

Linux is a general-purpose computer operating system, originally released in 1991 by Linus Torvalds. Linux is defined by its kernel, which is the core component of the system. The kernel interacts with the computer hardware to allow software and other hardware to exchange information. Linux was inspired by MINIX which, in turn, was inspired by UNIX. And Linux is based on a philosophy that software and operating systems should be free. Both, free of cost and freely modifiable.

The software license which allows this, in the case of the Linux kernel, is called the GNU General Public License.



While the Linux kernel is the same across nearly all these installations of Linux, the software that surrounds the kernel that provides capabilities like software package management, control of services, and the location of configuration files differs between them. Many of the tools that come packaged with Linux come from the GNU Project and aren't a part of Linux. And so, taken together, the combination of the kernel and these common tools is often referred to as GNU Linux. Different groups of software and configuration choices that are maintained by individuals or groups of people are called distributions, or distro's.

Most major distributions of Linux fall into four categories based on the original distribution from which they were derived. There's **Arch, Debian, Red Hat, and Slackware**, and any number of other smaller distributions.

Linux Mint, Ubuntu, Elementary OS, and Kali Linux are all derived from Debian.

CentOS, Fedora, and Red Hat Enterprise Linux are derived from Red Hat.

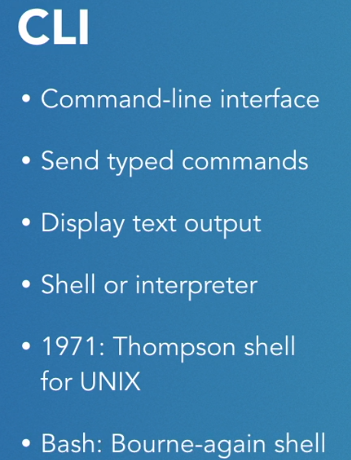


**To install git:**

sudo apt install git

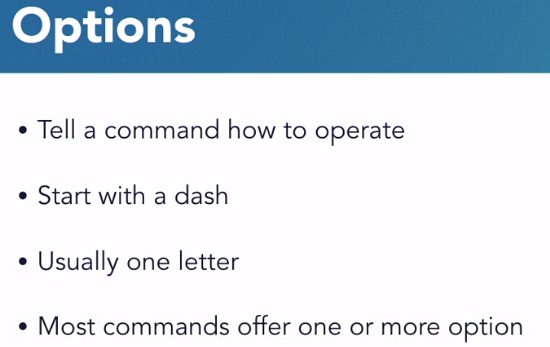
**To clone a repository from git:**

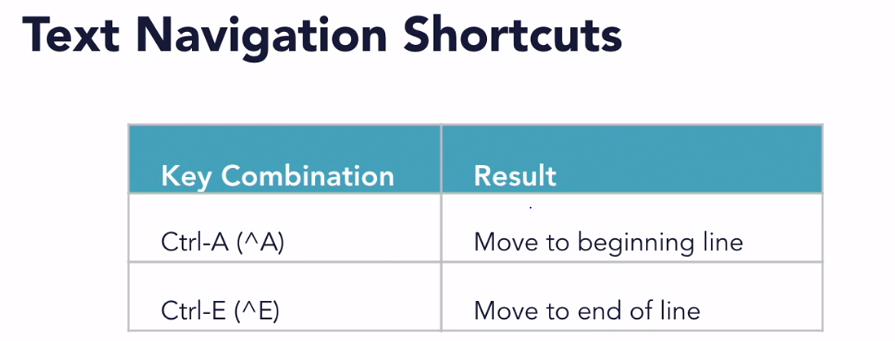
Example: git clone https://github.com /scottsimpson/commandlinebasics



Command is the program that we are running



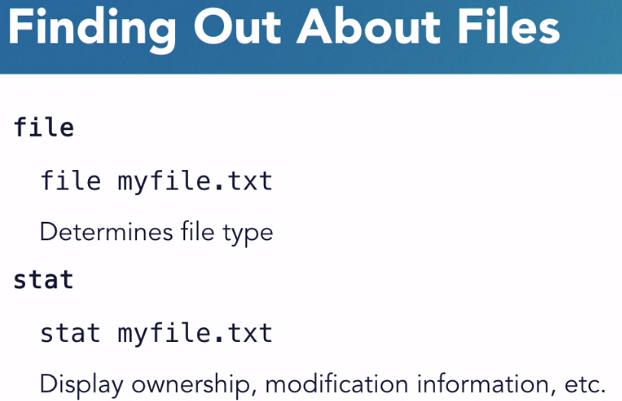




If we don’t know any command. We can type **apropos** to find the details of all command

Ex: apropos “list”

All commands with description containing list.



p-w-d, for print working directory

To escape a character **“\”** is used.

