Talk Tech to Me: 10 Common Linux File Management Commands

Monday, March 21, 2022 | By Damon M. Garn

The Linux command line takes a little getting used to, especially for those relatively new to the IT industry and the SysAdmin role. Many commands are covered by training materials, such as CompTIA CertMaster Learn for A+ and Linux+, and it is often handy to have a coherent series of example commands to work with, whether or not you are pursuing certification.

Let's take a look at the 10 common commands that Linux users encounter daily when managing files. The commands and examples are organized logically so that you can move from one to the next in a demonstration format.

1. Where Am I Now? The pwd Command

Navigating the Linux file system from the command line can be daunting. In a graphical user interface (GUI), it is pretty easy to see your current location and follow a path to a particular folder. The `pwd` command displays the path from the root of the filesystem (depicted as a single forward slash) to the directory you're currently browsing.

For example, many Linux distributions place you in your home directory upon login, but the pwd command tells you that the directory is located along the following absolute path: `/home/user1` (assuming user1 is your account name). Knowing where you are is the key to knowing where to go next.

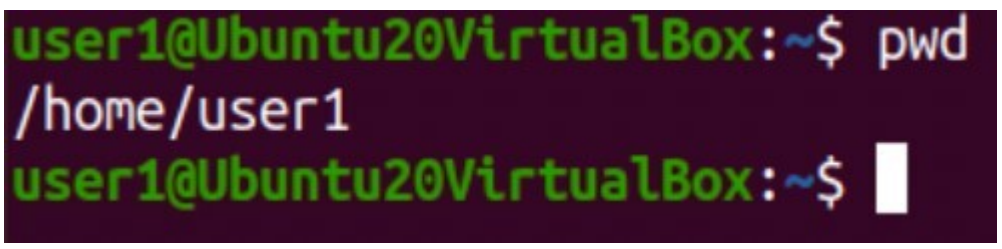

Figure 1: Display the present working directory

2. Organize Files: The mkdir Command

At this point, you know your current location in the filesystem, thanks to `pwd`. What about creating new directories to organize your data? Use the `mkdir` command for this. For example, `mkdir projects` creates a directory named "projects" in the current folder.

You can also use `mkdir` to create an entire hierarchy of folders. Let's say you want to create a directory named Q1 as a subdirectory of a folder named 2022, but the 2022 directory doesn't yet exist. In that case, type `mkdir -p 2022/Q1`, and the 2022 directory is created, and the Q1 subdirectory is created inside it.

For the purposes of this demonstration, you have now created three directories: "projects" and "2022", plus a subdirectory of 2022 named Q1.

Create one more directory in `/home/user1` with `mkdir` named “resources”.

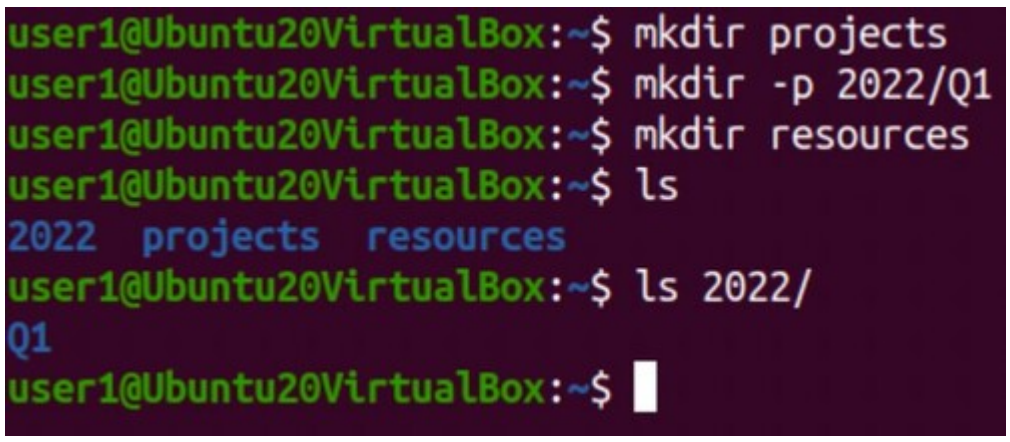

Figure 2: Create directories

3. Remove Directories: The rmdir Command

Maybe you’ve changed your mind about a directory, and you want to delete it. The `rmdir` command removes empty directories. Be careful—there is no undo action with this command. Here’s an example of using `rmdir` to delete the resources directory:

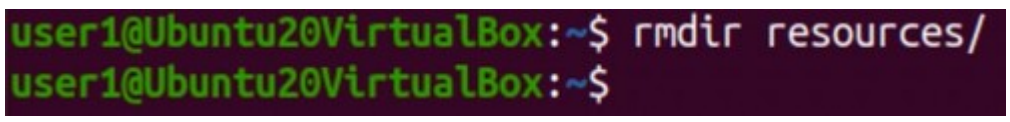

Figure 3: Delete directories

What if the directory contains files? In that case, use a modified version of the `rm` command (discussed below) to delete the directory and its contents. To delete the resources directory created earlier—assuming it has files in it—type `rm -R resources`.

Be aware that if there are files in a directory, you may be prompted to delete each one. This can be very cumbersome. If you add the `-f` option to the command, it forces the deletion of files without confirmation. That’s dangerous, but handy if you’re sure of yourself.

At this point, the resources directory is gone, but the projects, 2022, and 2022/Q1 directories remain.

4. Move Between Directories: The cd Command

Use the `cd` (change directory) command to change from one directory to another. Depending on what you need, use either the absolute path or the relative path.

To change from your current directory to the /etc directory (where most configuration files are stored), type `cd /etc`. Note that you must have the execute permission to a directory to change into it with `cd`.

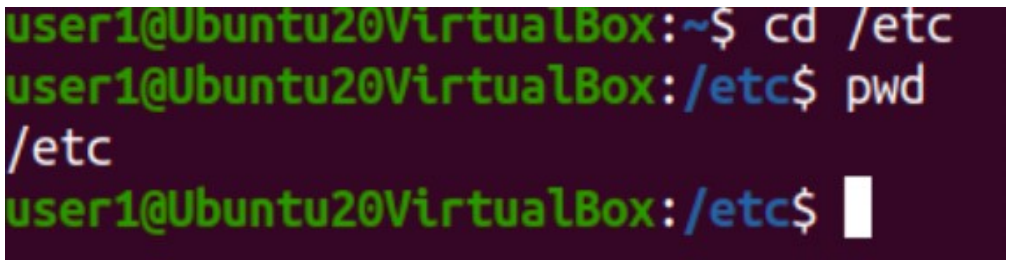

Figure 4: Change directories

Linux users often think of their home directory as “home base” or the jumping-off point for navigating the filesystem. Therefore, it is useful to know how to get home.

The obvious method is to use the `cd` command and type the absolute path, such as `cd /home/user1`. That certainly works, but there are easier ways. Recall that the tilde character (~) represents the home directory of the currently logged-on user. That means that `cd ~` also returns you to your home directory. However, you can just type `cd` with no argument, and the system assumes you wish to go to your home folder.

As part of the demonstration, use the `cd` command to change to the /etc directory, and then use `cd /home/user1` to return to your home directory. Repeat the process by moving back to the /etc directory and then using the `cd ~` and then `cd` commands to practice the shortcuts. Use `pwd` to confirm your location with each test.

```
user1@Ubuntu20VirtualBox:~$ cd /etc
user1@Ubuntu20VirtualBox:/etc$ pwd
/etc
user1@Ubuntu20VirtualBox:/etc$ cd /home/user1/
user1@Ubuntu20VirtualBox:~$ pwd
/home/user1
user1@Ubuntu20VirtualBox:~$ cd /etc
user1@Ubuntu20VirtualBox:/etc$ cd ~
user1@Ubuntu20VirtualBox:~$ pwd
/home/user1
user1@Ubuntu20VirtualBox:~$ cd /etc
user1@Ubuntu20VirtualBox:/etc$ cd
user1@Ubuntu20VirtualBox:~$ pwd
/home/user1
user1@Ubuntu20VirtualBox:~$
```

Figure 5: Change to your home directory

Finally, type `cd projects` from your home directory to change to the projects directory and prepare for the next example.

```
user1@Ubuntu20VirtualBox:~$ cd projects
user1@Ubuntu20VirtualBox:~/projects$ pwd
/home/user1/projects
user1@Ubuntu20VirtualBox:~/projects$
```

Figure 6: Change to the projects directory

5. Create Files: The touch Command

This article does not cover using text editors such as Vim or Nano to create files. However, it's sometimes useful to be able to create empty files to work with. The `touch` command does exactly that. Technically, `touch` updates the timestamp on existing files. However, if the command is issued and the specified file does not exist, it is created.

