# Linux essentials writeup

## 0. Foreword

In order to pass my exams I kinda have to study but its just something I just don’t do.. I’m the type of person who doesn’t care about studying and just likes to do put everything into practical use and that’s what I’ll do for this writup of Linux Essentials.

In here I’ll write down all commands with screenshots, code and explanations in order for other people and myself to learn from it.

## 1. Working with GUI

### 1.0 Introduction

In here nothing really important was mentioned we mostly went over what Linux is and how to set up a virtual machine..

### What is Linux

#### 1.1.1 What Is Linux

In here its once again about what Linux is but there is some useful stuff to look at.

So obviously, Linux is an operating system like Windows, Android and MacOS.

Graphical user interface, application

Description automatically generated  
Linux comes from Unix, Unix is also an operating system. It supports multitasking and multi-user functionality. It has a graphical user interface just like Windows to support easy navigation and support.

Linux was build up through 3 factors:

* MINIX Operating system (Open source Unix like OS)
* GNU Project (Unix clone with bad kernel)
* Linus Torvalds (Some random IT student that wanted to flex)

#### 1.1.2 Linux Kernel

Linux on its own is actually only a Kernel while an Operating system like Ubuntu or Kali is a combination of Linux and extra software.

The Kernel is the core of an OS and is for example responsible for communicating with the hardware.

