CHAPTER 1

A WEBMASTER'S INTRODUCTION TO UNIX

Classic. A book which people praise and don't read.

Samuel Langhorn Clemens
Pudd'nhead Wilson (1864)
Pudd'nhead Wilson's Calendar, Ch. 16

CHAPTER OBJECTIVES

In this chapter, you will learn about:

While the UNIX operating system is ancient by Internet standards, a few years shy of 30 years old at this writing[1], it is still the foundation of the Internet and warrants some serious attention. In an online world that has become cluttered with point-and-click, UNIX is a blessed relief to the harried Webmaster who wants to make a simple, immediate change. Once mastered at almost any level, UNIX is indeed the classic of the Internet. Not only does it serve as the cleanest and most efficient means of accomplishing many tasks, but a solid understanding of the system will allow you to progress more confidently into the areas of Web and system administration in the future. It requires a small amount of effort to learn if you have no previous experience with a command-line operating environment, but you will appreciate it in the end. As most students have found, you may hate it now, but you will love it later.

SELECTING YOUR LOGIN ID AND PASSWORD

To do the exercises and projects in this workbook, you will need a shell account on a UNIX machine running the Apache Web server. This can be acquired from many ISPs,[2] and in most cases, is not terribly expensive. During the process of setting up your account, you will be asked to select a login ID and then, in most cases, be assigned a password.

Your login ID identifies who you are to the computer and must be unique. Many people like to use some variation of their name for this purpose, depending on what is available on the host machine.

A password is, in essence, the key to your account. In the UNIX environment, it is six to eight characters in length and should contain a combination of upper- and lower-case letters, numbers, and special characters. It is important that your password not be "guessable" and not contain words that can be found in the dictionary. For instance, names of your children or pets are not good password choices, nor is any variation of your address or phone number.

All that said, how does one come up with a password that will be safe from hacking, but easily remembered? Our suggestion is to think of a phrase that you are not likely to forget and generate a password using the first letters of the words in that phrase. For instance, the phrase "UNIX is a Four-Letter Word"[3] might yield the password "U!a4Lw". See the connection?

[1] The UNIX operating system is a product of AT&T and was developed in the early 1970's.

[2] Consult your local Yellow Pages for ISPs in your area.

[3] Christopher C. Taylor, "UNIX is a Four Letter Word . . . and Vi is a Two Letter Abbreviation".

Finally, as if enough has not been said about passwords, there are two more "shoulds" that need to be mentioned. First, it is not a good idea to use the same password for multiple accounts. In the rare event that your password is guessed on one machine, you do not want the culprit to gain access to your other accounts. Second, you should discipline yourself to change your password(s) periodically, and *always* change it if you suspect that it has been compromised in any way.

LAB 1.1

LOGGING IN USING FTP AND TELNET

There are two methods of accessing your shell account once it has been established and you have connected according to the instructions from your ISP.

FTP

As previously discussed, the File Transfer Protocol (FTP) is a method of copying files from one computer to another. There are a number of FTP programs available. Fetch is a popular shareware[4] program for the Macintosh. WS-FTP and CuteFtp are good choices for Windows and NT platforms. UNIX and DOS users are likely to find that FTP is available directly from the command line.

Regardless of the program you choose, there are a few basic things you need to know. First and foremost, you need to know your destination, or target, address. This can be a machine name, URL, and/or IP address, depending on how the destination machine is set up. You will also need to know your user ID and password at the destination, unless you are going to an anonymous FTP site[5].

Once you are connected, you are ready to copy files back and forth between your local machine and the remote machine. First, you will need to specify the type of file(s) you are transferring, indicating binary for images and compiled programs and ASCII for text files such as your HTML documents. Then, keeping in mind that the local machine (the computer running the FTP program) is the actor, you will use the following commands:

[4] All shareware and freeware programs mentioned in this document are currently available for download at www.tucows.com.

[5] While they are not as prolific as they used to be, anonymous FTP sites are a popular means of making documents and software available for download by the general public. More often than not, the login ID for any anonymous site will be the word "anonymous" and the password will be your valid electronic mail address.

`get` Copy a file from the remote machine to the local machine.
Example: `get myfile.html`

`mget` Copy multiple files from the remote machine to the local machine.
Example: `mget*.html`

(* is what is known as a wildcard, something that will be discussed in greater detail later on. In this case, we have asked FTP to transfer all files that end with the extension `.html`.)

