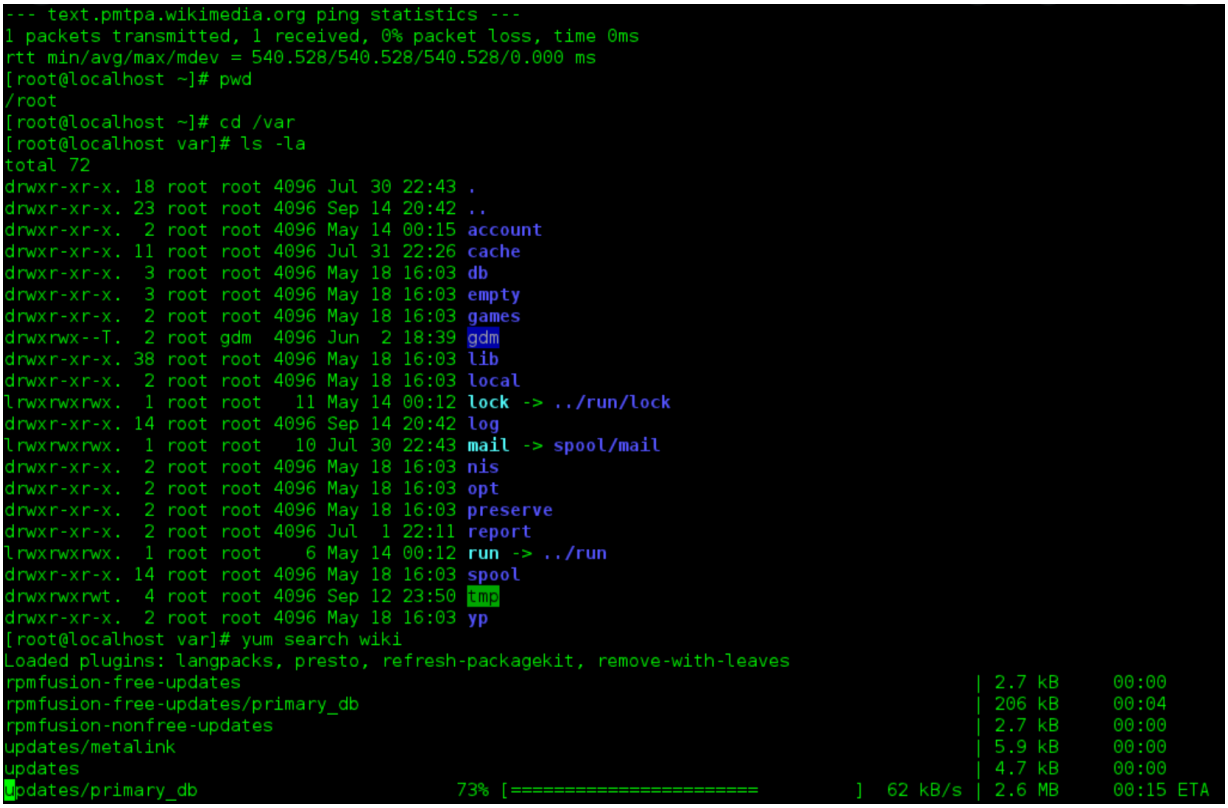
How to use the CLI (Command Line Interface)

What is the CLI??



The command line is an interface that allows a user to directly interface with the computer's directories and files. The CLI or shell is a terminal that allows for the use of developer commands, access to run programs, manipulate files, interact with other

computers and so on.

The CLI gives you more control over your machine than a GUI if you know how to use it. However, you don't need to become a whiz to use it. You can use the CLI even you only know how to use a few commands.

There are multiple different CLI program that can be used. Here are just a few.

Bash
https://www.gnu.org/software/bash/
This is the default for unix systems. Linux and Mac OS both run this by default.

