cybercademy

INTRODUCTION TO LINUX FOR CYBERSECURITY CRASH COURSE NOTES

TABLE OF CONTENTS:

INTRODUCTION AND NOTES FORMATTING

Introduction:

Having a foundational understanding of Linux is important when it comes to cybersecurity. Often times in the industry we hear our fellow professionals, colleagues, and our peers telling us to "Learn Linux" if you want to be successful in cybersecurity… which is true to some extent. Do you have to learn Linux to be in cybersecurity? No… Will it help you if you have a good understanding of what Linux is and how to use it? Yes! But before we transition into the specific uses Linux provides us for security uses, it's important to know the basics. In this crash course, we go over the basics of Linux with a security perspective in mind, however, most (if not all) of the content can be used for general purposes.

Here's a few key things to keep in mind when you are following along with these notes and learning:

- It's important to apply what you are learning.
  Use this crash course and these notes as momentum to start your own learning journey for Linux.
- Practice, practice, practice.
  Following along and completing this course is only part of the learning journey. You need to practice in order to retain what you have learned.
- Teach someone else!
  One way you can retain this information is by teaching someone else. It doesn't matter who it is (can even be yourself by making a video).

I hope this course and these notes can help assist you along your learning journey for cybersecurity 😊

Grant C.

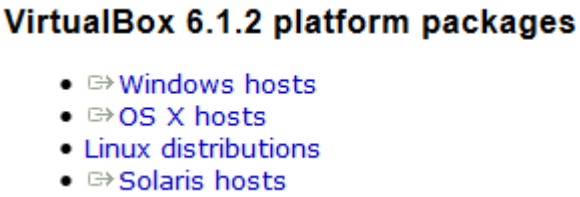
Notes Formatting:

Anything highlighted in the `Courier New` font is a command syntax.

You will commonly see the commands followed by the descriptions in tables.

INSTALLING ORACLE VM VIRTUALBOX MANAGER

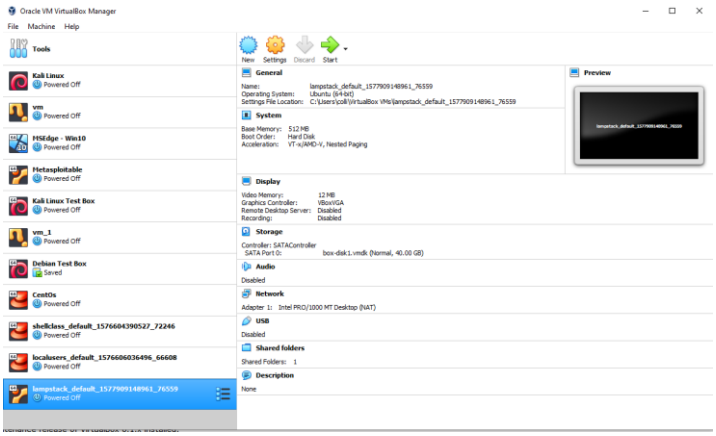
Step 1: Go to https://www.virtualbox.org/wiki/Downloads and choose the installation under platform packages.



Step 2: Follow the download instructions provided in the wizard.

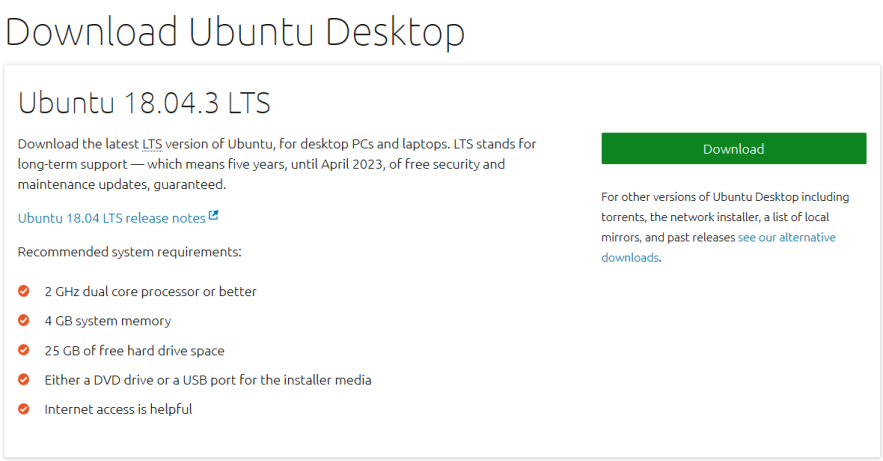
Step 3: Open VirtualBox from the desktop icon (if you allowed the shortcut) or search for VirtualBox on your Operating System.

Note: You will not see any of the following VMs if you are downloading a fresh install of VirtualBox.



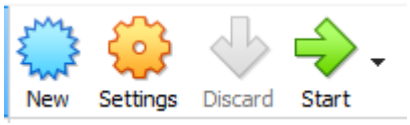
INSTALLING UBUNTU & MOUNTING UBUNTU TO NEW VM

Step 1: Go to https://ubuntu.com/download/desktop and click the Download button under Ubuntu Desktop.

