Chapter 3

Linux File Processing

In this chapter, you will learn the following to World Class standards:

1. **Common Commands in Linux File Processing**

Processing Files in Linux

There are a number of different commands and phrases we can use to view and manipulate files in Linux. We will go through some of the commonly used ones.

First we will look at the **Stream Editor** command. This phrase allows us to see a specific line in a text file. In the example, the number 2 means the second line of the file will be displayed, and file designates where we should enter the file we wish to view.

```
sed –n 2p file
```

The **Translator** command displays the contents of a file in a specific case. The first example shows how to display the file in all capital letters. The second shows how to display it in completely lower-case.

```
tr [a-z] [A-Z] < file

tr [A-Z] [a-z] < file
```

We can display files in a couple of other ways using the **Translator** command. In the first example here, we use **–d** to display the file without the characters **a**, **c**, or **f**. in the next example we use **–s** to substitute any space in the file with a comma.

```
tr –d “acf” < file

tr –s “,” “ “ < file
```

There are a few commands that we can use to save and quit files. The first example here, **:w**, saves the file we are working on but does not exit it. The second two commands, **:x** and **:q!** quit the file, but neither saves it.

```
:w file
:x
:q!
```

We can use the **Cut** command to take entire fields of a database and save them, in this case, to temporary files. The first example cuts column 2 (or field 2) from the database, designated as file, to the temporary file **temp1**. The second example cuts field 5 and saves it to **temp2**.

```
cut  -f2 -d :file > temp1

cut  -f5 -d :file > temp2
```

The **paste** command can be used to place the temporary files we created with the **cut** command into another file. In the example we paste both temporary files into another file. To view the newly-pasted contents use the **more** command followed by the filename.

```
paste temp1 temp2 > file

more file
```

We can use the **more** command again in conjunction with the list command to see the entire contents of the large directory **etc**, which represents any directory you wish to view. We can then view more of the content by hitting the spacebar. In the second example we added a **sort –r** command to list the contents of the **etc** directory in reverse order; we can still use the spacebar to view more of the content.

```
ls – l /etc | more

ls /etc | sort –r | more
```

The **grep** command is a very useful tool when trying to isolate certain files or lines in files. In this first example we use grep to search for a certain word, shown as text, in a directory, show as **etc/vnc.conf**. Again we add the more command to then end; we can advance through the results using the spacebar.

```
grep text/etc/vnc.conf |more
```

In this example we use **grep** to find the word **programmer** in every text file in the **ect** directory. Use an asterisk in place of a filename to search every text file in the directory. The more command again allows us to advance the results using the spacebar. The second example searches every file for the term **Linux**, not just .txt files.

```
grep programmer /etc/*.txt | more

grep Linux /etc/* | more
```

grep can also be used to search for phrases by putting our search phrase in quotation marks. In this example we search for the phrase **“maximum number”** in the etc directory.

```
Grep “maximum number” /etc/* | more
```

The Wildcard, or asterisk, can be used with the list command as well. In the first example we use it to list all of the text files in the current directory. In the second example we use it to list all of the **.conf** that have filenames beginning with s in the etc directory.

```
ls *.txt
ls /etc/s*.conf
```

There are a few operations we can execute with two files. The **uniq** command will duplicate the contents of file1 into file2. The **comm** command will compare the similarities between two files. The **diff** command will display the differences.

```
uniq file1 > file2
comm file1 file2
diff file1 file2
```

Finally, there are some operations we can use to see some file statistics. Using **wc –l** we can see the number of lines in a file. **wc-c** will show the number of bytes in a file, while **wc –w** will show how many words the file contains. **Wc –lcw** will display all three statistics for the designated file.

```
wc –l file
wc –c file
wc –w file
wc –lcw file
```