

INTRODUCTION TO RED HAT LINUX

Version 1.1

HIEU Q TRAN, PH.D.
Chemistry Computer Center
Department of Chemistry
UW-Madison

August 2004

Contents

1 Overview	3
2 Basic Concepts	3
2.1 Logging In	3
2.2 Logging Out	3
2.3 Graphical Interface	3
2.4 The Shell Environment	4
2.5 Getting Help	4
2.5.1 Manual Pages	4
2.5.2 Info Pages	6
2.5.3 Red Hat Documentation	6
2.5.4 Help Files	6
2.5.5 Internet Resources	6
2.6 Exercises	6
3 Working with Directories	6
3.1 The Root Directory - The <i>cd /</i> Command	6
3.2 Home Sweet Home - The <i>cd</i> Command	7
3.3 Where Am I - The <i>pwd</i> Command	7
3.4 Making New Directory - The <i>mkdir</i> Command	7
3.5 Listing contents of a directory - The <i>ls</i> Command	7
3.6 Going Up - The <i>cd ..</i> Command	7
3.7 Going Down - The <i>cd downdir</i> Command	7
3.8 Going Anywhere - The <i>cd anydir</i> Command	8
3.9 Going Nowhere - The <i>cd .</i> Command	8
3.10 Remove a Directory - The <i>rmdir</i> Command	8
3.11 Exercises	8

4	Working with Files	8
4.1	Manipulating Files with the <i>cat</i> Command	8
4.1.1	Redirecting Standard Output - The <i>></i> Operator	9
4.1.2	Redirecting Standard Input - The <i><</i> Operator	9
4.1.3	Appending Standard Output - The <i>>></i> Operator	10
4.1.4	Pipes - The <i> </i> Operator	10
4.2	Reading Text Files	11
4.2.1	The <i>cat</i> <i><filename></i> Command	11
4.2.2	The <i>more</i> <i><filename></i> Command	11
4.2.3	The <i>less</i> <i><filename></i> Command	11
4.2.4	The <i>head</i> <i><filename></i> Command	11
4.2.5	The <i>tail</i> <i><filename></i> Command	11
4.2.6	The <i>grep</i> Command	12
4.3	Manipulating Files at the Shell Prompt	12
4.3.1	Creating Files	12
4.3.2	Deleting Files - The <i>rm -i</i> Command	12
4.3.3	Deleting Directories - The <i>rmdir</i> Command	13
4.3.4	Copying Files - The <i>cp</i> Command	13
4.3.5	Moving or Renaming Files - The <i>mv</i> Command	13
4.3.6	Printing Files - The <i>lpr</i> Command	13
5	Using Text Editors	14
5.1	The <i>Vi</i> Editor	14
5.1.1	Invoking <i>vi</i>	14
5.1.2	Editing Mode	14
5.1.3	Escape Mode	15
5.2	The <i>NEdit</i> Editor	16
5.3	The <i>Pico</i> Editor	16
5.4	The <i>Emacs</i> Editor	16
5.5	Exercises	16
6	Getting Online	17
6.1	Logging on to a Remote System - The <i>ssh</i> Command	17
6.2	Transferring Files from/to a Remote System - The <i>ftp</i> Command	18
6.2.1	Ftp with your Username and Password	18
6.2.2	Ftp anywhere as <i>anonymous</i>	20
6.3	Checking Electronic Mail - The <i>pine</i> Email Program	20
6.3.1	Starting and Exiting <i>pine</i>	20
6.3.2	Composing and Sending Email	20
6.3.3	Sending Attachments with Email	20
6.3.4	Saving Messages & Attachments	21
6.4	Web Browsing - The Mozilla Web Browser	22
6.5	Exercises	22
7	The GNU Object Model Environment (GNOME) Desktop	22
7.1	What is GNOME?	22
7.2	Exercises	22

1	OVERVIEW	3
8	The K Desktop Environment (KDE)	23
8.1	What is KDE?	23
8.2	Exercises	23
9	Document History	23
9.1	Version History	23
9.2	Updates	23
10	Contact Information	23
11	References	23

1 Overview

This tutorial presents some fundamentals of the Red Hat (RH) Linux operating system. The tutorial targets beginning users who have no previous experience with Linux/UNIX; it is not meant to be comprehensive, nor is it a replacement of on-line manuals and help files.

The tutorial takes a practical and hands-on approach in presenting new material to the users. The users will start off with learning basic Linux concepts, learning to create text files using several popular text editors such as *Vi* or *NEdit*, then quickly move on to learning their way around the RH system. The tutorial finishes off with a brief introduction of the two most popular Graphical User Interfaces (GUIs) of RH: *K Desktop Environment (KDE)* and *GNU Network Object Model Environment (GNOME)*. The tutorial can be used as a workshop's companion as well as a self-study guide. The users are strongly recommended to finish all the exercises provided in this tutorial.

2 Basic Concepts

2.1 Logging In

The Red Hat operating system is available at several research computers at the Chemistry Department. The users must use their username-password combination (assigned by the Chemistry Computer Center) to log into their account.

