SELinux
Modify SELinux Policy
Cancel
OK

SELinux Setting: Permissive

Policy Type: targeted

- Name Service
  - [ ] Allow named to overwrite master zone fil
  - [✓] Disable SELinux protection for named dae
  - [ ] Disable SELinux protection for nscd daem
- NFS

Figure 50.1. Using the Security Level Configuration dialog box to change a runtime boolean.

If you want to control these settings with scripts, you can use the **`setenforce(1)`**, **`getenforce(1)`**, and **`selinuxenabled(1)`** commands.

50.2.7. Enable or Disable SELinux

Important

Changes you make to files while SELinux is disabled may give them an unexpected security label, and new files will not have a label. You may need to relabel part or all of the file system after re-enabling SELinux.

From the command line, you can edit the **`/etc/sysconfig/selinux`** file. This file is a symlink to **`/etc/selinux/config`**. The configuration file is self-explanatory. Changing the value of ***`SELINUX`*** or ***`SELINUXTYPE`*** changes the state of SELinux and the name of the policy to be used the next time the system boots.

```
~]# cat /etc/sysconfig/selinux
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
```

```
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - SELinux is fully disabled.
SELINUX=permissive
# SELINUXTYPE= type of policy in use. Possible values are:
#       targeted - Only targeted network daemons are protected.
#       strict - Full SELinux protection.
SELINUXTYPE=targeted

# SETLOCALDEFS= Check local definition changes
SETLOCALDEFS=0
```

Changing the Mode of SELinux Using the GUI

Use the following procedure to change the mode of SELinux using the GUI.

Note

You need administrator privileges to perform this procedure.

1. On the **System** menu, point to **Administration** and then click **Security Level and Firewall** to display the Security Level Configuration dialog box.

2. Click the **`SELinux`** tab.

3. In the **`SELinux Setting`** select either **`Disabled`**, **`Enforcing`** or **`Permissive`**, and then click **`OK`**.

4. If you changed from **`Enabled`** to **`Disabled`** or vice versa, you need to restart the machine for the change to take effect.

Changes made using this dialog box are immediately reflected in **`/etc/sysconfig/selinux`**.

50.2.8. Changing the Policy

This section provides a brief introduction to using customized policies on your system. A full discussion of this topic is beyond the scope of this document.

To load a different policy on your system, change the following line in **`/etc/sysconfig/selinux`**:

```
SELINUXTYPE=<policyname>
```

where <*policyname*> is the policy name directory under **`/etc/selinux/`**. This assumes that you have the custom policy installed. After changing the ***`SELINUXTYPE`*** parameter, run the following commands:

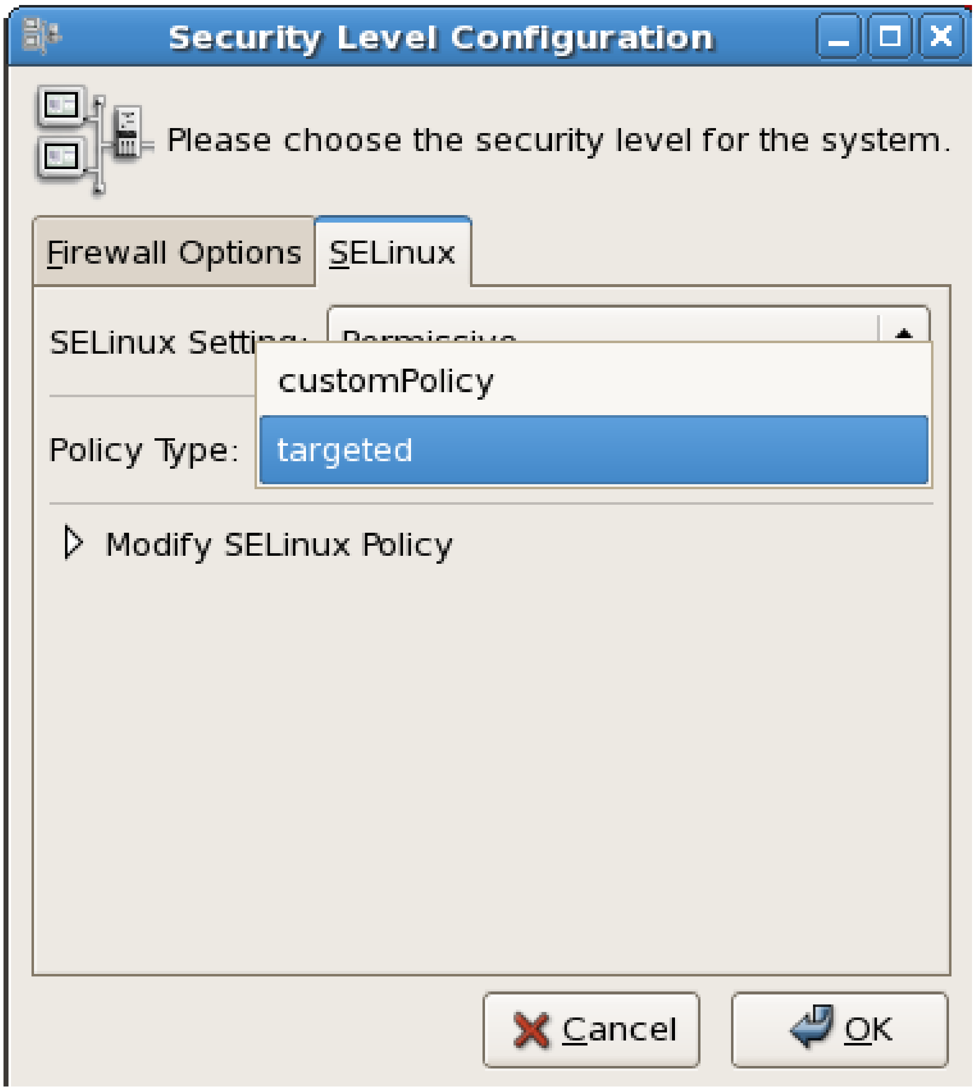
```
touch /.autorelabel
reboot
```

Use the following procedure to load a different policy using the **system-config-selinux** utility:

Note

You need administrator privileges to perform this procedure.

1. Ensure that the complete directory structure for the required policy exists under **`/etc/selinux`**.
2. On the **System** menu, point to **Administration** and then click **Security Level and Firewall** to display the Security Level Configuration dialog box.
3. Click the **`SELinux`** tab.
4. In the **`Policy Type`** list, select the policy that you want to load, and then click **`OK`**. This list is only visible if more than one policy is installed.
5. Restart the machine for the change to take effect.


Security Level Configuration

Please choose the security level for the system.

Firewall Options | SELinux

SELinux Setting: Permissive

Policy Type: customPolicy / targeted

Modify SELinux Policy

Cancel | OK

Figure 50.2. Using the Security Level Configuration dialog box to load a custom policy.

50.2.9. Specifying the Security Context of Entire File Systems

You can use the **`mount -o context=`** command to set a single context for an entire file system. This might be a file system that is already mounted and that supports xattrs, or a network file system that obtains a genfs label such as **`cifs_t`** or **`nfs_t`**.

For example, if you need the Apache HTTP Server to read from a mounted directory or loopback file system, you need to set the type to **`httpd_sys_content_t`**:

```
mount -t nfs -o context=system_u:object_r:httpd_sys_content_t \
 server1.example.com:/shared/scripts /var/www/cgi
```

Note

When troubleshooting **`httpd`** and SELinux problems, reduce the complexity of your situation. For example, if you have the file system mounted at **`/mnt`** and then symbolically linked to **`/var/www/html/foo`**, you have two security contexts to be concerned with. Because one security context is of the object class file and the other of type lnk_file, they are treated differently by the policy and unexpected behavior may occur.

50.2.10. Changing the Security Category of a File or User

Refer to Section 49.5.5, "Assigning Categories to Files" and Section 49.5.4, "Assigning Categories to Users" for information about adding and changing the security categories of files and users.

50.2.11. Running a Command in a Specific Security Context

You can use the **`runcon`** command to run a command in a specific context. This is useful for scripting or for testing policy, but care should be taken to ensure that it is implemented correctly.

