1 About this document

This is a collection of good introductions into using the Linux or unix command line. They are changed and editied slightly from there web counter parts, feel free to use the orginals as provided via the links.

Note that Copyright permission was recieced to produce this for the course from the different sources.

2 Quick Intro Into Linux

This is from `http://members.toast.net/art.ross/Linux\_2003/Linux\_Newbie\_Manual/part1/command1.html`

Welcome to Super-DOS!

That thing that looks like MS-DOS is a Linux terminal (sometimes called a "console"). It's here that all Linux commands are entered. If you're familiar with DOS, then get ready for Super-DOS!

3 The BASH shell

Other shells are available, but BASH, the Bourne Again Shell, is the Linux default. It incorporates features of its namesake, the Bourne shell, and those of the Korn, C and TCSH shells.

The BASH built-in command history remembers the last 500 commands entered by default. They can be viewed by entering history at the command prompt. A specific command is retrieved by pressing the UP ARROW or DOWN ARROW at the command prompt.

3.1 Guide to Linux File Command Mastery

By Sheryl Calish `http://www.oracle.com/technology/pub/articles/calish\_file\_commands.html` A crash course in Linux file commands for the newly initiated

Although GUI desktops such as KDE and GNOME help users take advan-

tage of Linux features without functional knowledge of the command-line interface, more power and flexibility are often required. Moreover, a basic familiarity with these commands is still essential to properly automate certain functions in a shell script.

This article is a "crash course" in Linux file commands for those who are either new to the operating system or simply in need of a refresher. It includes a brief overview of the more useful commands as well as guidance regarding their most powerful applications. Combined with a little experimentation, the information included here should lead to an easy mastery of these essential commands. (Note: When a kernel tweaked with Oracle Cluster File System (OCFS2) is involved, some of these commands may behave somewhat differently. In that case, Oracle provides an OCFS2 toolset that can be a better alternative for file command purposes.)

Note that all the included examples were tested on SUSE Linux 8.0 Professional. While there is no reason to believe they will not work on other systems, you should check your documentation for possible variations if problems arise.

3.2 Background Concepts

Before delving into specifics, let's review some basics.

3.2.1 Files and Commands

Everything is treated as a file in the Linux/UNIX operating system: hardware devices, including the keyboard and the terminal, directories, the commands themselves and, of course, files. This curious convention is, in fact, the basis for the power and flexibility of Linux/UNIX.

Most commands, with few variations, take the form:

```
command [option] [source file(s)] [target file]
```

3.2.2 Getting Help

Among the most useful commands, especially for those learning Linux, are those that provide help. Two important sources of information in Linux are the on-line reference manuals, or `man` pages, and the `whatis` facility. You

can access an unfamiliar command's `man` page description with the `whatis` command.

```
whatis echo
```

To learn more about that command use:

```
man echo
```

If you do not know the command needed for a specific task, you can generate possibilities using `man -k`, also known as `apropos`, and a topic. For example:

```
man -k files
```

One useful but often-overlooked command provides information about using `man` itself:

```
man man
```

You can scroll through any `man` page using the SPACEBAR; the UP ARROW will scroll backward through the file. To quit, enter q,!, or CTRL-Z.

4 Using info, the Documentation Tools

`http://www.linuxtopia.org/online\_books/centos\_linux\_guides/centos\_linux\_developer\_tools\_guide/s1-using-info.html`

`info` provides the sources for documentation for the GNU tools.

4.1 Reading info Documentation

Browse through the documentation using either Emacs[1] or the `info` documentation browser program. The information is in nodes, which correspond to the sections of a printed book. Follow them in sequence, as in books, or, using the hyperlinks, find the node that has the information you need. `info` has hot references (if one section refers to another section, `info` takes you directly to that other section, but gives you the ability to return easily. You can also search for particular words or phrases. Use `info` by typing its name at a shell prompt; no options or arguments are necessary. Check that `info` is in your shell path. If you have problems running `info`, contact your system administrator.

[1]The best or worst text editor to ever be created. Depending on your point of view

To get help with using `info`, type `h` for a programmed instruction sequence, or [Ctrl]-[h] for a short summary of commands. To stop using `info`, type [q].

4.2 About Paths

Whenever I mention a path, I'm speaking about the route taken to get to a file or directory. (Don't confuse this with PATH which is an "environment variable" - discussed later.)

One thing I should mention - to prevent confusion - is that drive letters don't exist in Linux. There is no A:, C: or D:, only the root directory (/), which contains everything else. You'll also notice that the backslash (\) is never used in Linux, only the forward-slash (/).