A typical graphical login screen is shown in Figure 1 (page 4).

2.2 Logging Out

To log out your graphical desktop session, select MainMenu ⇒ LogOut.

When the confirmation dialog appears as shown in Figure 2 (page 4), select the **Logout** option and click the **Yes** button. To save the configuration of your desktop, as well as any programs which are running, check the **Save current setup** option.

2.3 Graphical Interface

Once you start the X Window System, you will find a graphical interface known as a desktop similar to Figure 3 (page 5).

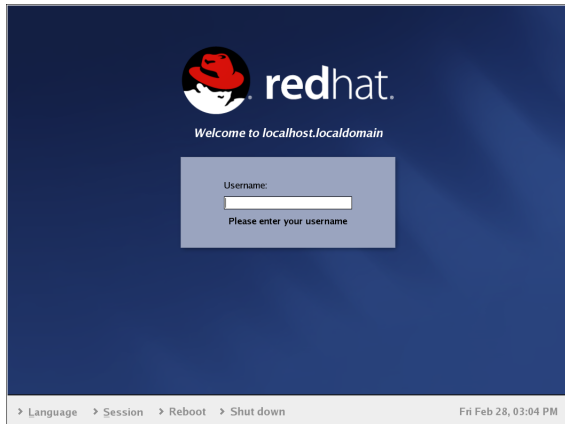


Figure 1: Graphical log-in screen

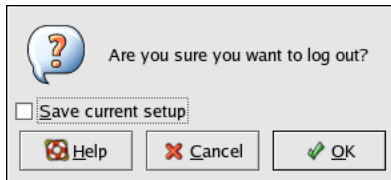


Figure 2: Graphic log-out screen

2.4 The Shell Environment

The desktop offers access to a shell prompt, an application that allows you to type commands instead of using a graphical interface. More details on Shell Prompt Basics will be provided later on.

You can open a shell prompt by selecting `MainMenu` ⇒ `SystemTools` ⇒ `Terminal`. You can also start a shell prompt by *right-clicking* on the desktop and choosing *New Terminal* from the menu.

To exit a shell prompt, click the `X` button on the upper right corner of the shell prompt window, or type `exit` at the prompt, or press `[Ctrl]-[D]` at the prompt.

2.5 Getting Help

There are several ways to get more information on your Red Hat Linux system. Along with the Red Hat Linux documentation there are *manual pages* (aka *man pages*), *INFO pages*, help files, and Internet resources.

2.5.1 Manual Pages

Applications, utilities, and shell prompt commands usually have corresponding manual pages that show the reader available options and values of file or executable.

Information on syntax or use of applications and shell prompt commands is provided in their corresponding manual pages (also called 'man pages'). Manual Pages can be accessed via shell prompt by typing the command `man` and the name of the executable. For example, to access the man page for the `cal` command (i.e. to print calendar), type the following:

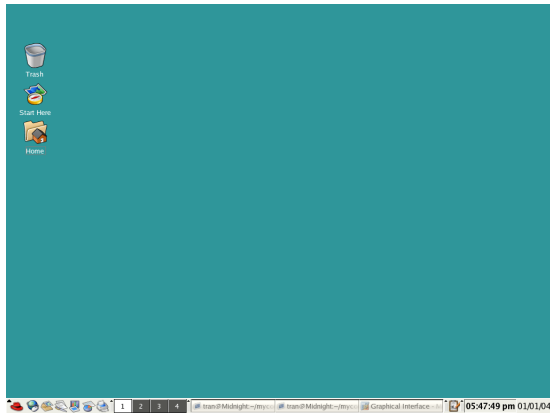


Figure 3: The graphical desktop

man cal

The output is as follows (incomplete):

```

CAL(1)                                BSD General Commands Manual          CAL(1)
NAME
cal - displays a calendar

SYNOPSIS
cal [-smjy13] [[month] year]

DESCRIPTION
Cal displays a simple calendar. If arguments are not specified, the cur-
rent month is displayed. The options are as follows:

-1 Display single month output. (This is the default.)

-3 Display prev/current/next month output.

-s Display Sunday as the first day of the week. (This is the
default.)

-m Display Monday as the first day of the week.

-j Display Julian dates (days one-based, numbered from January 1).

```

The NAME field shows the executable's name and a brief explanation of what function the executable performs. The SYNOPSIS field shows the common usage of the executable. The DESCRIPTION field shows available options and values associated with a file or executable.

2.5.2 Info Pages

Similar to *man*, information on a certain command can be obtained using *info*.
info cal

2.5.3 Red Hat Documentation

Red Hat provides in-depth information on the installation and use of its application in the documentation packages. You can access them at any time by clicking `MainMenu` ⇒ `Documentation`.

2.5.4 Help Files

Help files are included in the main menubar of graphical applications. You can access them at any time by clicking `MainMenu` ⇒ `Help`.

