Handbook

SNU Coding Club

September 30, 2022

Contents

1 Linux Cheat Sheet

1.1 The *very very useful ones*

1.1.1 `man`

- `man <command name>`
- The main command returns a helpful help page that gives you a brief description of what exactly a command does and how to use it.
- In case that man doesn't work, make sure that you have `mandocs` or `man` installed.
- The manual entries can also be accessed via a browser.

1.1.2 `--help` or `-h` flag

- Sometimes a command is too niche to warrant a page in the manual. In such cases you can use the help flags. There is no set standard but they're usually one of the two
- `<command> --help` or `<command> -h`
- If one doesn't work, try the other.

1.1.3 `sudo` and `su`

- `su` opens the current shell as root while `sudo` runs a specified command as root.
- In Linux root priviliges are very similiar to Adminstrator priviliges in Windows
- Be very careful when running anything as root. It can break your system.--
- It also requires the current user to know the root password as well be a part of the sudoers list

1.2 The Essentials

1.2.1 `cd`

- `cd` is used to traverse the filesystem from the terminal
- `cd <path>` will move the terminal to the defined path
- `cd ..` will move the terminal to the parent directory if it exists
- `cd /` will move the terminal to the root directory
- `cd ~` or just `cd` will move the terminal to the home directory

1.2.2 `ls`

- `ls` shows all the visible files and folders in directory.
- `ls -a` shows all the hidden and visible files and folders in the directory.
- `ls -A` shows **almost** all the hidden and visible files and folders in the directory(It excludes the `.` which loops back to the current directory and the `..` which points to the parent directory)
- `ls -l` displays all the visible files and folders and lists them in tabular form with some extra information(like size, author, date modified, etc)

1.2.3 `touch`

- `touch <filename>` creates an empty folder with the name specified

1.2.4 `mkdir`

- `mkdir <foldername>` creates a folder with the specified name in the current location of the terminal.

- `mkdir -p <path_to_folder>` creates the folder in the path specified as well as all the missing folders in the path to the folder.

1.2.5 `mv`

- `mv <source_path> <destination_path>` moves a file from the source_path to the destination_path
- `mv <path>/old_name <path>/new_name` will rename a file named `old_name` in the location `<path>` to `new_name`

1.2.6 `cp`

- `cp <source_path> <destination_path>` copies a file from the source_path to the destination_path
- `cp -r` is used to copy folders by recursively copying their contents

1.2.7 `rm`

- `rm <file1> <file2> <file3>` deletes the files mentioned
- **The Recycle Bin or Trash does not exist when it comes to deleting things from the terminal** so be careful with this command.
- `rm -f <file>` force deletes the file and overrides any warnings.
- `rm -r <folder>` recursively deletes the contents of a folder and finally deletes the folder too.
- **UNDER NO CIRCUMSTANCE SHOULD YOU RUN ANY OF THE FOLLOWING UNLESS YOU ARE ABSOLUTELY SURE ABOUT WHAT YOU ARE DOING**
  - `sudo rm -rf /` this will delete everything in your root directory
  - `rm -rf ~` this will delete everything in your home directory
- Running `rm -rf` with elevated priviliges in a dangerous location will most probably break your OS.
- Exercise caution.

1.2.8 `grep`

- `grep` is used to search the content of a specified file or durectory for a given string or a regex
- `grep '<search_term>' <file>` searches for the `search_term` in the `<file>`
- `grep -i '<search_term>' <file>` will search for the `search_term` in a case insensitive way.
- `grep -r '<search_term>' <folder_path>` will search for the `search_term` recursively within the specified directory.

1.2.9 `cat`

- `cat` allows the user to execute basic text modification from the terminal
- It is not a full blown editor like Vim or Emacs but it can read and append to files
- `cat <file>` will display the contents of the file
- `cat >> <file>` will allow you to enter some text into the terminal. The entered text is then appeneded to the end of the file.

1.3 The Extras

1.3.1 pwd

- `pwd` returns the path to the active directory

1.3.2 top

- `top` displays the processes running in real time. It also displays resource utilization and other information regarding the process.

1.3.3 pkill

- `pkill <pattern>` kills the first processes with the string `<pattern>` in their name

1.3.4 pgrep

- `pgrep <pattern>` returns the PID of all processes with the string `<pattern>` in their name

1.3.5 kill

- `kill <PID>` terminates the process with the PID specified

1.4 Useful for Bash Scripting

1.4.1 echo

- `echo "Hello, World!"` will print Hello, World! on the terminal.

1.4.2 wc

- `wc <file>` will print the number of words in a file
- `wc -l <file>` will print the number of lines in a file

1.4.3 Piping data

- We can pass the output of one command to the inpint of another by using the | operator.
- `grep -r '<search_term>' <folder_name> | wc -l` will return the number of entries that match the search