For example, you could use the following command to run a script to test for mislabeled content. The arguments that appear after the command are considered to be part of the command. (In this example, **`~/bin/contexttest`** is a user-defined script.)

```
runcon -t httpd_t ~/bin/contexttest -ARG1 -ARG2
```

You can also specify the entire context, as follows:

```
runcon user_u:system_r:httpd_t ~/bin/contexttest
```

50.2.12. Useful Commands for Scripts

The following is a list of useful commands introduced with SELinux, and which you may find useful when writing scripts to help administer your system:

`getenforce`

This command returns the enforcing status of SELinux.

`setenforce [ *Enforcing* | *Permissive* | *1* | *0* ]`

This command controls the enforcing mode of SELinux. The option **`1`** or **`Enforcing`** tells SELinux to enter enforcing mode. The option **`0`** or **`Permissive`** tells SELinux to enter passive mode. Access violations are still logged, but not prevented.

`selinuxenabled`

This command exits with a status of **`0`** if SELinux is enabled, and **`1`** if SELinux is disabled.

```
~]# selinuxenabled
~]# echo $?
0
```

`getsebool [-a] [`*`boolean_name`*`]`

This command shows the status of all booleans (**`-a`**) or a specific boolean (**`<boolean_name>`**).

`setsebool [-P] <boolean_name> value | bool1=val1 bool2=val2 ...`

This command sets one or more boolean values. The **`-P`** option makes the changes persistent across reboots.

`togglesebool boolean ...`

This command toggles the setting of one or more booleans. This effects boolean settings in memory only; changes are not persistent across reboots.

50.2.13. Changing to a Different Role

You use the **`newrole`** command to run a new shell with the specified type and/or role. Changing roles is typically only meaningful in the strict policy; the targeted policy is generally restricted to a single role. Changing types may be useful for testing, validation, and development purposes.

```
newrole -r <role_r> -t <type_t> [-- [ARGS]...]
```

The **`ARGS`** are passed directly to the shell specified in the user's entry in the **`/etc/passwd`** file.

Note

The **`newrole`** command is part of the **`policycoreutils-newrole`** package, which is required if you install the strict or MLS policy. It is not installed by default in Red Hat Enterprise Linux.

50.2.14. When to Reboot

The primary reason for rebooting the system from an SELinux perspective is to completely relabel the file system. On occasion you might need to reboot the system to enable or disable SELinux.

50.3. Analyst Control of SELinux

This section describes some common tasks that a security analyst might need to perform on an SELinux system.

50.3.1. Enabling Kernel Auditing

As part of an SELinux analysis or troubleshooting exercise, you might choose to enable complete kernel-level auditing. This can be quite verbose, because it generates one or more additional audit messages for each AVC audit message. To enable this level of auditing, append the ***`audit=1`*** parameter to your kernel boot line, either in the **`/etc/grub.conf`** file or on the GRUB menu at boot time.

This is an example of a full audit log entry when **`httpd`** is denied access to **`~/public_html`** because the directory is not labeled as Web content. Notice that the time and serial number stamps in the audit(...) field are identical in each case. This makes it easier to track a specific event in the audit logs:

```
Jan 15 08:03:56 hostname kernel: audit(1105805036.075:2392892): \
 avc:  denied  { getattr } for  pid=2239 exe=/usr/sbin/httpd \
 path=/home/auser/public_html dev=hdb2 ino=921135 \
 scontext=user_u:system_r:httpd_t \
 tcontext=system_u:object_r:user_home_t tclass=dir
```

The following audit message tells more about the source, including the kind of system call involved, showing that httpd tried to stat the directory:

```
Jan 15 08:03:56 hostname kernel: audit(1105805036.075:2392892): \
 syscall=195 exit=4294967283 a0=9ef88e0 a1=bfecc0d4 a2=a97ff4 \
 a3=bfecc0d4 items=1 pid=2239 loginuid=-1 uid=48 gid=48 euid=48 \
 suid=48 fsuid=48 egid=48 sgid=48 fsgid=48
```

The following message provides more information about the target:

```
Jan 15 08:03:56 hostname kernel: audit(1105805036.075:2392892): \
 item=0 name=/home/auser/public_html inode=921135 dev=00:00
```

The serial number stamp is always identical for a particular audited event. The time stamp may or may not be identical.

Note

If you are using an audit daemon for troubleshooting, the daemon may capture audit messages into a location other than **`/var/log/messages`**, such as **`/var/log/audit/audit.log`**.

50.3.2. Dumping and Viewing Logs

The Red Hat Enterprise Linux 5 implementation of SELinux routes AVC audit messages to **`/var/log/messages`**. You can use any of the standard search utilities (for example, **`grep`**), to search for lines containing **`avc`** or **`audit`**.

[22] LVM is the grouping of physical storage into virtual pools that are partitioned into logical volumes.

Chapter 51. Customizing SELinux Policy

51.1. Introduction

In earlier releases of Red Hat Enterprise Linux it was necessary to install the **`selinux-policy-targeted-sources`** packages and then to create a **`local.te`** file in the **`/etc/selinux/targeted/src/policy/domains/misc`** directory. You could use the **`audit2allow`** utility to translate the AVC messages into allow rules, and then rebuild and reload the policy.

The problem with this was that every time a new policy package was released it would have to execute the Makefile in order to try to keep the local policy.

In Red Hat Enterprise Linux 5, this process has been completely revised. The "sources" rpm packages have been completely removed, and policy packages are treated more like the kernel. To look at the sources used to build the policy, you need to install the source rpm, **`selinux-policy-XYZ.src.rpm`**. A further package, **`selinux-policy-devel`**, has also been added, which provides further customization functionality.

51.1.1. Modular Policy

Red Hat Enterprise Linux introduces the concept of *modular policy*. This allows vendors to ship SELinux policy separately from the operating system policy. It also allows administrators to make local changes to policy without worrying about the next policy install. The most important command that was added was **`semodule`**.

`semodule` is the tool used to manage SELinux policy modules, including installing, upgrading, listing and removing modules. You can also use **`semodule`** to force a rebuild of policy from the module store and/or to force a reload of policy without performing any other transaction. **`semodule`** acts on module packages created by **`semodule_package`**. Conventionally, these files have a .pp suffix (policy package), although this is not mandated in any way.

51.1.1.1. Listing Policy Modules

To list the policy modules on a system, use the **`semodule -l`** command:

```
~]# semodule -l
amavis  1.1.0
ccs     1.0.0
clamav  1.1.0
dcc     1.1.0
evolution       1.1.0
iscsid  1.0.0
mozilla 1.1.0
mplayer 1.1.0
nagios  1.1.0
oddjob  1.0.1
pcscd   1.0.0
pyzor   1.1.0
razor   1.1.0
ricci   1.0.0
smartmon        1.1.0
```

Note

This command does not list the base policy module, which is also installed.

The **`/usr/share/selinux/targeted/`** directory contains a number of policy package (*.pp) files. These files are included in the **`selinux-policy`** rpm and are used to build the policy file.

51.2. Building a Local Policy Module

The following section uses an actual example to demonstrate building a local policy module to address an issue with the current policy. This issue involves the **`ypbind init`** script, which executes the **`setsebool`** command, which in turn tries to use the terminal. This is generating the following denial:

```
type=AVC msg=audit(1164222416.269:22): avc:  denied  { use } for  pid=1940
comm="setsebool" name="0" dev=devpts ino=2 \
 scontext=system_u:system_r:semanage_t:s0
tcontext=system_u:system_r:init_t:s0 tclass=fd
```

Even though everything still works correctly (that is, it is not preventing any applications form running as intended), it does interrupt the normal work flow of the user. Creating a local policy module addresses this issue.

51.2.1. Using audit2allow to Build a Local Policy Module

The **`audit2allow`** utility now has the ability to build policy modules. Use the following command to build a policy module based on specific contents of the **`audit.log`** file:

```
ausearch -m AVC --comm setsebool | audit2allow -M mysemanage
```

The **`audit2allow`** utility has built a type enforcement file (**`mysemanage.te`**). It then executed the **`checkmodule`** command to compile a module file (**`mysemanage.mod`**). Lastly, it uses the **`semodule_package`** command to create a policy package (**`mysemanage.pp`**). The **`semodule_package`** command combines different policy files (usually just the module and potentially a file context file) into a policy package.