2.5.5 Internet Resources

Along with other web sites that provide information on the RH systems, HTML and PDF versions of the Red Hat Linux manuals are available on the Red Hat Linux Documentation CD and online at <http://www.redhat.com/docs/>.

2.6 Exercises

Read the information provided from the command **man cal** above, then perform the following tasks:

1. Print the calendar of the current year.
2. Print the calendar of the current month.
3. Print the month and year of your birthdate to find out what day of the week you were born.

3 Working with Directories

Linux files are organized into a hierarchical directory structure. The components of a directory are separated by slashes. The term *pathname* is often used to refer to this slash-separated list. This section will show the users how to move their files around the RH system and organize their work environment with directories.

3.1 The Root Directory - The *cd* / Command

The Linux file system starts with a directory called ROOT. From this directory, other directories and files are expanded like the branches of a tree. The only exception is going from anywhere in the file system to ROOT, one would GO UP (not go down as for a real tree!) To get to the top of the file system, use the command:

```
cd /
```

3.2 Home Sweet Home - The *cd* Command

Each user has a home directory which has the same name as the *username*. To get back to your home directory, use the command:

```
cd
```

Later you will see that a **cd** command followed by a directory name will have a different meaning.

3.3 Where Am I - The *pwd* Command

To see where you are currently in the file system, type:

```
pwd
```

3.4 Making New Directory - The *mkdir* Command

To create a new directory, type:

```
mkdir newdir
```

Where *newdir* is the name of the directory you want to create. Each directory (or file) in the same directory must have a unique name.

3.5 Listing contents of a directory - The *ls* Command

1. **ls** list only the names of the files and directories existing in the current directory
2. **ls -l** list the name, size, and other information (except hidden files)
3. **ls -la** list the name, size, and other information of all files, including hidden files

3.6 Going Up - The *cd ..* Command

To go up (closer to ROOT) one directory, use the command:

```
cd ..
```

3.7 Going Down - The *cd downdir* Command

To go down (farther away from ROOT) one directory, use the command:

```
cd downdir
```

Of course, *downdir* must exist in the current directory, or an error will occur.

3.8 Going Anywhere - The *cd anydir* Command

To go anywhere in the file system, use the command:

```
cd anydir
```

Of course, *anydir* must exist (either in the current directory, or anywhere in the file system), or an error will occur. If it is not in the current directory, then the whole *pathname* must be provided.

3.9 Going Nowhere - The *cd .* Command

To go nowhere in the file system, use the command:

```
cd .
```

This command will lead you to nowhere but the current directory. The command is not useless, however, since sometimes a programmer wants to specifically instruct the system to look for a file in the current directory, not in the directory that has the executable currently running. This is particularly important when applications are running on a network. If this does not make any sense to you, don't worry about it.

3.10 Remove a Directory - The *rmdir* Command

To remove (delete) any directory (except for the one you are currently in), use the command:

```
rmdir pathname
```

This command will only remove *empty* directories. More on this later.

3.11 Exercises

1. Go to the ROOT directory and list its contents.
2. Go to a directory listed in the ROOT directory and list its contents in three different ways.
3. Go to your home directory, then create a new directory called *rhtutor*.
4. Go to the directory created above (*rhtutor*), create four new directories called *one*, *two*, *three*, and *not-wanted*.
5. Go back to your home directory, then delete the directory *notwanted* created in the exercise above.

4 Working with Files

4.1 Manipulating Files with the *cat* Command

The *cat* command (short for *concatenate*, which means to combine files) is a utility which shows you contents of files, information about your system, or help you to gather (combine) files together. In this section, we will take a look at several ways of using the *cat* command with and without *redirection*.

4.1.1 Redirecting Standard Output - The > Operator

First off, let us consider the concept of *redirecting*. Any computer system consists of at least two main components: a device for receiving the user's input and one for displaying computing results. The input device is usually a keyboard and the output - the computer screen. The user would provide instructions to the computer via a keyboard, for the computer to perform the requested tasks and display the results on the computer monitor (or sometimes, the printer). The keyboard is then considered *standard input* and the monitor *standard output*. *Redirection* means causing the shell to change what it considers to be standard input and output. In this section, we will look at redirecting standard output.

To redirect standard output, use the > symbol. Placing > after the **cat** command directs its output to the filename following the symbol.

To illustrate this point, let us type the commands shown below, make observations, and take good notes of the results. (The \$ sign indicates the Shell Prompt. Hit at the end of each command line as shown. Type [Ctrl]-[D] to stop.)

NO REDIRECTING OUTPUT:

```
$ cat 
Hello. 
What is your name? 
Why don't you answer me? 
Stop repeating what I said! 
[Ctrl]-[D]
```

REDIRECTING OUTPUT:

```
$ cat > oops.txt 
Hello. 
What is your name? 
Why don't you answer me? 
Stop ignoring me! 
[Ctrl]-[D]
```

THE SOLUTION:

To see what happened, type these commands:

```
$ ls 
$ cat oops.txt 
```

4.1.2 Redirecting Standard Input - The < Operator

In the previous section, if you say the input was *redirected to the file oop.txt* in the second case, then you are correct. In other words, the direction of data flow is *keyboard* → *a file* (named *oops.txt* in this case).