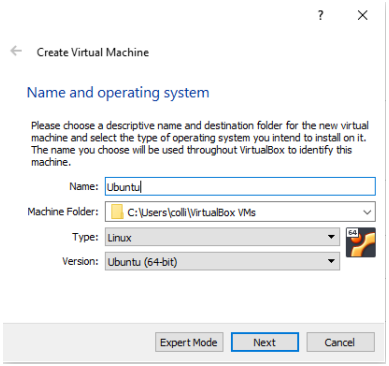


Step 2: Wait for the .iso installation to finish. After completion, make sure you know where to locate the file.

Step 3: Open VirtualBox and click *New*.



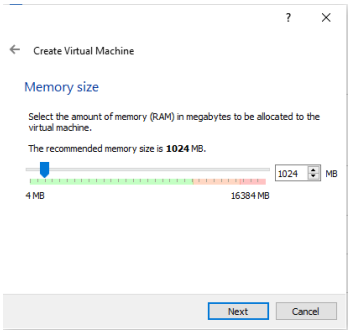
Step 4: Title the VM (here I am simply naming it Ubuntu). Make sure the Type and Version are Linux and 64-bit respectively.



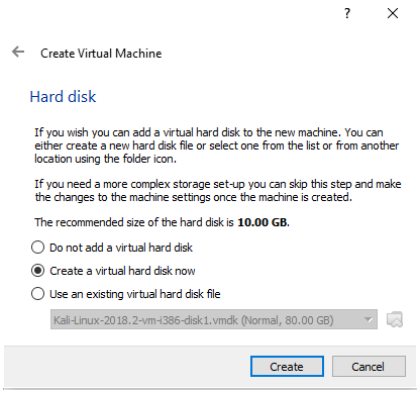
Step 5: Allocate memory size (also known as RAM) for VM.

Depending on how much RAM you have, you can increment the amount of RAM allocated in order to make this VM run more smoothly. Below is a simple table conversion from megabytes to gigabytes up to 6 GBs.

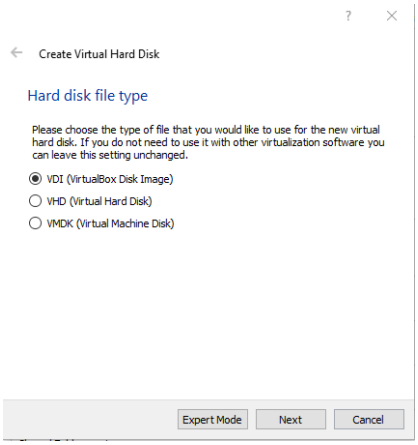
1024 MB	1GB
2048 MB	2 GB
3072 MB	3 GB
4096 MB	4 GB
5210 MB	5 GB
6144 MB	6 GB



Step 6: Choose *Create a virtual hard disk now* and click *Create*.



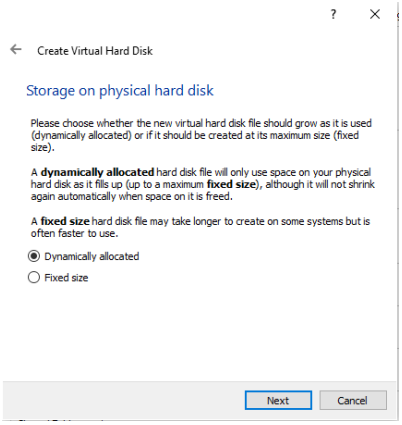
Step 7: Choose *VDI (VirtualBox Disk Image)* and click *Next*.



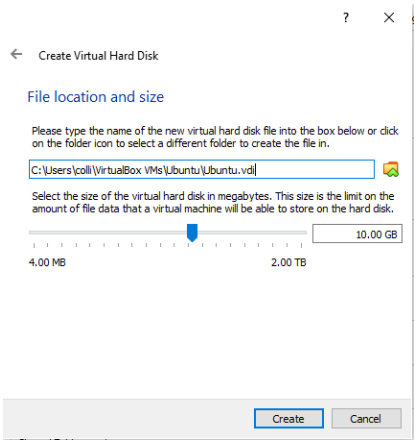
Step 8: Select what type of storage you want for the VM. I recommend dynamically allocated.

Dynamically allocated: Your disk size will increase as you create new files, directories, etc within the VM.

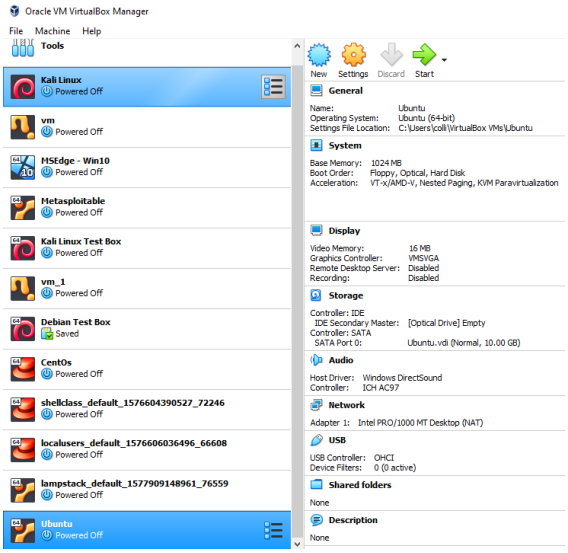
Fixed Size: The VM will not exceed designated size.