**ls -R folder\_name**: Recursively print the subfolder details inside folder\_name

**cd -**: Previously used folder. So, we ca switch back from current and last used folder

**ls -lh**: To know the size of the file

**mkdir -p folder1/folder2/folder3**: It will create nested folder. Within current folder first it will create folder1, inside that folder2 and inside that folder3.

**Copy a file**

cp poems.txt poems2.txt

**Move a file:**

mv folder1\file1.txt folder2\file2.txt

**Rename a file:**

mv file1.txt file2.txt

**Single dot (.) means current directory. To move a file from others directory to current directory:**



\*: Any number of character

?: Single character

**To move all files ending with .txt to other folder:**



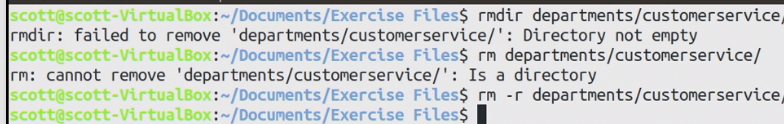
**To move everything from other folders to current folder:**



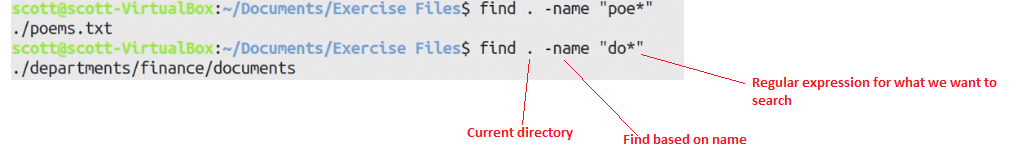
**Delete the files using? Exactly 1 character.**

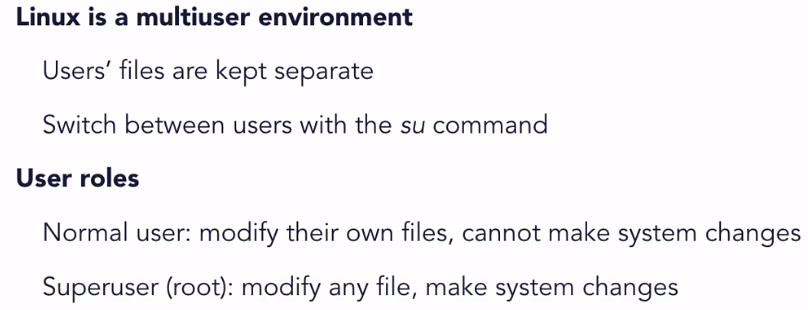


**To delete directory within a directory:**



**Find Files from the command line**





At the command line, we can switch between users with the **su** command, which is variously referred to as **set user**, **switch user**, or **substitute user**. To use su, we write that command followed by the name of the user we want to switch to.

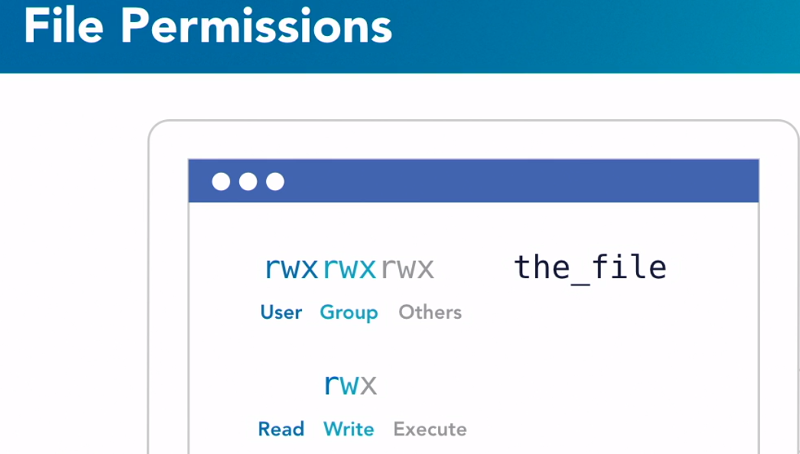
Probably the most common use of switching users at the command line is to do some system administration tasks.