A file can also be the *standard input*. Try this:

REDIRECTING INPUT:

```
$ cat < oops.txt 
```

In this case, the input comes from the file (*oops.txt*), the standard output is our monitor. The keyboard is bypassed.

What about skipping both the keyboard and the monitor? *You bet!* Try this:

```
$ cat < oops.txt > oopsout.txt
```

What this does is to read the input from the file *oops.txt* and instead of being displayed on the monitor, the contents of *oops.txt* is written on the new file *oopsout.txt*.

4.1.3 Appending Standard Output - The \gg Operator

As we learned earlier, the **ls** command lists the contents of a directory *on the monitor*. What do you think this next command will do? Try it.

```
$ ls > oops.txt 
```

Look at the contents of the file *oops.txt*. (*HINT: Use cat.*)

Now, try this:

```
$ ls -la > oops.txt 
```

And look at the contents of the file *oops.txt* again. I can tell you that the contents of the file has been overwritten by the **ls -la** command.

And this. Pay close attention:

```
$ ls > oops2.txt 
```

Look at the contents of the file *oops2.txt*.

Finally:

```
$ ls -la  $\gg$  oops2.txt 
```

Then look at the contents of the file *oops2.txt* again. (*NOTE: This time, we use the redirecting symbol \gg .*)

Compare two files *oops.txt* and *oops2.txt*. What do you think \gg did?

4.1.4 Pipes - The | Operator

In Linux, pipes connect the standard output of one command to the standard input of another command.

For instance, you want to print the contents of a directory:

```
$ ls -la | lpr
```

The output of the command **ls -la** is being fed to the command **lpr**. As a result, the contents of the current directory is printed.

4.2 Reading Text Files

4.2.1 The *cat* <filename> Command

From the previous section, we have learned how to use the **cat** command with redirection. Without the redirecting symbols, **cat** can be used to look at the contents of text files at the Shell Prompt without explicitly opening the files. For instance, to look at the contents of *oops.txt*:

```
$ cat oops.txt
```

4.2.2 The *more* <filename> Command

Because *oops.txt* is a small file, the whole contents fits in the monitor screen. For longer files, using **cat** will cause the whole contents to swiftly scroll down the monitor screen. (A fly may be able to read it that way, but we humans simply cannot!) Consider this:

```
$ cat mywishlist.txt (can't read!)
```

As expected, the wish list is too long, and simply flying down the screen. To be able to read it, use the following command:

```
$ more mywishlist.txt (much better!)
```

So, the take-home message is, "*Because the wish list is too long, you need **more!***"

What an irony! That may be the reason why someone (later(?)) invented the **less** command, as shown in the next section.

4.2.3 The *less* <filename> Command

Less is similar to **more**. However, **less** allows both backward and forward movement. See **man** page for more details. For instance:

```
$ less bigfile.txt
```

4.2.4 The *head* <filename> Command

The *head* command displays the first ten lines of a file. It is useful to quickly check the title and abstract of an on-line article. You can also look at a number of lines different than ten, by indicating the number of lines to read (as shown in the second example.)

```
$ head oops.txt
```

```
$ head -1 oops.txt (shows first line)
```

4.2.5 The *tail* <filename> Command

The **tail** command is similar to **head**, except it displays the *last* ten lines of a file.

```
$ tail oops.txt
```

```
$ tail -1 oops.txt (shows last line)
```

4.2.6 The *grep* Command

The **grep** (global regular expression print) command searches for a specified string of characters in one or more files. Suppose you want to go out to lunch one day, and decide to call ahead of time to make a reservation. Here is how:

First, create a file to store the names and phone numbers of several restaurants in town.

```
$ cat >> MadDiners.txt 
Hong Kong Cafe 259-1669 
Bahn Thai 233-3900 
Ginza of Tokyo 286-1878 
[Ctrl]-[D]
```

And suppose you want to call 'Hong Kong Cafe':

```
$ grep 'Hong Kong' MadDiners.txt
```

Try and see it for yourself!

(The quotation marks are used to combine two separate words in the search string together. Without them, the command still works, but not 'as beautifully'.)

4.3 Manipulating Files at the Shell Prompt

Now that you have learned how to read and search files, it's time to learn how to create and delete them.

4.3.1 Creating Files

You have, in fact, learned to create files from the Shell prompt (using redirected **cat** command), as described above. Another way to create a file is to use a text editor. We will take a look at several popular text editors later. One interesting command used in Linux to create files is **touch <filename>**. Try this:

```
$touch empty.txt
$ls -la empty.txt
$cat empty.txt
```

You can see that the size of *empty.txt* is zero and it is empty.

