

Lab: Review of the File System: Manipulating Files & Directories

If you have ever owned a computer before, then you know how important it is to be able to copy and move files around. That's why I dedicated an entire lab to talk just about that: copying, moving, and deleting files.

Note: Make sure to add following text in `/home/elliott/cats.txt` before starting the lab.

```
I love cars!
I love cats!
I love penguins!
```

Copying one file

Sometimes you need to copy a single file. Luckily this is a simple operation on the command line. I have a file named [cats.txt] in my home directory:

```
elliott@ubuntu-linux:~$ cat cats.txt
I love cars!
I love cats!
I love penguins!
elliott@ubuntu-linux:~$
```

I can use the [cp] command to make a copy of [cats.txt] named [copycats.txt] as follows:

```
elliott@ubuntu-linux:~$ cp cats.txt copycats.txt
elliott@ubuntu-linux:~$ cat copycats.txt
I love cars!
I love cats!
I love penguins!
elliott@ubuntu-linux:~$
```

As you can see, the copied file [copycats.txt] has the same content as the original file [cats.txt].

I can also copy the file [cats.txt] to another directory. For example, I can copy the file [cats.txt] to [/tmp] by running the [cp cats.txt /tmp] command:

```
elliott@ubuntu-linux:~$ cp cats.txt /tmp
elliott@ubuntu-linux:~$ cd /tmp
elliott@ubuntu-linux:/tmp$ ls cat*
cats.txt
elliott@ubuntu-linux:/tmp$
```

Notice that the copied file has the same name as the original file. I can also make another copy in **[/tmp]** with a different name:

```
elliott@ubuntu-linux:~$ cp cats.txt /tmp/cats2.txt
elliott@ubuntu-linux:~$ cd /tmp
elliott@ubuntu-linux:/tmp$ ls cat*
cats2.txt  cats.txt
elliott@ubuntu-linux:/tmp$
```

Copying multiple files

You may also want to copy multiple files at once. To demonstrate, let's begin by creating three files [apple.txt], [banana.txt], and [carrot.txt] in Elliot's home directory:

```
elliott@ubuntu-linux:~$ touch apple.txt banana.txt carrot.txt
elliott@ubuntu-linux:~$ ls
apple.txt carrot.txt copycats.txt dir1
banana.txt cats.txt Desktop
elliott@ubuntu-linux:~$
```

To copy the three newly created files to [/tmp], you can run the [cp apple.txt banana.txt carrot.txt /tmp] command:

```
elliott@ubuntu-linux:~$ cp apple.txt banana.txt carrot.txt /tmp
elliott@ubuntu-linux:~$ cd /tmp
elliott@ubuntu-linux:/tmp$ ls
apple.txt banana.txt carrot.txt cats2.txt cats.txt
elliott@ubuntu-linux:/tmp$
```

Child's play! In general, the [cp] command follows the syntax:

```
cp source_file(s) destination
```

Copying one directory

You may also want to copy an entire directory; that's also easily accomplished. To demonstrate, create a directory named [cities] in your home directory, and inside [cities], create three files [paris], [tokyo], and [london] as follows:

```
elliott@ubuntu-linux:~$ mkdir cities
elliott@ubuntu-linux:~$ cd cities/
elliott@ubuntu-linux:~/cities$ touch paris tokyo london
elliott@ubuntu-linux:~/cities$ ls
london paris tokyo
```

Now if you want to copy the [cities] directory to [/tmp], you have to pass the recursive [-r] option to the [cp] command as follows:

```
elliott@ubuntu-linux:~/cities$ cd ..
elliott@ubuntu-linux:~$ cp -r cities /tmp
```

You will get an error message if you omitted the [-r] option:

```
elliott@ubuntu-linux:~$ cp cities /tmp
cp: -r not specified; omitting directory 'cities'
```

You can verify that the [cities] directory is copied to [/tmp] by listing the files in [/tmp]:

```
elliott@ubuntu-linux:~$ cd /tmp
elliott@ubuntu-linux:/tmp$ ls
apple.txt banana.txt carrot.txt cats2.txt cats.txt cities
```

```
elliott@ubuntu-linux:/tmp$ ls cities
london paris tokyo
```

Copying multiple directories

You can also copy multiple directories the same way you copy multiple files; the only difference is that you have to pass the recursive [-r] option to the [cp] command.

To demonstrate, create the three directories [d1], [d2], and [d3] in Elliot's home directory:

```
elliott@ubuntu-linux:~$ mkdir d1 d2 d3
```

Now you can copy all three directories to [/tmp] by running the [cp -r d1 d2 d3 /tmp] command:

```
elliott@ubuntu-linux:~$ cp -r d1 d2 d3 /tmp
elliott@ubuntu-linux:~$ cd /tmp
elliott@ubuntu-linux:/tmp$ ls
apple.txt banana.txt carrot.txt cats2.txt cats.txt cities d1 d2 d3
```

Moving one file

Sometimes, you may want to move a file (or a directory) to a different location instead of copying and wasting disk space.