51.2.2. Analyzing the Type Enforcement (TE) File

Use the **`cat`** command to inspect the contents of the TE file:

```
~]# cat mysemanag.te
module mysemanage 1.0;

require {
 class fd use;
 type init_t;
 type semanage_t;
 role system_r;
};

allow semanage_t init_t:fd use;
```

The TE file is comprised of three sections. The first section is the **`module`** command, which identifies the module name and version. The module name must be unique. If you create an **`semanage`** module using the name of a pre-existing module, the system would try to replace the existing module package with the newly-created version. The last part of the module line is the version. **`semodule`** can update module packages and checks the update version against the currently installed version.

The next block of the TE file is the **`require`** block. This informs the policy loader which types, classes and roles are required in the system policy before this module can be installed. If any of these fields are undefined, the **`semodule`** command will fail.

Lastly are the allow rules. In this example, you could modify this line to **`dontaudit`**, because **`semodule`** does not need to access the file descriptor.

51.2.3. Loading the Policy Package

The last step in the process of creating a local policy module is to load the policy package into the kernel.

Use the **`semodule`** command to load the policy package:

```
~]# semodule -i mysemanage.pp
```

This command recompiles the policy file and regenerates the file context file. The changes are permanent and will survive a reboot. You can also copy the policy package file (**`mysemanage.pp`**) to other machines and install it using **`semodule`**.

The **`audit2allow`** command outputs the commands it executed to create the policy package so that you can edit the TE file. This means you can add new rules as required or change the **`allow`** rule to **`dontaudit`**. You could then recompile and repackage the policy package to be installed again.

There is no limit to the number of policy packages, so you could create one for each local modification you want to make. Alternatively, you could continue to edit a single package, but you need to ensure that the "require" statements match all of the allow rules.

Chapter 52. References

The following references are pointers to additional information that is relevant to SELinux and Red Hat Enterprise Linux but beyond the scope of this guide. Note that due to the rapid development of SELinux, some of this material may only apply to specific releases of Red Hat Enterprise Linux.

Books

SELinux by Example

Mayer, MacMillan, and Caplan

Prentice Hall, 2007

Tutorials and Help

Understanding and Customizing the Apache HTTP SELinux Policy

http://docs.fedoraproject.org/selinux-apache-fc3/

Tutorials and talks from Russell Coker

http://www.coker.com.au/selinux/talks/ibmtu-2004/

Generic Writing SELinux policy HOWTO

https://sourceforge.net/docman/display_doc.php?docid=21959[amp]group_id=21266

Red Hat Knowledgebase

http://kbase.redhat.com/

General Information

NSA SELinux main website

http://www.nsa.gov/research/selinux/index.shtml

NSA SELinux FAQ

http://www.nsa.gov/research/selinux/faqs.shtml

Fedora SELinux FAQ

http://docs.fedoraproject.org/selinux-faq/

SELinux NSA's Open Source Security Enhanced Linux

http://www.oreilly.com/catalog/selinux/

Technology

An Overview of Object Classes and Permissions

http://www.tresys.com/selinux/obj_perms_help.html

Integrating Flexible Support for Security Policies into the Linux Operating System (a history of Flask implementation in Linux)

http://www.nsa.gov/research/_files/selinux/papers/freenix01/freenix01.shtml

Implementing SELinux as a Linux Security Module

http://www.nsa.gov/research/selinux/index.shtmlpapers/module-abs.cfm

A Security Policy Configuration for the Security-Enhanced Linux

http://www.nsa.gov/research/_files/selinux/papers/policy/policy.shtml

Community

SELinux community page

http://selinux.sourceforge.net

IRC

irc.freenode.net, #rhel-selinux

History

Quick history of Flask

http://www.cs.utah.edu/flux/fluke/html/flask.html

Full background on Fluke

http://www.cs.utah.edu/flux/fluke/html/index.html

Part VIII. Red Hat Training And Certification

Red Hat courses and certifications are indisputably regarded as the best in Linux, and perhaps in all of IT. Taught entirely by experienced Red Hat experts, our certification programs measure competency on actual live systems and are in great demand by employers and IT professionals alike.

Choosing the right certification depends on your background and goals. Whether you have advanced, minimal, or no UNIX or Linux experience whatsoever, Red Hat Training has a training and certification path that is right for you.

Chapter 53. Red Hat Training and Certification

53.1. Three Ways to Train

Open Enrollment

Open enrollment courses are offered continually in 50+ locations across North America and 125+ locations worldwide. Red Hat courses are performance—based—students have access to at least one dedicated system, and in some courses, as many as five. Instructors are all experienced Red Hat Certified Engineers (RHCEs) who are intimately familiar with course curriculum.

Course schedules are available at http://www.redhat.com/explore/training

Onsite Training

Onsite training is delivered by Red Hat at your facility for teams of 12 to 16 people per class. Red Hat's technical staff will assist your technical staff prior to arrival to ensure the training venue is prepared to run Red Hat Enterprise Linux, Red Hat or JBoss courses, and/or Red Hat certification exams. Onsites are a great way to train large groups at once. Open enrollment can be leveraged later for incremental training.

For more information, visit http://www.redhat.com/explore/onsite

eLearning

Fully updated for Red Hat Enterprise Linux 4! No time for class? Red Hat's e—Learning titles are delivered online and cover RHCT and RHCE track skills. Our growing catalog also includes courses on the latest programming languages, scripting and ecommerce.

For course listings visit http://www.redhat.com/explore/elearning

53.2. Microsoft Certified Professional Resource Center

Tailored info and offers for Microsoft® Certified Professionals looking to add a Red Hat certification to their personal portfolio.

Check it out today: http://www.redhat.com/explore/manager

Chapter 54. Certification Tracks

Red Hat Certified Technician® (RHCT®)

Now entering its third year, Red Hat Certified Technician is the fastest-growing credential in all of Linux, with currently over 15,000 certification holders. RHCT is the best first step in establishing Linux credentials and is an ideal initial certification for those transitioning from non-UNIX®/ Linux environments.

Red Hat certifications are indisputably regarded as the best in Linux, and perhaps, according to some, in all of IT. Taught entirely by experienced Red Hat experts, our certification programs measure competency on actual live systems and are in great demand by employers and IT professionals alike.

Choosing the right certification depends on your background and goals. Whether you have advanced, minimal, or no UNIX or Linux experience whatsoever, Red Hat Training has a training and certification path that is right for you.

Red Hat Certified Engineer® (RHCE®)

Red Hat Certified Engineer began in 1999 and has been earned by more than 20,000 Linux experts. Called the "crown jewel of Linux certifications," independent surveys have ranked the RHCE program #1 in all of IT.

Red Hat Certified Security Specialist (RHCSS)

An RHCSS has RHCE security knowledge plus specialized skills in Red Hat Enterprise Linux, Red Hat Directory Server and SELinux to meet the security requirements of today's enterprise environments. RHCSS is Red Hat's newest certification, and the only one of its kind in Linux.

Red Hat Certified Architect (RHCA)

RHCEs who seek advanced training can enroll in Enterprise Architect courses and prove their competency with the newly announced Red Hat Certified Architect (RHCA) certification. RHCA is the capstone certification to Red Hat Certified Technician (RHCT) and Red Hat Certified Engineer (RHCE), the most acclaimed certifications in the Linux space.

54.1. Free Pre-assessment tests

Test your Linux smarts and identify your Red Hat course level with our automated pre-assessment tests.

Completely free, no obligations, 10 minutes of your time. http://www.redhat.com/explore/assess

Chapter 55. RH033: Red Hat Linux Essentials

http://www.redhat.com/training/rhce/courses/rh033.html

55.1. Course Description

The first course for both RHCT and RHCE certification tracks, RH033 is ideal for individuals who have never used Linux or UNIX, and who have no prior command line experience in any other operating system. You are taught the basics of a Red Hat Enterprise Linux environment, and it prepares you for your future role as a system administrator.

55.1.1. Prerequisites

User-level experience with any computer system, use of menus, use of any graphical user interface.

55.1.2. Goal

A Red Hat Enterprise Linux power user who can be productive in using and customizing a Red Hat system fo common command line processes and desktop productivity roles, and who is ready to learn system administration (RH133).

55.1.3. Audience

Users who are new to Linux and have no prior UNIX or command line skills, who want to develop and practice the basic skills to use and control their own Red Hat Linux system.

55.1.4. Course Objectives

1. Understand the Linux file system
2. Perform common file maintenance
3. Use and customize the GNOME interface
4. Issue essential Linux commands from the command line
5. Perform common tasks using the GNOME GUI
6. Open, edit, and save text documents using the vi editor
7. File access permissions
8. Customize X Window System
9. Regular expression pattern matching and I/O redirection
10. Install, upgrade, delete and query packages on your system
11. Network utilities for the user
12. Power user utilities

55.1.5. Follow-on Courses

RH133 Red Hat Linux Sys. Admin.