`put` Copy a file from the local machine to the remote machine.
Example: `put myfile.html`

`mput` Copy multiple files from the local machine to the remote machine.
Example: `mput *.html`

How you issue these commands will differ depending on the program you are using. You should consult the documentation and help sections of the software for specifics. The following is an annotated sample session that illustrates using FTP in either DOS or UNIX.

A SAMPLE FTP SESSION

```
northshore% ftp webclass.merrimack.edu
Connected to webclass.merrimack.edu.
220-
220-        Eco Software, Inc. Unauthorized access is
prohibited.
220-
220 sparky4 FTP server (Version wu-2.4(1) Mon Jul 29
18:00:46 EDT 1996) ready.
```

Login.

```
Name (webclass.merrimack.edu:arlyn): webclass
331 Password required for webclass.
Password:
230 User webclass logged in.
```

Print working directory to see where we are.

```
ftp pwd
257 "/home/web/webclass.merrimack.edu" is current di-
rectory.
ftp cd virtual_html/ahubbell
250 CWD command successful.
```

List the directory.

```
ftp ls
200 PORT command successful.
150 Opening ASCII mode data connection for file list.
textcounter.tar
welcome.shtml
226 Transfer complete.
32 bytes received in 0.032 seconds (0.99 Kbytes/s)
```

Set transfer protocol to binary.

```
ftp bi
200 Type set to I.
```

Copy `textcounter.tar` from remote to local server.

```
ftp get textcounter.tar
200 PORT command successful.
150 Opening BINARY mode data connection for
textcounter.tar (30720 bytes).
226 Transfer complete.
local: textcounter.tar remote: textcounter.tar
30720 bytes received in 0.1 seconds (2.9e+02
Kbytes/s)
```

Set transfer protocol to ASCII.

```
ftp as
200 Type set to A.
```

Copy `welcome.shtml` from remote to local server.

```
ftp get welcome.shtml
200 PORT command successful.
150 Opening ASCII mode data connection for
welcome.shtml (174 bytes).
226 Transfer complete.
local: welcome.shtml remote: welcome.shtml
182 bytes received in 0.058 seconds (3.1 Kbytes/s)
```

Copy `test.html` from local to remote server.

```
ftp put test.html
200 PORT command successful.
150 Opening ASCII mode data connection for test.html.
226 Transfer complete.
local: test.html remote: test.html
1953 bytes sent in 0.019 seconds (1e+02 Kbytes/s)
200 PORT command successful.
150 Opening ASCII mode data connection for file list.
```

```
test.html
textcounter.tar
welcome.shtml
226 Transfer complete.
43 bytes received in 0.047 seconds (0.89 Kbytes/s)
```

Quit FTP session.

```
ftp quit
221 Goodbye.
northshore%
```

TELNET

Whereas FTP facilitates moving files back and forth between computers, telnet allows you to login directly to a remote machine and do your work there. Again, there are many programs available: CRT, for example, for Windows, and NCSA Telnet for the Mac. As with the various FTP clients, you will need to review the help documentation for your particular program regarding how to use it. A generic UNIX/DOS-style telnet session looks as follows:

```
shell3% telnet webclass.merrimack.edu
Trying 206.243.176.49 . . .
Connected to webclass.merrimack.edu.
Escape character is '^]'.

Shore.Net, a service of Eco Software, Inc. Access
restricted to Shore.Net hosting customers only.
Unauthorized access is prohibited.


SunOS UNIX (sparky4)

login: webclass
Password:
Last login: Wed Jun 30 21:09:34 from 216.20.65.70
                 Welcome to the Shore.Net Virtual
Host server
                                      sparky4
        All use subject to terms and conditions of
membership agreement.
             Any access to other networks through
Shore.Net must
             comply with the rules appropriate for
that network.
```

```
        If you do not agree with these rules, disconnect
       now.
         For questions or assistance, mail support@shore.net
               or call (781) 593-3110.
           Maintenance Windows and System Notes:
       Window Type             Day             Time
       Regular maintenance     Tues/Thur       0400 - 0600
       Extended maintenance    Sun             0400 - 0800
       Emergency               Any             0400 - 0600
    see http://www.shore.net/services/policies for more
    information


  o  The Membership Agreement is available at:
     http://www.shore.net/services/policies/agreement.html
        webclass.merrimack.edu%
```

LAB 1.1 EXERCISES

1.1.1 LOGGING IN USING FTP AND TELNET

a) What does FTP stand for?