To do this, you can use the [mv] command. For example, you can move the file [copycats.txt] from Elliot's home directory to [/tmp] by running the [mv copycats.txt /tmp] command:

```
elliott@ubuntu-linux:~$ mv copycats.txt /tmp
elliott@ubuntu-linux:~$ ls
apple.txt  carrot.txt  cities d2  Desktop  Downloads
banana.txt  cats.txt   d1      d3  dir1     Pictures
elliott@ubuntu-linux:~$ cd /tmp
elliott@ubuntu-linux:/tmp$ ls
apple.txt  carrot.txt  cats.txt  copycats.txt  d2
banana.txt  cats2.txt  cities   d1            d3
```

Notice that [copycats.txt] is now gone from Elliot's home directory as it relocated to [/tmp].

Moving multiple files

You can also move multiple files the same way you can copy multiple files. For example, you can move the three files [apple.txt], [banana.txt], and [carrot.txt] from [/tmp] to [/home/elliott/d1] as follows:

```
elliott@ubuntu-linux:/tmp$ mv apple.txt banana.txt carrot.txt /home/elliott/d1
elliott@ubuntu-linux:/tmp$ ls
cats2.txt cats.txt cities copycats.txt d1 d2 d3
elliott@ubuntu-linux:/tmp$ cd /home/elliott/d1
elliott@ubuntu-linux:~/d1$ ls
apple.txt banana.txt carrot.txt
elliott@ubuntu-linux:~/d1$
```

As you can see, the three files [apple.txt], [banana.txt], and [carrot.txt] are no longer located in [/tmp] as they all moved to [/home/elliott/d1]. In general, the [mv] command follows the syntax:

```
mv source_file(s) destination
```

Moving one directory

You can also use the [mv] command to move directories. For example, if you want to move the directory [d3] and put it inside [d2], then you can run the [mv d3 d2] command:

```
elliott@ubuntu-linux:~$ mv d3 d2
elliott@ubuntu-linux:~$ cd d2
elliott@ubuntu-linux:~/d2$ ls
d3
elliott@ubuntu-linux:~/d2$
```

Notice that you don't need to use the recursive [-r] option to move a directory.

Moving multiple directories

You can also move multiple directories at once. To demonstrate, create a directory named [big] in Elliot's home directory:

```
elliott@ubuntu-linux:~$ mkdir big
```

Now you can move the three directories [d1], [d2], and [cities] to the [big] directory as follows:

```
elliott@ubuntu-linux:~$ mv d1 d2 cities big
elliott@ubuntu-linux:~$ ls big
cities d1 d2
elliott@ubuntu-linux:~$
```

Renaming files

You can also use the [mv] command to rename files. For example, if you want to rename the file [cats.txt] to [dogs.txt], you can run the [mv cats.txt dogs.txt] command:

```
elliott@ubuntu-linux:~$ mv cats.txt dogs.txt
elliott@ubuntu-linux:~$ cat dogs.txt
I love cars!
I love cats!
I love penguins!
elliott@ubuntu-linux:~$
```

If you want to rename the directory [big] to [small], you can run the [mv big small] command:

```
elliott@ubuntu-linux:~$ mv big small
elliott@ubuntu-linux:~$ ls small
cities d1 d2
elliott@ubuntu-linux:~$
```

In summary, here is how the [mv] command works:

1. If the destination directory exists, the [mv] command will move the source file(s) to the destination directory.
2. If the destination directory doesn't exist, the [mv] command will rename the source file.

Keep in mind that you can only rename one file (or one directory) at a time.

Hiding files

You can hide any file by renaming it to a name that starts with a dot.

Let's try it; you can hide the file [dogs.txt] by renaming it to [.dogs.txt] as follows:

```
elliott@ubuntu-linux:~$ ls
apple.txt banana.txt carrot.txt dogs.txt Desktop dir1 small
elliott@ubuntu-linux:~$ mv dogs.txt .dogs.txt
elliott@ubuntu-linux:~$ ls
apple.txt banana.txt carrot.txt Desktop dir1 small
elliott@ubuntu-linux:~$
```

As you can see, the file [dogs.txt] is now hidden as it got renamed to [.dogs.txt]. You can unhide [.dogs.txt] by renaming it and removing the leading dot from the filename:

```
elliott@ubuntu-linux:~$ mv .dogs.txt dogs.txt
elliott@ubuntu-linux:~$ ls
apple.txt banana.txt carrot.txt dogs.txt Desktop dir1 small
elliott@ubuntu-linux:~$
```

Yes, Sir! You can also hide and unhide directories in the same manner. I will leave that for you to do as an exercise.