RH253 Red Hat Linux Net. and Sec. Admin

RH300 Red Hat Linux RHCE Rapid Track

"I would enthusiastically recommend this course to anyone interested in Linux."——Mike Kimmel, ITT Systems Division

Chapter 56. RH035: Red Hat Linux Essentials for Windows Professionals

http://www.redhat.com/training/rhce/courses/rh035.html

56.1. Course Description

Designed for Windows® professionals with no prior UNIX or Linux experience, this course teaches fundamental Red Hat Enterprise Linux system administration skills. The first day provides a conceptual and practical transition for individuals to successfully add Linux management competencies to their portfolio. The remaining four days combines with the highly-acclaimed RH033 course, immersing individuals in the basics of a Red Hat Enterprise Linux environment and preparing them for future roles as cross-platform system administrators. The course also serves as the first course in the RHCT and RHCE tracks.

56.1.1. Prerequisites

Have experience with job tasks using Windows OS products at technician or system administrator level; experience as an IT professional; no prior UNIX or Linux experience required.

56.1.2. Goal

A Red Hat Enterprise Linux power user familiar with common command line processes who can perform some system administration tasks using graphical tools. The individual will also be ready to develop a deeper understanding of Red Hat Enterprise Linux system administration (RH133).

56.1.3. Audience

The typical student will be a Windows technician who prefers to manage servers using a graphic user interface. The individual will also possess a desire to effectively manage Red Hat Enterprise Linux systems and broaden their individual skill set.

56.1.4. Course Objectives

1. Learn to install software, configure the network, configure authentication, and install and configure various services using graphical tools
2. Understand the Linux file system
3. Issue essential Linux commands from the command line
4. Understand file access permissions
5. Customize X Window System
6. Use regular expression pattern matching and I/O redirection

56.1.5. Follow-on Courses

RH133 Red Hat Linux Sys. Admin. (p. 8)

RH253 Red Hat Linux Net. and Sec. Admin. (p. 9)

RH300 Red Hat Linux RHCE Rapid Track (p. 10)

"All in all I would rate this training experience as one of the best I have ever attended, and I've been in this industry for over 15 years." — Bill Legge, IT Consultant

Chapter 57. RH133: Red Hat Linux System Administration and Red Hat Certified Technician (RHCT) Certification

http://www.redhat.com/training/rhce/courses/rh133.html

57.1. Course Description

RH133 focuses on skills in systems administration on Red Hat Linux, to a level where you can attach and configure a workstation on an existing network. This 4.5-day course provides intensive hands-on training on Red Hat Enterprise Linux, and includes the RH202 RHCT Certification Lab Exam on the last day.

57.1.1. Prerequisites

RH033 Red Hat Linux Essentials or equivalent experience with Red Hat Linux.

57.1.2. Goal

Upon successful completion of this course, students will possess basic Linux system administrator knowledge which can be proved by passing the RHCT Exam. The exam is a performance-based lab exam that tests actual ability to install, configure, and attach a new Red Hat Linux system to an existing production network.

57.1.3. Audience

Linux or UNIX users who understand the basics of Red Hat Linux and desire further technical training to begir the process of becoming a system administrator.

57.1.4. Course Objectives

1. Install Red Hat Linux interactively and with Kickstart
2. Control common system hardware; administer Linux printing subsystem
3. Create and maintain the Linux filesystem
4. Perform user and group administration
5. Integrate a workstation with an existing network
6. Configure a workstation as a client to NIS, DNS, and DHCP services
7. Automate tasks with at, cron, and anacron
8. Back up filesystems to tape and tar archive
9. Manipulate software packages with RPM
10. Configure the X Window System and the GNOME d.e.
11. Perform performance, memory, and process mgmt.
12. Configure basic host security

57.1.5. Follow-on Courses

RH253 Red Hat Linux Net. and Sec. Admin. (p. 9)

Chapter 58. RH202 RHCT EXAM - The fastest growing credential in all of Linux.

http://www.redhat.com/training/rhce/courses/rh202.html

1. RHCT exam is included with RH133. It can also be purchased on its own for $349
2. RHCT exams occur on the fifth day of all RH133 classes

58.1. Course Description

The RHCT (Red Hat Certified Technician) is a hands-on, performance-based exam testing candidates actual skills in installing, configuring, and troubleshooting Red Hat Enterprise Linux. The Certification Lab Exam is bundled with RH133, but individuals who have mastered the content of RH033 and RH133 can take just the exam.

58.1.1. Prerequisites

Candidates should consider taking RH033 and RH133 in preparation for the exam, but they are not required to take it.

Chapter 59. RH253 Red Hat Linux Networking and Security Administration

59.1. Course Description

RH253 arms students with in-depth knowledge needed to configure common Red Hat Enterprise Linux network services. Network and local security tasks are also topics of this course.

59.1.1. Prerequisites

RH133 Red Hat Linux System Administration or equivalent experience with Red Hat Enterprise Linux, LAN/WAN fundamentals or equivalent, internetworking with TCP/IP or equivalent.

59.1.2. Goal

Upon completion of this course, individuals can set up a Red Hat Enterprise Linux server and configure common network services and security at a basic level.

59.1.3. Audience

Linux or UNIX system administrators who already have some real-world experience with Red Hat Enterprise Linux systems administration, want a first course in networking services and security, and want to build skills at configuring common network services and security administration using Red Hat Enterprise Linux.

59.1.4. Course Objectives

1. Networking services on Red Hat Linux server-side setup, configuration, and basic administration of common networking services: DNS, NIS, Apache, SMB, DHCP, Sendmail, FTP. Other common services: tftp, pppd, proxy.
2. Introduction to security
3. Developing a security policy
4. Local security
5. Files and filesystem security
6. Password security
7. Kernel security
8. Basic elements of a firewall
9. Red Hat Linux-based security tools
10. Responding to a break-in attempt
11. Security sources and methods
12. Overview of OSS security tools

59.1.5. Follow-on Courses

RH302 RHCE Certification Exam

"This course was excellent. The teacher was fantastic—his depth of knowledge is amazing."——Greg Peters, Future Networks USA

Chapter 60. RH300: RHCE Rapid track course (and RHCE exam)

The fastest path to RHCE certification for experienced UNIX/Linux users.

http://www.redhat.com/training/rhce/courses/rh300.html

60.1. Course Description

Five days in duration, this course provides intensive hands-on training on Red Hat Linux, and includes the RHCE Certification Exam on the last day.

60.1.1. Prerequisites

RH033, RH133, RH253 or equivalent experience with UNIX. Please do not register for RH300 unless you are experienced with systems administration or are a power user in UNIX or Linux environments.

60.1.2. Goal

Upon successful completion of this course, individuals will be a Red Hat Linux system administrator who has been trained and then tested using the RHCE Exam.

60.1.3. Audience

UNIX or Linux system administrators who have significant real-world experience and who want a fast-track course to prepare for the RHCE Exam.

60.1.4. Course Objectives

1. Hardware and Installation (x86 architecture)
2. Configuration and administration
3. Alternate installation methods
4. Kernel services and configuration
5. Standard networking services
6. X Window system
7. User and host security
8. Routers, Firewalls, Clusters and Troubleshooting

60.1.5. Follow-on Courses

Enterprise Architect curriculum and RHCA certification

Chapter 61. RH302 RHCE EXAM

1. RHCE exams are included with RH300. It can also be purchased on its own.
2. RHCE exams occur on the fifth day of all RH300 classes

http://www.redhat.com/training/rhce/courses/rhexam.html

61.1. Course Description

RHCE stands apart from many other certification programs in the IT sector because of its emphasis on hands-on, performance-based testing of actual skills in Red Hat Linux installation, configuration, debugging, and setup of key networking services.