_____

_____

b) When would you use FTP?

_____

_____

c) List the two transfer modes for FTP.

_____

_____

d) List the four major commands used to copy files and what they are used for.

e) What is telnet?

LAB 1.1 EXERCISE ANSWERS

1.1.1 ANSWERS

a) What does FTP stand for?

Answer: File Transfer Protocol.

b) When would you use FTP?

Answer: FTP is used to copy files from one machine to another.

c) List the two transfer modes for FTP.

Answer: *bi – Binary*

as – ASCII

d) List the four major commands used to copy files and what they are used for.

Answer:

`get` *Copies one file from the remote to the local machine.*

`mget` *Copies multiple files from the remote to the local machine.*

`put` *Copies one file from the local to the remote machine.*

`mput` *Copies multiple files from the local to the remote machine.*

e) What is telnet?

Answer: Telnet is an Internet protocol that allows the user to connect and login to a remote machine.

LAB 1.1 SELF-REVIEW QUESTIONS

1) Which of the following would qualify as a safe password?
   a) Hubbell
   b) 33StateStreet
   c) TaNSTa2F!
   d) All of the above
   e) None of the above

2) A password should not contain
   a) Special characters
   b) Upper- and lower-case letters
   c) English words
   d) Numbers

3) An example of an ASCII file is
   a) An image
   b) A compiled program
   c) An HTML document

4) Passwords should
   a) Never be changed
   b) Be changed periodically
   c) Be the same for all of your accounts

Quiz answers appear in the Appendix, Section 1.1.

Regardless of the telnet method you use, once you have logged in, you are ready to start working in the UNIX environment. This brief tutorial is by no means meant to replace a full study of the UNIX operating system and its capabilities, but let's take a look at some of the basic commands you will be using.

LAB 1.2

BASIC UNIX COMMANDS

man Displays UNIX manual pages.

man is the UNIX equivalent of "help," and will display detailed information about any given command that has a man page (see Figure 1.1).

EXAMPLE:

```
Shell3% man whoami
```

pwd Displays (prints) the working directory.

This command answers the question, "Where am I?" with a directory path. This information will become more important to you as you start setting up multiple directories in your Web site. A common problem is losing track of where you are during a telnet session; `pwd` gives you the answer.

EXAMPLE:

```
shell3% pwd
/home/2/a/arlyn
shell3%
```

ls Lists the current directory.

The `ls` command by itself displays all of the user file and directory names in your current directory.

Note that directory names are followed by a '/'.

ls -alg Lists the current directory with details.

`ls -alg` displays the contents of your current directory, including system (dot) files that con-

