1. Introduction

Linux: history and introduction

Linux is a version of UNIX operating system, the original ancestor of Linux. UNIX is a command-driven operating system in which the user has to type in commands at the computer console in order to operate the computer ("Introduction to Linux, 2001). UNIX is one of the most popular operating systems worldwide because of its large support base and distribution. It was originally developed in the 1970's at AT&T as a multitasking system for minicomputers and mainframes. Since then, it has grown to become one of the most widely-used operating systems.

In 1991, Linux Torvalds, a student at the University of Helsinki, sought to create a new version of UNIX; therefore, he joined forces with a group of programmers to create a new operating system—Linux. Linux is a free version of UNIX that continues to be developed by the cooperative efforts of volunteer groups of programmers, primarily on the Internet, who exchange code, report bug, and fix problems in an open-ended environment. As a result, the world now has a powerful, robust, and full-featured operating system that continues to change and grow.

Linux is known as an ***open source*** operating system and also called ***free software*** because everything about Linux is accessible to the public and is freely available to anyone. Since the Linux source code is available, anyone can copy, modify, and distribute this software. This allows for various companies such as SuSE, Red Hat, Caldera and others to sell and distribute Linux; however, at the same time, these companies must keep their Linux distribution code open for public inspection, comment, and changes. Despite of the command-line origins of Linux, these distributing companies are working to make the **Graphical User Interface** (GUI), the primary means of user interface; thus more user friendly.

Linux has mostly been used in small to medium-sized companies and is now on its way to being used in larger-sized companies. "In a survey run by Networking Computer Magazine, 75% of respondents were already using Linux and another 14% were evaluating it"

In addition, Linux servers account for a majority of the information available on the Internet. "A survey run by Net craft reports that 43% of all web sites use Linux servers running the Apache Web server"

Linux Distributions

A Linux **distribution** is an assemblage of software with its own packaging schemes, defaults and configuration methods. The following is a list of some of the major Linux distributions (LeBlanc, 2001, p. 17)

- **Corel Linux**: This is a new Linux distribution that has made an impressive entrance. The installation process is simple and does a great job of detecting and automatically configuring many sound and video card adapters. It also comes together with Corel's WordPerfect word processing package which has been ported to run on Linux.

- **Debian GNU/Linux**: This distribution is one of the oldest and recognized favorites among advanced technical groups. It is relatively difficult to install due to the very high number of installation options.

- **OpenLinux** (Caldera): The OpenLinux distribution has shrink-wrapped software packages that include the first graphical Linux installation. This distribution allows the user to play a game in the foreground while the computer loads software in the background during installation.

- **Red Hat**: Red Hat is the first company to mass market the Linux operating system. They have validated Linux by packing the GNU/Linux tools in shrink wrapped packages and have included valued-added features to their product such as: telephone support, training, and consulting services.

- **Slackware**: Of all of the surviving Linux distributions, Slackware has been around the longest. The installation interface had remained the same since its beginning, until a couple of years ago.

- **SuSE***:* This distribution derives from Germany. SuSE works closely with the XFree86 project (the free X graphical server component of all Linux distributions). As a result, they have a terrific graphical configuration tool called SaX.

- **TurboLinux**: This distribution provides a great graphical desktop environment along with a few tools for configuring the system. TurboLinux has lead the way in the turnkey installations by providing CD installations exclusive to Server, Workstation, and Clusters.

Features of Linux.

Linux system has many features. The main basic features are as;

1. **The operating system loads quickly and shuts down quickly**
2. **The operating system is very low cost**
3. **Upgrades are FREE!**
4. **Browsers like Firefox, Opera, and Chrome work well in Linux**
5. **Linux offers an amazing number of free programs**
6. **Linux can run Window Office**
7. **Linux runs Open Office**
8. **Linux looks enough like Windows you won't feel like you're in a strange place.**
9. **New open source software is available.**
10. **Older open source software is upgraded FREE!**
11. **You are not pestered about "Genuine Linux"**
12. **When you want to add software it is easy to find what you need**
13. **You don't have to clean the registry often – I've never felt the need in Linux**
14. **Linux can run Internet Explorer**
15. **You have access to support without paying for it**
16. **There are several places to find support**
17. **You may have access to a support group of living people in your community**
18. **You can learn Linux at your own pace**

Advantages of Linux over Other operating system.

a. Full access vs. no access

Having access to the source code is probably the single most significant difference between Linux and Windows. The fact that Linux belongs to the GNU Public License ensures that users (of all sorts) can access (and alter) the code to the very kernel that serves as the foundation of the Linux operating system. You want to peer at the Windows code? Good luck. Unless you are a member of a very select (and elite, to many) group, you will never lay eyes on code making up the Windows operating system.

b. Licensing freedom vs. licensing restrictions

Along with access comes the difference between the licenses. I'm sure that every IT professional could go on and on about licensing of PC software. But let's just look at the key aspect of the licenses (without getting into legalese). With a Linux GPL-licensed operating system, you are free to modify that software and use and even republish or sell it (so long as you make the code available). Also, with the GPL, you can download a single copy of a Linux distribution (or application) and install it on as many machines as you like. With the Microsoft license, you can do none of the above. You are bound to the number of licenses you purchase, so if you purchase 10 licenses, you can legally install that operating system (or application) on only 10 machines.

c. Online peer support vs. paid help-desk support

This is one issue where most companies turn their backs on Linux. But it's really not necessary. With Linux, you have the support of a huge community via forums, online search, and plenty of dedicated Web sites. And of course, if you feel the need, you can purchase support contracts from some of the bigger Linux companies (Red Hat and Novell for instance).

On the other side of the coin is support for Windows. Yes, you can go the same route with Microsoft and depend upon your peers for solutions. There are just as many help sites/lists/forums for Windows as there are for Linux. And you can purchase support from Microsoft itself. Most corporate higher-ups easily fall victim to the safety net that having a support contract brings. But most higher-ups haven't had to depend up on said support contract. Of the various people I know who have used either a Linux paid support contract or a Microsoft paid support contract, I can't say one was more pleased than the other. This of course begs the question "Why do so many say that Microsoft support is superior to Linux paid support?"

d. Full vs. partial hardware support

One issue that is slowly becoming nonexistent is hardware support. Years ago, if you wanted to install Linux on a machine you had to make sure you hand-picked each piece of hardware or your installation would not work 100 percent. I can remember, back in 1997-ish, trying to figure out why I couldn't get Caldera Linux or Red Hat Linux to see my modem. After much looking around, I found I was the proud owner of a Winmodem. So I had to go out and purchase a US Robotics external modem because that was the one modem I *knew* would work. This is not so much the case now. You can grab a PC (or laptop) and most likely get one or more Linux distributions to install and work nearly 100 percent. But there are still some exceptions. For instance, hibernate/suspend remains a problem with many laptops, although it has come a long way.

With Windows, you know that most every piece of hardware will work with the operating system. Of course, there are times (and I have experienced this over and over) when you will wind up spending much of the day searching for the correct drivers for that piece of hardware you no longer have the install disk for. But you can go out and buy that 10-cent Ethernet card and know it'll work on your machine (so long as you have, or can find, the drivers). You also can rest assured that when you purchase that insanely powerful graphics card, you will probably be able to take full advantage of its power.

e. Command line vs. no command line

No matter how far the Linux operating system has come and how amazing the desktop environment becomes, the command line will always be an invaluable tool for administration purposes. Nothing will ever replace my

favorite text-based editor, ssh, and any given command-line tool. I can't imagine administering a Linux machine without the command line. But for the end user -- not so much. You could use a Linux machine for years and never touch the command line, Same with Windows. You can still use the command line with Windows, but not nearly to the extent as with Linux. And Microsoft tends to obfuscate the command prompt from users. Without going to Run and entering cmd (or command, or whichever it is these days), the user won't even know the command-line tool exists. And if a user does get the Windows command line up and running, how useful is it really?

f. Centralized vs. non centralized application installation

The heading for this point might have thrown you for a loop. But let's think about this for a second. With Linux you have (with nearly every distribution) a centralized location where you can search for, add, or remove software. I'm talking about package management systems, such as Synaptic. With Synaptic, you can open up one tool, search for an application (or group of applications), and install that application without having to do any Web searching (or purchasing).

Windows has nothing like this. With Windows, you must know where to find the software you want to install, download the software (or put the CD into your machine), and run setup.exe or install.exe with a simple double-click. For many years, it was thought that installing applications on Windows was far easier than on Linux. And for many years, that thought was right on target. Not so much now. Installation under Linux is simple, painless, and centralized.

g. Flexibility vs. rigidity

I always compare Linux (especially the desktop) and Windows to a room where the floor and ceiling are either movable or not. With Linux, you have a room where the floor and ceiling can be raised or lowered, at will, as high or low as you want to make them. With Windows, that floor and ceiling are immovable. You can't go further than Microsoft has deemed it necessary to go.

Take, for instance, the desktop. Unless you are willing to pay for and install a third-party application that can alter the desktop appearance, with Windows you are stuck with what Microsoft has declared is the ideal desktop for you. With Linux, you can pretty much make your desktop look and feel exactly how you want/need. You can have as much or as little on your desktop as you want. From simple flat Fluxbox to a full-blown 3D Compiz experience, the Linux desktop is as flexible an environment as there is on a computer.

h. Automated vs. nonautomated removable media

I remember the days of old when you had to mount your floppy to use it and unmount it to remove it. Well, those times are drawing to a close -- but not completely. One issue that plagues new Linux users is how removable media is used. The idea of having to manually "mount" a CD drive to access the contents of a CD is completely foreign to new users. There is a reason this is the way it is. Because Linux has always been a multiuser platform, it was thought that forcing a user to mount a media to use it would keep the user's files from being overwritten by another user. Think about it: On a multiuser system, if everyone had instant access to a disk that had been inserted, what would stop them from deleting or overwriting a file you had just added to the media? Things have now evolved to the point where Linux subsystems are set up so that you can use a

removable device in the same way you use them in Windows. But it's not the norm. And besides, who doesn't want to manually edit the */etc/fstab*fle?

i. Multilayered run levels vs. a single-layered run level

I couldn't figure out how best to title this point, so I went with a description. What I'm talking about is Linux' inherent ability to stop at different run levels. With this, you can work from either the command line (run level 3) or the GUI (run level 5). This can really save your socks when X Windows is fubared and you need to figure out the problem. You can do this by booting into run level 3, logging in as root, and finding/fixing the problem.

With Windows, you're lucky to get to a command line via safe mode -- and then you may or may not have the tools you need to fix the problem. In Linux, even in run level 3, you can still get and install a tool to help you out (hello apt-get install APPLICATION via the command line). Having different run levels is helpful in another way. Say the machine in question is a Web or mail server. You want to give it all the memory you have, so you don't want the machine to boot into run level 5. However, there are times when you do want the GUI for administrative purposes (even though you can fully administer a Linux server from the command line). Because you can run the *startx* command from the command line at run level 3, you can still start up X Windows and have your GUI as well. With Windows, you are stuck at the Graphical run level unless you hit a serious problem.

FATS, NTFS, EXT files system

FAT: fat is one of a few different file system used with windows over the years. Almost every computer user has used FAT at one time or another, since it was the sparse base operating system at the heart of all window operating system.

NTFS: NTFS is the next generation of HPES. It comes with all version of Microsoft operating system beginning with window NT. Unlike FAT, it is a b-tree file system, meaning it has a performance and reliability advantage, including journaling, and support for encryption and compression, over FAT.

EXT: the ext file system is the extendable file system. Linux support the various version of ext as ie Ext1, ext2, ext3, ext4.ext4 is the new version of fedora Linux. Ext4 provides all the features of ext2 and ext3, and also features journaling and backward compatibility with ext3 and ext2.

Culture of Free Software

Software available FREE with source code under GPL license to use, change, improve and redistribute it modified or unmodified. Open source software can be used freely without having to pay license fees to its developers. Contrasts with the commercial model of software distribution in which the source code remains the developer's closely guarded private property.

The open source movement is a worldwide effort to promote an open style of software development more aligned with the accepted intellectual style of science than the proprietary modes of invention that have been characteristic of modern business. Both open source software & open access to research represent innovative

responses to the particular restrictions placed on the sharing and exchange of software code & research publications, respectively, imposed by current intellectual property economics.

Idea is to keep the SW advances openly available for everyone to use, understand, and improve, as in 1960-70 engineers did. Based on innovating, sharing, collaborating, community involvement, freedom, and winning together.

OSS Strengths

1. **Ability to fit local needs:** Availability of the source code means that you can modify and enhance the software to more closely fit your own needs.
2. **No restrictions on use:** no restrictions on how the software is used and no invoices for each user license.
3. **Low cost:** no charge for the software itself. If other libraries share their efforts, each user's cost is reduced. Pay only for needed support or any additional products & services if required. Even then huge savings than commercial SW.
4. **Innovation:** with open source code , users keep-up innovating, improving which means often much faster development cycle when compared to proprietary software
5. **User-driven:** Traditional vendors focus on providing functionality meeting needs of the majority of their customers. In contrast, OSS features emerge from the community of users.This makes OSS development user-driven: you decide what features are important and deserve attention rather than a vendor.
6. **Collaboration:** vibrant local, national and global user groups collaborate in creativity, development and trouble shooting.
7. **Transfer of Technical Know-How:** Being active member and part of OSS community, your team members will learn the minimum required know-how of SW & technologies in use.
8. **Reliability:** OSS is peer-reviewed software, exposed to extreme scrutiny, with problems being found and fixed instead of being kept secret until the wrong person discovers. So the code base is more reliable than closed, proprietary software. Mature open-source code is as bulletproof as software ever gets. OSS evolves at astonishing speed. People improve it, people adapt it, people fix bugs.
9. **Security:** Proprietary software, with 'closed' source code, support and future development rely solely on the resources of a single vendor. If the vendor goes down, so does your product support?
10. In contrast, OSS relies on stable code bases developed and supported by many providers worldwide. As a result, libraries using OSS have more support options than those using proprietary software.

2. Basic of Linux

2.1. Command

- alias Create an alias •
- apropos Search Help manual pages (man -k)
- apt-get Search for and install software packages (Debian/Ubuntu)
- aptitude Search for and install software packages (Debian/Ubuntu)
- aspell Spell Checker
- awk Find and Replace text, database sort/validate/index

b

- basename Strip directory and suffix from filenames
- bash GNU Bourne-Again SHell
- bc Arbitrary precision calculator language
- bg Send to background
- bind Set or display readline key and function bindings •
- break Exit from a loop •
- builtin Run a shell builtin
- bzip2 Compress or decompress named file(s)

c

- cal Display a calendar
- case Conditionally perform a command
- cat Concatenate and print (display) the content of files
- cd Change Directory
- cfdisk Partition table manipulator for Linux
- chattr Change file attributes on a Linux file system
- chgrp Change group ownership
- chmod Change access permissions
- chown Change file owner and group
- chroot Run a command with a different root directory
- chkconfig System services (runlevel)
- cksum Print CRC checksum and byte counts
- clear Clear terminal screen
- cmp Compare two files
- comm Compare two sorted files line by line
- command Run a command - ignoring shell functions •
- continue Resume the next iteration of a loop •
- cp Copy one or more files to another location
- cron Daemon to execute scheduled commands
- crontab Schedule a command to run at a later time
- csplit Split a file into context-determined pieces
- curl Transfer data from or to a server
- cut Divide a file into several parts

d

- date Display or change the date & time
- dc Desk Calculator

e

- echo Display message on screen •
- egrep Search file(s) for lines that match an extended expression
- eject Eject removable media
- enable Enable and disable builtin shell commands •
- env Environment variables
- ethtool Ethernet card settings
- eval Evaluate several commands/arguments
- exec Execute a command
- exit Exit the shell
- expect Automate arbitrary applications accessed over a terminal
- expand Convert tabs to spaces
- export Set an environment variable
- expr Evaluate expressions

f

- false Do nothing, unsuccessfully
- fdformat Low-level format a floppy disk
- fdisk Partition table manipulator for Linux
- fg Send job to foreground
- fgrep Search file(s) for lines that match a fixed string
- file Determine file type
- find Search for files that meet a desired criteria
- fmt Reformat paragraph text
- fold Wrap text to fit a specified width.
- for Expand *words*, and execute *commands*
- format Format disks or tapes
- free Display memory usage
- fsck File system consistency check and repair
- ftp File Transfer Protocol
- function Define Function Macros
- fuser Identify/kill the process that is accessing a file

g

- gawk Find and Replace text within file(s)
- getopts Parse positional parameters
- grep Search file(s) for lines that match a given pattern
- groupadd Add a user security group
- groupdel Delete a group
- groupmod Modify a group
- groups Print group names a user is in
- gzip Compress or decompress named file(s)

h

- hash Remember the full pathname of a name argument
- head Output the first part of file(s)
- help Display help for a built-in command •