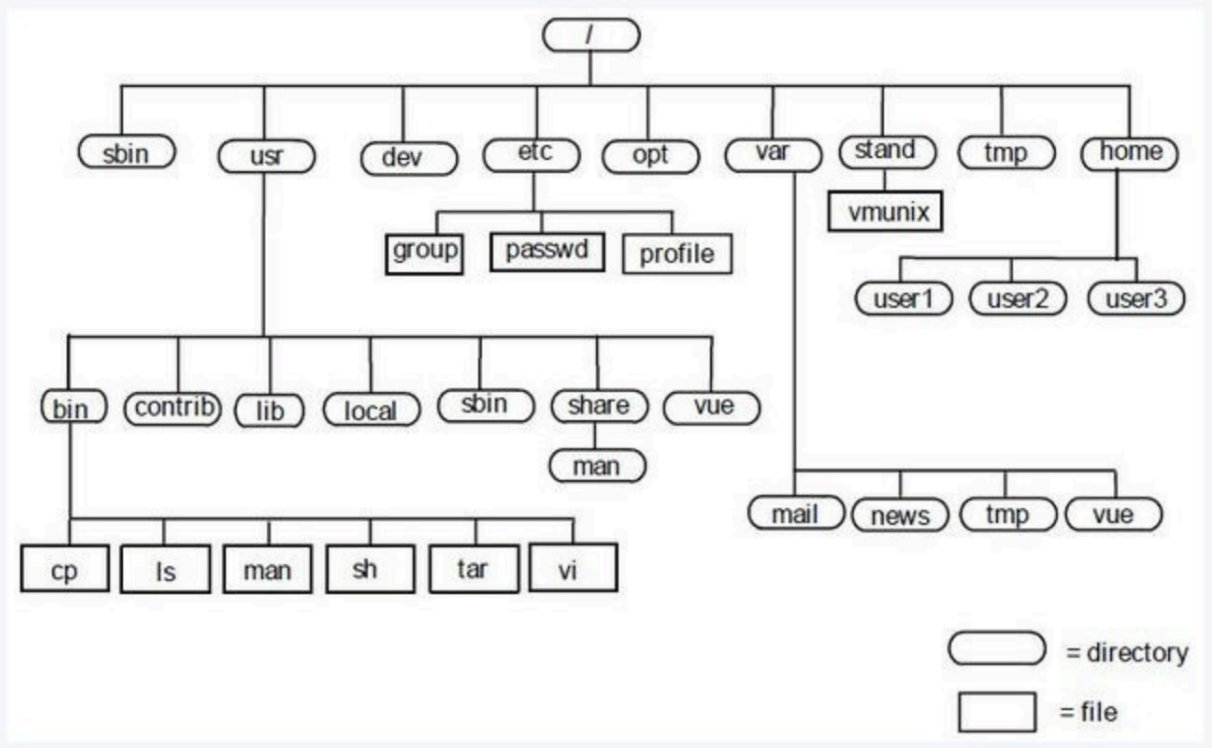
ZSH
https://www.zsh.org/
Depending on what OS you are using this may be preinstalled. To check (and use it) type "zsh" into the command line. ZSH is a shell but it can also be used for scripting.

OhMyZSH
https://ohmyz.sh/
OhMyZSH is an add-on to ZSH and provides a lot of plugins and visual queues to ZSH.

Navigation

When using the command line interface. The first thing you need to understand is how to navigate around in the computer. To begin to understand this, we must 1st understand how a computer's file system works.

Think of a computer's file system like a tree. Every tree has a root. The root is where the tree begins. A computer's file system is no different. It has a root and that location is represented by a /



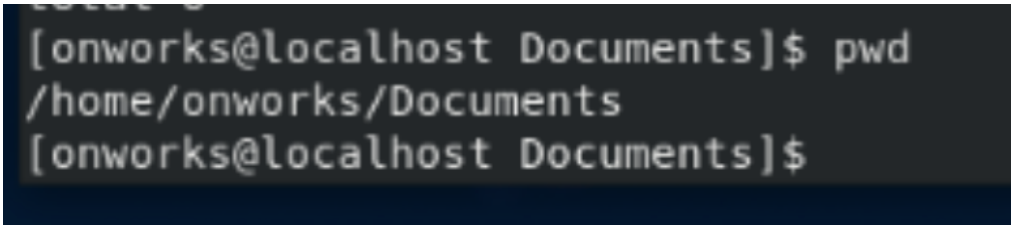
With this being the case, to navigate around in command line we have to move from folder to folder, up and down in this file tree. Closer and further to and from the root.

pwd - print working directory

To see where we currently are in the file tree of the computer, we can use the pwd command. You will use this a lot as well unless you use OhMyZSH with the location plugin.