4.3.2 Deleting Files - The *rm -i* Command

Now, let's delete that useless *empty.txt* file.

```
$rm -i empty.txt (Say 'Yes' to remove it)
$ls -la empty.txt (gone!)
```

In the **rm -i** command, the *flag -i* stands for *interactive*. This gives you a chance to change your mind, in case you accidentally type the command without meaning to. It is wise to **always** use this flag when

removing your files!

4.3.3 Deleting Directories - The *rmdir* Command

A directory can be deleted using the **rmdir** command. However, if the directory is not empty, some flags must be used. Read the *man* pages for details.

```
$rmdir emptydir  
$rm -rf NOTEMPTYdir (BE VERY CAREFUL!)
```

*The **rm -rf** command will delete EVERYTHING in the NOTEMPTYdir directory, including the directory itself. Therefore, do not use this command, unless you know exactly what you are doing.*

4.3.4 Copying Files - The *cp* Command

This command is very straightforward. For instance,
\$ cp original.txt duplicate.txt

will make a copy of *original.txt* and name it *duplicate.txt*. (Check for the existence of *duplicate.txt* before doing this, or you will overwrite the current file!)

4.3.5 Moving or Renaming Files - The *mv* Command

The **mv** *source target* can be used in two different ways. For example,

```
$mv f1.txt f2.txt  
(Change the filename from f1.txt to f2.txt. No directory change is involved. Therefore, no moving is involved.)
```

```
$mv f1.txt ../  
(The target is a directory, this command moves the files f1.txt to the directory right above the current one without changing filename.)
```

```
$mv f1.txt ../f2.txt  
(This command does both actions: moving the file and changing its name.)
```

4.3.6 Printing Files - The *lpr* Command

To print the file *myfile.txt* to a default printer:

```
$ lpr myfile.txt
```

To print the file *myfile.txt* to the printer named *Midnight*:

```
$ lpr -PMidnight myfile.txt
```

To print the files *myfile.txt* and *yourfile.txt* to the printer named *Midnight*:

```
$ lpr -PMidnight myfile.txt yourfile.txt
```

To see what printing jobs are currently in the printer queue:

```
$ lpq
```

Midnight is ready and printing

Rank	Owner	Job	File(s)	Total Size
active	tran	17	myfile.txt	1024 bytes

This indicates that the user **tran** has a job (number 17) that is currently being printed at the printer named *Midnight*.

To remove the job from the printing queue and stop it from printing:

```
$ lprm 17
```

```
$ lpq
```

```
Midnight is ready
no entries
```

(It's gone!)

5 Using Text Editors

5.1 The Vi Editor

The **vi** editor is a powerful, interactive, and visually oriented editor. Because **vi** is so large and powerful, this section will only describe some of its features. More information can be found in on-line or published documentation.

5.1.1 Invoking *vi*

Command	Meaning
vi <i>file</i>	Invoke vi editor on <i>file</i>
view <i>file</i>	Invoke vi editor on <i>file</i> in read-only mode
vi -r <i>file</i>	Recover <i>file</i> and recent edits after system crash

HANDS-ON:

```
$ vi learnvi
```

This command will invoke **vi**. Type **i** to start Editing Mode.

5.1.2 Editing Mode

You must be in *editing mode* to be able to *edit* text. One way to start editing mode:

- Start **vi**
- Type **i**

Command	Meaning
←, ↓, ↑, →	Move left, down, up, right
<i>Backspace</i>	Delete letters one by one

The bizarre thing about editing with *vi* is that a lot of editing (copy, paste, etc.) commands are not carried out in editing mode. Instead, they are done *outside editing mode*. In order to exit *editing mode*, hit the `escape` key.

An official document would divide command types into several modes, such as *editing*, *movement*, *ex*, etc. The trouble is this kind of classification is rather confusing due to the reason mentioned in the previous paragraph. To make the tutorial more efficient, we use a very simple scheme: you are either in *editing mode* or not. We call the latter *escape mode* (invoked by hitting `escape`).

5.1.3 Escape Mode

You are in *Escape Mode* is when you are *not* in editing mode.

Hit `escape` to invoke it.

While in *Escape Mode*, you can do quite a few things: save, save as..., cut, copy, paste, undo, move around the screen, scroll down, scroll up, to name a few.

MOVEMENT COMMANDS:

Command	Meaning
h, j, k, l	Move left, down, up, right (or use arrows ←, ↓, ↑, →)
0, \$	First, last position of current line
H	Top line of screen
L	Last line of screen
nH	n (number) of lines after top line
nL	n (number) of lines before last line
<i>Ctrl - F</i> , <i>Ctrl - B</i>	Scroll forward, backward one screen
<i>Ctrl - D</i> , <i>Ctrl - U</i>	Scroll forward, backward one-half screen

'COPY-PASTE-DELETE' COMMANDS:

Sure, the name may sound weird, but at least you know exactly what is being talked about. Here is a general rule: after copy, cut, or delete some text, one can paste it.

