Session 1

Ibrahim

November 9, 2020

1 Introduction

I believe you should be able to learn what your computer does. You should be able to make your software do what you want it to do (within the reasonable limits of its capabilities, of course). The key to attaining this power lies in under- standing the fundamentals of what the software does and how it works You should never have to fight with a computer. Linux is a great platform for learning because it doesn't try to hide any- thing from you. In particular, most system configuration can be found in plaintext files that are easy enough to read. The only tricky part is figuring out which parts are responsible for what and how it all fits together.

2 Basic file management commands

Understanding The File System Tree

Like Windows, a Unix-like operating system such as Linux organizes its files in what is called a hierarchical directory structure. This means that they are organized in a tree-like pattern of directories (sometimes called folders in other systems), which may contain files and other directories. The first directory in the file system is called the root directory. The root directory contains files and subdirectories, which contain more files and subdirectories and so on and so on. Note that unlike Windows, which has a separate file system tree for each storage device, Unix-like systems such as Linux always have a single file system tree, regardless of how many drives or storage devices are attached to the computer. Storage devices are attached (or more correctly, mounted) at various points on the tree according to the whims of the system administrator, the person (or persons) responsible for the maintenance of the system.

File and filenames

- file names can use any character
  !abc?#*.a2   G*#UC
  But it's not wise to use special characters in filenames
- spaces are accepted but not recommended, use dashes or underscores instead
  text results.txt   text_results.txt   text-results.txt
- File names are case sensitive

- file names starting with a dot are hidden
  .bashrc .profile .config

- no concept of file extension , the dot is just another character
  file.txt.old file.doc.origin

Pathes

Linux has a directory hierarchy that starts at /, sometimes called the root directory. The directory separator is the slash (/), not the backslash (\). There are several standard subdirectories in the root directory, such as /usr

- Directory names separated by slahes "/"
  /usr/src/files/text/myfile.txt

- Can be absolute or relative

  - Absolute : does not depend on where you are
    /home/ibrahim/GUC ~/GUC
  - Relative : depends on where you are in the tree
    ./myproject/report ../../myproject/report
    "." means the current dir
    ".." means the parent dir

Commands

```
ibrahim@free$<command> [options] [arguments]
```

This is called a shell prompt and it will appear whenever the shell is ready to accept input. While it may vary in appearance somewhat depending on the distribution, it will usually include your username@machinename, followed by the current working directory (more about that in a little bit) and a dollar sign. If the last character of the prompt is a pound sign ("#") rather than a dollar sign, the ter- minal session has superuser privileges. This means either we are logged in as the root user or we selected a terminal emulator that provides superuser (administrative) privileges.

- commands can be issued on thier own

```
$ls
```

```
 $pwd
```

```
 $cd
```

```
 $tree
```

- The options are normally options , they usually start with "-" or "–"

```
$ls -a
$ls --all
Multiple options
$ls -a -R
$ls -aR
$ls -Ra
$ls --all --recurisve
```

- arguments are those info passed to the command such as filenames

```
$rm -rf ./project-data

($1 to $n) $1 is the first arguments, $2 is second
   argument till $n  n th  arguments. From 10  th
   argument, you must need to inclose them in braces like
   ${10}, ${11} and so on

$0 the name of the command

$$ Process id of current shell

 $?  Exit status id of last command

$# Total number of arguments passed to script
```

basic commands

- ls to list files

```
$ls
$ls -a
$ls -l
$ls <dir> to list the content of the mentioned dir
$ls <file> to list mentioned
```

- pwd print current dir
- cd to change dir

```
$cd
$cd [destnation]
$cd -
$cd ../../usr/files/.conf
```

- mkdir to make new dir

```
$mkdir project/images
$mkdir projects images
$mkdir -p projects/images
```

- cp to copy files

```
$cp ~/books/science.doc ~/school/science.doc
$cp ~/books/science.doc ~/school/
$cp -r <dir>
$cp file1 file2
$cp /etc/passwd . copy the file here
```

- mv move and rename files
- rm remove files
- touch create files
- less (pager)

3 Try to solve

- create a tree of dirs includes your courses each course includes 10 weeks each week includes lec and tut dir
- create a dir called src and another called out copy all java files into src and all class files into out
- create a dir called webpages that have 100 project each have html css js images/jpj images/png fonts and then create multiple html files then copy them inside css dir then remove the old ones