There are two basic user roles in Linux. There's the normal user and the superuser. The difference here is one of privilege.

The normal user can modify, create, delete, and move their own files, but they can't make changes to the system. They can't install software, they can't make changes to system files, and, they can't browse other users' home folders. The superuser, which is called root, can make changes to the system.

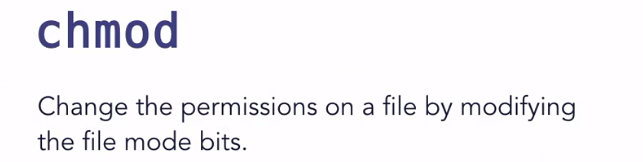
It can install software, it can start and stop services, and so on. Normal users can be granted the ability to temporarily use root's power through a command called sudo. It's uncommon and its bad practice to log into the root user directly to do normal work

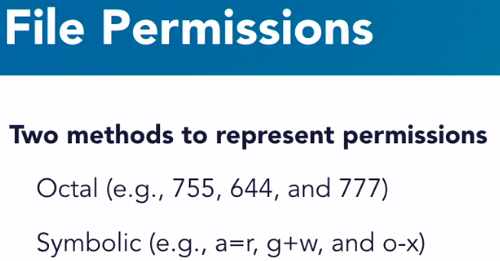
**Sudo tells the system to run whatever command is after it with superuser privileges instead of the normal user's privileges.**



The sequence of letters breaks down into three groups. The first represents the **user**, or the owner of the file. The second group of three represents the **group** that owns the file, and the third group represents **all other users not in the group** that owns the file.

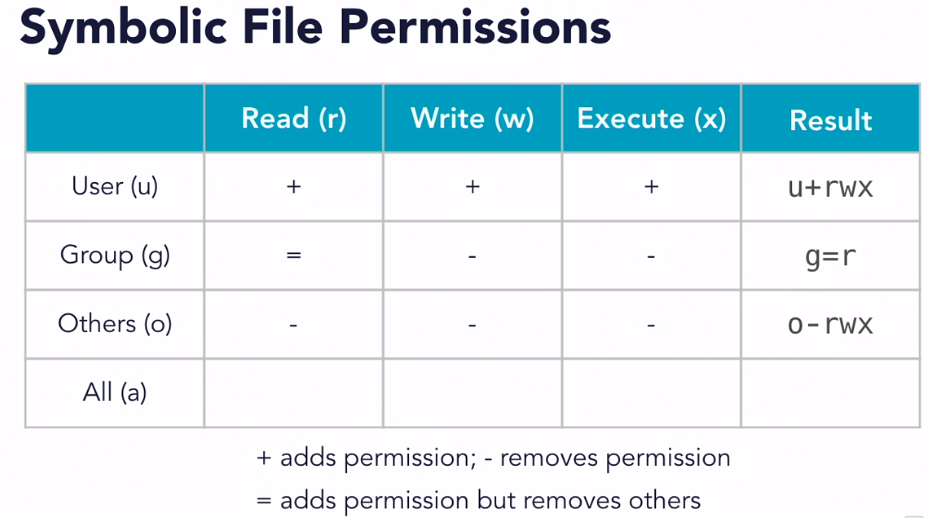
Each of the groups of three breaks down into three individual letters, which stand for Read, Write, and execute.