Command	Meaning
x	Delete character after the cursor
dw	Delete word after the cursor
dd	Delete current line (where the cursor is)
yy	Copy current line (where the cursor is)
p	Paste text after cursor
u	Undo last edit

'SAVE - SAVE AS...' COMMANDS:

Command	Meaning
ZZ	Write (save) and quit file
:w	Write (save) file
:w <i>newfile</i>	Save as ... <i>newfile</i>
:w!	Overwrite [read-only] file
:q	Quit (without saving)

Of course, there are more commands than those listed above. However, this would be enough to get things done. Refer to other sources for a more complete set of commands in *vi*.

5.2 The *NEdit* Editor

NEdit is a Graphical User Interface (GUI) style text editor that is similar to those available in Macintosh and MS Windows systems. Beginning users may be able to feel very comfortable even at first use of NEdit.

To invoke NEdit, type the following command at the Shell prompt:

```
$nedit
```

NEdit also provides the interface for users with programming experience to interact with the Linux systems through Shell programming and macros.

5.3 The *Pico* Editor

Pico is an easy-to-use text editor which can be invoked by the following command:

```
$pico
```

The beginners will feel comfortable even the first time they try this editor, thanks to the menu-oriented interface of this editor.

5.4 The *Emacs* Editor

Emacs, along with **vi**, is one of the default text editors that come with the Linux operating systems. **Emacs** is popular among programmers since it enables doing several different tasks concurrently between the Linux shells and other applications. To invoke **emacs**, type the following command:

```
$emacs filename
```

5.5 Exercises

1. Use *vi* to create a text file the contents of which are shown below. Save the file under the name *daily.html*.
2. Now use another text editor of your choice to modify the file mentioned in the previous question. *Make sure you only add text in the indicated section.* Save the file as *daily.html* when done.

```

— Starts —
<html>
<head><title>Daily Assertion</title></head>
<body bgcolor="#ffccff">
<center>
<h1><font color="#ff0000">Daily Assertion</font></h1>
<p>
<br> by
<font color="#0000ff">
<br>
<b>
<!-- Type your name in this space -->
<!-- STARTS HERE -->
HT
<!-- ENDS HERE -->
</b>
</font>
</b>
<p>
<hr width=20%>
<p>
<h3>I know I can learn Linux.</h3>
<h3>Because I am smart enough.</h3>
<h3>Because I am good enough.</h3>
<h3>Because I am pretty enough.</h3>
<h3>And people like me!</h3>
</center>
</body>
</html>
— Ends —

```

Figure 4 (page 18) shows what the file looks like on my computer.

6 Getting Online

This section shows the users how to log in, transfer (upload and download) files, and check mail, at the shell prompt. The users will also learn how to use a web browser to surf the Internet. The setting up and configuration of the Internet access is already available at all of the department's computers. Users who are interested in learning how to set up for their home computers should consult the RH Documentation for more details.

6.1 Logging on to a Remote System - The *ssh* Command

ssh (SSH client) is a program for logging into and executing commands on a remote machine. It is intended to replace **rlogin** and **rsh** which are considered insecure. More information can be found in the *man* pages. In this section, we only briefly introduce to you how to use **ssh** to log in the Chemistry email machine (aka *fizzie*) to check email, as follows:

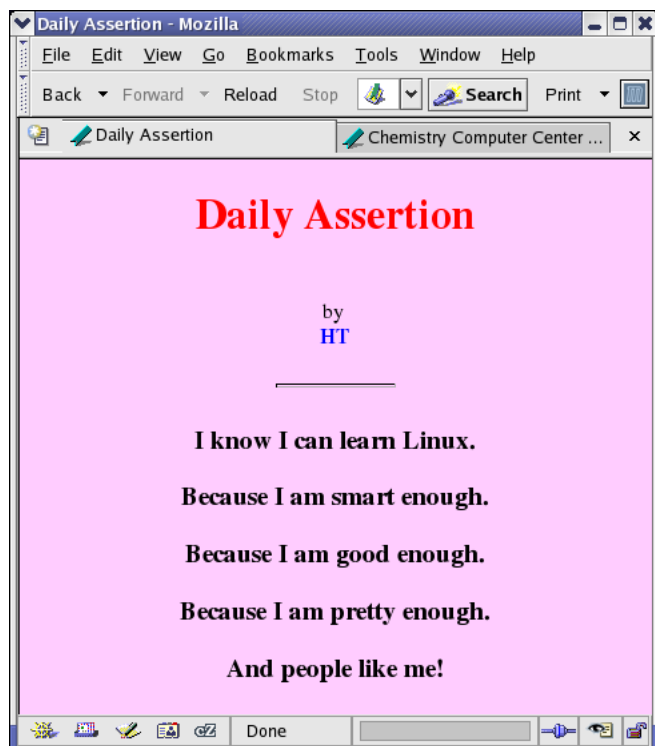


Figure 4: Daily Assertion

```
ssh -l tran fozzie.chem.wisc.edu
```

In the command above, the user **tran** tries to log in to the machine *fozzie*. The *flag -l* (letter 'el') stands for 'login'. Replace the *italics* text with the appropriate values (i.e. your username and the name of the machine you want to log in). You will be prompted to provide a password in order to get in.