- dd Convert and copy a file, write disk headers, boot records
- ddrescue Data recovery tool
- declare Declare variables and give them attributes •
- df Display free disk space
- diff Display the differences between two files
- diff3 Show differences among three files
- dig DNS lookup
- dir Briefly list directory contents
- dircolors Colour setup for `ls'
- dirname Convert a full pathname to just a path
- dirs Display list of remembered directories
- dmesg Print kernel & driver messages
- du Estimate file space usage
- history Command History
- hostname Print or set system name
- htop Interactive process viewer

i

iconv Convert the character set of a file
id Print user and group id's
if Conditionally perform a command
ifconfig Configure a network interface
ifdown Stop a network interface
ifup Start a network interface up
import Capture an X server screen and save the image to file
install Copy files and set attributes
iostat Report CPU and i/o statistics
ip Routing, devices and tunnels

j

jobs List active jobs •
join Join lines on a common field

k

kill Kill a process by specifying its PID
killall Kill processes by name

l

less Display output one screen at a time
let Perform arithmetic on shell variables •
link Create a link to a file
ln Create a symbolic link to a file
local Create a function variable •
locate Find files
logname Print current login name
logout Exit a login shell •
look Display lines beginning with a given string
lpc Line printer control program
lpr Off line print
lprint Print a file
lprintd Abort a print job
lprintq List the print queue
lprm Remove jobs from the print queue
lsattr List file attributes on a Linux second extended file system
lsblk List block devices
ls List information about file(s)
lsof List open files

m

make Recompile a group of programs
man Help manual
mkdir Create new folder(s)
mkfifo Make FIFOs (named pipes)
mkfile Make a file
mkisofs Create an hybrid ISO9660/JOLIET/HFS filesystem
mknod Make block or character special files
mktemp Make a temporary file
more Display output one screen at a time
most Browse or page through a text file
mount Mount a file system
mtools Manipulate MS-DOS files
mtr Network diagnostics (traceroute/ping)
mv Move or rename files or directories
mmv Mass Move and rename (files)

n

nc Netcat, read and write data across networks
netstat Networking connections/stats
nice Set the priority of a command or job
nl Number lines and write files
nohup Run a command immune to hangups
notify-send Send desktop notifications
nslookup Query Internet name servers interactively

o

open Open a file in its default application
op Operator access

p

passwd Modify a user password
paste Merge lines of files
pathchk Check file name portability
ping Test a network connection
pgrep List processes by name
pkill Kill processes by name
popd Restore the previous value of the current directory
pr Prepare files for printing
rm Remove files
rmdir Remove folder(s)
rsync Remote file copy (Synchronize file trees)

s

screen Multiplex terminal, run remote shells via ssh
scp Secure copy (remote file copy)
sdiff Merge two files interactively
sed Stream Editor
select Accept keyboard input
seq Print numeric sequences
set Manipulate shell variables and functions
sftp Secure File Transfer Program
shift Shift positional parameters
shopt Shell Options
shutdown Shutdown or restart linux
sleep Delay for a specified time
slocate Find files
sort Sort text files
source Run commands from a file '.'
split Split a file into fixed-size pieces
ss Socket Statistics
ssh Secure Shell client (remote login program)
stat Display file or file system status
strace Trace system calls and signals
su Substitute user identity
sudo Execute a command as another user
sum Print a checksum for a file
suspend Suspend execution of this shell •
sync Synchronize data on disk with memory

t

tail Output the last part of file
tar Store, list or extract files in an archive
tee Redirect output to multiple files
test Evaluate a conditional expression
time Measure Program running time
timeout Run a command with a time limit
times User and system times
touch Change file timestamps
top List processes running on the system
tput Set terminal-dependent capabilities, color, position
traceroute Trace Route to Host
trap Execute a command when the shell receives a signal •
tr Translate, squeeze, and/or delete characters
true Do nothing, successfully
tsort Topological sort
tty Print filename of terminal on stdin
type Describe a command •

u

ulimit Limit user resources •
umask Users file creation mask
umount Unmount a device
unalias Remove an alias •
uname Print system information
unexpand Convert spaces to tabs
uniq Uniquify files
units Convert units from one scale to another
unrar Extract files from a rar archive
unset Remove variable or function names
unshar Unpack shell archive scripts

```
  printcap Printer capability database
  printenv Print environment variables
  printf   Format and print data •
  ps       Process status
  pushd    Save and then change the current directory
  pv       Monitor the progress of data through a pipe
  pwd      Print Working Directory
q
  quota    Display disk usage and limits
  quotacheck Scan a file system for disk usage
r
  ram      ram disk device
  rar      Archive files with compression
  rcp      Copy files between two machines
  read     Read a line from standard input •
  readarray Read from stdin into an array variable •
  readonly Mark variables/functions as readonly
  reboot   Reboot the system
  rename   Rename files
  renice   Alter priority of running processes
  remsync  Synchronize remote files via email
  return   Exit a shell function
  rev      Reverse lines of a file
who      Print all usernames currently logged in
  whoami   Print the current user id and name (`id -un')
  wget     Retrieve web pages or files via HTTP, HTTPS or
FTP
  write    Send a message to another user
x
  xargs    Execute utility, passing constructed argument
list(s)
```

```
  until    Execute commands (until error)
  uptime   Show uptime
  useradd  Create new user account
  userdel  Delete a user account
  usermod  Modify user account
  users    List users currently logged in
  uuencode Encode a binary file
  uudecode Decode a file created by uuencode
v
  v        Verbosely list directory contents (`ls -l -b')
  vdir     Verbosely list directory contents (`ls -l -b')
  vi       Text Editor
  vmstat   Report virtual memory statistics
w
  wait     Wait for a process to complete •
  watch    Execute/display a program periodically
  wc       Print byte, word, and line counts
  whereis  Search the user's $path, man pages and source
files for a program
  which    Search the user's $path for a program file
  while    Execute commands


xdg-open Open a file or URL in the user's preferred
application.
  xz       Compress or decompress .xz and .lzma files
  yes      Print a string until interrupted
  zip      Package and compress (archive) files.
  .        Run a command script in the current shell
  !!       Run the last command again
  ###      Comment / Remark
```

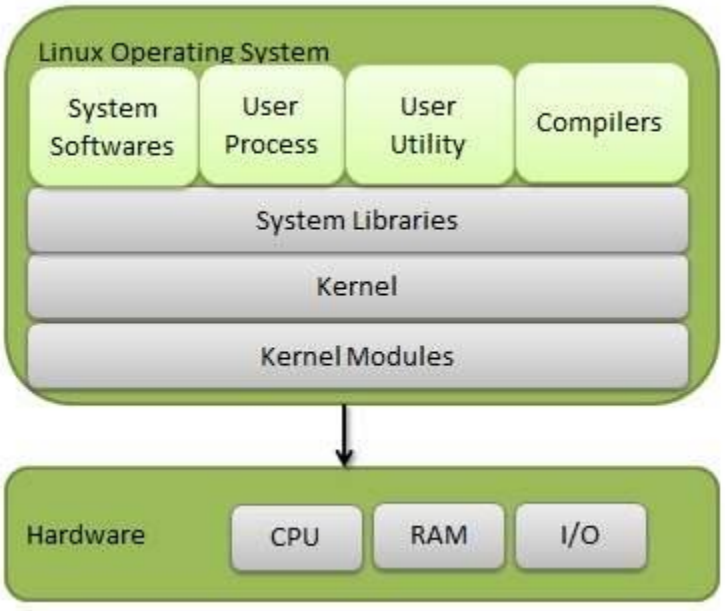
2.2. Shell Csh, Ksh,Bash

Shell is a UNIX term for the interactive user interface with an operating system. The shell is the layer of programming that understands and executes the commands a user enters. In some systems, the shell is called command syntax. The kernel is the essential center of a computer operating system, the core that provides basic services for all other parts of the operating system. A synonym is *nucleus*. A kernel can be contrasted with a shell, the outermost part of an operating system that interacts with user commands. *Kernel* and *shell* are terms used more frequently in Unix operating systems than in IBM mainframe or Microsoft Windows systems, Command interpreter. A shell usually implies an interface with command syntax.

Components of Linux system.

Linux Operating System has primarily three components

- **Kernel** − Kernel is the core part of Linux. It is responsible for all major activities of this operating system. It consists of various modules and it interacts directly with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.



- **System Library** − System libraries are special functions or programs using which application programs or system utilities accesses Kernel's features. These libraries implement most of the functionalities of the operating system and do not require kernel module's code access rights.

- **System Utility** − System Utility programs are responsible to do specialized, individual level tasks.

Linux operating system also offers several shells. These command-line interfaces provide powerful environments for software development and system maintenance. Though shells have many commands in common, each type has unique features. Over time, individual programmers come to prefer one type of shell over another; some develop new, enhanced shells based on previous ones. Unix also has an ecosystem of different shells; Linux carries this practice into the open-source software arena.

sh

The Bourne shell, called "sh," is one of the original shells, developed for Unix computers by Stephen Bourne at AT&T;'s Bell Labs in 1977. Its long history of use means many software developers are familiar with it. It offers features such as input and output redirection, shell scripting with string and integer variables, and condition testing and looping.

bash

The popularity of sh motivated programmers to develop a shell that was compatible with it, but with several enhancements. Linux systems still offer the sh shell, but "bash" -- the "Bourne-again Shell," based on sh -- has become the new default standard. One attractive feature of bash is its ability to run sh shell scripts unchanged. Shell scripts are complex sets of commands that automate programming and maintenance chores; being able to reuse these scripts saves programmers time. Conveniences not present with the original Bourne shell include command completion and a command history.

csh and tcsh

Developers have written large parts of the Linux operating system in the C and C++ languages. Using C syntax as a model, Bill Joy at Berkeley University developed the "C-shell," csh, in 1978. Ken Greer, working at Carnegie-Mellon University, took csh concepts a step forward with a new shell, tcsh, which Linux systems now offer. Tcsh fixed problems in csh and added command completion, in which the shell makes educated "guesses" as you type, based on your system's directory structure and files. Tcsh does not run bash scripts, as the two have substantial differences.

ksh

David Korn developed the Korn shell, or ksh, about the time tcsh was introduced. Ksh is compatible with sh and bash. Ksh improves on the Bourne shell by adding floating-point arithmetic, job control, command aliasing and command completion. AT&T; held proprietary rights to ksh until 2000, when it became open source.

2.3. Text Editor- vi,pico

It's almost impossible to use Red Hat Linux for any period of time and not need to use a text editor. If you are using a GUI, you can run gedit, which is fairly intuitive for editing text. Most Red Hat Linux shell users will use either the vi or emacs commands to edit text files. The advantage of vi or emacs over a graphical editor is that you can use it from any shell, a character terminal, or a character-based connection over a network (using telnet or ssh, for example). No GUI is required

Exploring Other Text Editors

There are dozens of text editors available to use with Linux. Here are a few contained in Red Hat Linux that you can try out if you find vi to be too taxing:

- **gedit** — The GNOME text editor that runs in the GUI.
- **jed** — This screen-oriented editor was made for programmers. Using colors, jed can highlight code you create so you can easily read the code and spot syntax errors. Use the Alt key to select menus to manipulate your text.
- **joe** — The joe editor is similar to many PC text editors. Use control and arrow keys to move around. Type Ctrl+C to exit with no save or Ctrl+X to save and exit.
- **kedit** — A GUI-based text editor that comes with the KDE desktop.
- **mcedit** — With mcedit, function keys help you get around, save, copy, move, and delete text. Like jed and joe, mcedit is screen-oriented

Usingvi text-Editor

Vi or vim is a text editor that is used to edit all configuration files and other general files. It is used to edit the program configuration files. While using vi editor.

Syntax.

Vi filename<enter>

Then press I to insert the text. After editing press Esc to stop the text writing, then we have following options.

;wq -> to save the current writing and exit.

;w-> save the current file but continue editing.

;q-> quite the current file. This work only if you don’t have any unsaved changes.

;q!->quite the current file and don’t save the change you just made to the file.

2.4. Files System of linux

Linux is a very flexible operating system that has a long history of interoperability with other operating system on a number of different hardware platforms.

Linux supports the following file system.

1. **Ext file system:** the ext file system is the extendable file system. Linux support the various version of ext as ie Ext1, ext2, ext3, ext4.ext4 is the new version of fedora linux. Ext4 provides all the features of ext2 and ext3, and also features journaling and backward compatibility with ext3 and ext2.
2. **Reiserfs:** the reiserfs file system is a journaling file system designed for fast server performance, especially in directories containing thousands of files. It is more space efficient than most of file system, because it does not takes up a minimum of one block per files.
3. **Systemv:** Linux currently provides read support for systemV partitions, and write support is experimental. The system file system driver currently supports APS/EAFS/EFS.
4. **Ufs:**usf is used in Solaris and early BSD operating system. Linus provides read support, and write support is experimental.

5. **FAT:** fat is one of a few different file system used with windows over the years. Almost every computer user has used FAT at one time or another, since it was the sparse base operating system at the heart of all window operating system.
6. **NTFS:** NTFS is the next generation of HPES. It comes with all version of Microsoft operating system beginning with window NT. Unlike FAT, it is a b-tree file system, meaning it has a performance and reliability advantage, including journaling, and support for encryption and compression, over FAT.
7. **IBM JFS:** IBM JFS is an easy to use journaling file system created by IBM. It is designed for high throughput server environments.
8. **SGI XFS:** SGI's Extended File System (XFS) is SGI'S newest file system for all all silicon Graphics system, from workstation to its supercomputer line.

2.5. Directories and their purpose

The Red Hat Linux file system is the structure in which all the information on your computer is stored. Files are organized within a hierarchy of directories. Each directory can contain files, as well as other directories.
Some of the Red Hat Linux directories that may interest you include the following:

- /bin — Contains common Linux user commands, such as ls, sort, date, and chmod.
- /boot — Has the bootable Linux kernel and boot loader configuration files (GRUB).
- /dev — Contains files representing access points to devices on your systems. These include terminal devices (tty*), floppy disks (fd*), hard disks (hd*), RAM (ram*), and CD-ROM (cd*). (Users normally access these devices directly through the device files.)
- /etc — Contains administrative configuration files.
- /home — Contains directories assigned to each user with a login account.
- /mnt — Provides a location for mounting devices, such as remote file systems and removable media (with directory names of cdrom, floppy, and so on).
- /root — Represents the root user's home directory.
- /sbin — Contains administrative commands and daemon processes.
- /tmp — Contains temporary files used by applications.
- /usr — Contains user documentation, games, graphical files (X11), libraries (lib), and a variety of other user and administrative commands and files.
- /var — Contains directories of data used by various applications. In particular, this is where you would place files that you share as an FTP server (/var/ftp) or a Web server (/var/www). It also contains all system log files (/var/log).

3. Installation of Linux

3.1. Partitioning

Choose your partitioning strategy. we have two choices related to how your disk is partitioned for Red Hat Linux installation:

Automatically partition— With this selection, all Linux partitions on all hard disks are erased and used for the installation. The installation process automatically handles the partitioning. (It does give you a chance to review your partitioning, however.)

Manually partition with Disk Druid — With this selection, the Disk Druid utility is run to let you partition your hard disk.

Choose partitioning. If you selected to have the installer automatically partition for us, we can choose from the following options:

Remove all Linux partitions on this system — Windows and other nonLinux partitions remain intact with this selection.

Remove all partitions on this system — This erases the entire hard disk.

Keep all partitions and use existing free space — This only works if you have enough free space on your hard disk that is not currently assigned to any partition.

If you have multiple hard disks, you can select which of those disks should be used for your Red Hat Linux installation. Turn the Review check box on to see how Linux is choosing to partition your hard disk. Click Next to continue.

After reviewing the Partitions screen, you can change any of the partitions you choose, providing you have at least one root (/) partition that can hold the entire installation and one swap partition. A small /boot partition (about 100MB) is also recommended. The swap partition is usually twice the size of the amount of RAM on your computer (for example, for 128MB RAM you could use 256MB of swap). Having less than doubled your physical memory for swap can cause performance problems.

3.2. Installation of Linux

1. **Insert the first CD-ROM in the CD-ROM drive.** If you are installing from a local hard disk or network and you have a floppy disk drive but no CD-ROM drive, you can insert an installation floppy boot disk instead. Refer to the section on creating install disks for information on making the disk (or disks) you need.
2. **Start your computer.** If you see the Red Hat Linux installation screen, continue to the next step.
3. **Start the boot procedure.** At the boot prompt, press Enter to start the boot procedure in graphical mode. If for some reason your computer will not let you install in graphical mode (16-bit color, 800x600 resolution, framebuffer), refer to the "Choosing Different Install Modes" sidebar. Different modes let you start network installs and nongraphical installs (in case your video card can't be detected).
4. **Media check.** At this point, you may be asked to check your installation media. If so, press Enter to check that the CD is in working order. If one of the CDs is damaged, this step saves you the trouble of getting deep into the install before failing. Repeat this step for each CD in the set; then select Skip to continue.
5. **Continue.** When the welcome screen appears, click Release Notes to see information about this version of Red Hat Linux. Click Next when you're ready to continue.
6. **Choose a language.** When prompted, indicate the language that you would like to use during the installation procedure by moving the arrow keys and selecting Next. (Later, you will be able to add additional languages.) You are asked to choose a keyboard.
7. **Choose a keyboard.** Select the correct keyboard layout (U.S. English, with Generic 101-key PC keyboard by default). Some layouts enable dead keys (on by default). Dead keys let you use characters with special markings (such as circumflexes and umlauts).
8. **Add a mouse.** When prompted, indicate the kind of mouse and click Next.
9. **Select Monitor Configuration.** Scroll down the list to find your monitor's manufacturer; then click the plus sign to choose the model. When you select the model, the correct horizontal and vertical sync rates are added, or you can type your own values. If your model is not found, consult the monitor's manual. Then try a Generic CRT or type the monitor's horizontal and vertical sync rates into the appropriate box. Click Next to continue.
10. **Choose install type.** Select either "Install Red Hat Linux" for a new install or "Upgrade an existing installation" to upgrade an existing version of Red Hat.
11. Although most computers will enable you to install Red Hat Linux in the default mode (graphical), there may be times when your video card does not support that mode. Also, though the install process will detect most computer hardware, there may be times when your hard disk, Ethernet card, or other critical piece of hardware cannot be detected and will require you to enter special information at boot time.

12. The following is a list of different installation modes you can use to start the Red Hat Linux install process. You would typically only try these modes if the default mode failed (that is, if the screen was garbled or hardware wasn't detected). For a list of other supported modes, refer to the /usr/share/doc/anaconda*/command-line.txt file or press F2 to see short descriptions of some of these types.

.

13. **Choose your partitioning strategy.** You have two choices related to how your disk is partitioned for Red Hat Linux installation:
    - **Automatically partition**— With this selection, all Linux partitions on all hard disks are erased and used for the installation. The installation process automatically handles the partitioning. (It does give you a chance to review your partitioning, however.)
    - **Manually partition with Disk Druid** — with this selection, the Disk Druid utility is run to let you partition your hard disk.

Click Next to continue.