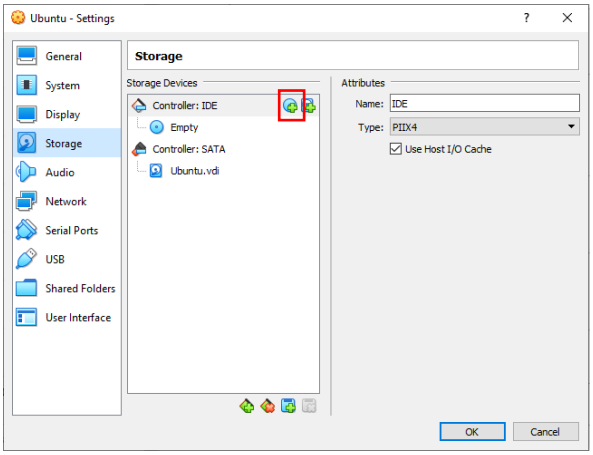
Step 9: Select initial amount of storage to be allocated to VM. I recommend starting out with the default (10 GB).

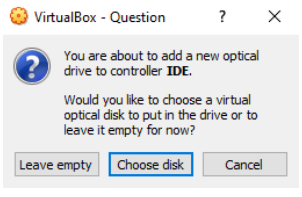


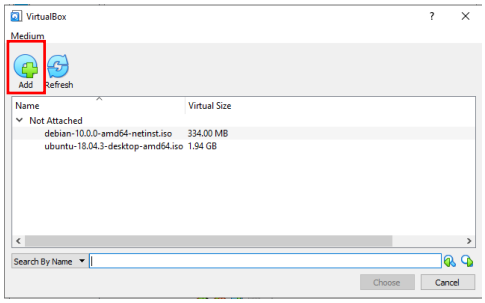
Step 10: Select the newly created VM and select *Settings*.



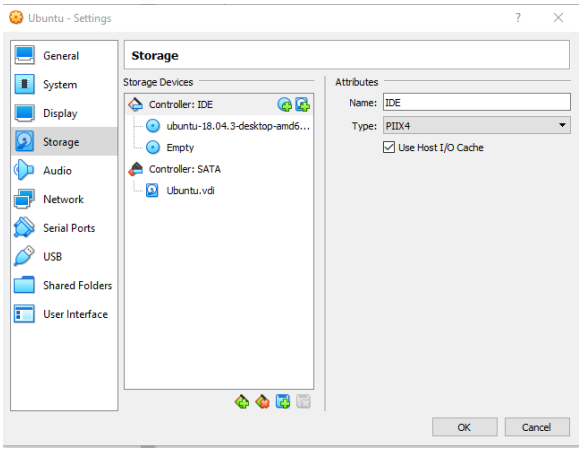
Step 11: Select *Storage* and click the circle disk icon (boxed in red below). Select choose disk and locate Ubuntu iso file.

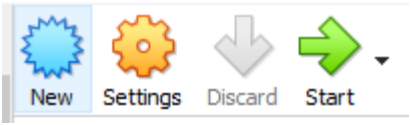






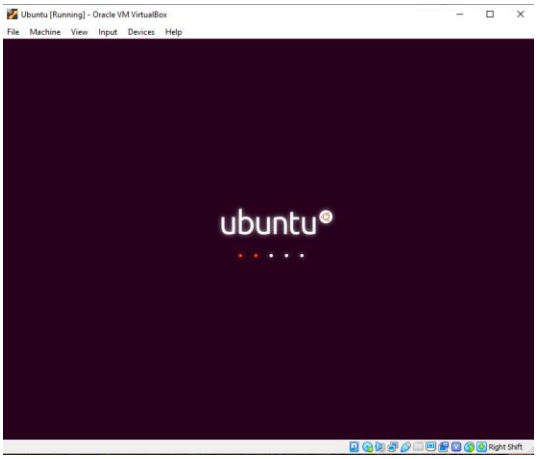
Step 12: Click *OK* and start the VM.