Instead of using the name *fozzie.chem.wisc.edu*, one can also use the IP address of that machine, as follows:

```
ssh -l tran 128.104.68.11
```

The results are the same.

6.2 Transferring Files from/to a Remote System - The *ftp* Command

Ftp is the user interface to the Internet standard File Transfer Protocol. The program allows a user to transfer files to and from a remote network site. The man pages provide a rather straightforward description of this utility. Here we are providing you with some practical commands that would make your daily tasks easier.

6.2.1 Ftp with your Username and Password

This is the case when you have an account at the server you want to transfer files (for instance, the Chemistry Department). Consider these commands at the Shell prompt and read the footnotes for their meanings:

```

>ftp open fozzie.chem.wisc.edu 1
Connected to fozzie.chem.wisc.edu (128.104.68.11).
220 chem.wisc.edu FTP server (Version wu-2.6.1-16) ready.
Name (fozzie.chem.wisc.edu:tran):tran Enter 2
331 Password required for tran.
Password: ***** 3
230 User tran logged in.
Remote system type is UNIX.
Using binary mode to transfer files. 4
ftp>bin 5
ftp>ascii 6
ftp>cd /home/tran/computing/rh 7
ftp>lcd /local/computing/rh 8
ftp>ls /home/tran/computing/rh 9
ftp>lls Oops! This command does not exist!

```

Now, let's do some transferring:

```

ftp>cd /remotedir 10
ftp>lcd /localdir 11
ftp>bin
ftp>put upfile1 12
ftp>get downfile1 13
ftp>ascii
ftp>put upfile2.txt 14
ftp>get downfile2.txt 15

```

Now, let's get productive:

```

ftp>mput Mylife-chapter*.txt 16
ftp>bin
ftp>mget HowToPassOrganicCumes-Volume*.mov 17

```

So, there you have it! At this point there is only one thing that needs addressing. What if you want to snoop around some other ftp sites for which you don't have an account? The next section will cover this.

¹Here you invoke an ftp to *fozzie.chem.wisc.edu*

²You only have to type the username if you are not currently logged in, i.e. if you are not **tran** in this case.

³Provide the password

⁴Binary mode is for transferring binary files. In fact, this mode works in every case. Read on.

⁵This command is used to set *binary mode*. *We know. It's already in binary mode, but we have to tell you sometime.*

⁶This command is used to set *text mode*, e.g. for simple text files

⁷Change the REMOTE directory to /home/tran/computing/rh

⁸Change the LOCAL directory to /local/computing/rh

⁹List the contents of the REMOTE directory /home/tran/computing/rh

¹⁰Change the REMOTE directory to /remotedir

¹¹Change the LOCAL directory to /localdir

¹²Uploading a binary file

¹³Downloading a binary file

¹⁴Uploading an ascii (text) file

¹⁵Downloading an ascii (text) file

¹⁶Uploading all the chapters of my life in one command!

¹⁷Downloading all the movies clips on how to pass organic cumulative exams in one command!

6.2.2 Ftp anywhere as *anonymous*

Generally speaking, the login process is very much like that described in the previous section. The only differences are you will log in as *anonymous* (or *guest*) as your username, and you are usually only allowed to *download* files. The password is usually nothing or your email account, depending on where you ftp to. These sites simply want to collect some information (e.g. email address) of its users for various reasons.

Finally, we'll provide you a link to quite a list of ftp sites. It is <http://www.ftp-sites.org/>.

6.3 Checking Electronic Mail - The *pine* Email Program

6.3.1 Starting and Exiting *pine*

To start *pine*, type the following command at the Shell prompt:

```
$ pine
```

The program will start and the main menu will appear, as shown in Figure 5 (page 21).

Pine comes with a very straightforward interface and rather comprehensive on-line help files. Beginners can easily find their ways around to learn to use *pine*. Therefore, the next sections will briefly describe several important features of this program. The rest of the features will be left to the users as a practice.

6.3.2 Composing and Sending Email

To compose and send an email (after opening *pine*, as described above), do the following:

1. Fill in the email headers
 - To:** Recipient's email address (e.g., *paul@chem.wisc.edu*)
 - Cc:** Other Recipients' email address (e.g., *barnet@chem.wisc.edu* - *cc* stands for *carbon copy*)
 - Attchmnt:** Any files attached to the messages (see below)
 - Subject:** A brief memo of what the message is about
2. Type the message text, starting at the line after — *Message Text* —
3. When done, type [Ctrl-X] to send the message, or [Ctrl-C] to cancel

In the example above, I sent a message to *paul@chem.wisc.edu* as the main recipient. I also included *barnet@chem.wisc.edu* as recipient.