14. **Choose partitioning.** If you selected to have the installer automatically partition for you, you can choose from the following options:
    - **Remove all Linux partitions on this system** — Windows and other nonLinux partitions remain intact with this selection.
    - **Remove all partitions on this system** — this erases the entire hard disk.
    - **Keep all partitions and use existing free space** — this only works if you have enough free space on your hard disk that is not currently assigned to any partition.

If you have multiple hard disks, you can select which of those disks should be used for your Red Hat Linux installation. Turn the Review check box on to see how Linux is choosing to partition your hard disk. Click Next to continue.

After reviewing the Partitions screen, you can change any of the partitions you choose, providing you have at least one root (/) partition that can hold the entire installation and one swap partition. A small /boot partition (about 100MB) is also recommended. The swap partition is usually twice the size of the amount of RAM on your computer (for example, for 128MB RAM you could use 256MB of swap). Having less than doubled your physical memory for swap can cause performance problems.

Click the Next button (and select OK to accept any changes) to continue.

15. **Configure boot loader.** All bootable partitions and default boot loader options are displayed. By default, the install process will use the GRUB boot loader, install the boot loader in the master boot record of the computer, and choose Red Hat Linux as your default operating system to boot.
16. The names shown for each bootable partition will appear on the boot loader screen when the system starts. Change a partition name by clicking it and selecting Edit. To change the location of the boot loader, click "Configure advanced boot loader options" and continue to the next step. (If the defaults are okay, skip the next step.)
17. To choose where to store the boot loader, select one of the following:
    - **Master Boot Record (MBR)** — This is the preferred place for GRUB. It causes either GRUB or LILO to control the boot process for all operating systems installed on the hard disk.
    - **First Sector of Boot Partition** — If another boot loader is being used on your computer, you can have GRUB installed on your Linux partition (first sector). This lets you have the other boot loader refer to your GRUB boot loader to boot Red Hat Linux.

You can choose to add Kernel Parameters (which may be needed if your computer can't detect certain hardware). You can select to use linear mode (which was once required to boot from a partition on the disk that is above cylinder 1024, but is now rarely needed).

18. **Configure networking.** At this point, you are asked to configure your networking. This applies only to configuring a local area network**.** If you will use only dial-up networking, skip this section by clicking next. If your computer is not yet connected to a LAN, you should skip this section.

19. **Choose language support.** Your installation language should be selected automatically as your default language on this screen. You can select to install support for additional languages by clicking the check boxes next to the languages you want. You can click the Select All button to install all supported languages to your system.
20. **Choose a time zone.** Select the time zone from the list of time zones shown. Either click a spot on the map or choose from the scrolling list. To see a more specific view of your location, click World and choose your continent. From the UTC Offset tab, you can choose a time zone according to the number of hours away from Greenwich Mean Time (GMT), known as the UTC offset.
21. **Set root password.** You must choose a password for your root user at this point. The root password provides complete control of your Red Hat Linux system. Without it, and before you add other users, you will have no access to your own system. Enter the Root Password, and then type it again in the Confirm box to confirm it. (Remember the root user's password and keep it confidential! Don't lose it!) Click Next to continue.
22. **Select Packages.** You are presented with groups of packages at this point. Which packages are selected by default depends on the type of installation you chose earlier. In general, either more workstation-oriented or server-oriented packages are selected. Select the ones you want and click next.
23. Because each group represents several packages, you can click the Details button next to each group to select more specifically the packages within that group. Because workstation and personal desktop installations don't add any server packages, this is a good opportunity to add server packages for the services you expect to use.
24. **About to Install.** A screen tells you that you are about to begin writing to hard disk. You can still back out now, and the disk will not have changed. Click Next to proceed. (To quit without changes, eject the CD and restart the computer.) Now the file systems are created and the packages are installed.
25. **Create boot disk.** If you want to create a boot disk, insert a blank floppy and click Next to create the boot floppy. I recommend that you do this if you have a floppy disk drive, and label the disk with the name of the computer, operating system, and release number. (You can still skip it by clicking no.)
26. **Finish installing.** When you see the "Congratulations" screen, you are done. Note the links to Red Hat Linux information eject the CD and click Exit.

3.3. Troubleshooting of installation

After you have finished installing Red Hat Linux, you can check how the installation went by checking your log files. There are three places to look once the system comes up:

- **/tmp/upgrade.log** — When upgrading packages, output from each installed package is sent to this file. You can see what packages were installed and if any failed.
- **/var/log/dmesg** — This file contains the messages that are sent to the console terminal as the system boots up, including messages relating to the kernel being started and hardware being recognized. If a piece of hardware isn't working, you can check here to make sure that the kernel found the hardware and configured it properly.

- **/var/log/boot.log** — This file contains information about each service that is started up at boot time. You can see if each service started successfully. If a service fails to start properly, there may be clues in this file that will help you learn what went wrong.

Here are lists of utilities you can use to reconfigure different features that were set during installation:

- **Changing or adding a mouse:** redhat-config-mouse
- **Changing a keyboard language:** redhat-config-keyboard

- **Adding or deleting software packages:** redhat-config-packages or rpm
- **Partitioning:** fdisk or sfdisk
- **Boot loader:** /boot/grub/grub.conf (for GRUB); lilo and /etc/lilo.conf (for LILO)
- **Networking (Ethernet & TCP/IP):** redhat-config-network
- **Time zone:** timeconfig or firstboot
- **User accounts:** useradd or redhat-config-users
- **Sound:** redhat-config-soundcard
- **X Window System:** redhat-config-xfree86

4. System Administration

4.1. Root login

The "root" account on a Linux computer is the account with full privileges. Root access is often necessary for performing commands in Linux, especially commands that affect system files. Because root is so powerful, it's recommended to only request root access when necessary, as opposed to logging in as the root user. This can help prevent accidental damage to important system files.

The root user has complete control of the operation of your Red Hat Linux system. That user can open any file or run any program. The root user also installs software packages and adds accounts for other people who use the system.

The home directory for the root user is /roots. That and other information associated with the root user account is located in the /etc/passwd file. Here is what the root entry looks like in the /etc/passwd file:

root:x:0:0:root:/root:/bin/bash

This shows that for the user named root, the user ID is set to 0 (root user), the group ID is set to 0 (root group), the home directory is /root, and the shell for that user is /bin/bash. You can change the home directory or the shell used, if you like, by editing the values in this file

4.2. Super user

In addition to the two user types, there is the *superuser*, or *root* user, that has the ability to override any file ownership and permission restrictions. In practice, this means that the superuser has the rights to access anything on its own server. This user is used to make system-wide changes, and must be kept secure.

It is also possible to configure other user accounts with the ability to assume "superuser rights". In fact, creating a normal user that has sudo privileges for system administration tasks is considered to be best practice.The answer is that you can use the su command. From any Terminal window or shell, you can simply type:

```
$ su
Password: ******
#
```

We can use the sudo command to login in super user

The different between su and sudo is as

The su command switches to the super user – or root user – when you execute it with no additional options. You'll have to enter the root account's password. This isn't all the su command does, though – you can use it to switch to any user account. If you execute the **su bca** command, you'll be prompted to enter bca's password and

the shell will switch to bca's user account.Once you're done running commands in the root shell, you should type **exit** to leave the root shell and go back to limited-privileges mode.

Sudo runs a single command with root privileges. When you execute **sudo command**, the system prompts you for your current user account's password before running **command** as the root user. By default, Ubuntu remembers the password for fifteen minutes and won't ask for a password again until the fifteen minutes are up.

Using sudo for assigning administrative privilege

One way to give full or limited root privileges to any non-root user is to set up the sudo facility. That simply entails adding the user to /etc/sudoers and defining what privilege you want that user to have. Then the user can run any command he or she is privileged to use by preceding that command with the sudo command.

The following is an example of how to use the sudo facility to cause any users that are added to the wheel group to have full root privileges:

1. As the root user, edit the /etc/sudoers file by running the visudo command:
2. `# **visudo**`
   By default, the file is opened in vi, unless your EDITOR variable happens to be set to some other editor acceptable to visudo (for example, export EDITOR=gedit) The reason for using visudo is that the command will lock the /etc/sudoers file and do some basic sanity-checking of the file to ensure it was edited correctly.

3. Uncomment the following line to allow users in the group named wheel to have full root privileges on the computer:

   ```
   %wheel     ALL=(ALL)     ALL
   ```

   The previous line causes the user to be prompted for a password to be allowed to use administrative commands. To allow users in the wheel group to have that privilege without using a password, uncomment the following line instead:

   ```
   %wheel     ALL=(ALL)     NOPASSWD: ALL
   ```

4. Save the changes to the /etc/sudoers file (in vi, type **ZZ**).
5. Still as root user, open the /etc/group file in any text editor and add the users you want to have root privilege to the wheel line. For example, if you were to add the users mary and jake to the wheel group, the line would appear as follows:

   ```
   wheel:x:10:root,suresh,bca
   ```

At this point, the users suresh and bca can run the sudo command to run commands, or parts of commands, that are normally restricted to the root user. The following is an example of a session by the user jake after he has been assigned sudo privileges:

```
[bca]$ sudoumount /mnt/win

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these two things:

        #1) Respect the privacy of others.
```

```
        #2) Think before you type.

Password: *********
[bca]$ mount /mnt/win
mount: only root can mount /dev/hda1 on /mnt/win
[bca]$ sudo mount /mnt/win
[bca]$
```

In the above session, the user bca runs the sudo command so he can unmount the /mnt/win file system (using the umount command). He is given a warning and asked to provide his password (this is bca's password, *not* the root password).

4.3. Configuration of hardware with kudzu

When you add or remove hardware from your computer and reboot Red Hat Linux, a window appears during the reboot process advising that hardware has either been added or removed and asking if you want to reconfigure it. The program that detects and reconfigures your hardware is called kudzu.

The kudzu program is a hardware autodetection and configuration tool that runs automatically at boot time. If you like, you can also start kudzu while Red Hat Linux is running. In either case, here is what kudzu does:

1. It checks the hardware connected to your computer.
2. It compares the hardware it finds to the database of hardware information stored in the /etc/sysconfig/hwconf file.
3. It prompts you to change your system configuration, based on new or removed hardware that was detected.

The following is a list of hardware that kudzu can detect (according to the kudzu README file), followed by a description of what kudzu does to configure the device. Other devices may be detected as well (such as USB devices).

- **Network devices** — Adds an Ethernet interface alias (eth0, eth1, etc.) if necessary and either migrates the old device configuration or creates a new one.
- **SCSI** — Adds an alias for scsi_hostadapter.
- **Sound card** — Runs the sndconfig command to configure and test the sound card.
- **Mouse** — Links the new mouse device to /dev/mouse and runs the mouseconfig command to configure and test the mouse.
- **Modem** — Links the new modem device to /dev/modem.
- **CD-ROM** — Links the CD-ROM device to /dev/cdrom.
- **Scanner** — Links the new scanner device to /dev/scanner.
- **Keyboard** — Runs the kbdconfig command to reconfigure the keyboard. Also, if you are using a serial console, it makes sure /etc/inittab and /etc/securetty are configured to be used by a serial console.

The following is a list of actions kudzu takes when a device is removed:

- **Network** — Removes the alias for the Ethernet interface (eth0, eth1, etc.).
- **SCSI** — Removes the alias for the SCSI host adapter (scsi_hostadapter).
- **Mouse** — Removes the link to /dev/mouse.
- **Modem** — Removes the link to /dev/modem.
- **CD-ROM** — Removes the link to /dev/cdrom.
- **Scanner** — Removes the link to /dev/scanner.

To run kudzu, either reboot (during the reboot, kudzu is run automatically) or switch to a virtual terminal (Ctrl+Alt+F2), log in as root, and run the kudzu command. For any hardware that has been added or removed since the last time kudzu was run, you are asked if you want to configure it, not configure it, or do nothing.

4.4. Checking system space

Displaying system space with df

You can display the space available in your file systems using the df command. To see the amount of space available on all of the mounted file systems on your Linux computer, type df with no options:

$ df

```
Filesystem  1k-blocks     Used   Available  Use%    Mounted on
/dev/hda3    30645460  2958356    26130408   11%    /
/dev/hda2       46668     8340       35919   19%    /boot
/dev/fd0         1412       13        1327    1%    /mnt/floppy
```

Checking disk usage with du

To find out how much space is being consumed by a particular directory (and its subdirectories), you can use the ***du*** command. With no options, ***du*** lists all directories below the current directory, along with the space consumed by each directory. At the end, ***du*** produces total disk space used within that directory structure.

The du command is a good way to check how much space is being used by a particular user (du /home/user1) or in a particular file system partition (du /var). By default, disk space is displayed in 1K block sizes. To make the output more friendly (in kilobytes, megabytes, and gigabytes), use the -h option as follows:

```
$ du -h /home/bca
114k    /home/bca/httpd/stuff
234k    /home/bca/httpd
137k    /home/bca/uucp/data
701k    /home/bca/uucp
1.0M    /home/bca
The output shows the disk space used in each directory under the home directory
```

Finding disk consumption with find

The find command is a great way to find file consumption of your hard disk using a variety of criteria. You can get a good idea of where disk space can be recovered by finding files that are over a certain size or were created by a particular person.

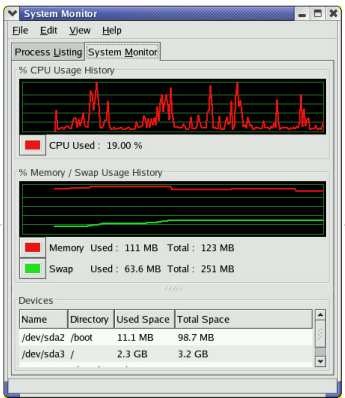
In below example, the find command searches the root file system (/) for any files owned by the user named bca (-user bca) and prints the filenames. The output of the find command is then listed with a long listing in size order (ls -ldS). Finally that output is sent to the file /tmp/bca. When you read the file /tmp/bca, you will find all of the files that are owned by the user bca, listed in size order. Here is the command line:

```
# find / -user bca -print -xdev | xargs ls -ldS > /tmp/jake
```

4.5. Monitoring system performance

Watch computer usage with System Monitor

If you like visual representations of your system use, the System Monitor provides a great way to see how much your system is being used. To open the System Monitor from the Red Hat menu, click System Tools → System Monitor.



Monitoring CPU usage with top

Start the top utility in a Terminal window, and it displays the top CPU consuming processes on your computer. Every five seconds, top will determine which processes are consuming the most CPU time and display them in descending order on your screen.

By adding the -S option to top, you can have the display show you the cumulative CPU time that the process, as well as any child processes that may already have exited, has spent. If you want to change how often the screen is updated, you can add the -d secs option, where secs is replaced by the number of seconds between updates.

By default, processes are sorted by CPU usage. You can sort processes numerically by PID (press N), by age (press A), by resident memory usage (press M), by time (press T), or back to CPU usage (press P).

4.6. Working with file system

File systems in Red Hat Linux are organized in a hierarchy, beginning from root (/) and continuing downward in a structure of directories and subdirectories. As an administrator of a Red Hat Linux system, it is your duty to make sure that all the disk drives that represent your file system are available to the users of the computer. It is also your job to make sure there is enough disk space in the right places in the file system for users to store what they need.

The organization of your file system begins when you install Linux. Part of the installation process is to divide your hard disk (or disks) into partitions. Those partitions can then be assigned to:

- A part of the Linux file system,
- Swap space for Linux, or
- Other file system types (perhaps containing other bootable operating systems)

For our purposes, I want to focus on partitions that are used for the Linux file system. To see what partitions are currently set up on your hard disk, use the fdisk command as follows:

```
# fdisk -l
Disk /dev/hda:  40.0 GB, 40020664320
255 heads, 63 sectors/track, 4825 cylinders
Units = cylinders of 16065 * 512 bytes = 8225280 bytes

   Device Boot    Start       End    Blocks   Id  System
/dev/hda1   *         1        13       104    b  Win95 FAT32
/dev/hda2            84        89     48195   83  Linux
/dev/hda3            90       522   3478072+  83  Linux
/dev/hda4           523       554    257040    5  Extended
/dev/hda5           523       554    257008+  82  Linux swap
```

Using the mkfs command to create a file system

It is possible to create a file system, for any supported file system type, on a disk or partition that you choose. This is done with the mkfs command. While this is most useful for creating file systems on hard disk partitions, you can create file systems on floppy disks or re-writable CDs as well.

4.7. Configuration modules

There are a few commands that allow you to maniuplate the kernel. Each is quickly described below, for more information say `man [command]`.

- **depmod** - handle dependency descriptions for loadable kernel modules.
- **insmod** - install loadable kernel module.
- **lsmod** - list loaded modules.
- **modinfo** - display information about a kernel module.
- **modprobe** - high level handling of loadable modules.
- **rmmod** - unload loadable modules.

Others commands are as following:

1. Listing loaded modules

To see which modules are currently loaded into the running kernel on your computer, you can use thelsmod command. Here's an example:

```
# lsmod
Module                  Size  Used by
sr_mod                 15120   0  (autoclean)
es1371                 26784   0  (autoclean)
ac97_codec              8704   0  (autoclean) [es1371]
gameport                1920   0  (autoclean) [es1371]
soundcore               4112   4  (autoclean) [es1371]
binfmt_misc             6272   1
nuscsitcp              17200   0  (unused)
autofs                 10816   1  (autoclean)
```

2. modules information.

To find information about any of the loaded modules, you can use the modinfo command. For example, you could type the following:

```
# modinfo -d es1371
"ES1371 AudioPCI97 Driver"
```

Loading modules

You can load any module that has been compiled and installed (to the /lib/modules directory) into your running kernel using the insmod command. The most common reasons for loading a module are that you want to use a feature temporarily (such as loading a module to support a special file system on a floppy you want to access) or to identify a module that will be used by a particular piece of hardware that could not be autodetected.