4.3 Absolute And Relative Paths

A relative path means just that, relative to the current directory. You can always tell a relative path because it never starts with a /. If you were in /etc and wanted to move to the directory /etc/defaults then instead of entering the absolute path:

```
cd /etc/defaults
```

...it's quicker just to enter the relative path:

```
cd defaults
```

Ok, now that you have some fundamental knowledge of paths, it's time to learn the two most important commands, `cd` and `ls`.

4.4 cd - change directory

4.4.1 Examples

```
cd
```

Move to home directory of the account you're logged into, e.g. `/home/guest`.

```
cd -
```

Move back to previous directory you were in.

`cd ..`

Move up to the parent directory, i.e. the directory that holds the directory you're currently in.

`cd /usr/docs`

Move to the directory docs, which is contained in the directory usr, which like everything else, is contained in the root directory (/).

`cd reports`

Move to the directory reports, contained in the directory you're currently in.

4.5 ls - list [a directory]

4.5.1 Examples

`ls`

Display the contents of the directory you're currently in.

`ls -a`

Display every file and directory in the current directory (i.e. including "hidden" files and directories).

`ls -l`

Display the files in the current directory, including (from left to right): the file type; permissions; the number of hard links; the owner name; the group name; the size in bytes; and the date the file was last saved.

To combine the above two examples you would enter:`ls -al`

`ls | less`

Pipe (pass) the listing of the current directory to the less program, allowing you to scroll up and down a long listing. (That — character is called a "pipe".)

4.6 Less Keys:

Down cursor = Down 1 line.

Up cursor = Up 1 line.

Spacebar = Down 1 screen.

b = Back up 1 screen.

q = Quit.

h = Activate help screen.

`ls -l /usr | less`

Pipe the listing of the `/usr` directory (passing all file details (-l)), to the `less` program.

`ls > list.of.files`

Either create or overwrite the file `list.of.files`, with a redirected (¿) listing of the current directory, as the file's content.

`ls >> list.of.files`

The same as above, except this time the listing is appended to the end of the file `list.of.files`.

5 Making and Removing Directories

This is from `http://members.toast.net/art.ross/Linux\_2003/Linux\_Newbie\_Manual/part1/command2.html` Learn by *doing* is the nature of this manual, and in keeping with that belief, here's a tutorial to teach you the `mkdir` and `rmdir` commands, to make and remove directories.

Rather than entering a tedious:

`cd` (Only required if you're not in your home directory.)

`mkdir a`

`mkdir a/sub`

`mkdir a/sub/subsub`

...simply enter:

```
cd; mkdir a a/sub a/sub/subsub
```

Note

Here you've just been introduced to a nice trick - using the semi-colon (;) to enter multiple commands in one go. This will work with anything you can think of (even aliases!).

To check that everything is, *as* it should be, enter:

```
tree a
```

...to display the following:

```
a

‘-- sub

    ‘-- subsub

2 directories, 0 files
```

Now that the three directories have been created, it's time to destroy them!

Enter:

```
rmdir a
```

5.1 Some Things You Can't Do

To prevent accidents, `rmdir` will refuse to remove directories that contains files and/or other directories. Which is why you received the error message:

```
rmdir:  a:  Directory not empty
```

This will also prevent you removing a parent directory, since by its *very* name, it*has* to contain your current working directory. Logic also dictates that you can't remove your current directory since this would involve you being lost in limbo'!

The `rmdir` has a handly little option, called `-p`. It's purpose is to ask the shell to not only remove a directory, but also all of its parent directories - *so long as* each parent is empty.

Enter:

```
rmdir -p a/sub
```

...oh dear, still an annoying error message:

```
rmdir:  a/sub:  Directory not empty
```

Now press the up cursor key to bring back up the last command, but this time add `/subsub` before pressing the Return key, like so:

```
rmdir -p a/sub/subsub
```

Bingo, all three directories have been removed!

6 Copy move and remove files

This is from `http://members.toast.net/art.ross/Linux\_2003/Linux\_Newbie\_Manual/part1/command3.html`

6.1 cp - copy

6.1.1 Examples

```
cp -r a b
```

Copy the entire contents of directory `a` into directory `b`.

```
cp a.txt backup
```

Copy the file `a.txt` across to the directory `backup`.

```
cp a.txt backup/a_bk.txt
```

Copy the file `a.txt` across to the directory `backup` and rename it `a_bk.txt` at the same time. If the file `backup/a_bk.txt` already existed then it will be overwritten without warning.

```
cp -i a.txt backup/a_bk.txt
```

The same as above but asks for confirmation before overwriting an existing file.