61.1.1. Prerequisites

See RH300 course prerequisites. For further information, please refer to the RHCE Exam Prep Guide: www.redhat.com/training/rhce/examprep.html

61.1.2. Content

1. Section I: Troubleshooting and System Maintenance (2.5 hrs)
2. Section II: Installation and Configuration (3 hrs.)

"Seriously, this was an outstanding class. I feel very well prepared for the test tomorrow." — Logan Ingalls, Web developer, Texterity Inc., USA

Chapter 62. RHS333: RED HAT enterprise security: network services

Security for the most commonly deployed services.

http://www.redhat.com/training/architect/courses/rhs333.html

62.1. Course Description

Red Hat Enterprise Linux has gained considerable momentum as the operating system of choice for deploying network services such as web, ftp, email, and file sharing. Red Hat's RHCE curriculum provides training in deploying these services and on the essential elements of securing them.

62.1.1. Prerequisites

RH253, RH300, or RHCE certification or equivalent work experience is required for this course. Course participants should already know the essential elements of how to configure the services covered, as this course will be focusing on more advanced topics from the outset.

62.1.2. Goal

This class advances beyond the essential security coverage offered in the RHCE curriculum and delves deeper into the security features, capabilities, and risks associated with the most commonly deployed services.

62.1.3. Audience

The audience for this course includes system administrators, consultants, and other IT professionals responsible for the planning, implementation, and maintenance of network servers. While the emphasis is on running these services on Red Hat Enterprise Linux, and the content and labs will assume its use, system administrators and others using proprietary forms of UNIX may also find many elements of this course relevant.

62.1.4. Course Objectives

1. Mastering basic service security
2. Understanding cryptography
3. Logging system activity
4. Securing BIND and DNS
5. Network user authentication security
6. Improving NFS security
7. The secure shell: OpenSSH
8. Securing email with Sendmail and Postfix
9. Managing FTP access
10. Apache security

11. Basics of intrusion response

62.1.5. Follow-on Courses

RH401 Red Hat Enterprise Deployment and System Mgmt. RH423 Red Hat Enterprise Directory Services and Authentication RH436 Red Hat Enterprise Storage Mgmt. RH442 Red Hat Enterprise System Monitoring and Performance Tuning

Chapter 63. RH401: Red Hat Enterprise Deployment and systems management

Manage Red Hat Enterprise Linux deployments.

http://www.redhat.com/training/architect/courses/rh401.html

63.1. Course Description

RH401 is a four-day intensive hands-on lab course in skills and methods critical to large-scale deployment and management of mission-critical Red Hat Enterprise Linux systems, including failover and load-balancing, CVS for system administrators, RPM rebuilding, and performance tuning for specific applications.

63.1.1. Prerequisites

RH253 at a minimum, RHCE certification preferred, or comparable skills and knowledge. All prospective course participants without RHCE certification are encouraged to verify skills with Red Hat's free online pre—assessment tests. Note: Persons should not enroll in RH401 without meeting the above prerequisites.

All prospective course participants who do not possess RHCE certification are strongly advised to contact Red Hat Global Learning Services for a skills assessment when they enroll.

63.1.2. Goal

RH401 trains senior system administrators to manage large numbers of Enterprise Linux servers in a variety of roles, and/or manage them for mission—critical applications that require failover and load-balancing. Further, RH401 is benchmarked on expert—level competencies in managing operating systems for enterprise roles—the course teaches how to implement and manage enterprise Red Hat Enterprise Linux deployments efficiently and effectively in ways that make the entire enterprise deployment manageable by a team.

63.1.3. Audience

Senior Red Hat Enterprise Linux system administrators and other IT professionals working in enterprise environments and mission-critical systems.

63.1.4. Course Objectives

1. Configuration management using CVS
2. Construction of custom RPM packages
3. Software management with Red Hat Network Proxy Server
4. Assembling a host provisioning and management system
5. Performance tuning and analysis
6. High-availability network load-balancing clusters
7. High-availability application failover clusters

63.1.5. Follow-on Courses

RHS333 Enterprise Security: Securing Network Services

RH423 Red Hat Enterprise Directory Services and Authentication

RH436 Red Hat Enterprise Storage Mgmt.

RH442 Red Hat Enterprise System Monitoring and Performance Tuning

"After taking RH401 I am completely confident that I can implement enterprise—scale high—availability solutions end-to-end."——Barry Brimer, Bunge North America

Chapter 64. RH423: Red Hat Enterprise Directory services and authentication

Manage and deploy directory services for Red Hat Enterprise Linux systems.

http://www.redhat.com/training/architect/courses/rh423.html

64.1. Course Description

RH423 is an intensive course that provides four days of instruction and labs on cross-platform integration of directory services to provide authentication or information service across the enterprise.

64.1.1. Prerequisites

RH253 at a minimum, RHCE certification preferred, or comparable skills and knowledge. All prospective course participants without RHCE certification are encouraged to verify skills with Red Hat's free online pre—assessment tests. Note: Persons should not enroll in RH423 without meeting the above prerequisites. All prospective course participants who do not possess RHCE certification are strongly advised to contact Red Hat Global Learning Services for a skills assessment when they enroll.

64.1.2. Goal

RH423 trains senior system administrators to manage and deploy directory services on and for Red Hat Enterprise Linux systems. Gaining an understanding of the basic concepts, configuration, and management of LDAP—based services is central to this course. Students will integrate standard network clients and services with the directory service in order to take advantage of its capabilities. We will also look at PAM, the Pluggable Authentication Modules system, and how it is integrated with services that require authentication and authorization.

64.1.3. Audience

Senior Red Hat Enterprise Linux system administrators and other IT professionals working in enterprise environments and mission-critical systems.

64.1.4. Course Objectives

1. Basic LDAP concepts
2. How to configure and manage an OpenLDAP server
3. Using LDAP as a "white pages" directory service
4. Using LDAP for user authentication and management
5. Integrating multiple LDAP servers

64.1.5. Follow-on Courses

RHS333 Enterprise Security: Securing Network Services

RH401 Red Hat Enterprise Deployment and Systems Management

RH436 Red Hat Enterprise Storage Mgmt. (p. 16)

RH442 Red Hat Enterprise System Monitoring and Performance Tuning

Chapter 65. SELinux Courses

65.1. RHS427: Introduction to SELinux and Red Hat Targeted Policy

http://www.redhat.com/training/security/courses/rhs427.html

1-day rapid intro to SELinux, how it operates within the Red Hat targeted policy, and the tools available for working with this powerful capability. RHS427 constitutes the first day of RH429.

65.1.1. Audience

Computer security specialists and others responsible for implementing security policies on a Linux computer. RHS429 requires RHCE or comparable knowledge.

65.1.2. Course Summary

Among the most significant features of Red Hat Enterprise Linux is SELinux (Security Enhanced Linux), a powerful, kernel-level security layer that provides fine-grained control over what users and processes may access and do on a system. By default, SELinux is enabled on Red Hat Enterprise Linux systems, enforcing a set of mandatory access controls that Red Hat calls the targeted policy. These access controls substantially enhance the security of the network services they target, but can sometimes affect the behavior of third-party applications and scripts that worked on previous versions of Red Hat Enterprise Linux.

65.2. RHS429: Red Hat Enterprise SELinux Policy Administration

http://www.redhat.com/training/security/courses/rhs429.html

Among the most significant features of Red Hat Enterprise Linux is SELinux (Security Enhanced Linux), a powerful, kernel-level security layer that provides fine-grained control over what users and processes may access and execute on a system. RHS429 introduces advanced system administrators, security administrators, and applications programmers to SELinux policy writing. Participants in this course will learn how SELinux works; how to manage SELinux; and how to write an SELinux policy.

Chapter 66. RH436: Red Hat Enterprise storage management

Deploy and manage Red Hat's cluster file system technology.

Equipment-intensive:

1. five servers
2. storage array

http://www.redhat.com/training/architect/courses/rh436.html

66.1. Course Description

RH436 provides intensive hands-on experience with the emerging Shared Storage technology delivered by Red Hat Global File System (GFS). This four-day course focuses on the implementation of native Red Hat Enterprise Linux technologies included in Red Hat Cluster Suite and GFS.

66.1.1. Prerequisites

RH253 at a minimum, RHCE certification preferred, or comparable skills and knowledge. All prospective course participants without RHCE certification are encouraged to verify skills with Red Hat's free online pre—assessment tests.

66.1.2. Goal

This course is designed to train people with RHCE-level competency on skills required to deploy and manage highly available storage data to the mission-critical enterprise computing environment. Complementing skills gained in RH401, this course delivers extensive hands-on training with the cluster file system, GFS.

66.1.3. Audience

Senior Red Hat Enterprise Linux system administrators and other IT professionals working in enterprise environments and mission-critical systems.