Here is an example of the insmod command being used to load the parport module. The parport module provides the core functions to share parallel ports with multiple devices.

```
# insmodparport
Using /lib/modules/2.4.20-2.48/kernel/drivers/parport/parport.o
```

Removing modules

You can remove a module from a running kernel using the rmmod command. For example, to remove the module parport_pc from the current kernel, type the following:

```
#rmmodparport_pc
```

If the module is not currently busy, the parport_pc module is removed from the running kernel.

5. User management

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

5.1. Creating user accounts

Every person who uses your Red Hat Linux system should have a separate user account. Having a user account provides each person with an area in which to securely store files, as well as a means of tailoring his or her user interface (GUI, path, environment variables, and so on) to suit the way that he or she uses the computer.

You can add user accounts to your Red Hat Linux system in several ways. This chapter describes how to use the *useradd* command to add user accounts to Red Hat Linux from the command line, and how to use the Red Hat User Manager window to add users from the desktop.

Adding users with *useradd*

The most straightforward method for creating a new user from the shell is with the *useradd* command. After opening a Terminal window with root permission, you simply invoke the *useradd* command at the command prompt, with details of the new account as parameters.

The only required parameter to *useradd* is the login name of the user, but you will probably want to include some additional information. Each item of account information is preceded by a single letter option code with a dash in front of it.
syntax
useradd username
useradd bca8
In creating the account for bca8, the *useradd* command performs several actions:

- Reads the /etc/login.defs file to get default values to use when creating accounts.
- Checks command-line parameters to find out which default values to override.
- Creates a new user entry in the /etc/passwd and /etc/shadow files based on the default values and command-line parameters.
- Creates any new group entries in the /etc/group file.
- Creates a home directory based on the user's name and located in the /home directory.
- Copies any files located within the /etc/skel directory to the new home directory. This usually includes login and application startup scripts

Adding users with Red Hat User Manager

If you prefer a graphical window for adding, changing, and deleting user accounts, you can use the Red Hat User Manager window. To open the window from the GNOME desktop, click System Settings → Users and Groups (or type redhat-config-users from a Terminal window as root user).

When you open the Red Hat User Manager window, you see a list of all regular users that are currently added to your computer. Administrative users (UID 1 through 100) are not displayed. For each user, you can see the user name, UID, primary group, full name, login shell, and home directory. Click on any of those headings to sort the users by that information.
To add a new user from the Red Hat User Manager window, do the following:

1. Click the Add User icon. A Create New User window appears,
2. Type the requested information in the following fields:
    - **User Name (** A single word to describe the user. Typically, the user name is eight characters, all lowercase, containing the user's real first name, last name, or (more often) a combination of the two.
    - **Full Name (The** user's full name (usually first name, middle initial, and last name). This name is typically just used for display, so using upper- and lowercase is fine.
    - **Password (**The user's initial password. (Ask the user to change this password the first time he or she logs in to the new account, using the passwd command.)
    - **Confirm Password (** Type the password again, to make sure you entered it correctly.
    - **Login Shell (** The default shell (for entering typed commands) that the user sees when first logging in to Red Hat Linux from a character display.
    - **Create home directory (** By default, this box is selected and the user's home directory (as indicated by the Home Directory field) is created automatically.
    - **Home Directory (** By default, the user is given a home directory of the user's name in the /home directory. (For example, the user sheree would be assigned /home/sheree as her home directory.) Change this field if you want to assign the user to a different home directory.
    - **Create a private group for the user (** Check this box if you want a group by the same name as the user, created for this user. The name is added to the /etc/group file. This feature is referred to as user private groups (UPGs).

5.2. Setting user defaults

The useradd command and Red Hat User Manager window both determine the default values for new accounts by reading the /etc/login.defs file. You can modify those defaults by either editing that file manually with a standard text editor or by running the useradd command with the -D option. If you choose to edit the file manually, here is what you face:

```
# *REQUIRED*
```

```
# Directory where mailboxes reside, _or_ name of file, relative to the
# home directory. If you _do_ define both, MAIL_DIR takes precedence.
# QMAIL_DIR is for Qmail
#
#QMAIL_DIR Maildir
MAIL_DIR    /var/spool/mail
#MAIL_FILE .mail

# Password aging controls:
#
# PASS_MAX_DAYS Maximum number of days a password may be used.
# PASS_MIN_DAYS Minimum number of days allowed between password changes.
# PASS_MIN_LEN  Minimum acceptable password length.
# PASS_WARN_AGE Number of days warning given before a password
# expires.
#
PASS_MAX_DAYS     99999
PASS_MIN_DAYS     0
PASS_MIN_LEN      5
PASS_WARN_AGE     7

#
# Min/max values for automatic uid selection in useradd
#
UID_MIN                  500
UID_MAX                60000

#
# Min/max values for automatic gid selection in groupadd
#
GID_MIN                  500
GID_MAX                60000

#
# If defined, this command is run when removing a user.
# It should remove any at/cron/print jobs etc. owned by
# the user to be removed (passed as the first argument).
#
#USERDEL_CMD /usr/sbin/userdel_local

#
# If useradd should create home directories for users by default.
# On RH systems, we do. This option is ORed with the -m flag on
# useradd command line.
#
CREATE_HOME yes
```

Blank lines and comments beginning with a pound sign (#) are ignored. All other lines contain keyword/value pairs. For example, the keyword MAIL_DIR is followed by some white space and the value /var/spool/mail. This tells useradd that the initial user e-mail mailbox is created in that directory. Following that are lines that enable you to customize the valid range of automatically assigned user ID numbers or group ID numbers. A comment section that explains that keyword's purpose precedes each keyword. Altering a default value is as simple as editing the value associated with that keyword and then saving the login.defs file.

If you want to view the defaults, type the useradd command with the -D option as follows:

```
# useradd -D
GROUP=100
```

```
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
```

You can also use the -D option to change defaults. When run with this flag, useradd refrains from actually creating a new user account; instead, it saves any additionally supplied options as the new default values in /etc/login.defs.

5.3. Providing supports to users

Creating a technical support mailbox

E-mail is a wonderful communication tool, especially for the overworked system administrator. People usually put more thought and effort into their e-mail messages than into the voice messages that they leave. A text message can be edited for clarity before being sent, and important details can be cut and pasted from other sources. This makes e-mail an excellent method for Red Hat Linux users to communicate with their system administrator.

In an office with only a few users, you can probably get away with using your personal mailbox to send and receive support e-mails. In a larger office, however, you should create a separate mailbox reserved only for technical support issues. This has several advantages over the use of your personal mailbox:

- Support messages will not be confused with personal, non-support-related messages.
- Multiple people can check the mailbox and share administrative responsibility without needing to read each other's personal e-mail.
- Support e-mail is easily redirected to another person's mailbox when you go on vacation. Your personal e-mail continues to go to your personal mailbox.

One easy solution is to simply create a support e-mail alias that redirects messages to an actual mailbox or list of mailboxes. For example, suppose you wish to create a support alias that redistributes e-mail to the user accounts for support staff members Joe, Mary, and Bob. You would log in as root, edit the /etc/aliases file, and add lines similar to the following:

```
# Technical support mailing list
support: joe, mary, bob
```

After saving the file, you need to run the newaliases command to recompile the /etc/aliases file into a database format. Now your users can send e-mail to the support e-mail address, and the message is automatically routed to everyone on the list. When a member of the list responds to that message, he or she should use the "Reply To All" option so that the other support staff members also see the message.

Resetting a user's password

One common (if not *the* most common) problem that your users will encounter is the inability to log in because:

- They have the Caps Lock key on.
- They have forgotten the password.
- The password has expired.

If the Caps Lock key is not on, then you probably need to reset the individual's password. You can't look up the password because Red Hat Linux stores passwords in an encrypted format. Instead, use the passwd command to assign a new password to the user's account. Tell the user what that new password is (preferably in person), but then set the password to expire soon so that he or she must choose one (hopefully, a new one that is more easily remembered).

If you must reset a user's password, do so with the passwd command. While logged in as root, type passwd followed by the login name you are resetting. You are prompted to enter the password twice. For example, to change the password for mary, type:

```
# passwd mary
```

After resetting the password, set it to expire so that the user is forced to change it the next time she logs in. You can use the chage command to set an expiration period for the password and to trick the system into thinking that the password is long overdue to be changed.

```
# chage -M 30 -d 0 mary
```

The -M 30 option tells the system to expire Mary's password every 30 days.

5.4. Modification accounts

The ***usermod*** command is similar to the ***useradd*** command and even shares some of the same options. However, instead of adding new accounts, it enables you to change various details of existing accounts. When invoking the ***usermod*** command, you must provide account details to change followed by the login name of the account. Table below lists the available options for the ***usermod*** command.

Table: usermod Options for Changing Existing Accounts

Options	Description
-c comment	Change the description field of the account. You can also use the chfn command for this. Replace comment with a name or other description of the user account, placing multiple words in quotes.
-d home_dir	Change the home directory of the account to the specified new location. If the -m option is included, copy the contents of the home directory as well. Replace home_dir with the full path to the new directory.
-e expire_date	Assign a new expiration date for the account, replacing expire_date with a date in MM/DD/YYYY format.
-f inactivity	Set the number of days after a password expires until the account is permanently disabled. Setting inactivity to 0 disables the account immediately after the password has expired. Setting it to -1 disables the option, which is the default behavior.
-g group	Change the primary group (as listed in the /etc/group file) that the user is in. Replace group with the name of the new group.
-G grouplist	Set the list of groups that user belongs to. Replace grouplist with a list of groups.
-l login_name	Change the login name of the account to the name supplied after the -l option. Replace login_name with the new name. This automatically changes the name of the home directory; use the -d and -m options for that.

Table: usermod Options for Changing Existing Accounts

Options	Description
-m	This option is used only in conjunction with the -d option. It causes the contents of the user's home directory to be copied to the new directory.
-o	This option is used only in conjunction with the -u option. It removes the restriction that user IDs must be unique.
-s shell	Specify a new command shell to use with this account. Replace shell with the full path to the new shell.
-u user_id	Change the user ID number for the account. Replace user_id with the new user ID number. Unless the -o option is used, the ID number must not be in use by another account.

Assume that a new employee named Jenny Barnes will be taking over Mary's job. We want to convert the mary account to a new name (-l jenny), new comment (-c "Jenny Barnes"), and home directory (-d /home/jenny). We could do that with the following command:

```
# usermod -l jenny -c "Jenny Barnes" -m -d /home/jenny mary
```

Furthermore, if after converting the account we learn that Jenny prefers the tcsh shell, we could make that change with the -s option (-s /bin/tcsh):

```
# usermod -s /bin/tcsh jenny
```

Instead, we could use the chsh command to change the shell. The following is an example:

```
# chsh -s /bin/tcsh jenny
```

The chsh command is handy because it enables a user to change his or her own shell setting. Simply leave the user name parameter off when invoking the command, and chsh assumes the currently logged-in user as the account to change.

5.5. Deleting user accounts

Deleting user accounts with userdel

The userdel command takes a single argument, which is the login name of the account to delete. If you supply the optional -r option, it also deletes the user's home directory and all the files in it. To delete the user account with login name mary, you would type this:

```
# userdel mary
```

To wipe out her home directory along with her account, type this:

```
# userdel -r mary
```

Files owned by the deleted user but not located in the user's home directory will not be deleted. The system administrator must search for and delete those files manually. The find command comes in very handy for this type of thing. I won't describe all the capabilities of the find command (that would take a very fat chapter of its

own). I do, however, provide a few simple examples of how to use find to locate files belonging to a particular user, even when those files are scattered throughout a file system.

5.6. Checking disk quotas

Using quota to check disk usage

A careless or greedy user can gobble up all the space on your hard disk and, possibly, bring your computer to a halt. By using disk quotas, you can limit the amount of disk resources a user or group can use up.

The quota package contains a set of tools that lets you limit the amount of disk space (based on disk blocks) and files (based on inodes) that a user can consume. Using quotas, you can limit the amount of usage (on a per-user and -group basis) for each file system on your computer. The general steps for setting disk quotas are:

- Creating quota files
- Editing the /etc/fstab file
- Creating quota rules
- Checking quotas

You set quotas on disk partitions listed in your /etc/fstab file. For computers that are shared by many users, there may be a separate /home or /var partition where users are expected to put all their data. That kind of partition boundary can prevent an entire disk from being consumed by one user. Quotas on the /home or /var partition can make sure that the space within those partitions are shared fairly among your computer's users.

Editing the /etc/fstab file

You need to add quota support to the file system. To do that, edit the /etc/fstab file and add the usrquota option to field number four of the partition for which you want to set quotas. Here is an example of a line from /etc/fstab:

```
/dev/hda2     /home      ext3    defaults,usrquota,grpquota       1 2
```

Here, the /home file system is used to allow disk quotas for all users' home directories under the /home directory.

Creating quota files

You need to have aquota.user and/or aquota.group files in the root directory of the partition on which you want to establish disk quotas. To add quotas based on individual users, you need an aquota.user file, while aquota.group is needed to set quotas based on groups. One way to create these files is with the quotacheck command. Here is an example of the quotacheck command to create an initial aquota.user file:

```
# quotacheck -c /home
```

Creating a quota startup script

If the quota package doesn't include a startup script (and it doesn't with the current Red Hat Linux), you can create your own. You want this script to check quotas (quotacheck command), start the quota service (quotastart command) and turn off the service (quotaoff command).

Open a new file called /etc/init.d/quota as root user, using any text editor.

Turn on the quota startup script

If you created a quota file, as described in the previous step, you need to make it executable and set it to start automatically when you start Red Hat Linux. To do those things, type the following as root user:

```
# chmod 755 /etc/init.d/quota
# chkconfig on quota
```

Using du to check disk use

You can discover the most voracious consumers of disk space using the du command. Invoke du with the -s option and give it a list of directories; it reports the total disk space used by all the files in each directory. Add an -h option to display disk space used in numbers, followed by kilobytes (k), megabytes (M), or gigabytes (G). The -c option adds a total of all requested directories at the end. The following checks disk usage for several home directories:

```
# du -h -c -s /home/tom /home/bill /home/tina /home/sally
```

This should result in a list of all of your users' home directories preceded by the number of kilobytes that each directory structure uses. It looks something like this:

```
339M    /home/tom
81M     /home/bill
31M     /home/tina
44k     /home/sally
450M    total
```

5.7. Sending mail to all users

Occasionally, you need to send messages to all users on your system. Warning users of planned downtime for hardware upgrades is a good example. Sending e-mail to each user individually is extremely time consuming and wasteful; this is precisely the kind of task that e-mail aliases and mailing lists were invented for. Keeping a mailing list of all the users on your system can be problematic, however. If you are not diligent about keeping the mailing list current, it becomes increasingly inaccurate as you add and delete users. Also, if your system has many users, the mere size of the alias list can become unwieldy.

The following script, called mailfile, provides a simple method of working around these problems. It grabs the login names from the /etc/passwd file and sends e-mail to all users.

6. Security and system handling

6.1. Understanding shell scripts

The shell is a user program or it is an environment provided for user interaction. It is a command language interpreter that executes commands read from the standard input device such as keyboard or from a file. The shell gets started when you log in or open a console (terminal).Quick and dirty way to execute utilities. The shell is not part of system kernel, but uses the system kernel to execute programs, create files etc. Several shells are available for Linux including:

1. BASH (Bourne-Again SHell) - Most common shell in Linux. It's Open Source.
2. CSH (C SHell) - The C shell's syntax and usage are very similar to the C programming language.
3. KSH (Korn SHell) - Created by David Korn at AT & T Bell Labs. The Korn Shell also was the base for the POSIX Shell standard specifications.
4. TCSH - It is an enhanced but completely compatible version of the Berkeley UNIX C shell (CSH).

Please note that each shell does the same job, but each understands different command syntax and provides different built-in functions. Under MS-DOS, the shell name is COMMAND.COM which is also used for the same purpose, but it is by far not as powerful as our Linux Shells are!

Shell Prompt

Process of writing and executing a script

- Open terminal.
- Navigate to the place where you want to create script using ‘cd‘command.
- Cd (enter) [This will bring the prompt at Your home Directory].
- touch hello.sh (Here we named the script as hello, remember the ‘.sh‘ extension is compulsory).
- vi hello.sh (nano hello.sh) [You can use your favourite editor, to edit the script].
- chmod 744 hello.sh (making the script executable).
- sh hello.sh or ./hello.sh (running the script)

Writing your First Script

```
#!/bin/bash
# My first script
echo "Hello World!"
```

Save the above lines on a text file, make it executable and run it, as described above.

```
Hello World!
```

Writing your Second Script

OK time to move to the next script. This script will tell you, your’s “username” and list the running processes.

```
#! /bin/bash
echo "Hello $USER"
echo "Hey i am" $USER "and will be telling you about the current processes"
echo "Running processes List"
ps
```

Create a file with above codes, save it to anything you want, but with extension “.sh“, make it executable and run it, from you terminal.

Sample Output

```
Hello tecmint
Hey i am tecmint and will be telling you about the current processes
```

```
Running processes List
PID TTY          TIME CMD
1111 pts/0    00:00:00 bash
1287 pts/0    00:00:00 sh
1288 pts/0    00:00:00 ps
 World!
```

Writing your Third Script

Moving to, write our third and last script for this article. This script acts as an interactive script. Why don't you, yourself execute this simple yet interactive script and tell us how you felt.

```
#! /bin/bash
echo "Hey what's Your First Name?";
read a;
echo "welcome Mr./Mrs. $a, would you like to tell us, Your Last Name";
read b;
echo "Thanks Mr./Mrs. $a $b for telling us your name";
echo "********************"
echo "Mr./Mrs. $b, it's time to say you good bye"
```

output

```
Hey what's Your First Name?
Suresh
welcome Mr./Mrs. Suresh, would you like to tell us, Your Last Name
Sharma
Thanks Mr./Mrs. Suresh Sharma for telling us your name
**************************************************
Mr./Mrs. Kumar, it's time to say you good bye
```

6.2. System start up and shutdown

System Initialization

When you turn on your computer, a lot happens even before Red Hat Linux starts up. Here are the basic steps that occur each time you boot up your computer to run Red Hat Linux:

1. **Boot hardware —** Based on information in the computer's read-only memory (referred to as the BIOS), your computer checks and starts up the hardware. Some of that information tells the computer which devices (floppy disk, CD, hard disk, and so on) to check to find the bootable operating system.
2. **Start boot loader —** Typically, the BIOS checks the master boot record on the primary hard disk to see what to load next. With Red Hat Linux installed, the GRUB boot loader is started, allowing you to choose to boot Red Hat Linux or another installed operating system.
3. **Boot the kernel —** Assuming that you selected to boot Red Hat Linux, the Linux kernel is loaded. That kernel mounts the basic file systems and transfers control to the init process. The rest of this section describes what happens after the kernel hands off control of system start-up to the init process.