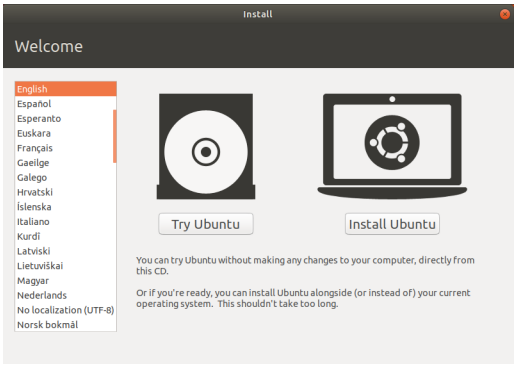




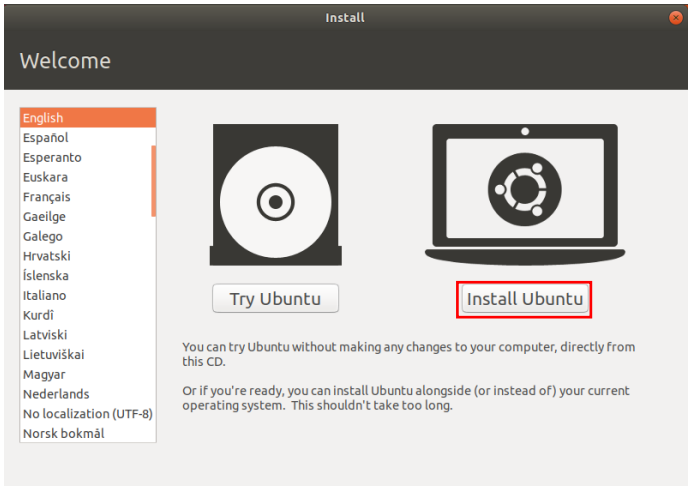
SETUP AND CONFIGURE UBUNTU

Step 1: Start the VM by pressing Start. A loading screen will appear, wait until the Welcome screen.



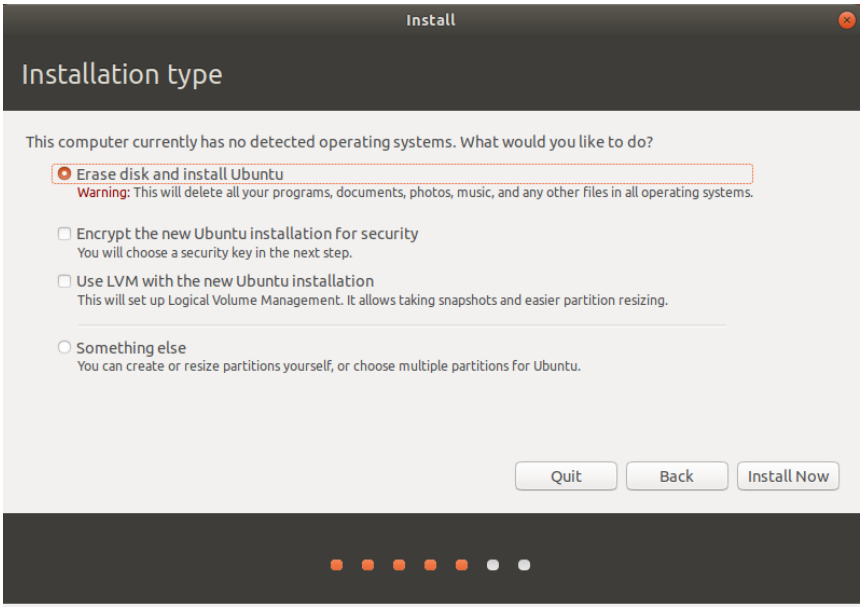


Step 2: Click Install Ubuntu and select language.



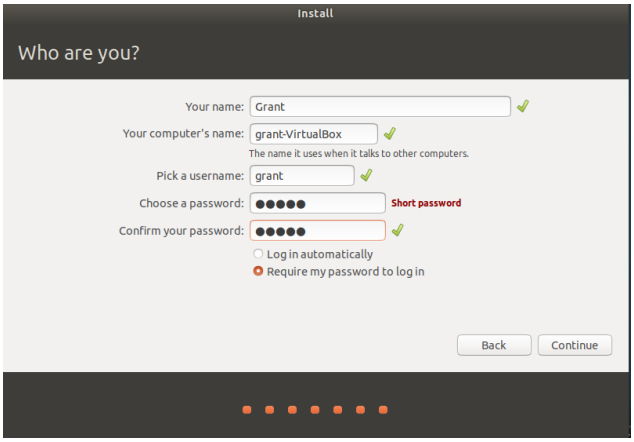
Step 3: Leave the selected defaults and click continue.

A warning will pop up after clicking Install Now, click continue.



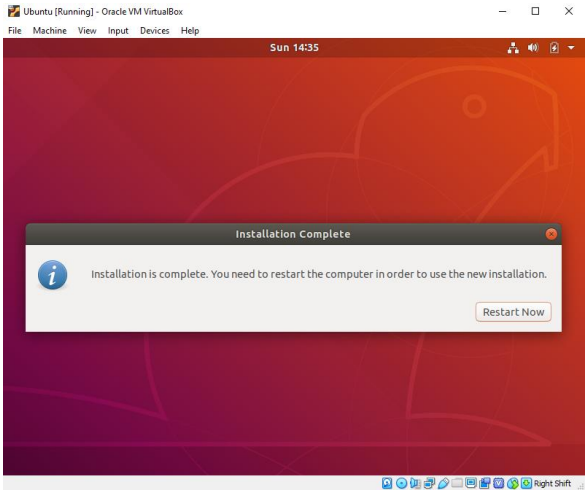
Step 4: Choose your time zone and click continue.