Example:

pwd - this will show us the current bread crumb of where we are. Below shows us we are in /home/onworks/Documents



cd - changedirectory

We can do this using the “cd” command. You will be using this command a lot as it is the way to move about your file system.

Examples:

cd - on its own will bring you to the root computer

cd ~ - will also bring you to the root of the

computer

cd ~username - will bring you to the folder of that specific user

cd . - will bring you to the current folder you are in

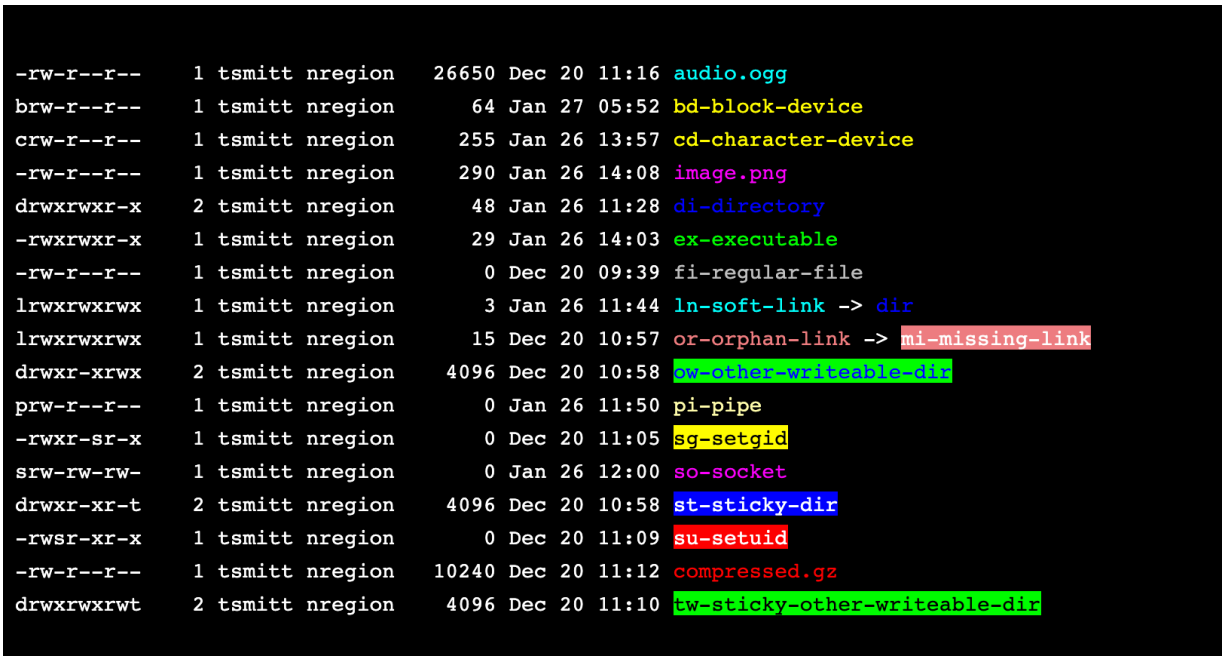
cd .. will bring you up one folder

cd foldername - will bring you into the folder in question

cd foldername/folder - will bring you into a specific location

(Note: the command chdir does the same exact thing as cd)

ls - list files and directories (dir in windows)



This command is used to list the files and directories in the current folder. This is also a command you will be using a lot and there are some optional arguments that can be used when using this command. The default sorting is alphabetically.