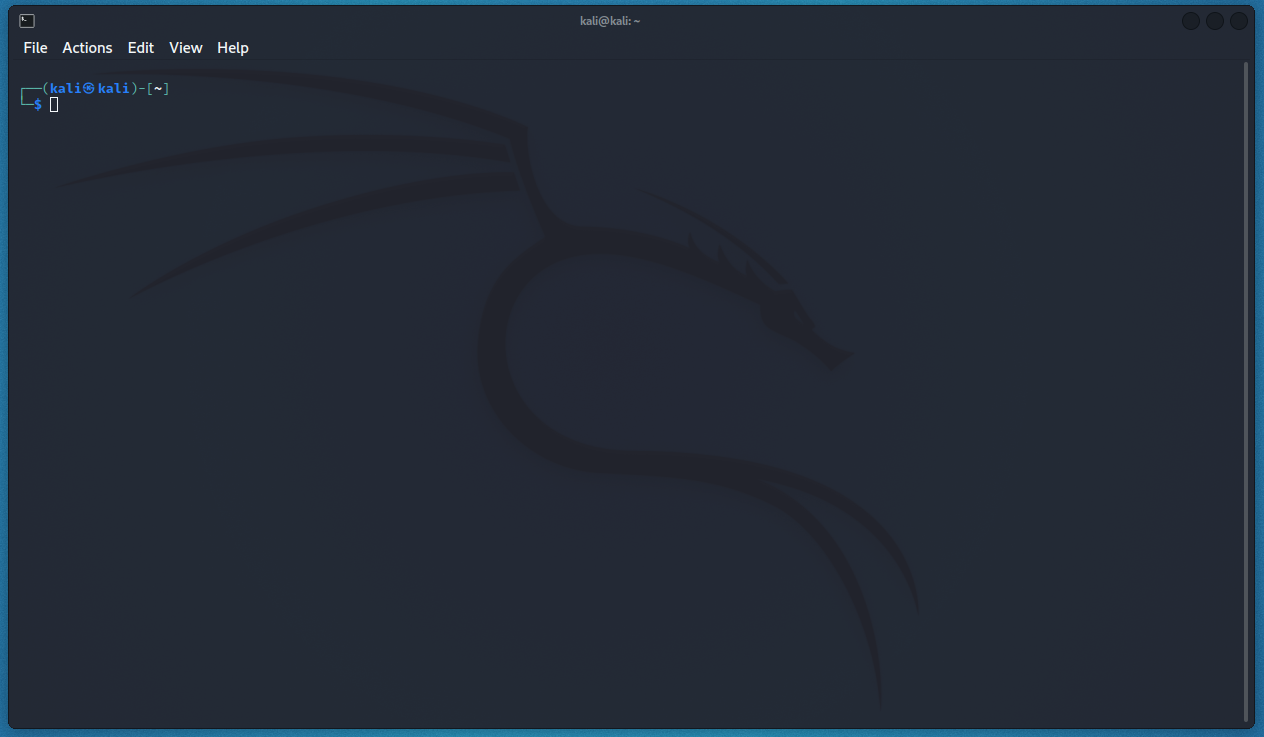
#### 1.1.3 Linux CLI

Linux CLI is a text interface, with it you have no need at all for a graphical interface. Using CLI is really powerful and its an important skill to master, OS like Arch Linux are completely build on only using the CLI and it gives its users a lot more power than you can get from a graphical interface.

Graphical user interface, diagram

Description automatically generated with medium confidence

CLI has 3 parts that should be pointed out right now.



Prompt

Shell

Terminal

Prompt: A piece of text given by the shell that tells when you can send a new command.

Shell: Shell is the text program that waits for text input that will handle all commands and programs.

Terminal: Application that shows the CLI.

We will mostly use Bash aka. Bourne Against Shell, this is also a CLI Shell but it also contains command language in which programming is possible.

## 2. Working with CLI

### 2.0 Special characters

Some characters are more difficult to find than others, on my Belgian keyboard it can be a pain in the ass so time to write those key combo’s down!

~ = ALT GR + ‘=’ + <Space>

` = ALT GR + ‘µ’ + <Space>

### 2.1 Man Pages

#### 2.1.1 What are Man Pages

We will first go over Man pages, this is short for Manual pages and in short we can use Man <command> To get an entire manual page of a command.

#### 2.1.2 Usage & Navigation

We can use Man ls to get the Man Page of the command ls.

A screenshot of a computer

Description automatically generated with medium confidence

Within the man page we can scroll but there are multiple other ways to navigate through the pages!

Within the man page we can use H to find all commands within the man page.

We can use the following things to navigate:

* **Space or f**: Move 1 page down
* d: Move down half a page
* d: Go back one page
* d: Go back half a page
* /: Jumps to the first word you type before you press enter after you press enter you can use
  + space or n to jump to the next
  + shift + n to move back
  + \btext\b to search for a specific word (when just searching for text it will also match with context, this wont)
* g: go to the first line of the man page
* Shift + g: go to the last line of the man page
* q: Quit the Man Page

#### 2.1.3 More info

Most Unix files have a man page, we will go over some with examples:

* Man <Command>: Man following a command will give out the manual with all variations | man ls
* Man <Config-File>: Most config files have its own manual page | man resolv.conf
* Man <daemon/root-binary>: man pages also exist for daemons | man system-networkd

We can also use man -k <string> to search through all Man Pages with the given string inside.

We can use whatis <path> to get a description | whatis route

We can use Whereis <path> to get the location of a Man Page | whereis passwd

There is also Man Man which will give the man page of man.  
In here we can see:

* 1 Executable programs or shell commands
* 2 System calls (functions provided by the kernel)
* 3 Library calls (functions within program libraries)
* 4 Special files (usually found in /dev)
* 5 File formats and conventions eg /etc/passwd
* 6 Games
* 7 Miscellaneous (including macro packages and conventions), e.g. man(7)
* 8 System administration commands (usually only for root)
* 9 Kernel routines [Non standard]

These are different sections within a man page.

We can search for a section in a man page by using man <sectionNr> <path> | man 5 passwd

### 2.2 Working With Directories

#### 2.2.1 What are Directories

In short: A Directory is a location for storing files on your computer. These sit in a hierarchical file system.

#### 2.2.2 Navigating through Directories

We can use pwd (Print Working Directory) to check the current directory you are in from the root (/) .

We use cd (Change Directory) to change our directory, when we use ls we can see to what Directory we can quickly change to |   
cd Downloads.

Cd has some special ways to path:

* ~ : Instantly go to the home directory | cd ~/Downloads
* .. : Move one folder up | cd ../Downloads
* . : Stay in the current directory | cd .
* - : Go to the directory you where previously in | cd -

Absolute and relative paths, to put it really simple, an absolute path always starts with ‘/’ in Linux, a relative never starts with ‘/’.