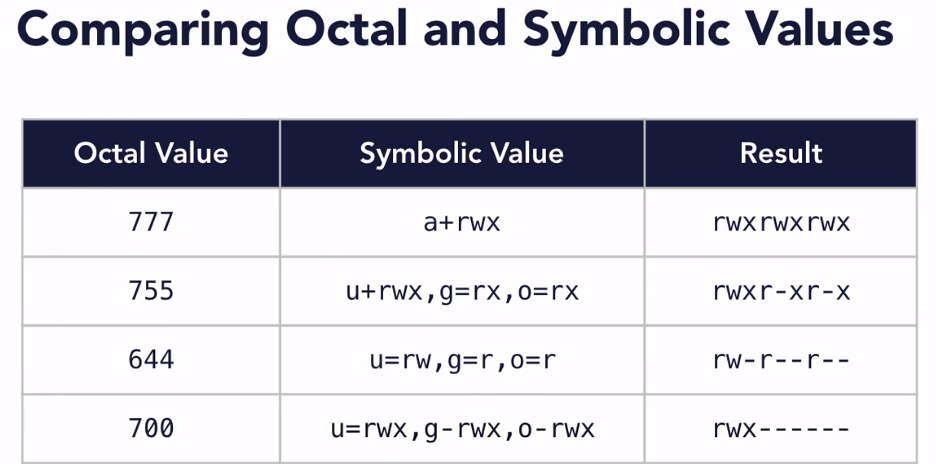


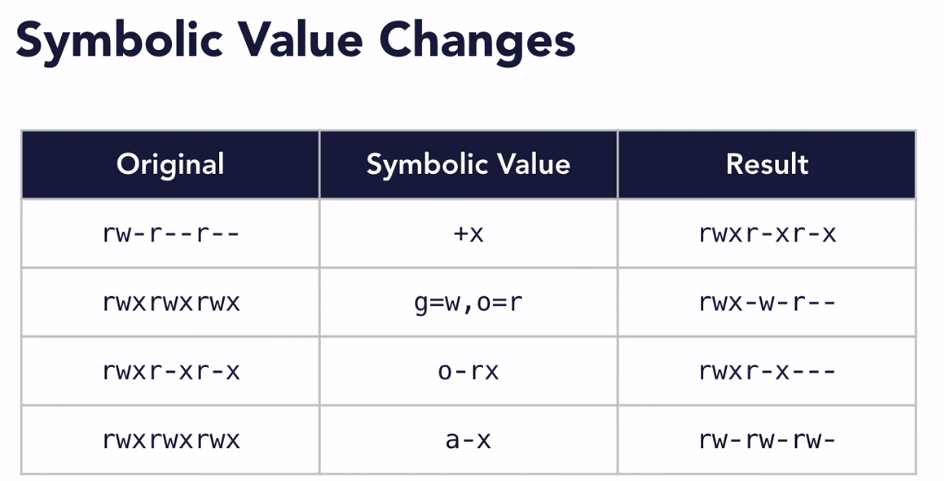


We can change the permissions of a file using the chmod command. Chmod changes the file mode bits on a file, and we can do that in two ways. The first is to use an octal notation, which uses three values to represent read, write, and execute





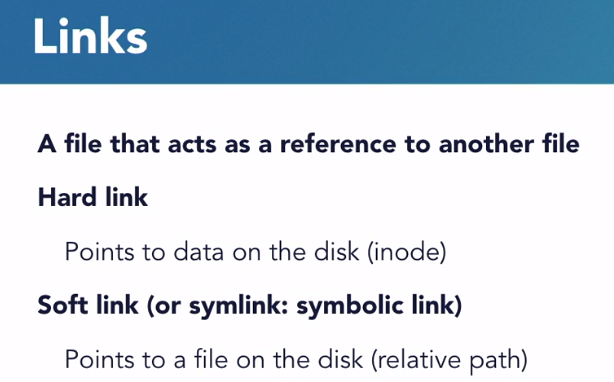




When a user creates a file in their home directory, it starts out with a permission mode of 644