Starting init

In the boot process, the transfer from the kernel phase (the loading of the kernel, probing for devices, and loading drivers) to init is indicated by the following lines:

```
INIT: version 2.84 booting
                        Welcome to Red Hat Linux
```

The init program, part of the SysVinit RPM package, is now in control. Known as "the father of all processes," the output from ps always lists init as PID (process identifier) 1. Its actions are directed by the /etc/inittab file, which is reproduced next.

During system start-up, a series of scripts are run to start the services that you need. These include scripts to start network interfaces, mount directories, and monitor your system. Most of these scripts are run from subdirectories of /etc/rc.d. The program that starts most of these services up when you boot and stops them when you shut down is the /etc/rc.d/rc script. The following sections describe run-level scripts and what you can do with them.

Starting run-level scripts

As previously mentioned, the /etc/rc.d/rc script is a script that is integral to the concept of run levels. Any change of run level causes the script to be executed, with the new run level as an argument. Here's a quick run-down of what the /etc/rc.d/rc script does:

- **Checks that run level scripts are correct.** The rc script checks to find each run level script that exists and excludes those that represent backup scripts left by rpm updates.
- **Determines current and previous run levels.** Determines the current and previous run levels to know which run-level scripts to stop (previous level) and start (current level).
- **Decides whether to enter interactive startup.** If the confirm option is passed to the boot loader at boot time, all server processes must be confirmed at the system console before starting.
- **Kills and starts run-level scripts.** Stops run-level scripts from the previous level, then starts run-level scripts from the current level.

In Red Hat Linux, most of the services that are provided to users and computers on the network are started from run-level scripts.

6.3. Scheduling system task

Frequently, you need to run a process unattended or at off-hours. The at facility is designed to run such jobs at specific times. Jobs you submit are spooled in the directory /var/spool/at, awaiting execution by the at daemon atd. The jobs are executed using the current directory and environment that was active when the job was submitted. Any output or error messages that haven't been redirected elsewhere are e-mailed to the user that submitted the job.

The following sections describe how to use the at, batch, and cron facilities to schedule tasks to run at specific times. These descriptions also include ways of viewing which tasks are scheduled and deleting scheduled tasks that you don't want to run anymore.

Using at.allow and at.deny

There are two access control files designed to limit which users can use the at facility. The file /etc/at.allow contains a list of users that are granted access, and the file /etc/at.deny contains a similar list of those who may not submit at jobs. If neither file exists, only the superuser is granted access to at. If a blank /etc/at.deny file exists (as in the default configuration), all users are allowed to utilize the at facility to run their own at jobs

Specifying when jobs are run

There are many different ways to specify the time at which an at job should run (most of which look like spoken commands)

Table : Samples for Specifying Times in an at Job

Command Line	When the Command Is Run
at now	The job is run immediately.
at now + 2 minutes	The job will start 2 minutes from the current time.
at now + 1 hour	The job will start one hour from the current time.
at now + 5 days	The job will start five days from the current time.
at now + 4 weeks	The job will start four weeks from the current time.
at now next minute	The job will start in exactly 60 seconds.
at now next hour	The job will start in exactly 60 minutes.
at now next day	The job will start at the same time tomorrow.
at now next month	The job will start on the same day and at the same time next month.
at now next year	The job will start on the same date and at the same time next year.
at now next fri	The job will start at the same time next Friday.
at teatime	The job will run at 4 p.m. They keywords noon and midnight can also be used.
at 16:00 today	The job will run at 4 p.m. today.
at 16:00 tomorrow	The job will run at 4 p.m. tomorrow.
at 2:45pm	The job will run at 2:45 p.m. on the current day.
at 14:45	The job will run at 2:45 p.m. on the current day.
at 5:00 Sep 14 2003	The job will begin at 5 a.m. on September 14, 2003.
at 5:00 9/14/03	The job will begin at 5 a.m. on September 14, 2003.

Submitting scheduled jobs

The at facility offers a lot of flexibility in how you can submit scheduled jobs. There are three ways to submit a job to the at facility:

- **Piped in from standard input**. For example, the following command will attempt to build the Perl distribution from source in the early morning hours while the machine is likely to be less busy:

  ```
  echo "cd /tmp/perl; make ; ls -al" | at 2am tomorrow
  ```

  An ancillary benefit to this procedure is that a full log of the compilation process will be e-mailed to the user that submitted the job.

- **Read as standard input.** If no command is specified, at will prompt you to enter commands at the special at> prompt, as shown in the following example. You must indicate the end of the commands by pressing Ctrl+D, which signals an End of Transmission (<EOT>) to at.
- `$ **at 23:40**`
- `at> **cd /tmp/perl**`
- `at> **make**`
- `at> **ls -al**`
- `at> ***<Ctrl-d>***`
- **Read from a file.** When the -f command-line option is followed by a valid filename, the contents of that file are used as the commands to be executed, as in the following example:
- `$ **at -f /root/bin/runme now + 5 hours**`

This runs the commands stored in /root/bin/runme in five hours. The file can either be a simple list of commands or a shell script to be run in its own subshell (that is, the file begins with #!/bin/bash or the name of another shell).

Viewing scheduled jobs

You can use the atq command (effectively the same as at -l) to view a list of your pending jobs in the at queue, showing each job's sequence number, the date and time the job is scheduled to run, and the queue in which the job is being run.

The two most common queue names are "a," which represents the at queue, and "b," which represents the batch queue. All other letters (upper- and lowercase) can be used to specify queues with lower priority levels. If the atq command lists a queue name as =, it indicates that the job is currently running. Here is an example of output from the atq command:

```
# atq
2    2003-09-02 00:51 a
3    2003-09-02 00:52 a
4    2003-09-05 23:52 a
```

Here you can see that there are three at jobs pending (job numbers 2, 3, and 4, all indicated as "a"). After the job number, the output shows the date and hour each job is scheduled to run.

Deleting scheduled jobs

If you decide that you'd like to cancel a particular job, you can use the atrm command (equivalent to at -d) with the job number (or more than one) as reported by the atq command. For example, using the following output from atq:

```
# atq
18      2003-09-01 03:00 a
19      2003-09-29 05:27 a
20      2003-09-30 05:27 a
21      2003-09-14 00:01 a
22      2003-09-01 03:00 a
```

you can remove the jobs scheduled to run at 5:27 a.m. on September 29 and September 30 from the queue with the command atrm

6.4. Backing up and restore

Selecting a Backup Strategy

There are several approaches to backing up your data. You need to ask yourself a few questions to decide which approach is best for you. Some things that you should consider are:

- In the event of a crash, how much downtime can I tolerate?
- Will I need to recover older versions of my files or is the most recent revision sufficient?
- Do I need to back up files for just one computer or for many computers on a network?

Your answers to these questions will help you decide how often to do full backups and how often to do incremental backups. If the data is particularly critical, you may even decide that you need to have your data

duplicated constantly, using a technique called *disk mirroring*. The following sections describe different backup methods.

Full backup

A full backup is one that stores every file on a particular disk or partition. If that disk should ever crash, you can rebuild your system by restoring the entire backup to a new disk. Whatever backup strategy you decide on, some sort of full backup should be part of it. You may perform full backups every night or perhaps only once every week; it depends on how often you add or modify files on your system, as well as the capacity of your backup equipment.

Incremental backup

An incremental backup is one that contains only those files that have been added or modified since the last time a more complete backup was made. You may choose to do incremental backups to conserve your backup media. Incremental backups also take less time to complete. This can be important when systems are in high use during the work week and running a full backup would degrade system performance. Full backups can be reserved for the weekend when the system is not in use.

Disk mirroring

Full and incremental backups can take time to restore, and sometimes you just can't afford that downtime. By duplicating your operating system and data on an additional hard drive, you can greatly increase the speed with which you can recover from a server crash.

With disk mirroring, it is usually common for the system to continuously update the duplicate drive with the most current information. In fact, with a type of mirroring called RAID 1, the duplicate drive is written to at the same time as the original, and if the main drive fails, the duplicate can immediately take over. This is called *fault-tolerant* behavior, which is a must if you are running a mission-critical server of some kind.

Network backup

All of the preceding backup strategies can be performed over a network. This is good because you can share a single backup device with many computers on a network. This is much cheaper and more convenient than installing a tape drive or other backup device in every system on your network. If you have many computers, however, your backup device will require a lot of capacity. In such a case, you may consider a mechanical tape loader, writable DVD drive or CD jukebox.

It is even possible to do a form of disk mirroring over the network. For example, a Web server may store a duplicate copy of its data on another server. If the first server crashes, a simple TCP/IP host name change can redirect the Web traffic to the second server. When the original server is rebuilt, it can recover all of its data from the backup server and be back in business.

Getting and installing mirrordir to clone directories

The mirrordir package is a way of doing hard-drive mirroring. Mirrordir is a powerful tool that enables you to make and maintain an exact copy of a hierarchy of directories. You can find its official Web site at http://mirrordir.sourceforge.net. It can be downloaded by selecting the Download RPM button from that site.

After downloading the file, install it in the same way you install any rpm. For example, if you have downloaded the rpm file to /tmp, you can type:

```
# rpm -i /tmp/mirrordir*rpm
```

Cloning a directory with mirrordir

Now that you have mirrordir installed, you can use it to clone a directory. Suppose you have a second hard drive with a partition large enough to hold a copy of your /home partition. Your first step is to create a directory to mount the partition on, and then mount it. Log in as root and type the following:

```
# mkdir -p /mirror/home
# mount /dev/hdb5 /mirror/home
```

In this example, you are mounting the fifth partition (5) of the second hard drive (hdb), hence the device name /dev/hdb5. The b refers to the second disk, and the 5 refers to the partition number. You may use a different drive or partition. If so, adjust the device name accordingly. Assuming the mount command successfully mounted the drive, you are ready to copy your /home partition. Enter the following command:

Restoring Backed Up Files

The restore command is used to retrieve files from a backup tape or other medium that was created by dump. You can use restore to recover an entire file system or to interactively select individual files. It recovers files from the specified media and copies them into the current directory (the one you ran the recover command in), re-creating subdirectories as needed. Much as with the dump command, the first parameter passed to the restore command is a list of single character option codes

6.5. Password protection

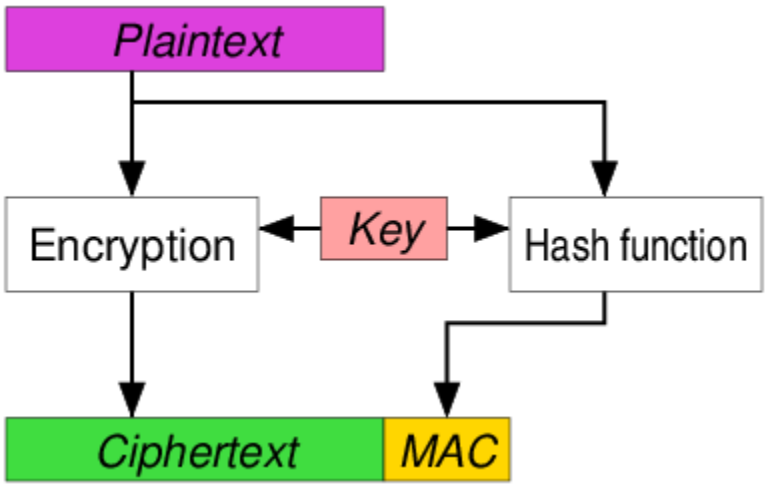
As operating systems, Linux/Unix put the user's privacy and safety above all. While this has resulted in a product that many people swear by, it's also led to certain features that may not be easy to discern at first sight. For instance, the possibility of password protecting directories without using encryption is something that many people don't know about. While encrypting is definitely useful, it also has certain issues associated with it, including:

1. Decreased performance

No matter how strong your system is, opening encrypted folders tends to take up a considerable amount of resources, resulting in a slower and more cumbersome computer that's not always fun to use.

2. Encryption prevents the folder's contents from being searchable/indexed

By its very nature, encryption hides content. Therefore, the files inside the folder you've encrypted will not show up on any search or index attempts, which can be quite annoying when you're looking for something specific.



As you can see, there are a number of issues that pop up whenever encryption is involved. While this kind of protection is necessary for sensitive material, a simple password would be enough to deter strangers from accessing folders of lesser importance. Luckily, Linux/Unix allow for this kind of password protection system as well, as evidenced by the following methods:

1. Changing file permissions

By modifying the permissions of certain files and folders, you can control who gets to access them. This way, they will only be readable by their owner. Anyone who'd want to change these permissions would have to type in a password, or sudo as root, which also requires a password. To change the permissions, just use the *"chmod og-rwx filename"* command on all the files you want to restrict access to.

2. Create a new user

You can also choose to create a new user for all your protected files and directories. Simply employ the *"chown $newuser filename directoryname"* and *"chmod og-rwx filename directoryname"* commands, taking care to replace *"$newuser"* with the new user account name. By using this method your files will be safe even if you forget to log out for any reason.

The methods described above do a good job of protecting your folders from unwarranted intrusions without resorting to encryption. Of course, a password protection system would be nothing without a good password, so take the time to come up with something that's easy to remember for you, but hard to guess for others. Ideally, you should shy away from using information about yourself as a base for any password, since such passwords can rather easily be detected through some basic social engineering techniques. In fact, for most people nowadays it's much easier to simply use a good password generator instead. These can be readily found online and generate passwords that are much harder to guess. Just be sure to write them down somewhere or use a dedicated password manager, lest you risk getting locked out of your own

6.6. File security

While Linux offers all the tools you need to secure you're the computer, if you and careless, someone can (and probably will) harm your system or try to steal your data. The following checklist covers a range of security measures to protect your Linux desktop or server.

- **Add users and passwords**: - creating separate user accounts (each with a good password) in your first line of defense in keeping your data secure. Users are protected from each other, as well as from an

outsider who takes over one user account. Setting up group accounts can extend the concept of ownership to multiple users. See chapter 11 for more on setting up users accounts and " using password protection" later in this chapter

- **Read, write, and execute permissions** :- every item in a Linux file system (including file directories, application and devices) can be restricted by read write and execute permission for that item's owner and group, as well as by all others. In this way, for example, you can let other users run a command or open a file, allowing them to change it. See chapter 4 for information on sitting file and directory permission.
- **Protect root**: - in standard Linux systems, the root user (as well as other administrative user account such as apache) have special abilities to use and change your Linux system. Protect the root account's password and don't use the root account when you don't need to. An open shell or desktop owned by the root user can be target for attack. Running system – config-* windows as a regular user (and then entering the root password as prompted) and running administrative commands using sudo can reduce exposure to attacks on your root account. See chapter 10 for information on handling the root user account.
- **User trusted software**: - while there are no guarantees with any open source software by using an established Linux distribution (such as fedora or RHEL). Software repositories where you get add-on package or updates should likewise be scrutinized. Using valid GPG public keys (which use signatures and encryption) you can ensure that the software you install comes from a vendor. And, of course, always be sure of the source of data files you receive before opening them in a Linux application.
- **Get software updates**: - as vulnerabilities and bugs are discovered in software packages, every Linux distribution (including RHEL and fedora) offers tools for getting and installing those updates, especially if you are using Linux as a server. See chapter 5 for information on using package Kit and yum to get software updates.
- **Use secure applications**:- even with software that is valid and working, some applications offer better protection from attack or invasion than others. For example, if you want to log in to a computer over the internet, the secure shell services (SSH) is considered more secure than rlogin or telnet services. Also, some services that are thought to be insecure if you expose them on the internet (such as samba and NFS), can be used more securely over the internet virtual private network (VPN) tunnels (such as IPsec or CIPE),
- **Use restrictive firewalls**: - a primary job of a firewall is to accept requests for services form a network that you want to allow and turn away the requests that you don't. And desktop system should refuse requests that come in on most ports. A server system should allow requests for a controlled set of ports. This chapter describes how to set up a firewall using iptables.
- **Enable only service you need**:- To offer services in Linux (such as web , file , or mail services0, a daemon process will listen on particular port number. Don't enable services you don't need. In fact, you shouldn't even install server software you don't need see chapter 12 for information on using system services
- **Limit access to services**: - you can restrict access for a service you want to have on to a particular host computer, domain, or network interface. For example, a computer with services such as NFS to computers on the LAN, but not offer those name services to the internet. Service may limit access in their own configuration files r using TCP/IP wrappers
- **Check your system**:-Linux has tons of tools available for checking the security of your system. After you install Linux, you can check access to its port using major watch network traffic using wireshark, you can also add popular security tools such as Nessus, to get a more complete view of your system security.

- **Monitor your system**: - you can log almost every type of activity on your Linux system. System log files, using syslogd and klogd facilities, can be configure to track as much as or as little of your system activity as you choose. The logwatch facility provides an easy to have the potential problems messages forwarded to your administrative email account. Linux loggings are described latter In this chapter. You can get add-on packages such as tripwire and port sentry, to check system for tampering and deal with someone scanning your ports respectively.
- **Use SELinux:-**SELinux is an extraordinarily rich (and complex) facilities for managing the access of nearly every aspect of a Linux system. It addresses the if-i-get-root-access-i-own-your-box shortcoming of Linux and UNIX system for highly secure environments. Red hat system offers a useful, limited set of SELinux policies that are turned on by default in fedora. Despite improvements, however, ,many Linux system administrators still just turn off SELinux because the find it difficult to use

7. Setting up web server

7.1. Introduction to web server

The World Wide Web, as it is known today, began as a project of Tim Berners-Lee at the European Center for Particle Physics (CERN). The original goal was to provide one consistent interface for geographically dispersed researchers and scientists who needed access to information in a variety of formats. From this idea came the concept of using one client (the Web browser) to access data (text, images, sounds, video, and binary files) from several types of servers (HTTP, FTP, SMTP, Gopher, NNTP, WAIS, Finger, and streaming-media servers).