Examples:

ls - will list the files and folders in the current dir. However, this command will NOT display hidden files (.somefile).

ls -a - will list ALL files within a folder including hidden files.

la - an alias of ls -a

l - an alias of la

ls -l - will display the files and folders in long format line by line.

ls -R - will display items and items within sub folders.

ls -t - sort by when modified.

ls -u - sort by last time file was accessed.

Creating / Deleting / Moving Folders and Files

mkdir - makedirectory

This is used to make a folder.

Examples:

mkdir nameoffolder - this will make the new folder named “nameoffolder” in this case.

mkdir folder1 folder2 - this will make 2 folders that are siblings, one named “folder1” and the other named “folder2”.

touch - create a (blank) file

To create a file we can use the touch command and give the file a name and file extension (the kind of file we want it to be). This will make us a file that has no content, we could then open it up and add content to it in some program.

Examples:

touch index.html - this will make us an HTML file named “index” (index.html).

touch index.html about.html contact.html - this will make us a bunch of blank files at the same time. The files index.html about.html and contact.html will all be created in the same folder at the same time.

cat - create files / View the contents of a file / concatenate files

The command cat comes from the word concatenate which is to connect or chain 2 things together. We can use the cat command to view the contents of files or to concatenate files together. We can also use cat to make a file and concatenate the contents into the file.

Examples:

cat index.html - this will show the content of

the file in the CLI.

cat > index.html <h1> Hello!!</h1> - this will create a file named “index.html” and it will insert <h1> Hello!!</h1> into the file. (Control + D to exit)

cat index.html text.txt > newfile.html - this will concatenate the contents of index.html and text.txt into a new file called newfile.html

cat file1.txt > file2.txt - Copy the contents of file1.txt into file2.txt
Creating files with

mv - move a file (like cut and paste)

The mv command is how we can move a file from one place to another.

Examples:

mv index.html public_html - this will move

the file index.html into the folder “public_html”. Note that the folder must be created already. If it is not created already then the mv command will assume you are trying to rename the file index.html as public_html. Also note that if public_html was a file that already existed, it would be overwriten.

mv website public_html - if “website” is a file or a directory, move the file or directory into the folder named “public_html”. If the folder “public_html” does not exist, it will rename the folder “website” as “public_html”.

mv index.html ../ - This will move the file up one directory.

cp - copy a file or a directory (like copy and paste)

If you want to copy something. We can do

that with the cp command.

Examples:

cp index.html about.html - this will copy the file “index.html” and name the copy “about.html”.

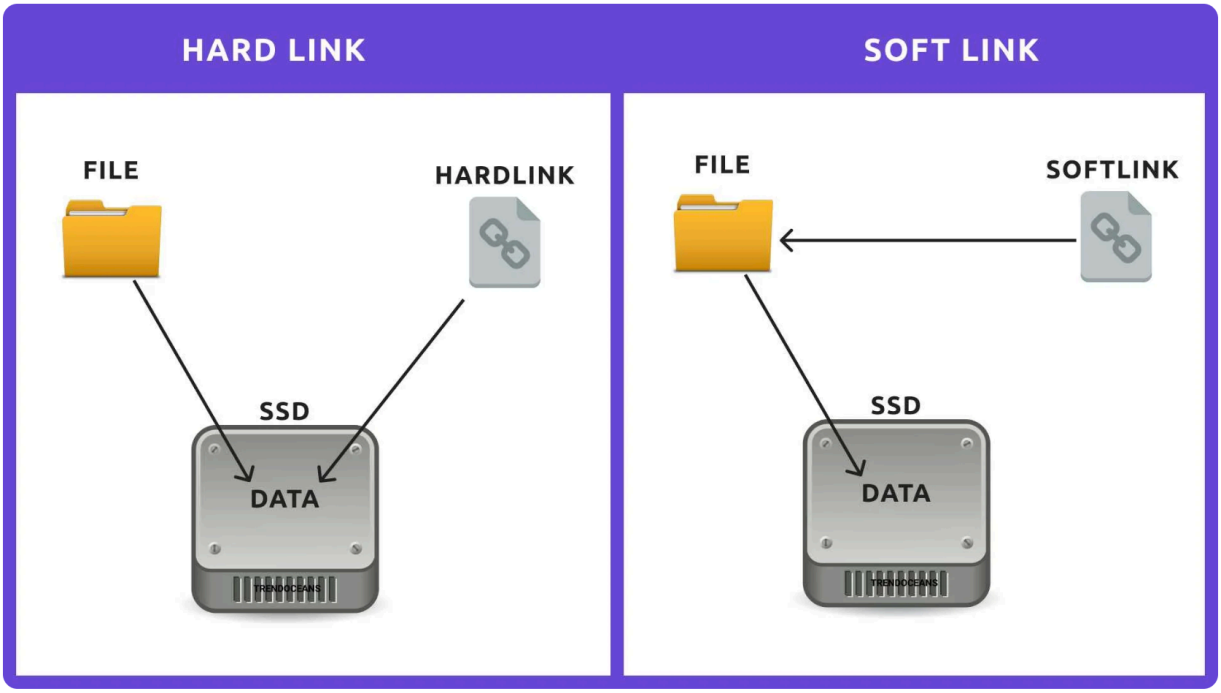
More examples here: https://www.computerhope.com/unix/ucp.htm