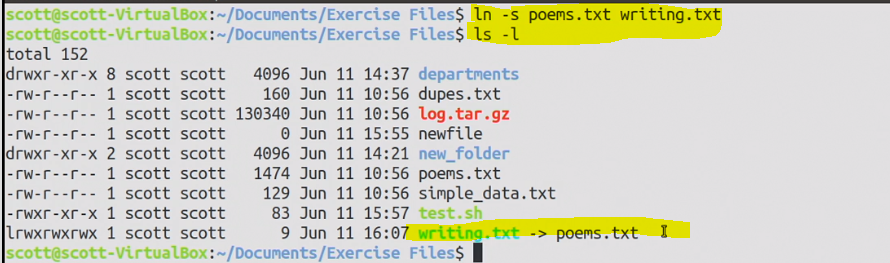
To change ownership to root user:

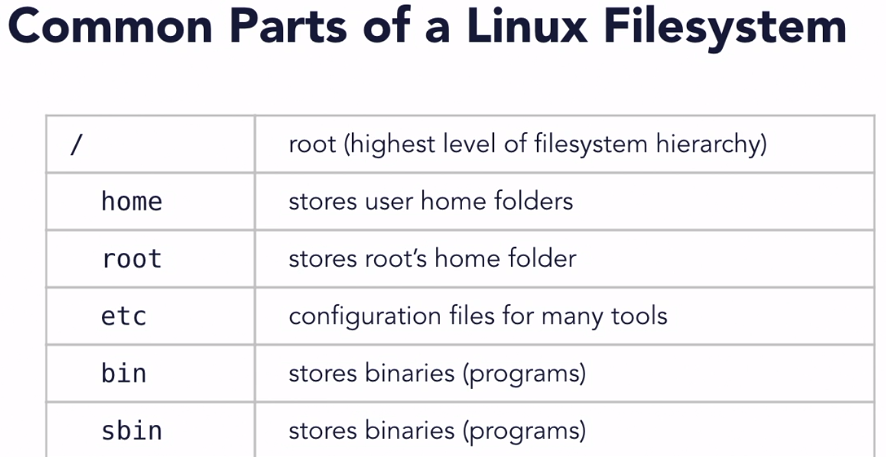




Links are files that are references to other files, and they're used to avoid having multiple copies of the same file in different places. You keep one file in a well-known location and then add a little pointer or a link to other places you want that file to appear to be.

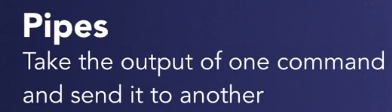
**Creating a Soft link:**

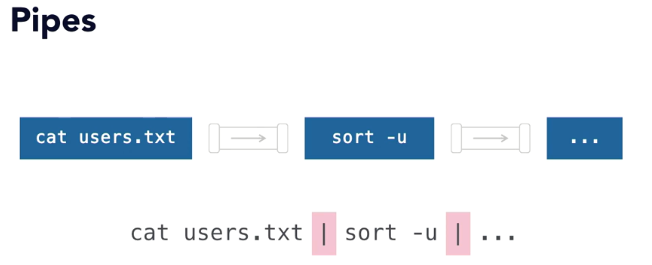




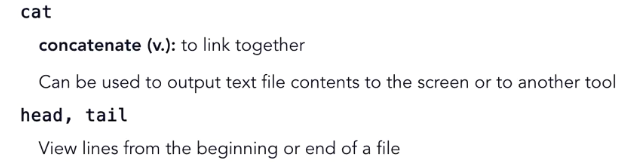


Linux Part 2:

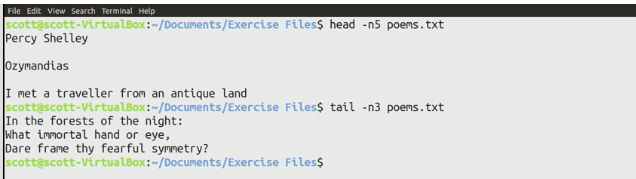




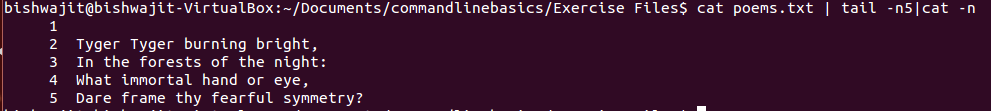




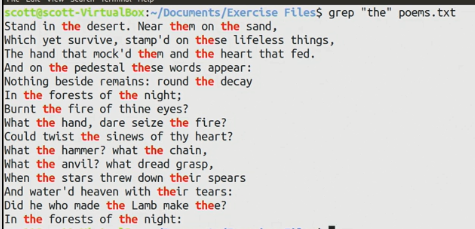
Ex:



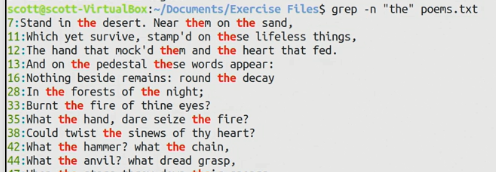
**Using Pipe display the last 5 lines of a file:**



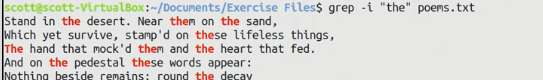




Number the row number

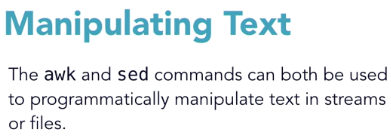


-i case insensitive



**-v** omits the matching line

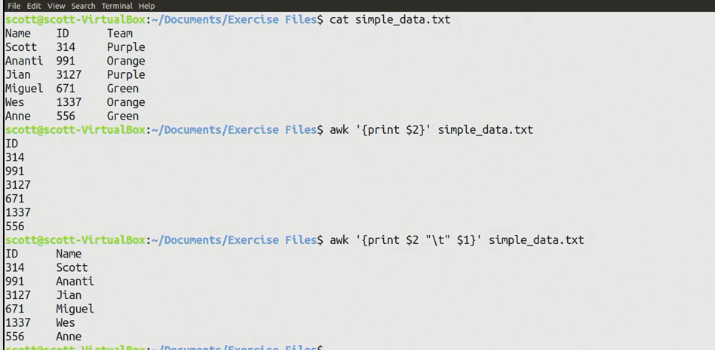
**-E** to specify that we are using regular expression



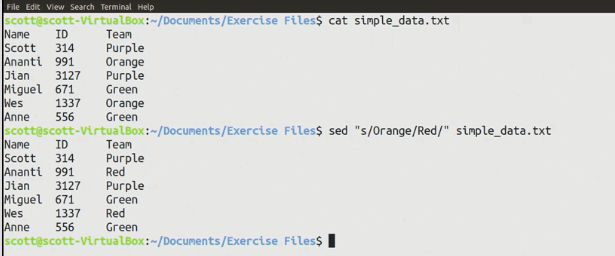
**Ctrl+A:** To move to the first character of a line

**Ctrl+E:** To move to the last character of a line

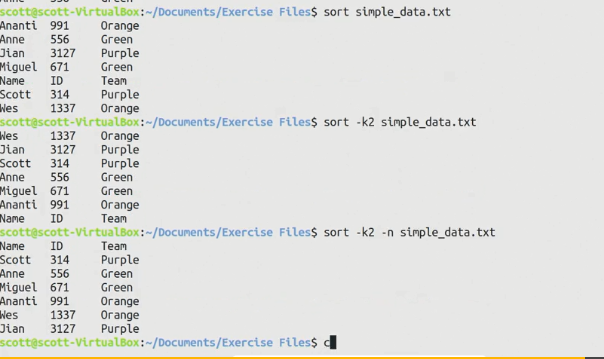
**Awk is responsible for extracting data**:



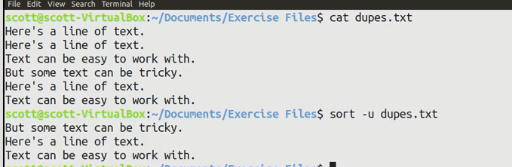
**Sed is for replacing one string with another. S stands for substitute**



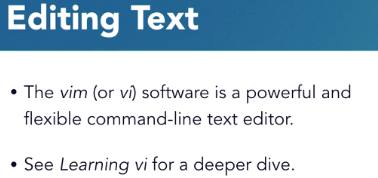
**Sorting: k2 stands for 2nd column, -n stands for numeric**

