Linux Cheat Sheet

Handy Command Line Tricks for Linux

1. Linux comes in several flavors. The following commands will help you determine which Linux distro is installed on your host, what's the version of your Linux kernel, the CPU model, processor speed, etc.

```
$ cat /etc/issue
$ cat /proc/version
$ cat /proc/cpuinfo
```

2. Find the total amount of RAM available on your Linux box and how much is free.

```
$ free -mto
```

3. The command cd.. takes you up one directory level but cd – will move you to the previous working directory. Or use the command pwd to print the full path of the current directory that you can copy-paste later into the shell.

```
$ cd -
$ pwd
```

4. The command *history* will show a list of all the recently executed commands and each will have an associated number. Use !<number> to execute that command again. Or, if the history is too long, use grep to search a particular command.

```
$ !<command number>
$ history | grep <some command name>
```

5. You can remove any particular command from the shell history by number.

```
$ history -d <command number>
```

6. If you made an error while typing a command name, just enter the correct command name and then use !* to reuse all the previous arguments.

```
$ <command> !*
```

7. Run the previous command but after replacing abc in the command with another string - xyz.

```
$ ^abc^xyz
```

8. This will list the size of all sub-folders of a directory in KB, MB or GB.

```
$ du -sh */
```

9. A better version of the ls command that displays file sizes in KB and MB.

```
$ ls -gho
```

10. You can use *man <command>* to learn more about the syntax of a command but what if you don't remember the name of the command itself? Use apropos then.

```
$ apropos <search phrase>
```

11. Compare the content of two text files to see what has changed.

```
$ diff wp-config.php wp-config.php.old
```

12. Find lines that are common in any two text files.

```
$ grep -Fx -f file-A.html file-B.html
```

13. Compare the content of two directories recursively.

```
$ diff –urp /old-wp-directory /new-wp-directory
```

14. Find all files under the current directory that are larger than 10 MB in size.

```
$ find . -size +10M -exec du -h {} \;
```

15. Find all files on the system that have been modified in the last 2 days.

```
$ find . –type f –mtime -2
```

16. Find all files on the system that were modified less than 10 minutes ago

```
$ find . –type f –mmin -10
```

17. Find all PHP files that contain a particular word or phrase.

```
$ find . -name "*.php" -exec grep -i -H "labnol" {} \;
```

18. When copying or moving files, Linux won't show a warning if you are overwriting an existing file. Therefore always use the –i switch to prevent overwrites.

```
$ cp –i abc.txt xyz.txt
```

19. Backup the content of the current folder into a tarball file using gzip compression.

```
$ tar zcfv backup.tar.gz /wp-directory/
```

20. Find processes with the highest CPU usage. Then use *kill –9 pid* to kill a process.

```
$ ps aux | sort -nrk 3 | head
```

21. Execute the following command in your Apache logs directory to determine hits coming from individual IP addresses.

```
$ cat access.log | awk '{print $1}' | sort | uniq -c | sort –n | tail
```

22. Monitor hits from Google bots to your website in real-time.

```
$ tail –f access.log | grep Googlebot
```

23. To find all files and web pages on your site that return a 404 error, run the following command in the Apache logs directory.

```
$ awk '$9 == 404 {print $7}' access.log | uniq -c | sort -rn | head
```

24. Find the 100 most popular pages of your site using Apache server logs again.

```
$ cat access.log | awk '{print $7}' |sort |uniq -c |sort -n |tail -n 100
```

25. Quickly find and replace a string in or more files.

```
$ find . -type f -name "*.php" -exec sed -i 's/wordpress/WordPress/' {} \;
```

The original article is available at labnol.org under Linux / WordPress.