link or ln - Hard and soft links

There are 2 different types of links:

Soft (symbolic) links: ln -s

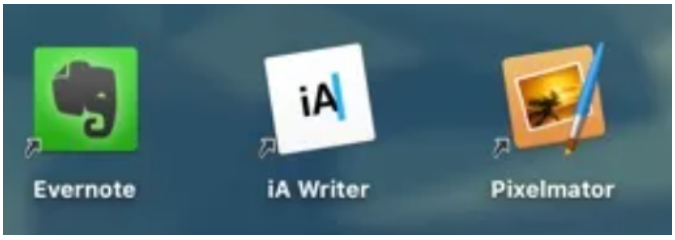
Hard links: ln



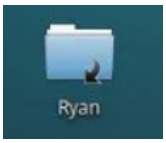
A soft link is like a shortcut. Ever see a some logo like this? in:

Windows

Mac OS



Linux



Those are soft links. Think of it like making a new file that points to another file. The shortcut will have its own inode number which will point to another file that has its own inode number which points to the actual data.

A hard link creates a link between 2 files or like having 2 different names for the exact same file. The hard link is a bit harder to understand and to explain this, we have to understand how data is saved on your computer.

Basically, in unix systems, files have:

An i-node table (which contains metadata of the file)

The actual data blocks (which is the actual data that is the file stored on the hard drive)

The I-node table consists of things like:

The chmod of the file (who can read, write, execute the file)