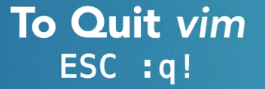


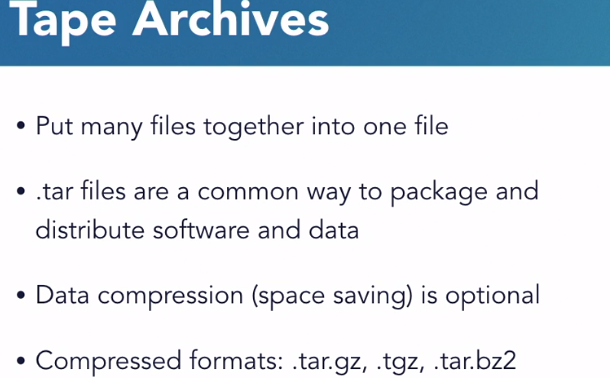
**Ignoring duplicates: -u stands for unique**











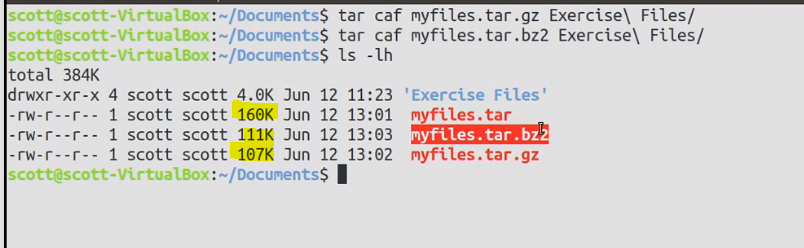
TAR files, short for Tape Archive files, are still incredibly common for software distribution across Linux and macOS. TAR files, unlike ZIP files, don't offer compression themselves, but there are ways to incorporate compression into a TAR file which you may see as you explore different software distribution styles. Compression tries to reduce the size of a file using some mathematical tricks. As far as compressed archives go there's tar.gz or .tgz, which is a TAR file with gzip compression.

There's also tar.bz2, which is a TAR file with bzip compression. And there are others. Gzip and bzip are two different methods for compressing data

**How to create a tar file:**

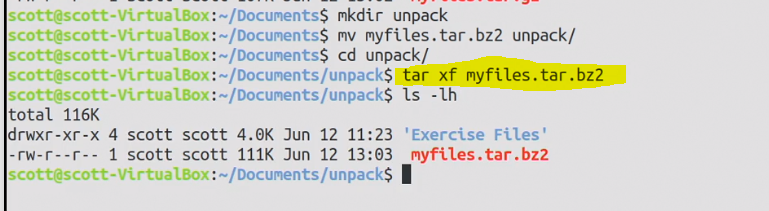


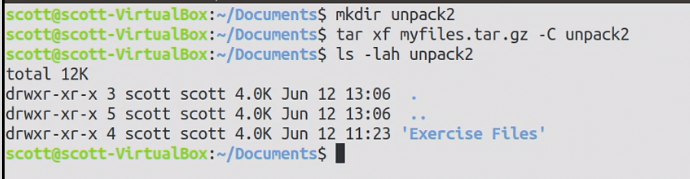
The C option says to create an archive. We'll use another option later to extract from an archive. The V tells the TAR tool to be verbose, that is to list out each file that gets added to the archive. This can be handy to create an index of the contents of the file. The F option tells the TAR command to output the archive to a file. Without that, the data that makes up the TAR file will be sent to the screen to the standard output, unless you pipe it somewhere else. After the F option comes the name of the archive.



The A option tells TAR to figure out what kind of compression to use based on the file extension.