The Web server usually has a simpler job: to accept HTTP (HyperText Transfer Protocol) requests and send a response to the client. However, this job can get much more complex (as the server can also), executing functions such as:

- Performing access control based on file permissions, user name/password pairs, and host name/IP address restrictions.
- Parsing a document (substituting appropriate values for any conditional fields within the document) before sending it to the client.
- Spawning a CGI (Common Gateway Interface) script or custom API (Application Program Interface) program to evaluate the contents of a submitted form, presenting a dynamically created document, or accessing a database.
- Sending a Java applet to the client.
- Logging any successful accesses, failures, and errors.

The Apache Web server

The Apache Web server was originally based on HTTPd, a free server from NCSA (the National Center for Supercomputing Applications). At the time, HTTPd was the most common server on the Internet. Unfortunately, the development of the server wasn't keeping up with the needs of Webmasters, and several security problems had been discovered. Many Webmasters had been independently applying their own features and fixes to the NCSA source code. In early 1995, a group of these developers pooled their efforts and created "a patchy server," initially just a collection of patches to the HTTPd code. Since then, the Apache Group has largely rewritten the code and created a stable, multiplatform Web server daemon.

Apache is also the base for several other Web servers, most of which use Apache's freely available source code and add improved security features such as SSL (Secure Sockets Layer) for encrypted data transfer or advanced authentication modules.

The main features of the Apache Web server include:

- The stability and rapid development cycle associated with a large group of cooperative volunteer programmers.
- Full source code, downloadable at no charge.
- Ease of configuration using plain-text files.
- Access-control based on client host name/IP address or user name/password combinations.
- Support for server-side scripting as well as CGI scripts.
- A custom API that enables external modules (for example, for extended logging capabilities, improved authentication, caching, connection tracking, and so on) to be utilized by the server daemon.

7.2. Starting apache web server

Here's a quick way to get your Apache Web server going. From here, you'll want to customize it to match your needs and your environment (as described in the section that follows).

1. Make sure that Apache is installed by typing the following from a Terminal window:

```
$ rpm -qa | grep httpd
redhat-config-httpd-1.1.0-1.1
httpd-devel-2.0.45-14
httpd-2.0.45-14
httpd-manual-2.0.45-14
```

   The version number you see may be different. You need only the httpd package to get started. I recommend httpd-manual because it has excellent information on the whole Apache setup. The httpd-devel package includes the apxs tool for building and installing extension modules. The redhat-config-httpd package contains a GUI-based Apache Configuration tool.

6. A valid host name is recommended for your Apache server (for example, abc.handsonhistory.com). If you don't have a real fully-qualified domain name, you can edit the /etc/httpd/conf/httpd.conf file and define the ServerName as your computer's IP address. Open the httpd.conf file (as the root user) in any text editor, search for the line containing ServerName new.host.name:80, and uncomment it. It should appear as follows:

```
ServerName new.host.name:80
```

   To make the Web server available to your LAN, you can use your IP address instead of new.host.name (for example, ServerName 10.0.0.1). The :80 represents the port number (which is the default). For a public Web server, get a real DNS host name.

7. Add an administrative e-mail address where someone can contact you in case an error is encountered with your server. In the /etc/httpd/conf/httpd.conf file, the default administrative address appears as follows:

```
ServerAdmin root@localhost
```

   Change root@localhost to the e-mail address of your Apache administrator.

8. Start the httpd server. As root user, type the following:

```
# /etc/init.d/httpd start
```

If all goes well, this message should appear: Starting httpd: [OK]. Now you're ready to go.

10. To have httpd start every time you boot your system, run the command as root user.

```
# chkconfig httpd on
```

12. To make sure that the Web server is working, open Mozilla (or another Web browser) and type the following into the location box and press Enter:

```
http://localhost/
```

14. You should see the Test Page for the Apache Web server, as shown in figure. To access this page from another computer, you will need to enter your Apache server's host name or IP address.

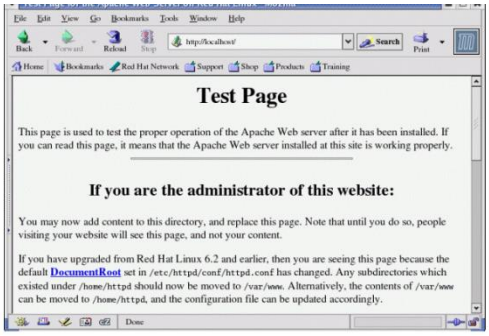


Figure : Appearance of the Test Page indicates that the Apache installation succeeded.

15. The Test Page is actually an error condition, indicating that you haven't added any content to your Web site yet. To get started, you should add an index.html file that contains your own home page content to the /var/www/html directory. Then you can continue to add your own content to that directory structure

7.3. Configuration the apache server

7.4. Monitoring server activities

8. Setting up DHCP and NIS

Setting up a Dynamic Host Configuration Protocol (DHCP) server allows you to centrally manage the addresses and other network information for client computers on your private network. With DHCP configured on your network, a client computer can simply indicate that it wants to use DHCP and the DHCP server can provide its IP address, network mask, DNS server, NetBIOS server, router (gateway), and other information needed to get up and running on the network

8.1. Introduction to DHCP

DHCP is used to control the network configuration of a host through a remote server. DHCP functionality comes installed as a default feature in most of the contemporary operating systems. DHCP is an excellent alternative to the time-consuming manual configuration of network settings on a host or a network device.

DHCP works on a client-server model. Being a protocol, it has it’s own set of messages that are exchanged between client and server

How DHCP Works?

Before learning the process through which DHCP achieves it's goal, we first have to understand the different messages that are used in the process.

1. DHCPDISCOVER

It is a DHCP message that marks the beginning of a DHCP interaction between client and server. This message is sent by a client (host or device connected to a network) that is connected to a local subnet. It's a broadcast message that uses 255.255.255.255 as destination IP address while the source IP address is 0.0.0.0

2. DHCPOFFER

It is DHCP message that is sent in response to DHCPDISCOVER by a DHCP server to DHCP client. This message contains the network configuration settings for the client that sent the DHCPDISCOVER message.

3. DHCPREQUEST

This DHCP message is sent in response to DHCPOFFER indicating that the client has accepted the network configuration sent in DHCPOFFER message from the server.

4. DHCPACK

This message is sent by the DHCP server in response to DHCPREQUEST recieved from the client. This message marks the end of the process that started with DHCPDISCOVER. The DHCPACK message is nothing but an acknowledgement by the DHCP server that authorizes the DHCP client to start using the network configuration it received from the DHCP server earlier.

5. DHCPNAK

This message is the exact opposite to DHCPACK described above. This message is sent by the DHCP server when it is not able to satisfy the DHCPREQUEST message from the client.

6. DHCPDECLINE

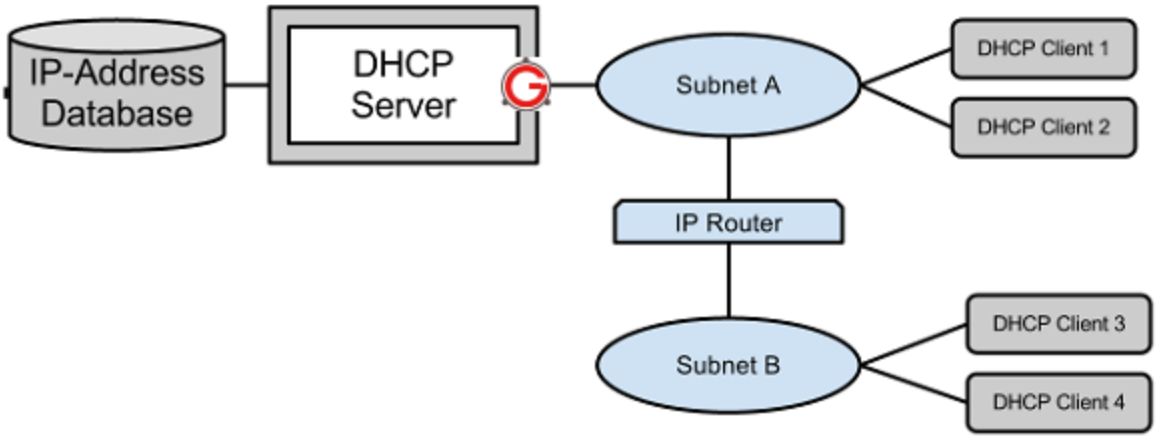
This message is sent from the DHCP client to the server in case the client finds that the IP address assigned by DHCP server is already in use.

7. DHCPINFORM

This message is sent from the DHCP client in case the IP address is statically configured on the client and only other network settings or configurations are desired to be dynamically acquired from DHCP server.

8. DHCPRELEASE

This message is sent by the DHCP client in case it wants to terminate the lease of network address it has be provided by DHCP server.



Now as we know about the various DHCP messages, it's time to go through the the complete DHCP process to give a better Idea of how DHCP works. Note that the steps mentioned below assume that DHCP functionality is enabled by default on the client side.

8.2. Setting up DHCP server

Assuming you have already set up the physical connections between your DHCP server and the client computers on your network (presumably an Ethernet LAN), the minimum tools you need to get the DHCP server working are:

- A firewall that allows DHCP access
- A configured /etc/dhcpd.conf file
- A running dhcpd server daemon (which can be started at boot time)

After the DHCP server is running, it broadcasts its availability as a DHCP server to the LAN. A client simply boots up (with an Ethernet network interface turned on and DHCP identified as its method of getting network addresses), and the information it needs to get up and running on the network is fed to it from the server.

Note The dhcpd.conf file allows an extraordinary amount of flexibility. To see the full set of options and parameters you can set in that file, refer to the dhcp-options and dhcpd.conf man pages (type man dhcp-options).

Opening your firewall for DHCP

The firewall on your DHCP server must be configured to allow access to UDP ports 67 and 68. If you are using iptables (and you did not open ports 67 and 68 during installation), you can add a new rule to iptables and then save the changes permanently. Type the following as root:

```
# iptables -I INPUT -I eth0 -p udp --sport 67:68 --dport 67:68 -j ACCEPT
```

In this example, requests are allowed to and from ports 67 and 68 on the eth0 interface (which is your first Ethernet card). If your DHCP server is also a routing firewall for your network, you want to make sure that you are only offering DHCP services to your LAN and not to the Internet. (You need to figure out if eth0, eth1, or some other card is connected to your LAN.)

If the rule was accepted (type **iptables -L** to make sure), you can save your entire firewall configuration so that the new rule is included permanently. To do that, type the following (as root user):

```
# iptables-save > /etc/sysconfig/iptables
```

This updates your /etc/sysconfig/iptables file so that all the current rules (including the one you just added) are included the next time iptables is restarted.

Configuring the dhcpd.conf file

Suppose you have a single pool of IP addresses that you want to distribute to a set of computers that are all on the same subnetwork. In other words, all the computers are connected to one hub (or a set of daisy-chained hubs). Here is an example of a simple dhcpd.conf file:

```
ddns-update-style interim;
ignore client-updates;

subnet 10.0.0.0 netmask 255.0.0.0 {

     option routers               10.0.0.1;
     option domain-name-servers     10.0.0.1;
     option subnet-mask          255.0.0.0;
     option domain-name          "handsonhistory.com";

     range dynamic-bootp 10.0.0.150 10.0.0.225;
     default-lease-time 21600;
     max-lease-time 43200;

     # Set name server to appear at a fixed address
     host ns {
          next-server ns1.handsonhistory.com;
          hardware ethernet 00:D0:B3:79:B5:35;
          fixed-address 10.0.0.1;
     }
}
```

In this example, this DHCP server is providing IP addresses for client computers on a small LAN. The first two lines tell the DHCP server not to update DNS records for the local domain based on the IP addresses it assigns.

The DHCP server is serving a single LAN: 10.0.0.0 network with a 255.0.0.0 netmask. Other data in this file define what information the DHCP server will hand out to clients on this LAN.

A single server at address 10.0.0.1 is used as the router (or gateway) and DNS server for the LAN. To ensure that this server always gets the fixed address of 10.0.0.1, a host entry is set to the hardware address (00:D0:B3:79:B5:35) for the Ethernet card on the host named ns.

The pool of addresses handed out by this DHCP server is 10.0.0.150 to 10.0.0.225, as set by the range dynamic-bootp line. (Using dynamic-bootp allows bootp and dhcp clients to get addresses.) Along with the IP address that each client is assigned, the client is also given the associated subnet-mask and domain name.

The IP addresses that the DHCP server hands out are leased to each client for a particular time. The default-lease-time (set to 21,600 seconds here, or six hours) is the time assigned if the client doesn't request a particular lease period. The max-lease-time (43,200 seconds here, or 12 hours) is the highest amount of time the server

will assign, if the client requests it. Clients can renew leases, so they don't have to lose the IP address while they are still using it.

Expanding the dhcpd.conf file

As I noted earlier, this is a very simple example that works well for a single network of client computers. Below are some examples of ways that you can expand your dhcpd.conf file.

- If you have multiple ranges of addresses on the same subnetwork, you can add multiple range options to a subnet declaration. Here is an example:

```
 subnet 10.0.0.0 netmask 255.0.0.0 {
   range 10.0.0.10 10.0.0.100;
   range 10.0.0.200 10.0.0.250;
}
```

  This example causes the DHCP server to assign IP addresses between the ranges of 0.0.10 and 0.0.100 and between 0.0.200 and 0.0.250 on network number 10.

- You can set fixed addresses for particular host computers. In particular, you would want to do this for your server computers so that their addresses don't change. One way to do this is based on the Ethernet hardware address of the server's Ethernet card. All information for that computer can be contained in a host definition, such as the following:

```
host pine {
   hardware ethernet 00:04:5A:4F:8E:47;
   fixed-address 10.0.0.254;
}
```

  Here, when the DHCP server encounters the Ethernet address, the fixed-address (10.0.0.254) is assigned to it. Type ifconfig -a on the server computer to see the address of its Ethernet hardware (while the interface is up). Within this host definition, you can add other options as well. For example, you could set the location of different routes (routers option).

- Many of the options let you define the locations of various server types. These options can be set globally or within particular host or subnet definitions. For example:

```
option netbios-name-servers 10.0.0.252;
option time-servers 10.0.0.253;
```

  In these examples, the netbios-name-servers option defines the location of the WINS server (if you are doing Windows file and print server sharing using Samba). The time-servers option sets the location of a time server on your network.

- The DHCP server can be used to provide the information an X Terminal or diskless workstation could use to boot up on the network. The following is an example of a definition you could use to start such a computer on your network:

```
host maple {
     filename "/dwboot/maple.nb";
     hardware ethernet 00:04:5A:4F:8E:47;
     fixed-address 10.0.0.150;
           }
```

In the previous example, the boot file used by the diskless workstation from the DHCP server is located at /dwboot/maple.nb. The hardware Ethernet value identifies the address of the Ethernet card on the client.

Starting the DHCP server

After the /etc/dhcpd.conf file is configured, you can start the DHCP server immediately. As root user from a Terminal window, type the following:

```
# service dhcpd start
```

Your DHCP server should now be available to distribute information to the computers on your LAN. If there are client computers on your LAN waiting on your DHCP server, their network interfaces should now be active.

If everything is working properly, you can have your DHCP server start automatically each time your computer boots by turning on the dhcpd service as follows:

```
# chkconfig dhcpd on
```

8.3. Setting DHCP Client

Configuring a network client to get addresses from your DHCP server is fairly easy. Different types of operating systems, however, have different ways of using DHCP. Here are examples for setting up Windows and Red Hat Linux DHCP clients.

Windows:

1. From most Windows operating systems (Windows 95, 98, 2000, ME, and so on), you open the Network window from the Control Panel (Start → Settings → Control Panel).
2. From the Configuration tab, click the TCP/IP interface associated with your Ethernet card (something like TCP/IP or ipv4→ Com EtherLink III).
3. Click Properties. The Properties window appears.
4. Click the IP Address tab and then select "Obtain an IP Address Automatically".
5. Click OK and reboot the computer so the client can pick up the new IP address.

Red Hat Linux:

1. While you are initially installing Red Hat Linux, click Configure using DHCP on the Network Configuration screen. Your network client should automatically pick up its IP address from your DHCP server when it starts up.
2. To set up DHCP after installation, open the Network Configuration window (neat command).
3. From the Network Configuration window:
    a. Click the Devices tab (on by default).
    b. Click your Ethernet device (probably eth0).
    c. Click Edit.
    d. Click the General tab.
    e. Click "Automatically obtain IP address Settings with" and select dhcp.
    f. Select OK.
    g. Select Apply.
4. Then, from a Terminal window, type:
5. `# /etc/init.d/network restart`

By default, a Red Hat Linux client will not accept all information passed to it from the DHCP server. The way that the Red Hat client handles DHCP server input is based on settings in the */etc/sysconfig/network-scripts/ifup script.* If the client has DHCP turned on, when the system starts up networking, the ifup script runs the *dhcpcd* command as follows:

- If the *dhcpcd* client process is currently running, the *dhcpcd* command sends a signal to it so that it asks the DHCP server to renew the lease on the IP address.
- If no host name is set on the client (or if the host name is set to localhost), the -H option is passed to *dhcpcd* to indicate that it should accept the host name supplied by the DHCP server. (If the host name is already set, the client will not reset the host name from the server.)
- Any new DNS server assignments are accepted by the client. If your DNS servers are already configured in the /etc/resolv.conf file, then you can have the -R option passed to *dhcpcd* to prevent it from updating that file with new DNS server information. (To do this, add PEERDNS=no to the */etc/sysconfig/network* file on the client.)