66.1.4. Course Objectives

1. Review Red Hat Enterprise Linux storage management technologies
2. Data storage design: Data sharing
3. Cluster Suite overview
4. Global File System (GFS) overview
5. GFS management
6. Modify the online GFS environment: Managing data capacity
7. Monitor GFS
8. Implement GFS modifications
9. Migrating Cluster Suite NFS from DAS to GFS
10. Re-visit Cluster Suite using GFS

66.1.5. Follow-on Courses

RHS333 Enterprise Security: Securing Network Services

RH401 Red Hat Enterprise Deployment and Systems Management

RH423 Red Hat Enterprise Directory Services and Authentication

RH442 Red Hat Enterprise System Monitoring and Performance Tuning

"The class gave me a chance to use some of the latest Linux tools, and was a reminder of the benefits of using Linux for high-availability systems."——Paul W. Frields, FBI — Operational Technology Division Quantico, VA, USA

Chapter 67. RH442: Red Hat Enterprise system monitoring and performance tuning

Performance tuning and capacity planning for Red Hat Enterprise Linux

http://www.redhat.com/training/architect/courses/rh442.html

67.1. Course Description

RH442 is an advanced four-day hands-on lab course covering system architecture, performance characteristics, monitoring, benchmarking, and network performance tuning.

67.1.1. Prerequisites

RHCT at a minimum, RHCE certification recommended, or comparable skills and knowledge. All prospective course participants without RHCE certification are encouraged to verify skills with Red Hat's free online pre—assessment tests.

67.1.2. Goal

RH442 is designed to teach the methodology of performance tuning and capacity planning for Red Hat Enterprise Linux. This class will cover:

1. A discussion of system architecture with an emphasis on understanding the implications of system architecture on system performance
2. Methods for testing the effects of performance adjustments (benchmarking)
3. Open source benchmarking utilities
4. Methods for analyzing system performance and networking performance
5. Tuning configurations for specific application loads

67.1.3. Audience

RH442 is aimed at senior Red Hat Enterprise Linux system administrators and other IT professionals working in enterprise environments and mission-critical systems.

67.1.4. Course Objectives

1. Overview of system components and architecture as they relate to system performance
2. Translating manufacturers' hardware specifications into useful information
3. Using standard monitoring tools effectively to gather and analyze trend information
4. Gathering performance-related data with SNMP
5. Using open source benchmarking utilities
6. Network performance tuning
7. Application performance tuning considerations

8. Tuning for specific configurations

67.1.5. Follow-on Courses

RHS333 Enterprise Security: Securing Network Services

RH401 Red Hat Enterprise Deployment and Systems Management

RH423 Red Hat Enterprise Directory Services and Authentication

RH436 Red Hat Enterprise Storage Mgmt.

Chapter 68. Red Hat Enterprise Linux Developer Courses

68.1. RHD143: Red Hat Linux Programming Essentials

http://www.redhat.com/training/developer/courses/rhd143.html

An intensive hands-on course designed to rapidly train staff in key skills for developing applications and programs on Red Hat Enterprise Linux. This five-day course provides hands-on training, concepts, demonstrations, with emphasis on realistic labs and programming exercises. Upon completion of the course, students will have learned and practiced the essential skills required to develop programs for Linux systems.

68.2. RHD221 Red Hat Linux Device Drivers

http://www.redhat.com/training/developer/courses/rhd221.html

This course is designed to teach experienced programmers how to develop device drivers for Linux systems. Upon completion of the course, students will understand the Linux architecture, hardware and memory management, modularization, and the layout of the kernel source, and will have practiced key concepts and skills for development of character, block, and network drivers.

68.3. RHD236 Red Hat Linux Kernel Internals

http://www.redhat.com/training/developer/courses/rhd236.html

This course is designed to provide a detailed examination of the Linux kernel architecture, including process scheduling, memory management, file systems, and driving peripheral devices. This five-day course provides hands-on training, concepts, and demonstrations, with emphasis on realistic labs and programming exercises.

68.4. RHD256 Red Hat Linux Application Development and Porting

http://www.redhat.com/training/developer/courses/rhd256.html

A four-day developer course for experienced programmers who are already familiar with development on a UNIX-like system and want to develop new applications as well as port existing applications to Red Hat Enterprise Linux.

Chapter 69. JBoss Courses

69.1. RHD161 JBoss and EJB3 for Java

http://www.redhat.com/training/jboss/courses/rhd161.html

Developers JBoss and EJB3 for Java Developers is targeted toward proficient Java developers who wish to extend their knowledge to EJB3 and J2EE middleware programming using the JBoss Application Server. This class is an in-depth introduction to EJB3 and J2EE using the JBoss Application Server. It provides a hands-or approach to EJB3 and J2EE application development, deployment and the tools necessary to facilitate both processes.

69.1.1. Prerequisites

Basic Java programming skills and knowledge of OOAD concepts are required. The student must have practical knowledge of, and/or experience with, the following:

1. The object-oriented concepts of inheritance, polymorphism and encapsulation
2. Java syntax, specifically for data types, variables, operators, statements and control flow
3. Writing Java classes as well as using Java interfaces and abstract classes

69.2. RHD163 JBoss for Web Developers

http://www.redhat.com/training/jboss/courses/rhd163.html

JBoss for Web Developers focuses on web tier technologies in the JBoss Enterprise Middleware System (JEMS) product stack. We cover details on JBoss Portal, how to create and deploy portlets, integrating portlets with other web tier frameworks such as JavaServer Faces JSF) and configuring and tuning the Tomcat web container embedded in JBoss Application Server. Familiarity with JSP and Servlet development and related specification is heavily recommended. No previous experience with Portlets or JSF is required.

69.2.1. Prerequisites

The prerequisite skills for this class are basic J2EE Web Container (Servlet/JSP) programming skills and some experience with J2EE Web-based and multi-tier application deployments on the JBoss Application Server in conjunction with the Tomcat container (whether embedded with Apache or integrated with the JBoss Application server). The student should have development experience with the following technologies:

1. JNDI
2. The Servlet 2.3/2.4 API
3. The JSP 2.0 API
4. J2EE application development and deployment on the JBoss Application Server
5. Deployment of a Web Application on embedded (stand alone) Tomcat or on integrated Tomcat (JBossWeb)
6. A working knowledge of JDBC and EJB2.1 or EJB3.0

while not a prerequisite, is helpful.

69.3. RHD167: JBOSS - HIBERNATE ESSENTIALS

http://www.redhat.com/training/jboss/courses/rhd167.html

69.3.1. Prerequisites

1. An understanding of the relational persistence model
2. Competency with the Java language
3. Knowledge of OOAD concepts
4. Familiarity with the UML
5. Experience with a dialect of SQL
6. Using the JDK and creating the necessary environment for compilation and execution of a Java executable from the command line
7. An understanding of JDB

No prior knowledge of J2EE or Hibernate is required. This training is based on Hibernate 3.2 series.

69.3.2. Course Summary

Hibernate Essentials is targeted toward Java developers who must become competent with the Hibernate or the Java Persistence API object/relational persistence and query service implementation. The primary audience is intended to be Java developers who work with SQL-based database systems or database developers who are looking for an introduction to object-oriented software development. Database administrators who are interested in how ORM may affect performance and how to tune the performance of the SQL database management system and persistence layer will also find this course of value. This course covers the JBoss, Inc. implementation of the JSR-220 sub-specification for Java Persistence and it covers the foundational APIs of version 3.x of the JBoss, Inc. Hibernate product, or simply, Hibernate 3.

69.4. RHD267: JBOSS - ADVANCED HIBERNATE

http://www.redhat.com/training/jboss/courses/rhd267.html

JBoss Advanced Hibernate training is targeted toward Java developers who wish to extract the full power of the Hibernate O/R Mapping framework. The primary target audience consists of Java developers who work with SQL-based database systems, database developers who are looking for an introduction to object-oriented software development and database administrators interested in how ORM affects performance and how to tune the performance of the SQL database management system and persistence layer. The training covers the new Hibernate 3 features.

69.4.1. Prerequisites

The prerequisite skills for this class are the following:

1. Basic Hibernate knowledge.
2. Competency with the Java language
3. Knowledge of OOAD concepts

4. Familiarity with the UML
5. Experience with a dialect of SQL
6. Using the JDK and creating the necessary environment for compilation and execution of a Java executable from the command line.
7. Experience with, or comprehensive knowledge of JNDI and JDBC.
8. Entity EJB2.1 or EJB3.0 knowledge, while not a prerequisite, is helpful.
9. Prior reading of the book Hibernate in Action, by Christian Bauer and Gavin King (published by Manning) is recommended.