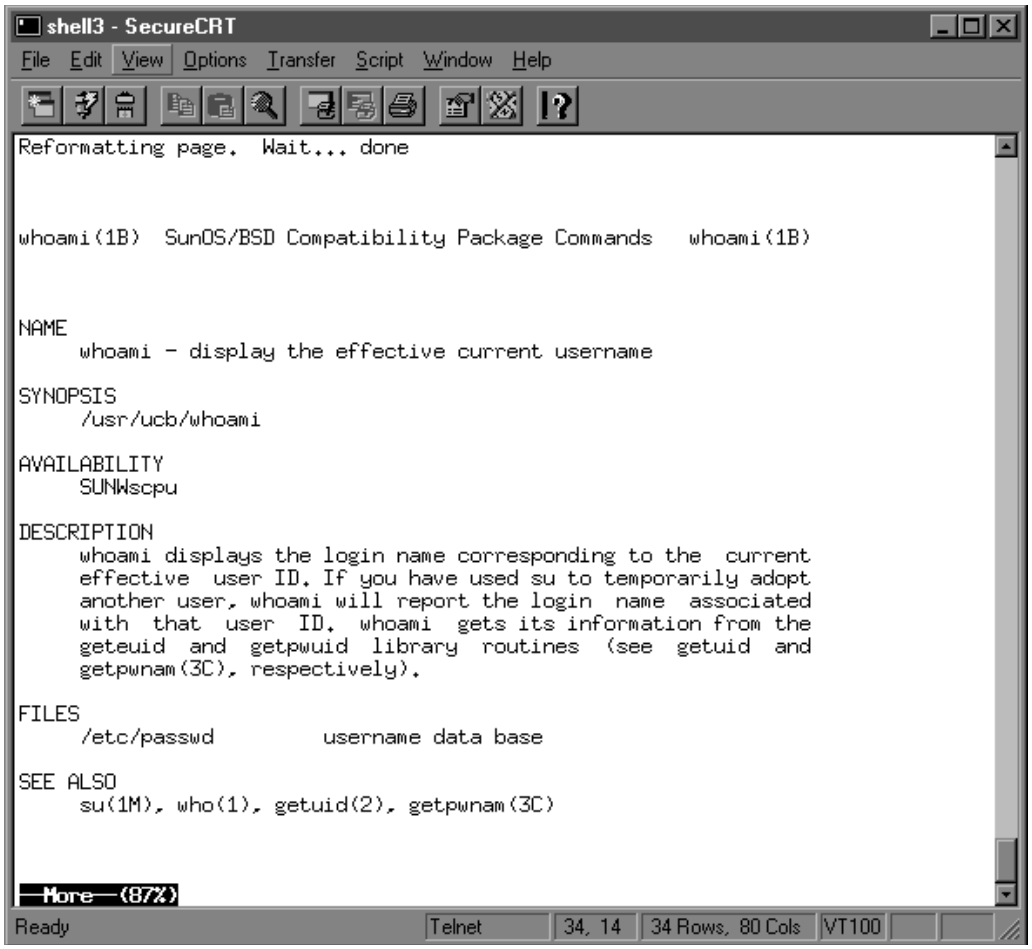

Figure 1.1 ■ The UNIX man page for `whoami`.

tain instructions for your account (you might want to leave these alone for the time being), as well as access permissions, owners, file sizes, and date and time last modified. See Figure 1.2 for an example.

`whoami` Who am I?

This command displays your user ID.