When looking for downloads from for example the home folder you will do:

* Absolute path: cd /home/<name>/Downloads
* Relative path: cd <name>/downloads

An Absolute path always starts from the root while a relative starts from the directory you are currently at.

We can see the current directories and files using ls inside of a directory.

Ls has some important arguments to remember:

* -a : This will show all files including hidden files | ls -a
* -1 : when adding -1 your folders will be listed vertically | ls -1
* -l : When adding -h the file sizes will be shown we add 1 to it to make the listing more visible | ls -l

#### 2.2.3 Tree

Tree is a package that isn’t on Linux by default, we can install this using sudo apt install tree more info about this later.

Now we can use the tree command and receive a directory tree, we can find in the man page many different parameters we can add to it using man tree.

In this example I made a tree of everything inside the home folder.

Text

Description automatically generated

#### 2.2.4 Create and remove directories

Obviously we can also create and remove directories

We can simply do mkdir <string> to create a directory with name | mkdir hello.

We can also use mkdir -p <string> to create a directory within parent directories, when adding -p mkdir will create the directories that don’t exist yet | mkdir -p ~/hello/hello/hello.

Text

Description automatically generated

You can delete empty directories using rmdir <path> |   
rmdir ~/hello/hello/hello

Text

Description automatically generated

You can also delete an entire path with rmdir -p <path> |   
rmdir -p hello/hello

Text

Description automatically generated

You can **not** add a ~ to your path cause with you tell rmdir to also remove your home folder like that!

### 2.3 Working With Files

#### 2.3.1 Files In Linux

Files in Linux are case sensitive do if you search for File1 by with the command cat file1 you won’t find it!

Its also important to know that basically everything on your OS is a file, even directories are some kind of special file and is still case sensitive!

If we want to see what type of file extension a file is we can use the file command | file /etc/passwd.

#### 2.3.2 File create, copy, remove and move

If we want to create a file e can simply do touch <string> in order to create one | touch hello.

You should keep in mind that touch can do this:

* -t : When you want to specify a time created instead of the current time | touch -t 201905050000 hi

If we want to remove a file e can simply do rm <path> in order to remove one | rm hello.

Rm has some interesting parameters to look at:

* -i : When adding -i to rm you will get asked for a yes or no if you really want to remove the file | rm -i hello
* .\* : Removes all hidden files \* stands for everything so this basically removes everything with a . in front of it | rm .\*
* {\*,.\*}: Removes all files and hidden files | rm {\*,.\*}
* -rf : This is a yeet away all, rm -r will not remove non-empty directories but when we add the -f parameter it will force it to also remove non-empty directories | rm -rf hello

In order to copy a file we can use the cp command | cp hi hi2:

* We can also copy a file into a directory with the same name as the target file | cp hi hiDirectory/
* -r : You can also copy an entire directory with -r |   
  cp -r dir1 dir2
* -i : when we add -i in front we can prevent cp from overriding any existing files | cp dir1 dir1BackupWhereFilesCantChange

We can Ofcorse also move and rename files for this we use the mv command with this we can chose a file and place it in a directory by doing mv <targetFile> <Directory> | mv file1 dir2/

What this does is delete file1 at 1 spot and placing it in another we can also use this to rename and simply say mv <targetFile> <newname> what this does is it will recreate the file at the current path, which is the same with a new name | mv file1 file2

There is also the rename command but it’s a little more tricky. It uses regular expressions but we will get into it. Imagine that we got a folder with .txt files and we want them to be .png files in order to do this we can do | rename ‘s/\.txt/.png/’ \*.txt

* First it does the rename command
* Then you specify .txt which is the thing you want to be changed | ‘s/\.txt
* Then you specify what you want to replace .txt with .png | /.png/’
* This happened between quotes to make sure its seen as one string | ‘s/\.txt/.png/’
* Then we specify our targets which is all files with .txt in the back | \*.txt