"The best part of the Advanced Hibernate course was networking with fellow engineers that had problems similar to my own, and working with a knowledgeable instructor to solve them."--Mike Pasternak, Consulting Engineer, United Switch & Signal

69.5. RHD261:JBOSS for advanced J2EE developers

http://www.redhat.com/training/jboss/courses/rhd261.html

JBoss for Advanced J2EE Developers is targeted toward J2EE professionals who wish to take advantage of the JBoss Application Server internal architecture to enhance the functionality and performance of J2EE applications on the JBoss Application Server. This course covers topics such as JMX and those beyond the J2EE specification such as Microkernel architecture, Security, Clustering, and Fine Tuning.

69.5.1. Prerequisites

It is highly recommended that students either complete the JBoss for Java Developers course OR take the Middleware Placement Exam prior to registering for the JBoss for Advanced J2EE Developers course. The developer should have practical experience with each of the following topics:

1. JNDI
2. JDBC
3. Servlets and JSPs
4. Enterprise Java Beans
5. JMS
6. The J2EE Security Model
7. J2EE application development and deployment on the JBoss Application
8. Experience with ANT and XDoclet or similar technologies.

While prior knowledge of JMX is helpful, it is not required. This training is based on the 4.x series of the JBoss Application Server.

"I thought the training materials were well-organized, including both the handbook and the labs. The instructor frequently asked for feedback on material and pace. It was apparent that he cared about our understanding of the material."--Jeremy Prellwitz, SiRAS.com, USA

69.6. RH336: JBOSS for Administrators

http://www.redhat.com/training/jboss/courses/rh336.html

69.6.1. Prerequisites

Basic working knowledge of the Windows or Linux (Unix-based) operating system. The student must have experience with the following:

1. Creating directories, files and modifying access rights to the file store
2. Installing a JDK
3. Configuring environment variables, such as JAVA_HOME, for an Operating system
4. Launching Java applications and executing an OS-dependent script that launches a Java application.
5. Creating and expanding a Java archive file (the jar utility)

No prior knowledge of J2EE or the JBoss Application Server is required. Some familiarity with supporting Java applications with XML configurations, however, is strongly recommended.

69.6.2. Course Summary

JBoss for Administrators is targeted toward application support individuals, such as system administrators, configuration management and quality assurance personnel who wish to become proficient in configuring and administrating the JBoss application server (3.2.x and 4.x series) and the applications deployed on the application server.

"The JBoss for Administrators course was a great balance of both lecture and labs. It is always nice to have hands on knowledge of the topics to make them seem more real and applicable."——Thomas Skowronek, Palm Harbor Homes, USA

69.7. RHD439: JBoss Clustering

http://www.redhat.com/training/jboss/courses/rhd439.html

Clustering is a 4-day training focusing on high availability services of JBoss Enterprise Middleware System (JEMS). You will learn how JBoss Application Server leverages JGroups and JBoss Cache for replication and fail-over, how to configure, tune and implement JGroups protocol stacks, how to leverage JBoss Cache in your own middleware application implementation and how to use and configure mod_jk for HTTP load balancing. We will also cover in some detail JBoss Application Server high availability services such as HA-JNDI, HA-JMS and HA-singleton.

69.7.1. Prerequisites

Completion of the JBoss for Advanced J2EE Developers course is strongly recommended before taking this course. It is also strongly recommended that the student has at minimum 18 month practical development experience using J2EE and other Java middleware technologies, and it is suggested that the student have some practical experience with JBoss Application Server. Solid Java programming experience (minimum 3 years) is required and understanding of basic TCP/IP topics is necessary.

The student must have the following skills:

1. JTA, Transactions, Java concurrency
2. EJB 2.1, JMS, reliable messaging technologies
3. Previous experience with Apache httpd and some exposure to mod_jk and/or mod_proxy

4. Familiar with JBoss AS microkernel and JMX
5. Familiarity with TCP/IP, UDP, Multicasting

"The JBoss for Administrators course was very informative. Our instructor did a great job at answering our questions (some very specific to the student) while maintaining the course direction. I am very excited about applying what I have learned in the course."——Andy Beier, Arizona Statue University, USA

69.8. RHD449: JBoss jBPM

http://www.redhat.com/training/jboss/courses/rhd449.html

69.8.1. Description

JBoss jBPM training is targeted for system architects and developers who work closely with business analysts and are responsible for bringing business processes into J2EE environment using jBPM as a BPM engine. In addition, The JBoss jBPM training will provide students with a thorough understanding of the BPM landscape, types of engines and positioning of the buzzwords.

Students will acquire practical hands on expertise and will be ready to start developing business processes with JBoss jBPM after the course. Another goal of the training is to provide a thorough preparation for comparing workflow engines.

69.8.2. Prerequisites

1. The student must have previous experience developing an Hibernate application. The student must know how to configure a simple Session Factory for Hibernate, utilize a Hibernate Session and transactional demarcation and how to perform basic queries on Hibernate objects.
2. Competency with Java application development.
3. Previous exposure to the concepts of workflow and business process modeling (BPM) is not required
4. Experience with JBoss Eclipse or the Eclipse IDE with the JBoss plugin is recommended but not required
5. Basic notions of JUnit test framework is recommended.

69.9. RHD451 JBoss Rules

http://www.redhat.com/training/jboss/courses/rhd451.html

The course covers the core engine for Drools 3 (JBoss Rules 3.0), as well as the various techniques and languages that can be used to manage business rules, and how the rule engine may be embedded in J2SE and J2EE applications. This course will be a complimentary course to any future courses on rule management using future releases of Jboss Rules.

69.9.1. Prerequisites

1. Basic Java competency
2. Some understanding of what constitutes an inferencing rule engine versus a scripting engine
3. Viewing of the Jboss Rules webinars and demos is recommended but not required

4. Java EE specific experience is not required for the course, but students who need to know how to integrate with Java EE will need the appropriate experience

Appendix A. Revision History

Revision 11-1 **Tue 30 Jun 2015** **Barbora Ančincová**

Updated the book with information about the POODLE vulnerability (CVE-2014-3566).

Revision 11-0 **Fri 12 Sep 2014** **Barbora Ančincová**

Resolve BZ#1121893: RHEL 5 Deployment Guide - Bonding Options - regarding max_bonds and debug options.
Resolve BZ#1104152: Insecure way of generating swapfile is proposed in our documentation.

Revision 10-0 **Tue 01 Oct 2013** **Jaromír Hradílek**

Resolve BZ#853938: RFE: The proc File System: Document restricting access to /proc/<PID>.
Resolve BZ#826891: RFE: Yum: Provide a procedure to upgrade systems with "yum update" using RHEL Installation ISO image.
Resolve BZ#961815: Document migration mysql5.1->mysql5.5.

Revision 9-6 **Tue Jan 08 2013** **Jaromír Hradílek**

Resolve BZ#810514: RFE: Document the Ext4 File System.
Resolve BZ#216687: RFE: Postfix - a standard, FHS-compliant place for virtual user mailboxes.
Resolve BZ#816177: RFE: Postfix MySQL map support.
Resolve BZ#810512: MinorMod: Automated Tasks: default log rotation cronjob causes problems in a RHEV shared storage environment.
Resolve BZ#840000: Kdump hangs on CCISS module loading on RHEL 5.8 - HP Proliant DL380 G6 (Smartarray 410i).
Resolve BZ#852604: Kdump failed with intel_iommu=on.
Resolve BZ#847292: MajorMod: Network Interfaces: Static routes and default gateway are interface-specific?
Resolve BZ#821302: Documentation of NFS restrictions for secure nfs mounts work over TCP/UDP.
Resolve BZ#852372: Deployment Guide References $ISA in PAM sections.
Resolve BZ#713417: /root is labelled system_u:object_r:default_t:s0 after switching to MLS.
Resolve BZ#821225: No information about, amount of RAM to reserve for kdump kernel in the documentation.