EXAMPLE:

```
shell3% whoami
arlyn
```

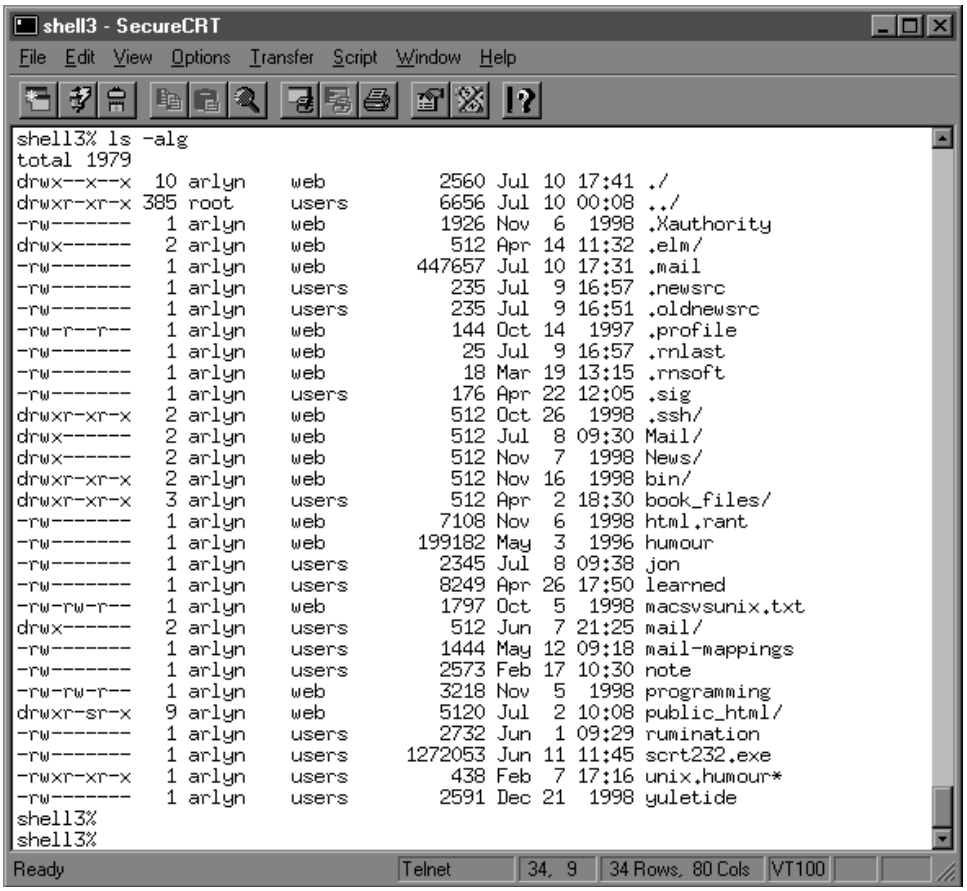

Figure 1.2 ■ The `ls -alg` command.

```
shell3%
```

`quota -v` Displays the quota, which is the amount of space you have available to you on the server.

`cd` Changes the directory.

`cd` allows the user to move among the directories to which they have access.

EXAMPLE:

```
shell3% cd book_files
shell3%
```

Note that the request to change to the directory `book_files` returns nothing but the UNIX prompt—in this case, `shell3%`. Many UNIX commands are of the attitude "No news is good news," and will not return a message unless an error has occurred. For example, if we try to move into a directory that does not exist as we've specified, we will receive the following:

```
shell3% cd nonexistent_directory
nonexistant_directory: No such file or
directory
shell3%
```

The variations of `cd` we will be using are all of the same attitude; they are only listed and defined herein. The output of each successful request will look exactly like the initial `cd book_files` shown above. If you are ever in doubt as to whether you are where you think you are, issue the `pwd` command.

`cd` Changes to your home directory, where you were when you first logged in.

`cd ..` Moves up one directory.

LAB 1.2 EXERCISES

1.2.1 BASIC UNIX COMMANDS

a) Write the command that will tell you where you are on a UNIX server.

_____

_____

b) What is the command that displays your user ID?

_____

_____

c) Write the command that will move you into a subdirectory named `images`.

d) Write the command that will move you up one directory level.

e) Write the command that will move you back to your login directory.

f) Obtain help for the `date` command.

LAB 1.2 EXERCISE ANSWERS

1.2.1 ANSWERS

a) Write the command that will tell you where you are on a UNIX server.

Answer: `pwd`

b) What is the command that displays your user ID?

Answer: `whoami`

c) Write the command that will move you into a subdirectory named `images`.

Answer: `cd images`

d) Write the command that will move you up one directory level.

Answer: `cd ..`

e) Write the command that will move you back to your login directory.

Answer: `cd`

f) Obtain help for the `date` command.

Answer: `man date`

LAB 1.2 SELF-REVIEW QUESTIONS

1) When using the `ls -alg` command, directory names are followed by a
- **a)** *
- **b)** .
- **c)** /
- **d)** ..

2) The `ls` commands displays
- **a)** Filenames
- **b)** Directory names
- **c)** Both of the above

3) `ls –alg` displays
- **a)** File and directory names
- **b)** Access permissions
- **c)** File sizes
- **d)** All of the above

4) UNIX will most often confirm the successful completion of a command.
- **a)** True
- **b)** False

Quiz answers appear in the Appendix, Section 1.2.

LAB 1.3

FILE AND DIRECTORY MANAGEMENT

cp Copies one file to another.

The copy command is used to copy one file to another and is another example of a UNIX command that does not display output unless there is a problem. It should also be noted that the command will not warn you if you are about to overwrite an existing file. UNIX, for better or worse, depending on how you feel about operating systems questioning your every command, usually assumes that you know what you are doing and does what you ask.

EXAMPLE:

```
shell3% cp toucans.GIF toucans.gif
shell3%
```

Notice the filenames `toucans.GIF` and `toucans.gif`. `toucans.GIF` is the original file and `toucans.gif` is the new file being created. The name of the new file was chosen to illustrate the case-sensitive nature of UNIX. `toucans.GIF` and `toucans.gif` are indeed two unique files.

```
shell3% ls toucans.*
toucans.GIF  toucans.gif
shell3%
```

This is going to be important to remember when you begin naming and accessing the files and directories on your Web site. The same case sensitivity applies to UNIX commands as well. `CP` is not the same as `cp`, and in fact, will generate an error.

cp -r Copies recursively.