Step 5: Create a username and password, click Continue

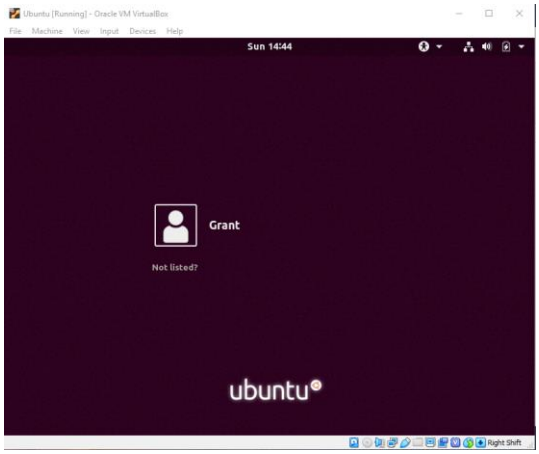


Allow Ubuntu to install.

Step 6: Click Restart Now after automatic installation has taken place.



After restart there will be a login screen for you to enter your password.

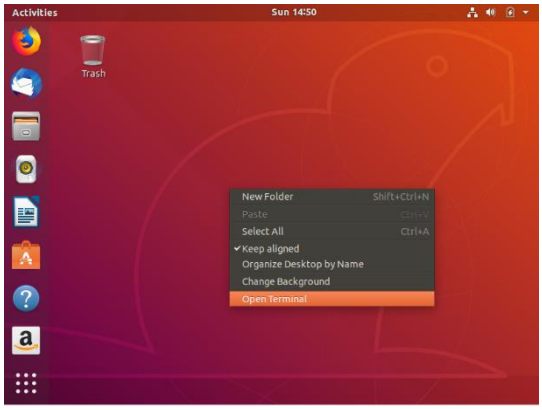


NAVIGATING TO THE TERMINAL & THE SHELL

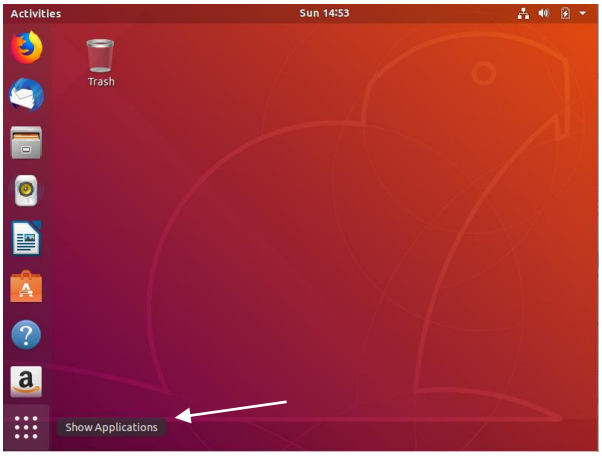
The command line, also known as the terminal, is what makes Linux powerful. The power of Linux resides in how well you can use and navigate the terminal – you will be using this almost every time you want to do something internally within the OS.

There are a few methods to open a new terminal.

Method One: Right click the desktop and select Open Terminal



Method Two: Navigate to Show Applications and type in “terminal”



The shell is nothing more than a program which accepts your commands and executes those commands. Also known as a command line interpreter.

The command line is more powerful and will always be here… you will use the command line to configure routers, firewalls, etc

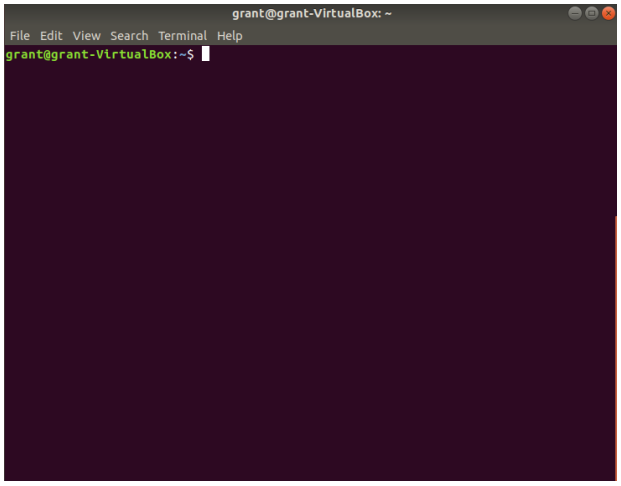
User permissions:

$	Denotes Regular User
#	Denotes Superuser

Anything can be done by root, whereas normal users have a limited amount of permissions.

By convention, only execute commands with root (or sudo) so you do not have to switch to a root account. Day to day activities should be performed using a normal account.

The shell

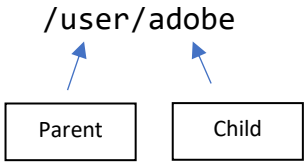


LINUX DIRECTORIES: MOST COMMON

Linux directories are commonly known as folders to the computer user. In Linux, we typically call these directories. There are many default directories in Linux, listed below are a few of the most common ones:

`/`	Known as the root directory
`/bin`	Binaries and other executable programs reside here
`/etc`	System configuration files
`/home`	Home directories
`/opt`	Optional or third-party software
`/tmp`	Temporary space, typically cleared on reboot
`/usr`	User related programs
`/var`	Variable data, most notably log files

Linux directories can have subdirectories which reside underneath the parent directory.