To change how the *dhcpcd* command works to accept information from the DHCP server, you can pass options to the *dhcpcd* command. Do this by adding arguments to the *DHCPCD*ARGS variable in the */etc/sysconfig/network* configuration file. For example, *DHCPCD*ARGS="-d" causes the ifup script to run *dhcpcd* in debug mode so that messages are sent to the /var/log/messages file. (Type **man *dhcpcd*** to see other *dhcpcd* options.)

8.4. Understanding NIS

The information you share with NIS comes from files that are used with UNIX systems and, therefore, compatible with other UNIX-like systems, such as Red Hat Linux. The group of computers that the master NIS server supports is referred to as an *NIS domain*. This domain is a defined set of host computers that may or may not be the same group of computers contained in a TCP/IP domain.

With NIS, an administrator creates information databases called *maps* from common UNIX (or Linux) system files. The NIS maps are created on the master NIS server and are accessible to other host computers from that server. Just in case the master server is down or inaccessible, one or more slave servers can be defined. The NIS slave servers contain copies of the NIS maps and can provide that information to client computers when the master is unavailable. However, NIS slave servers are not used to create the maps.

When the maps have been shared among the computers in the NIS domain, the main result is that all the computers share a common set of users and network configuration. The following is a list of files that are available for sharing by NIS (not all of them are set up for sharing by default).

- **/etc/group** — Defines the groups to which users on the computer belong.
- **/etc/passwd** — Defines the users who have accounts set up on the computer.
- **/etc/shadow** — Contains encrypted passwords for the users set up in the /etc/passwd file.
- **/etc/gshadow** — Contains encrypted passwords associated with groups contained in the /etc/groups file. (This file is optional and is usually not used.)
- **/etc/passwd.adjunct** — Secures password entries if your system doesn't use shadow passwords. (This file is used with SunOS systems.)
- **/etc/aliases** — Contains user aliases used with e-mail. It allows mail that is sent to a particular user name to be directed to a different user (or set of users). On some systems, this file may be /etc/mailaliases instead.

- **/etc/ethers** — Used by the RARP to map Ethernet addresses into IP numbers. This file is optional. (By default, RARP support is not configured into Red Hat Linux.)
- **/etc/bootparams** — Contains entries needed to start diskless workstations (typically used to boot Sun Microsystems diskless workstations).
- **/etc/hosts** — Contains the names and IP addresses of computers that can be reached on TCP/IP networks. (Often used to contain all the addresses for a private LAN, while Internet addresses would be determined from a DNS server.)
- **/etc/networks** — Used to attach a name to a network. In this way, you can refer to networks by name rather than by number.
- **/etc/printcap** — Contains printer definitions.
- **/etc/protocols** — Identifies numbers that are assigned to different Internet network protocols (such as IP, TCP, UDP, and others).
- **/etc/publickey** — Used on some UNIX systems to contain user names and associated public and private keys for secure networking in NFS and related features.
- **/etc/rpc** — Contains listings of supported Remote Procedure Call (rpc) protocols. These protocols are used with Sun Microsystems UNIX systems to allow requests for network services, such as NIS and others.
- **/etc/services** — Contains listings that identify port number and protocols for supported network services that are used with Internet protocols.
- **/etc/netgroup** — Used to define users (from particular hosts and domains) for permission-checking associated with remote mounts, remote shells, and remote logins.
- **/etc/netid** — Contains information that maps RPC network names to UNIX credentials.

9. Setting up a Database server

MySQL is a popular structured query language (SQL) database server. Like other SQL servers, MySQL provides the means of accessing and managing SQL databases. However, MySQL also provides tools for creating database structures, as well as for adding, modifying, and removing data from those structures. Because MySQL is a relational database, data can be stored and controlled in small, manageable tables. Those tables can be used in combination to create flexible yet complex data structures.

A Swedish company called MySQL AB is responsible for developing MySQL (www.mysql.com). MySQL AB has released MySQL as an open-source product, gaining revenue by offering a variety of MySQL support packages. The company also supports several application programming interfaces (APIs) to help application developers and Web content creators to access MySQL content.

Finding MySQL Packages

You need at least the mysql and mysql-server packages installed to set up MySQL using the procedures described in this chapter. The following MySQL packages come with the Red Hat Linux distribution:

- **mysql** — This software package contains a lot of MySQL client programs (in /usr/bin), several client shared libraries, the default MySQL configuration file (/etc/my.cnf), a few sample configuration files, files to support different languages, and documentation.
- **mysql-server** — This software package contains the MySQL server daemon (mysqld) and the mysqld start-up script (/etc/init.d/mysqld). The package also creates various administrative files and directories needed to set up the MySQL databases.

- **mysql-devel** — This software package contains libraries and header files required for developing MySQL applications.
- **php-mysql** — This software package contains a shared library that allows PHP applications to access MySQL databases. This basically allows you to add PHP scripts to your Web pages that can access your MySQL database.

9.1. Configuration database server

Creating the my.cnf configuration file

Global options that affect how the MySQL server and related client programs run are defined in the /etc/my.cnf file. The default my.cnf file contains only a few settings needed to get a small MySQL configuration going. The following is an example of the /etc/my.cnf file that comes with MySQL:

```
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock

[mysql.server]
user=mysql
basedir=/var/lib

[safe_mysqld]
err-log=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
```

Most of the settings in the default my.cnf file define the locations of files and directories needed by the mysqld server. Each option is associated with a particular group, with each group identified by a name in square brackets. The above options are associated with the mysqld daemon ([mysqld]), the MySQL server ([mysql.server]), and the safe_mysqld script that starts the mysqld daemon ([safe_mysqld]).

The default datadir value indicates that /var/lib/mysql is the directory that stores the mysql databases you create. The socket option identifies /var/lib/mysql/mysql.sock as the socket that is used to create the MySQL communications end-point associated with the mysqld server. The basedir option identifies /var/lib as the base directory in which the mysql software is installed. The user option identifies mysql as the user account that has permission to do administration of the MySQL service.

The err-log and pid-file options tell the safe_mysqld script the locations of the error log (/var/log/mysqld.log) and the file that stores the process ID of the mysqld daemon when it is running (/var/run/mysqld/mysqld.pid). The safe_mysqld script actually starts the mysqld daemon from the mysqld start-up script.

Starting the MySQL Server

For Red Hat Linux, the MySQL server is off by default. To turn it on, however, is fairly simple. The /etc/init.d/mysqld start-up script is delivered with the mysql-server package. To start the server, you can either run the mysqld start-up script manually or set it to start each time your system boots.

To start the MySQL server manually, type the following from a Terminal window as root user:

```
# service mysqld start
```

To have the MySQL server start each time the computer reboots, type the following (as root):

```
# chkconfig mysqld on
```

9.2. Checking the status

You can use the mysqladmin or mysqlshow commands to check that the MySQL server is up and running. Here's an example of how to check information about the MySQL server using the mysqladmin command.

```
# mysqladmin -u root -p version proc
Enter password: ********
mysqladmin  Ver 8.23 Distrib 3.23.57, for pc-linux-gnu on i686
Copyright (C) 2000 MySQL AB & MySQL Finland AB & TCX DataKonsult AB
This software comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to modify and redistribute it under the GPL license

Server version       3.23.57
Protocol version     10
Connection           Localhost via UNIX socket
UNIX socket          /var/lib/mysql/mysql.sock
Uptime:              2 days 10 hours 47 min 35 sec

Threads: 1  Questions: 184  Slow queries: 0  Opens: 1  Flush tables: 3
Open tables: 1 Queries per second avg: 0.001
+----+------+----------+------+-------+------+-----+------------------+
| Id | User | Host     | db   |Command| Time |State| Info             |
+----+------+----------+------+-------+------+-----+------------------+
| 52 | root | localhost| mysql| Query | 0    |     | show processlist |
+----+------+----------+------+-------+------+-----+------------------+
```

Each of the two options to mysqladmin shown here provides useful information. The version information shows the mysqladmin version is 8.23 and the number assigned to this distribution of the mysql server is 3.23.57. The binary package was created for PC versions of Linux/GNU on the i686 processor. The connection to the server is through a UNIX socket (mysql.sock) on the local host. The server has been up for 2 days, 10 hours, 47 minutes, and 35 seconds. Statistics show that there is one thread (connection to the server) currently active. There have been 184 requests to the server.

The proc option shows that one client is currently connected to the server. That client is logged into MySQL as the root user on the localhost. The client that has an Id of 52 (which you could use, as the server's administrator, if you wanted to disconnect the user) is currently querying the MySQL database.

If the server were not running at the moment, the mysqladmin command shown above would result in a failure message:

```
mysqladmin: connect to server at 'localhost' failed.
```

Recommended remedies are to try to restart the server (by typing /etc/init.d/mysqld restart) or to make sure that the socket exists (/var/lib/mysql/mysql.sock).

9.3. Working with database

It includes working as in other data base creating table, deleting data, inserting data, modifies data etc.

10. Setting up DNS Server

10.1. Introduction to DNS

The *Domain Name System* (*DNS*) is essentially a distributed database that translates host names into IP addresses (and IP addresses back to host names). That database also contains information related to each domain, such as how the domain is organized into zones, where to route mail for that domain, and who to contact with questions associated with the domain.

By setting up a DNS server, you become part of a hierarchy of DNS servers that make up the Internet. At the top of this hierarchy is the root server, represented by a dot ("."). Below the root server are the Top Level Domains or TLDs (such as .com, .org, and so on). Domains that individual organizations own and maintain lie below the TLDs. That's where you come in.

As someone who's setting up a DNS server, you're responsible for managing the host names and IP addresses for the computers in the domain (or domains) for which you're responsible. Keeping your DNS information correct means that people can access the services that you want to share, and the Internet as a whole works that much better as a result.

Besides using your DNS server to help people from the Internet find the public servers in your domain, you can also use DNS to provide name and IP address mapping for computers on your private network. The example in the "DNS name server example" section later in this chapter describes how to configure both private and public name and IP address records for a domain.

Caution Setting up a DNS server can be a complex and (these days) potentially dangerous undertaking. A compromised DNS server can cause requests for host addresses to be directed to a cracker's server. The sample DNS server in this section is one created as an example of a DNS server for a home or small office environment. For information on the many different ways to set up a DNS server, open the BIND 9 Administrator Reference manual in a Web browser: /usr/share/doc/bind-9.2.2/arm/Bv9ARM.html.

Understanding DNS

The basic function of a *name server* is to answer queries by providing the information that those queries request. A DNS name server primarily translates domain and host names into IP addresses. Each domain is typically represented by at least two DNS servers.

- **Primary (master) name server:** This name server contains authoritative information about the domains that it serves. In response to queries for information about its domains, this server provides that information marked as being authoritative. The primary is the ultimate source for data about the domain. The secondary name server only carries the same authority in that it has received and loaded a complete set of domain information from the primary.
- **Secondary (slave) name server:** This name server gets all information for the domain from the primary. As is the case for the primary, DNS considers the secondary's information about the domain that it serves authoritative. (You set secondary servers in the NS RR records for the zone in the named.conf file on the primary.)

NS records in the parent zone for a domain list the primary and one or more secondary name servers. This *delegation* of servers defines the servers that have authority for the zone.

Because zone records change as you add, remove, or reconfigure the computers in the zone, you assign expiration times for information about your zone. You set the expiration time in the time to live (TTL) field, in the named.conf file (which I describe later).

Other specialized types of DNS servers are possible as well. Although these types of servers don't have authority for any zones, they can prove useful for special purposes:

- **Caching name server** (This type of server simply caches the information it receives about the locations of hosts and domains. It holds information that it obtains from other authoritative servers and reuses that information until the information expires (as set by the TTL fields).
- **Forwarding name server** (Creating a server that's not authoritative for a zone but that can forward name server requests to other name servers may prove efficient. This server is essentially a caching name server, but is useful in cases where computers lie behind a firewall and in which only one computer can make DNS queries outside that firewall on behalf of all the internal computers.

Understanding authoritative zones

As an administrator of a DNS server, you need to configure several zones. Each zone represents part of the DNS namespace as you view it from your DNS server. Besides the one or more zones representing your domain, you have a zone that identifies your local host and possibly your local, private LAN.

If you configure a server as authoritative for a zone, that server has the last word on resolving addresses for that zone. Your master name server is authoritative for the domain you configure in the "DNS name server example" section but not for domains outside your domain.

Remember that the DNS server that you configure is the ultimate authority for your zone. Other zones don't know how you configure your host names and IP addresses unless you properly set up your DNS server to distribute that information across the Internet.

The definitive data that you set up for your domain exists in the form of resource records. Resource records consist of the data associated with all names below the authoritative point in the tree structure. When the DNS server uses these records to reply to queries, it sets the *authoritative answer* (*AA*) bit in the packet that includes the reply. The AA bit indicates that your name server has the best and most current information available about your domain.

Understanding BIND

In Red Hat Linux (and most other Linux and UNIX systems), you implement DNS services by using the *Berkeley Internet Name Domain* (*BIND*) software. The Internet Software Consortium maintains BIND (at www.isc.org/products/BIND). The particular version of BIND that I describe in this chapter is BIND 9.

The basic components of BIND include the following:

- **DNS server daemon** (/usr/sbin/named)**:** The named daemon listens on a port (port number 53 by default) for DNS service requests and then fulfills those requests based on information in the configuration files that you create. Mostly, named receives requests to resolve the host names in your domain to IP addresses.

  Note The named daemon actually launches from the /etc/init.d/named startup script. You need to set that script to start automatically after your DNS server is ready to go. You can also use the named

startup script to check the status of the named daemon.

- **DNS configuration files** (named.conf and /var/named/*)**:** The /etc/named.conf file is where you add most of the general configuration information that you need to define the DNS services for your domain. Separate files in the /var/named directory contain specific zone information.
- **DNS lookup tools:** You can use several tools to check that your DNS server is resolving host names properly. These include commands such as host, dig, and nslookup (which are part of the bind-utils software package).

To maintain your DNS server correctly, you can also perform the following configuration tasks with your DNS server:

- **Logging** (You can indicate what you want to log and where log files reside.
- **Remote server options** (You can set options for specific DNS servers to perform such tasks as blocking information from a bad server, setting encryption keys to use with a server, or defining transfer methods.

You don't need to give out DNS information to everyone who requests it. You can restrict access to those who request it based on the following:

- **Access control list** (This list can contain those hosts, domains, or IP addresses that you want to group together and apply the same level of access to your DNS server. You create acl records to group those addresses, then indicate what domain information the locations in that acl can or can't access.
- **Listen-on ports** (By default, your name server accepts only name server requests that come to port 53 on your name server. You can add more port numbers if you want your name server to accept name service queries on different ports.
- **Authentication** (To verify the identities of hosts that are requesting services from your DNS server, you can use keys for authentication and authorization. (The key and trusted-keys statements are used for authentication.)

DNS name server example

To get an idea of what you need to set up your DNS server, the following sections step you through an example of a DNS server for a domain called *yourdomain.com*. In the example, you're creating a DNS server for a small office network that includes the following:

- A private, local network that resides behind a firewall.
- A server providing DNS service and acting as a firewall between the LAN and the Internet.

In this office, other computers on the LAN are using the same Internet connection for outgoing communications. So the firewall on the server does network address translation (NAT) to enable the client computers to use the firewall as a router to the Internet. shows the configuration of the example *yourdomain.com* domain.

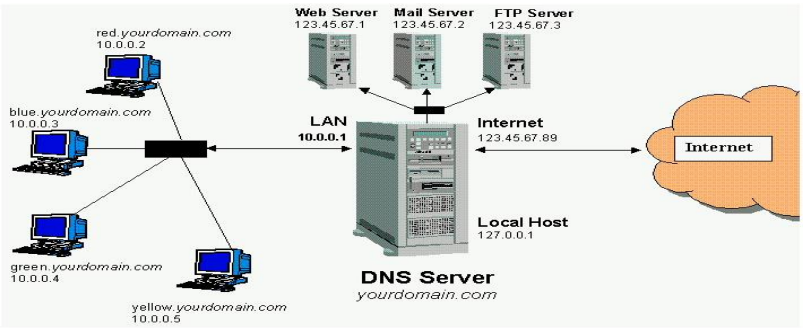


Figure above: The sample *yourdomain.com* DNS server has a combination of public servers and

private client computers.

Figure above illustrates a small office network that's sharing a single Internet connection. The DNS, Web, mail, and FTP servers all have public IP addresses. (These addresses are fictitious, so please don't try to use them.) Behind the DNS server (which is also operating as a firewall) are four client computers that have private IP addresses. (You can reuse these addresses and other private addresses that I describe in)

The job of the DNS server, in this configuration, is to map the names of the public servers (www.*yourdomain.com*, mail.*yourdomain.com*, ftp.*yourdomain.com*, and ns1.*yourdomain.com*) into the static IP addresses that the ISP assigns (123.45.67.1 through 123.45.67.4). The DNS server also provides DNS service from the private addresses on the LAN, so each computer can reach the others on the LAN without needing to store all computer names in their own /etc/hosts file.

A key feature to this example is that it divides the view of this domain between what the outside world can see and what the computers on the private network can see. Using the view feature of BIND, I create an *outside* view that lets queries from the Internet find only public servers (Web, Mail, and FTP) in the domain. Then I create an *inside* view that lets queries from the local LAN find both the public servers and private computers (red, blue, green and yellow) in the domain.

10.2. Setting up DNS and configuration

The DNS server software that comes with the current Red Hat Linux is Berkeley Internet Name Daemon (BIND) version 9. To configure BIND 9, you work with the following components:

- **Configuration file** (/etc/named.conf) (The main DNS server configuration file.
- **Zone directory** (/var/named) (The directory containing files that keep information about Internet root DNS servers (named.ca file) and information about the zones that you create for your DNS server.
- **Daemon process** (/usr/sbin/named) (The daemon process that listens for DNS requests and responds with information that the named.conf file presents.
- **Debugging tools** (named-checkconf, and named-checkzone) (What you use to determine whether you created your DNS configuration correctly.