This variation of `cp` is used to copy a directory and its contents to another directory. `cp` alone will return an error if used with directories:

```
shell3% cp book_files book_files_new
cp: book_files: is a directory
shell3%
```

On the other hand, `cp -r` successfully creates a new directory:


```
shell3% cp -r book_files book_files_new
shell3%
```

mkdir Makes a directory.

`mkdir` is used to create directories and subdirectories within your account.

EXAMPLE:

```
shell3% mkdir images
shell3%
shell3% ls
images/  toucans.GIF  toucans.gif
shell3%
```

mv Moves files and directories.

`mv` is used to move files and directories from one place to another, as well as for renaming. Going back to our example files, we can change the name of `toucans.GIF` to `toucans.Gif` in this way:

```
shell3% mv toucans.GIF toucans.Gif
shell3% ls toucans.*
toucans.Gif toucans.gif
shell3%
```

The `ls` command verifies that the name of the file has been changed.

Now, note what happens when we use `mv` to move `toucans.Gif` into the `images` directory:

```
shell3% mv toucans.Gif images
```

```
shell3% ls toucans.*
toucans.gif images/
shell3%
```

`toucans.Gif` no longer appears in our directory listing because the file has been relocated to `images`.

`rm` Removes files.

`rm` removes, or deletes, files and should be used with care as it does so without confirming your request. In UNIX, a request to remove a file is completed immediately; the file is not "marked" for removal at a later time, and there is no "trashcan" or "recycle bin" from which you can retrieve the file. Done is done.

EXAMPLE:

```
shell3% rm toucans.gif
shell3% ls toucans.gif
/usr/ucb/ls: No match.
shell3%
```

Notice that UNIX did not question our intent, and the subsequent `ls` does indeed confirm that our `toucans.gif` file is gone. The more severe consequences of using `rm` carelessly can be illustrated by combining the command with wildcards:

```
shell3% ls toucans.*
toucans.GIF toucans.Gif  toucans.gif
shell3% rm toucans.*
shell3% ls toucans.*
/usr/ucb/ls: No match.
shell3%
```

All three files are removed with nary a peep out of UNIX.

`rmdir` Removes a directory.

`rmdir` is used to remove an empty directory. If the target of the command contains files or subdirectories, UNIX will complain:

```
shell3% rmdir remove_me
```

```
rmdir: directory "remove_me": Directory not
empty
shell3% cd remove_me
shell3% ls
file1 file2  file3
shell3% rm *
shell3% cd ..
shell3% rmdir remove_me
shell3%
```


Once we `cd` into `remove_me`, `rm` the existing files, and `cd` back one level, `rmdir` honors our request.

If you have directories you wish to remove without first removing the files and subdirectories they contain, you may, but please do so with care. The command is `rm -r`, and for the above example, would have been issued as such:

```
shell3% rm -r remove_me
shell3%
```

`pico` Opens a simple full-screen editor on most UNIX servers.

A new file is opened by issuing the command with no argument, e.g., `pico` (see Figure 1.3). An existing file can be