Revision 8-0 **Tue Feb 21 2012** **Jaromír Hradílek**

Resolve BZ#749948: [Release Notes and Deployment Guide] Migration tooling from RHN Classic to Cert-based RHN for RHEL 5.
Resolve BZ#718608: MinorMod: FTP: Missing text fragment in vsftpd configuration documentation.
Resolve BZ#720387: MinorMod: The proc File System: Illogical parameter description.
Resolve BZ#720860: Update Deployment (Guide) in RHEL5 Build Tree.
Resolve BZ#760925: MinorMod: Network File System: Severely suboptimal timeo option in NFS mount examples (for TCP).
Resolve BZ#784754: MinorMod: Network Interfaces: typo - wrong tense in 15.3. Interface Control Scripts.
Resolve BZ#740916: MinorMod: The kdump Crash Recovery Service: Incorrect description of the crashkernel parameter.
Resolve BZ#767105: incorrect default action in kdump part.
Resolve BZ#714080: debug option for bonding can not be used in BONDING_OPTS in /etc/sysconfig/network-scripts/ifcfg-bondX.
Resolve BZ#769776: Documentation needs to be updated for "default shell" option in /etc/kdump.conf.
Resolve BZ#781441: /etc/securetty documentation is incorrect [rhel-5.7].

Revision 7-0 **Thu Jul 21 2011** **Jaromír Hradílek**

Resolve BZ#720382: MinorMod: Network Interfaces: LINKDELAY parameter needs to be added to "Interfac
Configuration Files".
Resolve BZ#632028: MajorMod: Redundant Array of Independent Disks (RAID): Document mdadm Usage.
Resolve BZ#720009: MinorMod: LVM: Update screenshots in the "Manual LVM Partitioning" section.
Resolve BZ#711162: MinorMod: Network Interfaces: Incorrect static routes configuration.
Resolve BZ#707238: broadcast is calculated with ipcalc, not ifcalc.
Resolve BZ#678316: HOTPLUG network config file option is not documented.
Resolve BZ#562018: Ch.4 Redundant Array of Independent Disks (RAID) - screenshots need updating.
Resolve BZ#485033: iptables -p ALL --dport not allowed according to man 8 iptables.

Revision 6-0	**Thu Jan 13 2011**	**Jaromír Hradílek**

Resolve BZ#249485: 'fsid=num' is listed under NFS client options, but it is a server-only option.
Resolve BZ#253659: additional commands required when adding machines to domain.
Resolve BZ#453242: guide does not tell you which packages you need to run an NFS server.
Resolve BZ#504250: cell should have newline characters, it shouldn't be all on one line.
Resolve BZ#520650: /proc/loadavg documentation error.
Resolve BZ#584075: vsftp typo for text_userdb_names.
Resolve BZ#625384: bonding configuration SLAVE=bond0 is invalid.
Resolve BZ#644617: misspelled word.
Resolve BZ#645123: spelling Errors in Deployment Guide II.
Resolve BZ#595366: RFE: document Shared Subtrees.

Revision 5-0	**Thu July 30 2010**	**Douglas Silas**

Resolve BZ#239313: document oom_adj and oom_score.
Resolve BZ#526502: correct quotaon instructions with proper, safe operating procedures.
Resolve BZ#551367: correct SELinux dhcpd_disable_trans description.
Resolve BZ#521215: clarify NFS interaction with portmapper, rpc.mountd, rpc.lockd and rpc.statd.
Resolve BZ#453875: various OpenSSH chapter corrections.
Resolve BZ#455162: correct zone example configuration file, description.
Resolve BZ#460767: make it a proper daemon.
Resolve BZ#600702: correct directories used for SSL key generation.

Revision 4-2	**Wed Sep 30 2009**	**Douglas Silas, Jaromír Hradílek, Martin Prpic**

Change heading titles to correspond with actual headings used in 'man rpm'.
Resolve BZ#499053: /usr/sbin/racoon is correct install path.
Remove any mention of 'pkgpolicy' in /etc/yum.conf as per BZ#237773.
Resolve BZ#455162: correct example zone file with regard to records, description.
Resolve BZ#510851: /proc/cmdline has confusing descriptions of sample output.
Resolve BZ#510847: page with multiple footnotes formatted incorrectly in online PDF.
Resolve BZ#214326: more detailed usage info concerning vsftpd banners and secueerity.
Resolve BZ#241314: formatting problems in screen elements.
Resolve BZ#466239: postfix connect-from-remote-host configuration fix.

Revision 4-1	**Mon Sep 14 2009**	**Douglas Silas**

Resolve BZ#214326: Server Security FTP Banner instructions: questions re: vsftpd.conf.
Resolve BZ#466239: insert line into Postfix config file to allow connecting remotely.
Resolve BZ#499053: path for racoon daemon is /usr/sbin/racoon, not /sbin/racoon.
Resolve BZ#510847: missing footnotes in PDF output.
Resolve BZ#510851: rewrite /proc/cmdline minor section to make more sense.
Resolve BZ#515613: correct location of RHEL5 GPG keys and key details.
Resolve BZ#523070: various minor fixes; --redhatprovides to rpm -q --whatprovides.

Revision 4-0	**Wed Sep 02 2009**	**Douglas Silas**

Resolve BZ#492539: "This directive is useful..." to "This directive must be used in machines containing mor
than one NIC to ensure...".
Resolve BZ#241314: re: kernel-pae and hugemem support on RHEL 4 and 5.
Resolve BZ#453071: incorrect tag use led to config files and other screen elements being displayed on sing
lines.
Resolve BZ#507987: clarify and correct statements about partitions being in use while resizing or removing
Resolve BZ#462550: recommended amount of swap space, according to
http://kbase.redhat.com/faq/docs/DOC-15252.
Resolve BZ#466239: line omitted from Postfix configuration meant connecting remotely failed
Resolving other MODIFIED BZs (fixed previously): 468483, 480324, 481246, 481247, 438823, 454841,
485187, 429989, 452065, 453466.

Revision 3-0	Wed Jan 28 2009	Michael Hideo Smith

Resolves: #460981
Changing 64GB *tested* support to support for 16GB.

Appendix B. Colophon

The manuals are written in DocBook XML v4.3 format.

Garrett LeSage created the admonition graphics (note, tip, important, caution, and warning). They may be freely redistributed with the Red Hat documentation.

Contributing Writers: John Ha (System Administration, Filesystems, Kernel), Joshua Wulf (Installation and Booting), Brian Cleary (Virtualization), David O'Brien (Security and SELinux), Michael Hideo (System Administration), Don Domingo (System Administration), Michael Behm (System Administration), Paul Kennedy (Storage), Melissa Goldin (Red Hat Network)

Honoring those who have gone before: Sandra Moore, Edward C. Bailey, Karsten Wade, Mark Johnson, Andrius Benokraitis, Lucy Ringland

Honoring engineering efforts: Jeffrey Fearn

Technical Editing: Michael Behm

Graphic Artist: Andrew Fitzsimon

The Red Hat Localization Team consists of the following people:

- East Asian Languages
  - Simplified Chinese
    - Tony Tongjie Fu
    - Simon Xi Huang
    - Leah Wei Liu
    - Sarah Saiying Wang
  - Traditional Chinese
    - Chester Cheng
    - Terry Chuang
    - Ben Hung-Pin Wu
  - Japanese
    - Kiyoto Hashida
    - Junko Ito
    - Noriko Mizumoto
    - Takuro Nagamoto
  - Korean
    - Eun-ju Kim
    - Michelle Kim
- Latin Languages

- French
  - Jean-Paul Aubry
  - Fabien Decroux
  - Myriam Malga
  - Audrey Simons
  - Corina Roe
- German
  - Jasna Dimanoski
  - Verena Furhuer
  - Bernd Groh
  - Daniela Kugelmann
  - Timo Trinks
- Italian
  - Francesco Valente
- Brazilian Portuguese
  - Glaucia de Freitas
  - Leticia de Lima
  - David Barzilay
- Spanish
  - Angela Garcia
  - Gladys Guerrero
  - Yelitza Louze
  - Manuel Ospina
- Russian
  - Yuliya Poyarkova

- Indic Languages
  - Bengali
    - Runa Bhattacharjee
  - Gujarati
    - Ankitkumar Rameshchandra Patel
    - Sweta Kothari
  - Hindi

  - Rajesh Ranjan
- Malayalam
  - Ani Peter
- Marathi
  - Sandeep Shedmake
- Punjabi
  - Amanpreet Singh Alam
  - Jaswinder Singh
- Tamil
  - I Felix
  - N Jayaradha