**How to extract a tar file:**



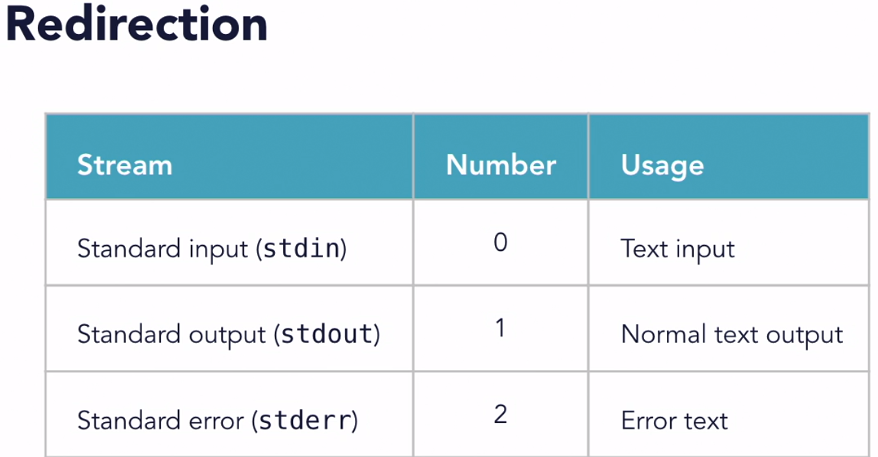


The capital C option specifies a directory to change into for unarchiving





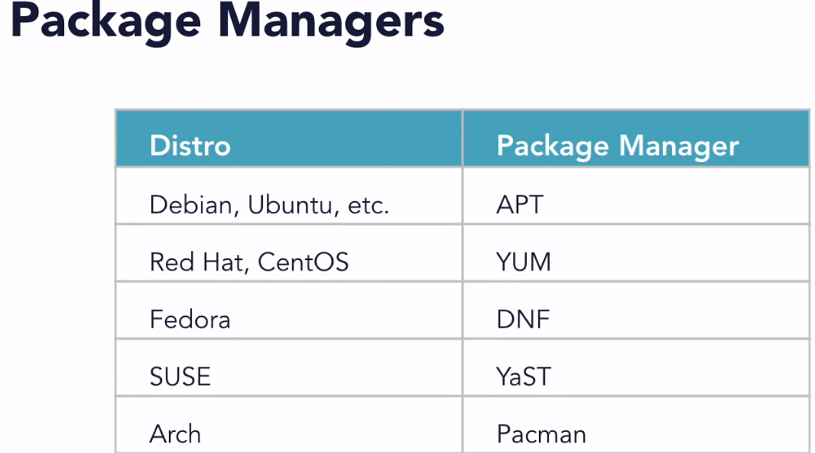
One note here, if we put the path to a folder, ZIP won't work the way we might expect. It'll give us a ZIP file containing an empty folder. To zip up a folder along with its contents, we'll need to add the -r option. And when I run that, I can see how much each of the files is reduced in size.



**To know about the Kernel:**



**To check the space: df -h**



One of the most common ways to add software to a Linux system is to use a package manager, and depending on the distro you're using, the package manager you'll use can vary.

Package managers have similar features but sometimes implement them in different ways.

**Sudo** (superuser do) is a utility for **UNIX**- and **Linux**-based systems that provides an efficient way to give specific users permission to use specific system commands at the root (most powerful) level of the system. **Sudo** also logs all commands and arguments.

Shell is a program that takes command from keyboard and give it to OS.

CLI-Command Line Interface

Terminal is a tool that is used to Shell command.

|  |  |
| --- | --- |
| **Command Name** | **Comments** |
| CTRL+ALT+T | Command to Open Terminal |
| pwd | Current working directory |
| cd / | Move to root directory |
| ls | List Directory contents |
| clear | Clear the command line. Just scroll up to see the contents |
| ls ~ | List contents of my home directory |
| .. | One folder Back |
| ls -l | List all files in long format. |
| ls -a | List all the hidden files |
| ls -al | Long list and hidden file combinedly |
| ls -lS | Sort the file in descending order of size |
| \* | Everything |
| ls -lS > out.txt | Create a file out.txt and save the results of ls -lS |
| ls -d \*/ | Display only the list of directories |
| man ls | Display all the commands available for ls. |
| ~ | Home Directory |
| / | Root Directory |
| .. | One folder above. Parent of the current folder |
| \ | Escape character |
| cat filename | Display the content of file |
| cat -b filename | Enter line number to nonblank row |
| cat -n filename | Enter line number to all the row |
| cat -s filename | Squeeze the blank rows to single blank row |
| cat -E filename | Add $ at the end of the line |

writes of owner: drwxr

d-directory

r-read

w-write

x-execute

r-read

Writes of group: xr