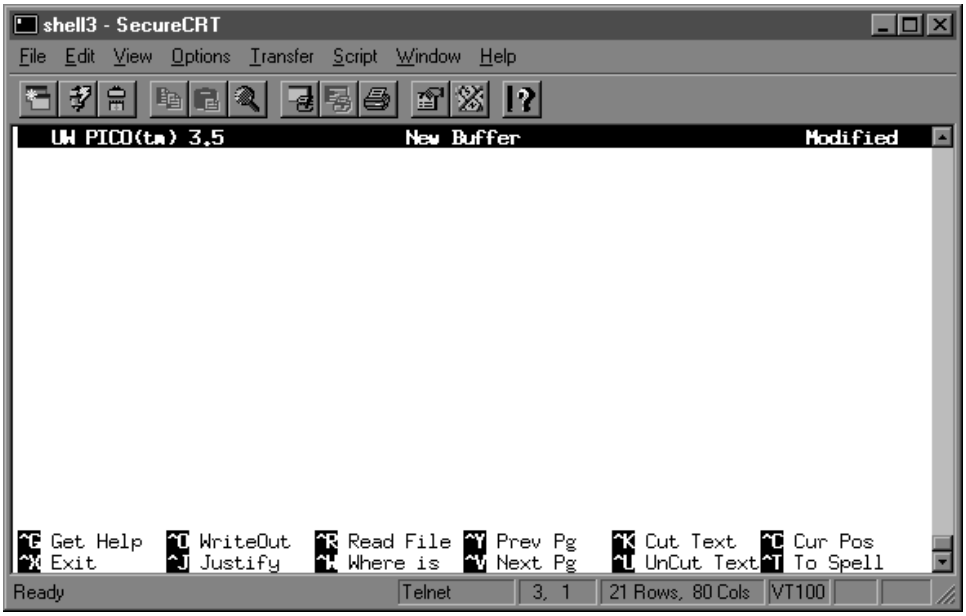

Figure 1.3 ■ A new file opened with `pico`.

opened by issuing the command followed by the name of the file you wish to modify, e.g., `pico myfile.html`. Your computer's arrow keys can be used to move among the text and control-character commands listed across the bottom of the editor window. The Help file, accessible via Control-G, details the editor's commands and their use.

Other text editors that are available on most UNIX servers are vi and Emacs. While it is entirely possible to create HTML files using an editor on a PC or Macintosh and then transfer to them to a UNIX server, it is strongly suggested that you become comfortable with the UNIX editor of your choice and work directly on the server. You will, in the long run, greatly appreciate bypassing the step of transferring a file every time it is changed.

LAB 1.3 EXERCISES

1.3.1 FILE AND DIRECTORY MANAGEMENT

a) What is the command to create a directory?

b) Copy a file named `Parrots` to another file named `Two_Parrots`.

c) Copy a directory named `Birds` to another directory named `Tweet`.

d) Rename the file `Parrots Two_Parrots.old`.

e) Copy `Two_Parrots.old` and `Two_Parrots` into the directory `Tweet`.

LAB 1.3 EXERCISE ANSWERS

1.3.1 ANSWERS

a) What is the command to create a directory?

Answer: `mkdir`

b) Copy a file named `Parrots` to another file named `Two_Parrots`.

Answer: `cp Parrots Two_Parrots`

c) Copy a directory named `Birds` to another directory named `Tweet`.

Answer: `cp –r Birds Tweet`

d) Rename the file `Parrots Two_Parrots.old`.

Answer: `mv Parrots Two_Parrots.old`

e) Copy `Two_Parrots.old` and `Two_Parrots` into the directory `Tweet`.

Answer: `cp Two.* Tweet`

LAB 1.3 SELF-REVIEW QUESTIONS

1) UNIX is case-sensitive.
- **a)** True
- **b)** False

2) `rmdir` will remove a directory and all of the files within it.
- **a)** True
- **b)** False

3) UNIX, by default, gives a warning if you are about to overwrite a file using `cp` or `mv`.
- **a)** True
- **b)** False

4) When `rm` is used with wildcards (*), UNIX will confirm the deletion of every file.
a) True
b) False

5) It is not possible to remove a directory unless the files it contains are removed first.
a) True
b) False

6) In UNIX, the commands `rm parrots` and `rm PARROTS` would remove the same file.
a) True
b) False

Quiz answers appear in the Appendix, Section 1.3.

LAB 1.4

FILE AND DIRECTORY PERMISSIONS

Perhaps one of the most important things to know in UNIX is how to grant and restrict access to your files and directories. For an HTML file to be readable over the Web, it must exist in a directory that is accessible to everyone and readable by everyone. The command used to change read, write, and execute permissions on files is `chmod`. Before looking at the command's syntax, however, let's take a quick look at a directory's and file's permissions as displayed with `ls -alg`.

The following two directory entries are for a directory named `parrots` and a file named `pomegranate.html`:

```
drwxr-xr-x  2 arlyn    users         512 Sep  2 22:08
parrots/
-rw-r--r--  1 arlyn    users         463 Jul 22 11:43
pomegranate.html
```

The permissions for the file and directory are shown in the first column and are broken down as follows:

Directory Flag	Owner Permissions	Group Permissions	World Permissions	
d	rwx	r-x	r-x	(parrots)
-	rw-	r--	r--	(pomegranate.html)

You'll note that the first column indicates whether the item is a file or a directory: `d` means it is a directory, `-` means it is a file. The remaining three columns detail the permissions for the owner of the file (`u`), the group that owns the file (`g`), and the rest of the world (`o`).