BIND 9 also includes tools for creating DNSSEC secured zones. By using these tools, you can create and generate keys to provide authentication and secure address resolution. The example illustrated in these sections doesn't include DNSSEC configuration.

The basic steps in creating a DNS server for your example are as follows:

- Identifying your DNS servers
- Creating DNS Configuration files (named.conf and /var/names/*)
- Starting the named daemon
- Monitoring named activities

In the example configuration, you set up a primary master DNS server and a slave DNS server. The primary holds the authoritative records for the domain. The secondary is there to share requests for information about the domain, particularly in case the primary goes down.

Identifying your DNS servers

If you didn't have your DNS servers set up at the time that you purchased your domain name with a registration authority, you might have just "parked" the domain name there until you configured your DNS servers. Whenever you're ready to set up your DNS servers, return to that registration authority and provide the following information about your DNS servers:

- DNS server IP addresses (the static IP addresses of your DNS servers, probably primary and slave)
- DNS server host names (often ns1.*yourdomain.com*, where you replace *yourdomain.com* with your domain name for the primary; the slave host name is ns2.*yourdomain.com*)

You should register both the primary and slave DNS servers. After you update this record, that information typically takes a day or two to propagate throughout the Internet. Once your DNS servers are registered, you also need to tell the registration authority to use those DNS servers as the authority for addresses in your domain. The registration authority probably offers an online form you can fill out to identify your DNS servers.

Creating DNS configuration files (named.conf and /var/names/*)

In configuring a DNS server, you're actually creating definitions that apply to a particular *zone* in the public DNS tree, as well as several local zones that apply to your computer and local network. To create a useful DNS server for your example small-office environment, you have the following zones:

- **Public DNS server zone (** The DNS server is authoritative for the domain that you're serving. This zone serves the names and IP addresses for your public servers. In the example named.conf file. you need to replace the name *yourdomain.com* with the domain that you're creating. These records become accessible to everyone on the Internet.
- **Private DNS server zone (** So each computer on the private network doesn't need to know the IP addresses for other computers on your private network, a zone is added in the example named.conf file to let the DNS server resolve these addresses. The names and IP addresses (which are private) are available only to computers on your LAN.

Note that by creating different views of these zones, different information will be returned to queries, depending on where the queries come from. For example, when someone from the Internet requests the address of the DNS server (ns1.*yourdomain.com*), they will get the address 123.45.67.89. However, when a query for ns1.yourdomain.com comes from inside the LAN, the address 10.0.0.1 is returned. Also, any queries from the Internet for addresses of private computers (red.*yourdomain.com*, blue.*yourdomain.com,* and so on) are rejected.

Editing /etc/named.conf

To begin, you configure the /etc/named.conf file on the primary master DNS server representing your example *yourdomain*.com domain. This example starts from the /etc/named.conf file that comes with the caching-nameserver package in Red Hat Linux. (Make sure that you install the caching-nameserver and bind packages before you continue.) Following are a few tips relating to editing the named.conf file:

- If a statement contains substatements, make sure that you end the last substatement with a semicolon.
- Comments can appear in the same formats that popular programming languages use. These languages include C (begin with /* and end with */), C++ (begin with // and go to the end of the physical line), and shell or Perl styles (begin from a # and go to the end of the physical line).
- A leading exclamation mark (!) negates an element. Putting !123.45.67.89 in a statement causes the IP address 123.45.67.89 not to match the element. (Just make sure that the negation occurs before a positive match or the positive match takes precedence.)

The edited version of the /etc/named.conf file is as follows:

```
options {
     directory "/var/named";
};

acl "mylan" {
    127/8; 10.0.0.0/24;
};

view "inside" {
     match-clients { "mylan"; };
     recursion yes;

     zone "." IN {
     type hint;
     file "named.ca";
     };

     zone "0.0.10.in-addr.arpa" IN {
     type master;
     file "yourlan.db";
     };

     zone "yourdomain.com" {
     type master;
     file "db.yourdomain.com.inside";
     allow-transfer { 10.0.0.2; };
          };
};

view "outside" {
     match-clients { any; };
     recursion no;

     zone "." IN {
     type hint;
     file "named.ca";
     };

     zone "yourdomain.com" {
     type master;
     file "db.yourdomain.com.outside";
     allow-transfer { 123.45.67.2; };
          };
};

include "/etc/rndc.key";
```

The options definition lies at the beginning of the /etc/named.conf file and identifies the /var/named directory as the location where the zone files reside. The acl lines define the mylan access-control list, which consists of host computers on the 10.0.0.0 local private network and the localhost (127/8). (You use this definition in the 0.0.10.in-addr.arpa zone to enable only users on the LAN to perform reverse lookups of names of computers on the LAN.)

The DNS server is broken up into two views: inside and outside. The inside view defines how IP addresses are resolved for requests that come from the private LAN and localhost (as defined in mylan). By having recursion on (recursion yes), the named daemon will allow name server queries from any computer on the LAN. The

outside view defines how queries coming from all other places (presumably, the Internet) are handled. With recursion off (recursion no), only queries from other name servers are honored.

Setting up the zone files

The /var/named directory contains the zone files that the /etc/named.conf file names. For the example, you need to create only three zone files from scratch. You can (and should) leave the named.ca file alone.

The zone files are where most of the real work of the domain name server occurs. In the example, the db.*yourdomain.com*.inside file contains the basic records for the *yourdomain*.com domain, including all private names and addresses. The following is an example of that file:

```
$TTL      86400
@         IN        SOA       yourdomain.com. hostmaster.yourdomain.com. (
                                          2003040701  ; Serial
                                          28800       ; Refresh
                                          14400       ; Retry
                                          3600000     ; Expire
                                          86400 )     ; Minimum
; Name servers
               IN    NS        ns1.yourdomain.com.
               IN    NS        ns2.yourdomain.com.

; Mail server for domain
               IN    MX  10    mail.yourdomain.com.

; Public servers
ns1            IN    A         10.0.0.1
ns2            IN    A         10.0.0.2
mail           IN    A         123.45.67.2
www            IN    A         123.45.67.3
ftp            IN    A         123.45.67.4

; Private clients on the LAN
red            IN    A         10.0.0.2
blue           IN    A         10.0.0.3
green          IN    A         10.0.0.4
yellow         IN    A         10.0.0.5

; EOF
```

The zone file for your "inside" yourdomain.com contains resource records that include information about the zone. Your DNS server uses the TTL (time-to-live) record to tell name servers that store the information that you provide for this domain how long they can keep the information before they need to throw it out and get fresh information. The first value is the default for the entire zone, and the time is in seconds. So a value of 86,400 seconds indicates that a client that is using the information should obtain fresh records about this domain every 24 hours.

The SOA line identifies the start of authority for the domain. The at sign (@) represents the *yourdomain*.com. name. The dot (.) must appear at the end of the domain name. The dot represents the root server of the Internet. If you leave the dot off, your DNS server appends the domain name, so the DNS server will use the name *yourdomain*.com.*yourdomain*.com. The hostmaster.*yourdomain*.com string indicates the e-mail address of the person who is to receive e-mail regarding the domain. (The first dot changes to an @ sign, resulting in hostmaster@*yourdomain*.com). Other information regarding the SOA record is as follows:

- **Serial:** Start with any number here. If the zone records change, increase the serial number to alert other servers that they need to get fresh data about your domain.

  Caution If you forget to increase the serial number after changing zone records, other servers that cache this data never pick up your changes. To help remember, use the date in the serial number. The number 2003082701 would be for August 27, 2003. The 01 represents the first change made on that day.

- **Refresh:** Defines how often the slave DNS server for the zone checks for changes. (Here, 28,800 seconds represents 8 hours.)
- **Retry:** If the slave can't reach the master, it tries again after this retry interval. (Here, 14,400 seconds represents 4 hours.)
- **Expire:** If the slave can't contact the master within the expire time (3,600,000 seconds, or 1,000 hours, here), the slave discards the data.
- **Minimum:** Defines the cache time to live for negative answers. (Here, it's 86,400 seconds, or 24 hours.)

The name server (NS) records define the name servers that represent this zone. In this case, NS records define hosts with the names ns1 and ns2 in *yourdomain.com*. The MX record indicates the location of the mail server for the domain, so that the DNS server can direct e-mail to users in *yourdomain.com*. The rest of the file defines IP addresses for the private clients and public servers that are associated with the domain. Notice that the server at address 10.0.0.2 serves as a client on the LAN and a slave DNS server.

For the "outside" *yourdomain.com* zone we made a db.*yourdomain.com*.outside file using the same information from the "inside" file, with the following exceptions:

- Removed all references to private clients on the LAN. That way, someone poking around from the Internet can't get information about your private computers.
- Changed the addresses of the primary and slave DNS servers (ns1 and ns2) to 123.45.67.1 and 123.45.67.2, respectively. In that way, only public addresses for name servers are seen by the public.

The other new file in the example is the yourlan.db file, which contains the information necessary to perform reverse IP lookups for the computers on your LAN. Here's an example:

```
$TTL      86400
@         IN        SOA     0.0.0.10.in-addr.arpa. hostmaster.yourdomain.com. (
                                        2002052701  ; Serial
                                        28800       ; Refresh
                                        14400       ; Retry
                                        3600000     ; Expire
                                        86400 )     ; Minimum
          IN        NS      0.0.0.10.in-addr.arpa.
1         IN        PTR     yourdomain.com.
2         IN        PTR     red.yourdomain.com.
3         IN        PTR     blue.yourdomain.com.
4         IN        PTR     green.yourdomain.com.
5         IN        PTR     yellow.yourdomain.com.

; EOF
```

The SOA record identifies 0.0.0.10.in-addr.arpa. as the start of authority for the zone. The NS line defines 0.0.0.10.in-addr.arpa. as the name server for the zone. Other records are pointers to host names that reverse-map

on the 10.0.0. network. The records represent the address for the DNS server (*yourdomain.com*) and each of the clients on the LAN (red, blue, green, and yellow).

After you finish creating your own zone files, you can use the named-checkzone command to make sure that each zone file is correctly formed. Here is how you'd run the named-checkzone command (as root user) to check the two *yourdomain.com* zone files:

```
# named-checkzone yourdomain.com /var/named/db.yourdomain.com.inside
zone yourdomain.com/IN: loaded serial 2003082701
OK
# named-checkzone yourdomain.com /var/named/db.yourdomain.com.outside
zone yourdomain.com/IN: loaded serial 2003082701
OK
```

The output indicates that both files are okay and that named-checkzone command is able to load the new serial numbers. In this case, the serial number represents the first serial number (01) on August 27, 2003 (2003082701).

Starting the named (DNS) daemon

To start the named daemon and see whether it's working, type the following (as root user):

```
# /etc/init.d/named start
```

If the named daemon starts successfully, clients of your DNS server should start getting information about your domain. To set the named daemon to start each time that the system boots up, type the following:

```
# chkconfig named on
```

Remember that, whenever you make changes to the named.conf or any of the zone files, you must increase the serial number for anyone checking your domain records to pick up those changes. After that, you should restart the named service too (as root user) as follows:

```
# /etc/init.d/named restart
```

If you see the Starting named message, your DNS server is probably up and running. If you want to make sure that your server is correctly resolving addresses, the following section describes some tools that you can use to check your DNS name server.

Checking that DNS is working

The best way to see if your DNS server is working correctly is to watch it in action. Here are a few commands you can use to check out your DNS server. The first example uses the host command to get the IP address for the host computer named blue in the local domain:

```
# host blue
blue.yourdomain.com has address 10.0.0.3
```

Instead of using the simple host name to get the computer's IP address, you can enter an IP address (instead of the name) or a fully qualified host name. In the following example, the dig command is used with a domain name to get information about the addresses for a domain:

```
# dig yourdomain.com
; <<>> DiG 9.2.1 <<>> yourdomain.com
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 43728
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 0

;; QUESTION SECTION:
;yourdomain.com.                IN     A

;; AUTHORITY SECTION:
yourdomain.com.          604800     IN     NS     ns1.yourdomain.com.
yourdomain.com.          604800     IN     NS     ns2.yourdomain.com.

;; Query time: 24 msec
;; SERVER: 10.0.0.1#53(10.0.0.1)
;; WHEN: Mon Apr   6  02:12:32 2003
;; MSG SIZE  rcvd: 129
```

Sections in the output from dig include a question section and an authority section. The results show name server assignments and addresses associated with the domain you're querying about. The nslookup command is another tool you can use to look up domain information. In the following example, nslookup looks up the server that is resolving ftp.*yourdomain.com*:

```
# nslookup -sil ftp.yourdomain.com

Server:        123.45.67.1
Address:       123.45.67.1#53

Name:   ftp.yourdomain.com
Address: 123.45.67.3
```

The output from the nslookup command includes the name of the computer fulfilling the request and its IP address, along with the name and address of the computer you're asking for. (The -sil prevents a message that nslookup might soon be removed from Red Hat Linux.) Try nslookup with an IP address (such as 10.0.0.1) to make sure reverse lookup works.

To check the status of the named server that is running on your local system, use the same script that starts named. Type the following to check the status of your DNS server daemon:

```
# /etc/init.d/named status
number of zones: 5
debug level: 0
xfers running: 0
xfers deferred: 0
soa queries in progress: 0
query logging is OFF
server is up and running
```

If you can't reach the computers that your DNS server is serving by name or IP a

11. ISP Simulation

11.1. Integration of servers; DNS, Web, Email etc

Objective: To have a router acting like an ISP for upcoming Lab Setup

I. ISP CONFIGURATION STAGE

<Configs>

First we will assign a hostname to our ISP Router

hostname ISP

Then we will create a loopback addresses to simulate the Public IP Address of the site were going to access

interface Loopback 1
description Global DNS
ip address 4.2.2.2 255.255.255.255
interface Loopback 2
description Google DNS
ip address 8.8.8.8 255.255.255.255
interface Loopback 3
description google.com
ip address 8.8.8.1 255.255.255.255

interface Loopback 4 description yahoo.com
ip address 98.139.183.24 255.255.255.255

We will then Assign IP Addresses to our interfaces

interface FastEthernet0/0
*description ***Link to R1****
ip address 77.77.77.1 255.255.255.252
duplex auto
speed auto
interface FastEthernet0/1
ip address 2.2.2.1 255.255.255.252
duplex auto
speed auto

Then we will route it via EIGRP

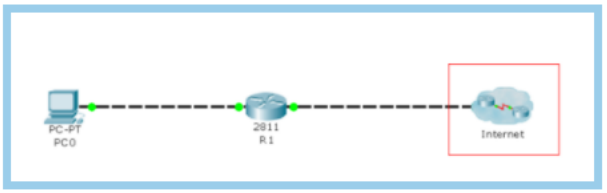
router eigrp 1
network 77.77.77.0 0.0.0.3
network 2.2.2.0 0.0.0.3
network 8.8.8.8 0.0.0.0
network 8.8.8.1 0.0.0.0
network 98.139.183.24 0.0.0.0

Now were going to create an access-list to prevent entry of Private Addresses to our ISP Router

ip access-list standard BLOCKED
deny 10.0.0.0 0.255.255.255
deny 172.0.0.0 0.255.255.255
deny 192.0.0.0 0.255.255.255
permit any

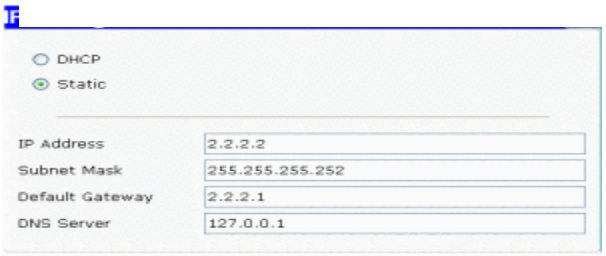
Put this command under interface fastehernet 0/0 to filter those addresses

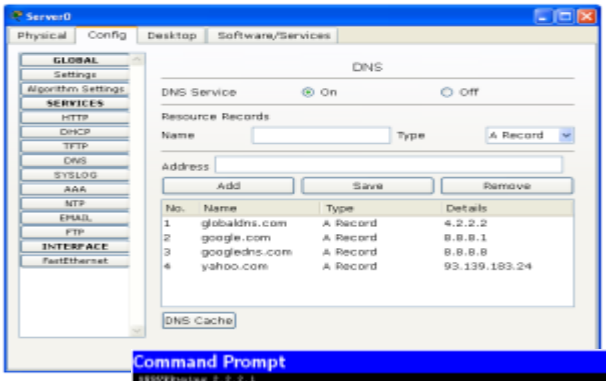
ip access-group BLOCKED in

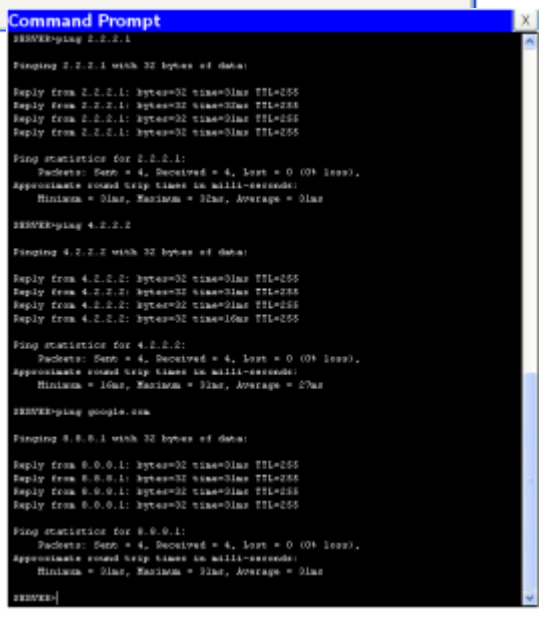


PACKET TRACER ISP

Overview: Configure a Router to act as an ISP







***II.* Configuring a DNS Server for the ISP**

Assign this IP Address to your Server.

IP