Lab 2 Report

Lucas Tavares

01-29-2024

1 Exercise 1

Using the `ls` to list the files in order of time I found out the most recent file modified in the lab directory was `check.sh`, which name I've written into `latest-file.info`. The directory `report`, showed up prior to `check.sh` (since I just modified `main.tex` and generated `main.pdf`, but since it is a directory and not a file, I disconsidered it.

Command:

```
ls -l -t
emacs latest-file.info
```

Disphering output:
From the output I looked at columns 6-9 because 6-8 says when the file was created/last modified and column 9 says the file name.

Options used:

- `-l` shows more information per file
- `-t` orders the files by modification time

Sources of informtion:

- `ls --help`: Found list of options
- `https://www.tutorialspoint.com/how-to-list-the-last-five-modified-files-in-linux#:~:text=Using%20ls%20Command&text=The%20%2Dt%20option%20sorts%20the,modified%20file%20at%20the%20top.`: Found on Google

2 Exercise 2

Using `ls` and `wc` connected with a pipe (|) I listed all items in my lab directory and counted the number of words on that list.

Command:

```
ls -A | wc -w
emacs count.info
```

Disphering output:
The output showed me the number of words on the list of all items in my lab directory. This number represents how many objects there are on my lab directory.

Options used:

- `-A` instructs `ls` to list all items, including hidden files
- `-w` tells `wc` to count words

Sources of informtion:

- `https://kodekloud.com/blog/file-count-in-directory-linux/#:~:text=The%20wc%20command%20stands%20for,number%20of%20files%20and%20directories.`: Found on Google

3 Exercise 3

Using `whereis` I could discover the full path to the `ls` library.

Command:

```
whereis -b ls
emacs ls.info
```

Disphering output:
The output gave me the full path to the binary of the program ls.

Options used:

- `-b` search only for binaries

Sources of informtion:

- `whereis --help`: Found list of options

4 Exercise 4

Using `find` I was able to find all the files with the name “readme.txt” in the whole Linux file systems which I had permission to access.

Command:

```
find / -iname "readme.txt" -print 2>/dev/null
emacs find.info
```

Disphering output:
The output displayed a list of the paths where there is a file named “readme.txt”. It was also not displaying the paths which I didn’t have permission granted to access.

Options used:

- `-iname` tells `find` what is the file name we are looking for, case insensitive
- `-print` print the full path with the file name on the standard output
- `2>/dev/null` tells my shell to redirect the permission denied messages (error messages) to /dev/null, hidding them

Sources of informtion:

- `man find`: Found list of options
- `https://www.hostinger.com/tutorials/how-to-use-find-and-locate-commands-in-linux/#:~:text=There%20are%20two%20Linux%20commands,only%20on%20your%20Linux%20database.`: Found on Google
- `https://www.cyberciti.biz/faq/bash-find-exclude-all-permission-denied-messages/`: Found on Google

5 Exercise 5

Using `htop` I was able to visualize the live information about running programs, CPU and memory usage.

Command:

```
htop
emacs taskmanager.info
```

Disphering output:
The output is a program which displays useful live updated information about your computer, system, and programs, such as CPU and memory usage and running programs. It is a program really similar to Windows' Task Manager.

Sources of informtion:

- `https://www.javatpoint.com/linux-task-manager#:~:text=If%20we%20wish%20for%20a,well%20as%20it%20appears%20good.`: Found on Google