BASIC LINUX COMMANDS: COMMANDS YOU USE ON THE DAILY

When navigating through the terminal, there quite a few commands you can perform. Below are the most common commands you will use when navigating through the terminal.

`ls`	List directory contents
`cd[dir]`	Changes current directory to dir
`pwd`	Displays the present working directory name. If you don't know what directory you are in do pwd
`cat [file]`	View contents of a file
`echo [argument]`	Displays argument to screen
`man`	Displays the online manual for a command
`exit, logout, or CTRL-D`	Exists the shell or current session
`clear`	Clears screen
`touch`	Creates or updates a file

LEARN MORE ABOUT A COMMAND USING HELP & MAN

There are a few ways to learn more about a command. You can use the "man" page which stands for manual. There are also help pages for commands which exist as a shell builtins.

To find out if a command is a shell builtin, simply type:

```
type -a [command]
```

Note: [command] is where you would type the command, for example `type -a ls`

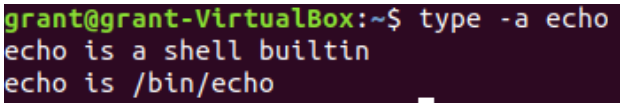
If a command is a shell builtin, then you can use the help command.

```
help [command]
```

If a command is not a shell builtin, use the man page.

```
man [command]
```

In this example, we can use both the help or man commands



When using the man pages, you can navigate around the page using the following commands below:

`Enter`	Move down a line
`Space`	Move down one page
`g`	Move to the top of the page
`G`	Move to the bottom of the page
`q`	Quit

WORKING WITH DIRECTORIES

Directories are simply containers for files and other directories. They provide a tree like structure for organizing the system.

Here are the most common directory commands:

`.`	This directory
`..`	Parent directory
`/`	Directory separator
`cd -` or `cd`	Takes you to previous directory
`mkdir [-p]`	Make new directory ([-p] parents option)
`rmdir [-p]`	Remove **empty** directories ([p] parents option)
`rm -rf directory`	Removes all directories and files - use with caution
`./command`	Execute command in this directory

LISTING DIRECTORIES

ls shows the output of files contents

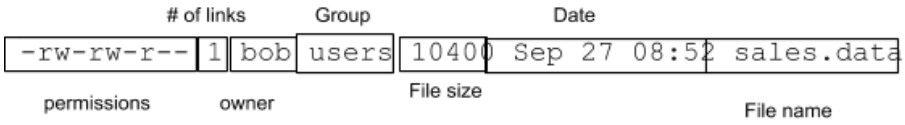
Here are some options you can use for ls:

`-a`	All files, included hidden files
`--color`	List files with colorized output
`-d`	List directory names and not their contents
`-l`	Long format
`-r`	Reverse order
`-R`	List files recursively
`-t`	Sort by time, most recent modified first
`-latr`	Long listing with time (most recent files displaying on screen)

To use these options, simply type in ls followed by the option (also known as a flag)

```
-ls -l
```

When typing in the ls -l option, the output will be a long listing format of the contents of each file. Here is an example of the long listing format and what each section means:



FILE AND DIRECTORY PERMISSIONS

```
-rw-r--r--
```

First character reveals file type:

-	Regular file
d	Directory
l	Symbolic Link

Types of permissions strings:

r	Read
w	Write
x	Execute

Permissions on files act as read, write, and execute.
Permissions on directories:

- Read: Allow file names to be read
- Write: Allows entries to be modified within the directory
- Execute: Allows access to contents and metadata for entries in the directory

Permissions Categories for Users:

u	User (user who owns the file)
g	Group (users that are in the file's group)
o	Other (users who do not own or are not in files group)

a	All - user, group, and other

Every user is a member of at least one group, called the primary group.
To find what group you are apart of use: groups or `id -Gn`

Hyphen (-) denotes user does not have the privilege

CHANGING FILE PERMISSIONS

The `chmod` command is used to change permissions on the categories (user, group, and other)

ugoa	u for user, g for group, o for other, a for all
+-=	+ to add permission, - to subtract, = set all
rwx	r for read, w for write, x for execute

Example of adding write permissions to group category: `chmod g+w file`

Example of adding write permissions to user & group category: `chmod ug+w file`

Example of adding read permissions to all categories: `chmod a=r file`

Changing File Permission (Octal Mode)

Represented in binary. 0 means off, 1 means on → must convert to base 10

Commonly used permissions (octal mode)

700	Files can be read, written, and executed by owner
755	Allows everyone on the system to execute, but only owner can modify
664	Allows a group of people to modify the file and let others read it
660	Allows a group of people to modify the file and not let others read it
644	Allows everyone on the system to read the file but only the owner can edit it

777 & 666 should be avoided

By default you are placed into your primary group. If you want to change your group, use:

```
chgrp group_name file
```