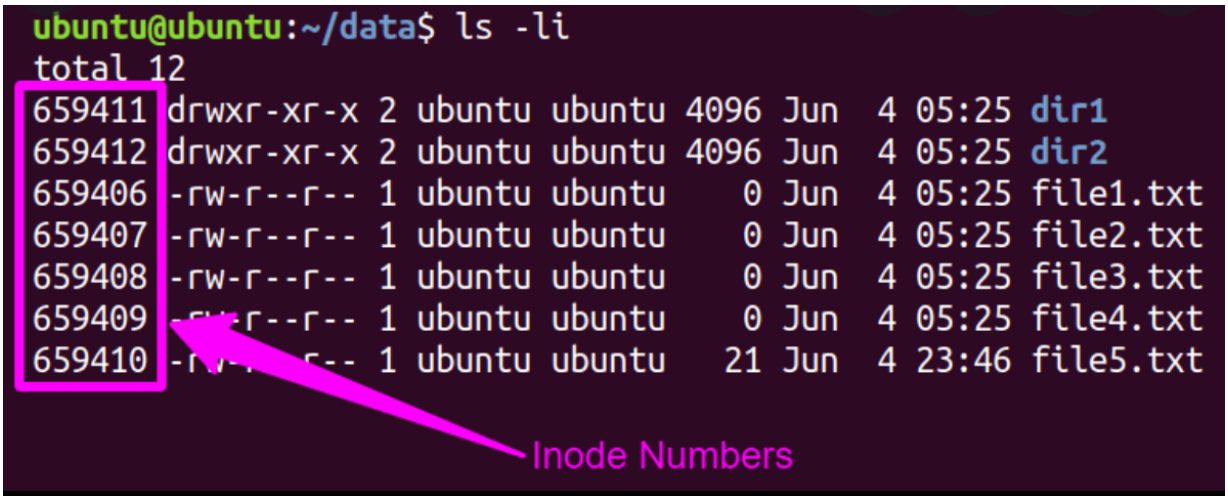
The owners of the file

The file type

The size of the file

The time stamp of the file (create and modification date/time)

The inode number which is the data blocks



With this being said:

The inode table points to where the file is in the data blocks (this is why when you delete something on your HD it's not actually deleted. The action of deleting a file just unlinks the inode table from the data block and thus, lets the computer know that it's ok to overwrite that data and in time the data gets overwritten but can technically be retrieved until it is).

So when we create a hard link, we are (to put it simply) making multiple files that point to the exact same data set.

We can make these in unix systems using the ln command.

Examples:

ln sourcefile.html hardlinkfile.html.html - this will hard link the “sourcefile.html” to the new link file “hardlinkfile.html”

ln -f sourcefile.html hardlinkfile.html - this will do the same as the above but it will unlink the file named “hardlinkfile.html” to its original link (from when it was created) so that it can be linked to the file “sourcefile.html” instead. The -f argument is what does this.

```
 96B Oct 11 10:15 dir1
 64B Oct 11 10:11 dir2
230B Oct 12 00:22 hardlinkfile.html
 15B Oct 12 00:19 shortcut.html -> sourcefile.html
230B Oct 12 00:22 sourcefile.html
```

ln -s sourcefile.html shortcut.html - this will

make a shortcut file named “shortcut.html” that will point to the file “sourcefile.html”.

rm - remove (delete) a file irreversable

The rm command deletes a file or directory. When using the rm command on its own, it is not reversible. So BE CAREFUL!

(May have to use the -d argument to delete a directory)

Examples:

rm index.html - This will remove a file.

rm index.html -f - This will remove a file and force it to delete bypassing any prompt.

rm index.html -i - (lower case i) Prompt before every removal. (Safer than using rm

on its own).

rm index.html -I - (uppercase I) Prompt once before removing more than 3 files.

rm index.html -r or -R - Remove directories and its contents recursively.

Note: there is a more safe way to delete stuff. Instead of using rm we can mv it to the tmp folder, in this case it will be deleted on next reboot:

mv file.txt /tmp

Documentation, help, history and clear

There are some things to remember that will help along the way:

man

The man command will bring up the manual

for a command.

Examples:

man ls - This will bring up the manual for the ls command.

man rm - This will bring up the manual for the rm command.

Press q to quit.

help or -h or —h

Any time you need help, type a command and **-h / —h / -help** and one of these will bring up some help for it.

```
└─[$]> curl -help
Usage: curl [options...] <url>
     --abstract-unix-socket <path> Connect via abstract Unix domain socket
     --alt-svc <file name> Enable alt-svc with this cache file
     --anyauth       Pick any authentication method
 -a, --append        Append to target file when uploading
```

History

Any time you want to view the history just type **history** and it will bring up a few 1000 lines of CLI history

```
2354  git clone https://github.com/AzimsTech/Android_Hacking.git
2355  ls
```

Clear

To clear off the screen use the **clear** command

Closing

There are many other things we can do with the CLI. The things discussed in this lecture is only scratching the surface. However, the things in this lecture are the fundamentals and should be practiced regularly. You may not remember all of these commands, or they may become a sort of faint memory on how to use them depending on how often you use them. But I find that the CLI is sort of like riding a bike. Once you learn the basics it's

easy to get back into it and do more. It is also easy to expand your abilities to use.

Don’t feel nervous about not remembering everything off by heart. It is a regular practice to look things up real quick to refresh your memory. Not everyone has a brain like a computer! Just keep trying things and have fun.