Value	Permission		Hexidecimal Value
r	Read		4
w	Write		2
x	Execute	(for a file)	1
	Enter	(for a directory)	

Based on the above, you can see that the directory `parrots` can be read by the owner, the group, and the world, only the owner can write to it, and everyone can enter it. The file `pomegranate.html` can be read by the world, but can be written to only by the owner.

Using `chmod`, one way to establish the permissions listed for the directory and file is as follows:


```
chmod 755 parrots
chmod 644 pomegranate.html
```

Take a minute to look at the `chmod` commands, the hex values of the permissions, and how they are grouped in the directory listings. In the case of the directory `parrots`, 755 is the result of adding together the numerical values for the owner permissions (rwx=4+2+1=7), the group permissions (rx=4+1=5), and the world permissions (rx=4+1=5). Using this logic, can you see the correlation between 644 and `rw-r--r--`for the file `pomegranate.html`?

Certainly, any number of permission variations can be established in this fashion, but 755 and 644 are the permission settings that are important for you to remember when making your HTML files available over the Web. 755 will allow browsers to enter and read or list directories, and 644 will allow them to read individual files. When creating new files and directories, always verify that the permissions are correct. While some servers will set them for you automatically, others will not, and this may lead to errors when a visitor tries to open your pages.

LAB 1.4 EXERCISES

1.4.1. FILE AND DIRECTORY PERMISSIONS

a) Based on the logic presented above, write the command to set the permissions for `pomegranate.html` such that only the owner of the file can read and write to it. The group and world will have no permissions at all.

b) Write the command to set permissions for `pomegranate.html` such that the owner, group, and world are only able to read the file.

c) Set the permissions for `parrots` such that only the owner can list and enter the directory.

d) Set the permissions for parrots such that the owner can list, write to, and enter the directory, and the group and world can only enter the directory.

LAB 1.4 EXERCISE ANSWERS

1.4.1 ANSWERS

a) Based on the logic presented above, write the command to set the permissions for `pomegranate.html` such that only the owner of the file can read and write to it. The group and world will have no permissions at all.

Answer: `chmod 600 pomegranate.html`

b) Write the comand to set permissions for `pomegranate.html` such that the owner, group, and world are only able to read the file.

Answer: `chmod 444 pomegranate.html`

c) Set the permissions for `parrots` such that only the owner can list and enter the directory.

Answer: `chmod 600 parrots`

d) Set the permissions for `parrots` such that the owner can list, write to, and enter the directory, and the group and world can only enter the directory.

Answer: `chmod 711 parrots`

LAB 1.4 SELF-REVIEW QUESTIONS

1) The numerical value of read+write+execute is
- **a)** 7
- **b)** 6
- **c)** 2

2) The numerical value of read+execute is
- **a)** 6
- **b)** 5
- **c)** 1

3) In most cases, files meant to be accessed by browsers over the Web must be readable by the
- **a)** Owner
- **b)** Group
- **c)** World

4) x means
- **a)** Executable
- **b)** Enter
- **c)** Both of the above

Quiz answers appear in the Appendix, Section 1.4.

CHAPTER 1

TEST YOUR THINKING

The projects in this section use the skills you've acquired in this chapter. The answers to these projects are available to instructors only through a Prentice Hall sales representative and are intended to be used in classroom discussion and assessment.

The only way to get comfortable with the UNIX operating system is to use it. As suggested, you should obtain a UNIX shell account and start working with the commands discussed in this chapter. Find out from your provider the specifics of setting up a Web site on their server. Practice creating directories and files, setting appropriate permissions, and transferring files back and forth between the server and your local computer.

1) Create a couple of directories in the web area of your account: Images and Content. Make sure the permissions are set so that you will be able to create and modify files within them, and others will be able to browse the files from the web.

2) Using the ftp program of your choice, transfer a few files from your local machine to the directories you created. Put graphics in your Images directory and plain text in Content.

3) Double-check the transferred files for proper permissions. Reset them if necessary.