`cp a b c d`

In the above form, every parameter but the last - are the files to copy, and the last parameter - is the directory to copy the files to.

6.1.2 Note

When copying more than one file you can't rename it at the same time.

6.2 mv - move

`mv` is a clever little command that will either move a file (`mv file directory`), or rename it (`mv oldfilename newfilename`).

`mv a b c work/reports`

Move the files `a, b,` and `c` into the directory `work/reports`.

`mv -i a b c work/reports`

The same as above, but asks for confirmation before overwriting any existing files.

`mv a.txt b.txt`

Rename the file `a.txt - b.txt`.

6.3 rm - remove

`rm a b c`

Remove the files `a, b` and `c`.

`rm -i a b c`

The same as above, but asks for confirmation before removing each file - *very* important when removing files using *wildcards*!

`rm -r work/reports`

Ther stands for "recursive" meaning *every* file and directory, in the directory `work/reports` will be removed,*without* warning - so be careful!

7 Viewing and Searching

`http://unix.cms.gre.ac.uk/general/viewing.html`

7.0.1 Note:

These tutorials assume you have a copy of the tutorial.txt file which can be copied from:

/home/wug01/public_html/general/tutorial.txt

Or alternatively, it can be downloaded from `http://unix.cms.gre.ac.uk/general/tutorial.txt` and saved in your home area. (Right click and Save As)

7.0.2 cat (concatenate)

The command `cat` can be used to display the contents of a file on the screen. Type:

```
cat tutorial.txt
```

As you can see, the file is longer than than the size of the window, so it scrolls past making it unreadable.

7.0.3 less

The command `less` writes the contents of a file onto the screen a page at a time. Type:

```
less tutorial.txt
```

Press the space-bar if you want to see another page, type q if you want to quit reading. As you can see, less is used in preference to cat for long files.

7.0.4 head

The `head` command writes the first ten lines of a file to the screen.

First clear the screen then type:

```
head tutorial.txt
```

Then type:

```
head -5 tutorial.txt
```

What difference did the -5 do to the `head` command?

7.0.5 tail

The `tail` command writes the last ten lines of a file to the screen.

Clear the screen and type:

```
tail tutorial.txt
```

How can you view the last 15 lines of the file?

7.0.6 Simple searching using less

Using `less`, you can search though a text file for a keyword (pattern). For example, to search through tutorial.txt for the word tutorial, type:

```
less tutorial.txt
```

then, still in `less` (i.e. don't press q to quit), type a slash followed by the word to search

```
/tutorial
```

As you can see, `/tutorial` finds and highlights the keyword. Type n to search for the next occurrence of the word.

7.0.7 grep

`grep` is one of many standard UNIX utilities. It searches files for specified words or patterns. First clear the screen, then type:

```
grep tutorial tutorial.txt
```

As you can see, `grep` has printed out each line containing the word tutorial.

Or has it????

Try typing:

```
grep tutorial tutorial.txt
```

The `grep` command is "case sensitive"; it distinguishes between tutorial and Tutorial. To ignore upper/lower case distinctions, use the `-i` option, i.e. type:

```
grep -i tutorial tutorial.txt
```

To search for a phrase or pattern, you must enclose it in single quotes (the apostrophe symbol). For example to search for spinning top, type:

```
grep -i 'spinning top' tutorial.txt
```

Some of the other options of `grep` are:

`-v` display those lines that do NOT match
`-n` precede each matching line with the line number
`-c` print only the total count of matched lines

Try some of them and see the different results. Don't forget, you can use more than one option at a time, for example, the number of lines without the words tutorial or Tutorial is:

```
grep -ivc tutorial tutorial.txt
```

7.0.8 wc

`wc` (word count)

A handy little utility is the `wc` command, short for word count. To do a word count on tutorial.txt, type:

```
wc -w tutorial.txt
```

To find out how many lines the file has, type:

```
wc -l tutorial.txt
```

7.0.9 Summary:

`cat` file display a file
`more` file display a file a page at a time
`head` file display the first few lines of a file
`tail` file display the last few lines of a file
`grep 'keyword'` file search a file for keywords
`wc` file count number of lines/words/characters in file

8 Further Reading:

I will eventually get these good documents incorprated into this document. But at this stage they will remain links:

Here is a good intro into regular expresions, sed and and find:

`http://www.grymoire.com/Unix/Regular.html`

`http://www.grymoire.com/Unix/Sed.html`

`http://www.grymoire.com/Unix/Find.html`

Note that they are old, but I have run through them myself and all work fine on linux.