Removing files

You can use the [rm] command to remove (delete) files. For example, if you want to remove the file [dogs.txt], you can run the [rm dogs.txt] command:

```
elliott@ubuntu-linux:~$ ls
apple.txt banana.txt carrot.txt dogs.txt Desktop dir1 small
elliott@ubuntu-linux:~$ rm dogs.txt
elliott@ubuntu-linux:~$ ls
apple.txt banana.txt carrot.txt Desktop dir1 small
```

You can also remove multiple files at once. For example, you can remove the three files [apple.txt], [banana.txt], and [carrot.txt] by running the [rm apple.txt banana.txt carrot.txt] command:

```
elliott@ubuntu-linux:~$ rm apple.txt banana.txt carrot.txt
elliott@ubuntu-linux:~$ ls
Desktop dir1 small
elliott@ubuntu-linux:~$
```

Removing directories

You can pass the recursive `[-r]` option to the `[rm]` command to remove directories. To demonstrate, let's first create a directory named `[garbage]` in Elliot's home directory:

```
elliott@ubuntu-linux:~$ mkdir garbage
elliott@ubuntu-linux:~$ ls
Desktop dir1 garbage small
```

Now let's try to remove the `[garbage]` directory:

```
elliott@ubuntu-linux:~$ rm garbage
rm: cannot remove 'garbage': Is a directory
elliott@ubuntu-linux:~$
```

Shoot! I got an error because I didn't pass the recursive `[-r]` option. I will pass the recursive option this time:

```
elliott@ubuntu-linux:~$ rm -r garbage
elliott@ubuntu-linux:~$ ls
Desktop dir1 small
```

Cool! We got rid of the `[garbage]` directory.

You can also use the `[rmdir]` command to remove only empty directories. To demonstrate, let's create a new directory named `[garbage2]` and inside it, create a file named `[old]`:

```
elliott@ubuntu-linux:~$ mkdir garbage2
elliott@ubuntu-linux:~$ cd garbage2
elliott@ubuntu-linux:~/garbage2$ touch old
```

Now let's go back to Elliot's home directory and attempt to remove `[garbage2]` with the `[rmdir]` command:

```
elliott@ubuntu-linux:~/garbage2$ cd ..
elliott@ubuntu-linux:~$ rmdir garbage2
rmdir: failed to remove 'garbage2': Directory not empty
```

As you can see, it wouldn't allow you to remove a nonempty directory. Therefore, let's delete the file `[old]` that's inside `[garbage2]` and then reattempt to remove `[garbage2]`:

```
elliott@ubuntu-linux:~$ rm garbage2/old
elliott@ubuntu-linux:~$ rmdir garbage2
elliott@ubuntu-linux:~$ ls
Desktop dir1 small
elliott@ubuntu-linux:~$
```

Boom! The `[garbage2]` directory is gone forever. One thing to remember here is that the `[rm -r]` command will remove any directory (both empty and nonempty). On the other hand, the `[rmdir]` command will only delete empty directories.

For the final example in this lab, let's create a directory named `[garbage3]`, then create two files `[a1.txt]` and `[a2.txt]` inside it:

```
elliott@ubuntu-linux:~$ mkdir garbage3
elliott@ubuntu-linux:~$ cd garbage3/
elliott@ubuntu-linux:~/garbage3$ touch a1.txt a2.txt
```

```
elliott@ubuntu-linux:~/garbage3$ ls
a1.txt a2.txt
```

Now let's get back to Elliot's home directory and attempt to remove [garbage3]:

```
elliott@ubuntu-linux:~/garbage3$ cd ..
elliott@ubuntu-linux:~$ rmdir garbage3
rmdir: failed to remove 'garbage3': Directory not empty
elliott@ubuntu-linux:~$ rm -r garbage3
elliott@ubuntu-linux:~$ ls
Desktop dir1 Downloads Pictures small
elliott@ubuntu-linux:~$
```

As you can see, the [rmdir] command has failed to remove the nonempty directory [garbage3], while the [rm -r] command has successfully removed it.

Nothing makes information stick in your head like a good knowledge-check exercise.

Knowledge check

For the following exercises, open up your Terminal and try to solve the following tasks:

1. Create three files [hacker1], [hacker2], and [hacker3] in your home directory.
2. Create three directories [Linux], [Windows], and [Mac] in your home directory.
3. Create a file named [cool] inside the [Linux] directory you created in task 2.
4. Create a file named [boring] inside the [Windows] directory you created in task 2.
5. Create a file named [expensive] in the [Mac] directory you created in task 2.
6. Copy the two files [hacker1] and [hacker2] to the [/tmp] directory.
7. Copy the two directories [Windows] and [Mac] to the [/tmp] directory.
8. Move the file [hacker3] to the [/tmp] directory.
9. Move the directory [Linux] to the [/tmp] directory.
10. Remove the file [expensive] from the [Mac] directory (in your home directory).
11. Remove the directory [Mac] from your home directory.
12. Remove the directory [Windows] from your home directory.
13. Remove the file [hacker2] from your home directory.
14. Rename the file [hacker1] to [hacker01].

True or false

1. The [cp] command can copy directories without using the recursive option [-r].
2. You have to use the recursive option [-r] when moving directories.
3. You can use the [mv] command to rename files or directories.
4. You can remove a non-empty directory with the [rmdir] command.
5. You can remove a non-empty directory with the [rm -r] command.