FINDING FILES AND DIRECTORIES

Use the `find` keyword to locate a file or directory.

```
find [path…] [expression]
```

`find . -name pattern`	Displays file names which matches pattern (case sensitive)
`find . -iname pattern`	Same as -name, but ignores case
`find . -ls`	Performs an ls on each of the found files or directories
`find . -mtime num_days`	Finds files that are num of days old
`find . -size num`	Find files that are size of num
`find . -newer file`	Finds files that are newer than file
`find . -exec command {} \;`	Run command against all the files that are found

In addition to find, you can also use the locate pattern keyword. Locate is indexed by updateb each day, so it may not reap the most accurate results

VIEWING AND EDITING FILES

Commands to display the contents of files to the screen:

`cat file`	Display entire file contents
`more file`	Browse through file, use spacebar to advance
`less file`	Similar to more but backward movements
`head file`	Output the beginning of the file
`tail file`	Output the ending of the file

cat does not update in real-time; tail & head are real-time updates to view a file

To view a file in real-time:

```
tail -f file
```

There are multiple editors you can choose when it comes to creating and editing files. The well-known editors in Linux are Nano, Vim, and Emacs…

Nano

- Very simple editor (not as complex as vi or emacs)
- To view a file in nano: nano file.txt
- To save your edits: CTRL-o
- To quit: CTRL-x

Vim

To edit a file in vi: `vi [file]` or `vim [file]` or `view [file]`

Vi has an updated version, Vim which standards for "VI Improved"

There are three modes in Vim:

- Command Mode: allows you to navigate the file, perform searches, delete, copy, paste text, etc
  - Enter command mode `[esc]`
- Insert Mode: insert, edit text
  - Enter Insert mode `[i, I, a, A]`
- Line Mode: perform specific types of saves, quit, etc
  - Enter Line mode `[:]`

Vi Command Cheat Sheet: [Link]

Editing Files with Emacs

Emacs is similar to VIM

To edit a file in emacs: `emacs [file]`

C-<char> → CTRL while pressing <char>
M-<char> → Meta Key or ALT key

Emacs Command Cheat Sheet: [Link]

DELETING, COPYING, MOVING, AND RENAMING FILES

Removing files:

`rm file`	Remove file
`rm -r dir`	Remove the directory and its contents
`rm -f file`	Force removal and never prompt for confirmation

Copying Files:

`cp source_file destination_file`	Copy source_file to destination_file
`cp src_file1 [...] dest_dir`	Copy source_files to destination_directory
`cp -r dir dest_dir`	Copy dir to dest_dir recursively

Moving Files:

`mv`	Move or rename files and directories
`mv source destination`	Move file from source to destination
`mv -i source destination`	Move interactive mode

Renaming Files:

The mv command can be used for renaming files.

```
mv filename1.txt filename2.txt
```

ARCHIVING AND COMPRESSING FILES

Creating a Collection of Files to Archive and easily transfer:

```
tar -czvf name-of-archive.tar.gz /path/to/directory-or-file
```

`tar`	Create, extract, or list contents of a tar archive using pattern if supplied
`tar -cf`	Create and name archive
`tar -v`	Enable verbose mode, see the contents being archived
`tar -z`	Enable gzip to tar

Compressing Files To Save Space:

`gzip`	Compress file
`gunzip`	Uncomlpress file
`gzcat`	Concatenates compressed files
`zcat`	Concatenates compressed files

INPUT AND OUTPUT REDIRECTION

Input, Output, and Redirection

- Three I/O Types
  - Standard Input / stdin / 0
  - Standard Output / stdout / 1
  - Standard Error / stderr / 2

Redirection:

`>`	Redirects standard output to a file
`>>`	Redirects standard output to a file. Appends to any existing content
`<`	Redirects input from a file to a command
`&`	Used with redirection to signal that a file descriptor is being used
`2>&1`	Combine stderr and standard output
`2>file`	Redirect standard error to a file
`>/dev/null`	Redirect output to a trash can

Why do we need to learn input and output redirection?

- Input / Output Redirection allows you to combine multiple commands together in one line, making it more efficient to use

SEARCHING FOR PATTERNS IN FILES

Use Grep to find text within a file:

`Grep pattern file_n`	Search for a pattern in a file
`grep -v pattern file_n`	Invert Match. Return files that does not match
`grep -i`	Perform a search, ignoring case
`grep -c`	Count the number of occurrences in a file
`grep -n`	Precede output with line numbers from file

PIPES

The Pipe (|) means take the standard output from the preceding command and pass it as the standard input to the following command. If the first command displays error messages those will not be passed to the second command.

- You can also chain pipe commands

`command-output | command-input`
Example: `strings BlueTrain.mp3 | grep -i john | head -1`
Common use of grep and piping: `cat file | grep pattern or grep pattern file`

ENVIRONMENT VARIABLES

Environment Variables are storage locations that have a name and a value.