6.3.3 Sending Attachments with Email

If you want to enclose a file to the message you send, you need to attach it to the message. In that case, do this:

- Use the arrow keys to move the cursor to the line **Attchmnt**.

```

PINE 4.33      MAIN MENU                               Folder: INBOX  857 Messages

?  HELP          -  Get help using Pine
C  COMPOSE MESSAGE -  Compose and send a message
I  MESSAGE INDEX -  View messages in current folder
L  FOLDER LIST   -  Select a folder to view
A  ADDRESS BOOK  -  Update address book
S  SETUP         -  Configure Pine Options
Q  QUIT          -  Leave the Pine program

Copyright 1989-2001.  PINE is a trademark of the University of Washington.

? Help          P PrevCmd          R RelNotes
0 OTHER CMDS > [Help]  N NextCmd        K KBlock

```

Figure 5: Pine Main Menu

- Hit [Ctrl-T] (To File: go to the file directory).
- Use the arrows (left, right, up, down) to get to the file you want to attach. If the file is buried under many levels of directories, keep moving the cursor to the appropriate directory, then hit `Enter`. This will open the directory for you to choose the right file. When done, hit `Enter`.
- After get to the right file, type **S to Select** or **E to Exit Browser**. Either way, you will get back to the message window.
- To attach more files, repeat the steps above.

When done with attachment, you can send the email just as described above.

This process will work with any files, including *viruses*. Viruses are simply executable files (i.e. programs) that get executed when you open the attachments. *Therefore, be very careful when you decide to open an attachment; especially when it comes from an unknown source.*

6.3.4 Saving Messages & Attachments

Suppose you want to save a message (including headers and all), do this:

- Hit **S**, then hit `Return` to the question, "SAVE Msg #856 to folder [saved-messages]:"
- You'll see [Message 856 copied to folder "saved-messages" and deleted]
- If you don't want to delete the message (after saving it), hit **u**. You will then see, "[Deletion mark removed, message won't be deleted]"

To save an attachment in a message sent to you, do this:

- While reading the message, hit **>** (ViewAtch).
- Highlight the attached files you want to save. Then just follow the instructions.

6.4 Web Browsing - The Mozilla Web Browser

Mozilla is a Web browser. As we all know, its features seem endless. Fortunately, the Internet has been around for almost a decade, so it is reasonable to assume that anybody must have used a Web browser at least once. To invoke **Mozilla**, type the following command at the Shell prompt (*note the lowercase*):

\$mozilla `Enter`

6.5 Exercises

1. Send *yourself* a message that has the file *daily.html* created above attached to it
2. Use Mozilla to check out the Department of Chemistry's web site (<http://www.chem.wisc.edu>)
3. As you know, you can use a Web browser to open a web site, or to open a file on your hard drive. Use Mozilla to open the web page *daily.html* you created in the *text editors* section.

7 The GNU Object Model Environment (GNOME) Desktop

7.1 What is GNOME?

The GNU Object Model Environment (GNOME) Desktop is the default when you start using your Linux workstation. However, users can also change to a different desktop such as KDE (see below). Working in a Windows graphical interface, GNOME allows the user to perform tasks (such as word processing, spreadsheets, and graphical applications) through the clicking of the computer mouse. Many users find GNOME much more user-friendly than the Shell environment. Those who are interested in learning more about GNOME can find more in-depth discussions in the *Red Hat Linux Reference Guide*.

7.2 Exercises

1. Open the Red Hat Linux Reference Guide and go to the GNOME section.
2. Read about GNOME in the Reference Guide to find out information about three major applications in GNOME: one for word processing, one for spreadsheet processing, and one for slide presentation.

8 The K Desktop Environment (KDE)

8.1 What is KDE?

K Desktop Environment (KDE) is a desktop environment which consists of a collection of programs and documentation, using a Windows interface. Some users may find KDE much more friendly than the Shell environment. Those who are interested in learning more about KDE can find more in-depth discussions in the *Red Hat Linux Reference Guide*.

8.2 Exercises

1. Open the Red Hat Linux Reference Guide and go to the KDE section.
2. Read about KDE in the Reference Guide to find out information about three major applications in KDE: one for word processing, one for spreadsheets processing, and one for slide presentation.
3. Are the three applications you found in the previous question the same or different from those in GNOME?

9 Document History

9.1 Version History

Version 1.1: August 2004 - Minor updates, more graphics

Version 1.0: January 2004

The on-line version can be found at <http://computing.chem.wisc.edu> (Go to: [Technical](#) ⇒ [Tutorials](#) ⇒ [Linux](#)).

9.2 Updates

Next scheduled update is August 2005.

10 Contact Information

Hieu Q Tran (tran@chem.wisc.edu 262-6936)

11 References

1. *Running Linux, Third Edition*. by M. Welsh, M. K. Dalheimer, and L. Kaufman. O'Reilly & Associates, Inc.
2. *Red Hat Linux 9: Red Hat Linux Getting Started Guide*. Red Hat Inc.

3. *Red Hat Linux 9: Red Hat linux Reference Guide*. Red Hat Inc.
4. *UNIX in a Nutshell, System V Edition*. by D. Gilly and the staff of O'Reilly & Associates, Inc. O'Reilly & Associates, Inc.