If you know the name of an environment variable use: `echo $VARIABLE_NAME`

- If you don't know, use: env or printenv

To set environment variables on start up, use: `export NAME="command"`

- Example:
  - `PAGER=less`
  - `TZ="US/Central"`

To unset environment variables on start up, use: `unset $NAME`

PROCESS CONTROL

Process Commands

`ps`	Display process status
`ps -e`	Displays everything
`ps -f`	Full format listing
`ps -u username`	Display processes running as username
`ps -p pid`	Display process information pid. A PID is a process ID
`ps -ef`	Display all processes
`ps -eH`	Display a process tree
`ps -e --forest`	Display a process tree
`ps -u username`	Display processes running as username
`top`	Interactive process viewer

CREATING AND SWITCHING USERS

One way to start a session as another user on a system is to use the su command.

`su [username]` → change user ID or become superuser

Common su Commands:
`su -` → Used to provide an environment similar to what the user would expect had the user logged in directly
`-c command` → Specify a command to be executed.

`whoami` → Displays the effective username

You can also use sudo to execute a command as another user, but you do not need to know the password of the other user.

Sudo Commands:

`sudo -l`	List available commands
`sudo command`	Run command as superuser
`sudo -u root command`	Sudo Command
`sudo -u user command`	Run command as user
`sudo su`	Switch to the superuser account
`sudo su -`	Switch to the superuser account
`sudo su - username`	Switch to the username account

To create a user, use: `useradd username`

To change user account: `sudo su - grant`

INSTALLING SOFTWARE

When you install software on a Linux system, you do so with a package. A package is a collection of files that make up an application.

- Package managers manage dependencies

Installing Software on CentOS, Fedora, and RedHat

Use yum command

`yum search search-string`	Search for search string
`yum install [-y] package`	Install package
`yum remove package`	Remove/uninstall package
`yum info [package]`	Display information about package

You can use the `rpm` command to interact with the package manager

Installing Software on Debian and Ubuntu

Ubuntu and Debian use Advanced Packaging Tool:

`apt-cache search search-string`	Search for search-string
`apt-get install [y] package`	Install package
`Apt-get remove package`	Remove/uninstall package
`apt-get purge package`	Remove/uninstall package
`apt-cache show package`	Display information about package

You can use the `dpkg` command to interact with the package manager

WORKING WITH IP TABLES

Linux Firewall

- Firewalls control network access
- The Linux Firewall is comprised of Netfilter + IPTables
- Netfilter: A kernel framework
- IPTables: A packet selection system
- Use **iptables** command to control the firewall

IPTables Structure

IPTables is comprised of Tables, Chains, and Rules.

There are 5 Default Tables

1. Filter * : most commonly used
2. NAT
3. Mangle
4. Raw
5. Security

There are 5 Default Chains

1. INPUT
2. OUTPUT
3. FORWARD
4. PREROUTING → allows altering of packets before sending to INPUT chain
5. POSTROUTING → allows altering of packets after they exit the OUTPUT Chain

Rules in IPTables

- Rules can be applied in IPTables: Rules = Match + Target
- Match on
    - Protocol
    - Source/Destination IP or network
    - Source/Destination Port
    - NIC

5 Default Targets

- ACCEPT
- DROP → will not notify sender that packet was rejected
- REJECT
- LOG
- RETURN

iptables / ip6tables: iptables handles IPv4 and ip6tables handles IPv6

IPTables Basic Commands

List / View

`iptables -L`	Display filter table
`iptables -t nat -L`	Display nat table
`iptables -nL`	Display using numeric output
`iptables -vL`	Display using verbose output
`iptables --line-numbers -L`	Use line numbers

Chain Policy / Default Targets

`iptables -P` *`CHAIN TARGET`*	Set default target for chain

Appending, Inserting, and Deleting Rules

`iptables -A` *`CHAIN RULE`*	Append a rule to a specific chain
`iptables -I` *`CHAIN RULE`*	Insert a rule into a chain
`iptables -D` *`CHAIN RULE`*	Delete Rule in a specific chain

Flushing (Deleting) all Rules

`iptables -t` *`table`* `-F` *`chain`*

Rule Specification Options

`-s SOURCE`	Source IP address
`-d DESTINATION`	Destination IP
`-p PROTOCOL`	Protocol
`-m MODULE`	Modules options (man iptables-extensions)

Example of IPTables Rules

```
iptables -A INPUT -p tcp --dport 22 -j DROP
```

Append to the INPUT Chain, to protocol TCP with a destination port of 22 and drop all of those packets

```
iptables -I INPUT -p tcp --dport 80 -j REJECT
```

Append to the INPUT Chain, to protocol TCP with a destination port of 80 and drop all of those packets

```
iptables -I INPUT -p tcp --dport 80 -m limit --limit 50/min --limit-
burst 200
```

Append to the INPUT Chain, to protocol TCP with a destination port of 80 and add a limit of 50 packets per minute

Saving Rules

By default, all IPTables rules will reset when the Linux system has reset. To save rules, install iptables-persistent (apt-get install iptables-persistent) on Debian / Ubuntu and save by using **netfilter-persistent save** and install iptables-services on CentOS / RedHat (yum install iptables-services) and save by using **service iptables save** command.