To use `touch`, simply type the command and the desired filename. To create a file named "tasklist" in the projects directory (assuming your `pwd` is the projects directory), type `touch tasklist`. Use `touch` to create two more files in the projects directory, one named "activities" and one named "content."

```
user1@Ubuntu20VirtualBox:~/projects$ touch tasklist
user1@Ubuntu20VirtualBox:~/projects$ touch activities
user1@Ubuntu20VirtualBox:~/projects$ touch content
user1@Ubuntu20VirtualBox:~/projects$ ls
activities  content  tasklist
user1@Ubuntu20VirtualBox:~/projects$
```

Figure 7: Use touch to create files

You now have three files in the projects directory: tasklist, activities and content.

6. What's in This Directory? The ls Command

The `ls` (list) command displays subdirectories and files in the current directory. However, filenames where the first character is a dot, such as .secretfile, are hidden and won't display with the `ls` command. However, by adding the `-a` (all) option, hidden files are shown.

Type the following commands to see an example (make sure you're in the projects folder):

```
user1@Ubuntu20VirtualBox:~/projects$ touch .secretfile
user1@Ubuntu20VirtualBox:~/projects$ ls
activities  content  tasklist
user1@Ubuntu20VirtualBox:~/projects$ ls -a
.  ..  activities  content  .secretfile  tasklist
user1@Ubuntu20VirtualBox:~/projects$
```

Figure 8: Display a hidden file

Is the .secretfile object displayed with `ls`? It should not be. Type `ls -a` and notice that the file is now displayed.

Add the `-l` option to display the contents of the directory in long format. The output includes permissions, ownership, timestamps, and other information pertaining to the files. Don't forget that you can combine options, too. Try `ls -la` to list all files in long format.

```
user1@Ubuntu20VirtualBox:~/projects$ ls -la
total 8
drwxrwxr-x 2 user1 user1 4096 Oct 28 10:33 .
drwxr-xr-x 4 user1 user1 4096 Oct 28 10:21 ..
-rw-rw-r-- 1 user1 user1    0 Oct 28 10:29 activities
-rw-rw-r-- 1 user1 user1    0 Oct 28 10:29 content
-rw-rw-r-- 1 user1 user1    0 Oct 28 10:33 .secretfile
-rw-rw-r-- 1 user1 user1    0 Oct 28 10:29 tasklist
user1@Ubuntu20VirtualBox:~/projects$
```

Figure 9: List all files in long format

If you're in the projects directory, the `ls -a` command should confirm the existence of four files: tasklist, activities, content and .secretfile.

7. Move and Rename: The mv Command

The `mv` command moves files from one directory to another. The syntax for moving a file is straightforward. Besides the actual `mv` command, simply specify the source and the destination directories for the file. I think of the syntax as *move the file from here to there*.

For example, to move content from the projects folder to the folder named 2022 in your home directory, type `mv content /home/user1/2022`. Use the `ls /home/user1/2022` command to ensure the process worked.

```
user1@Ubuntu20VirtualBox:~/projects$ mv content /home/user1/2022
user1@Ubuntu20VirtualBox:~/projects$ ls /home/user1/2022
content  Q1
user1@Ubuntu20VirtualBox:~/projects$
```

Figure 10: Move files

However, the `mv` command is also the rename command. To rename activities to labs, type `mv activities labs`. In other words, move it to the same location with a new name. Confusing but simple, once you grasp the concept.

```
user1@Ubuntu20VirtualBox:~/projects$ ls
activities  tasklist
user1@Ubuntu20VirtualBox:~/projects$ mv activities labs
user1@Ubuntu20VirtualBox:~/projects$ ls
labs  tasklist
user1@Ubuntu20VirtualBox:~/projects$
```

Figure 11: Rename files

8. Duplicate Files: The cp Command

A file can be copied to another location using the `cp` command. To copy tasklist from your projects directory location to the 2022/Q1 directory, type `cp tasklist ~/2022/Q1`. The syntax is the same as with `mv`; *copy from here to there*.

```
user1@Ubuntu20VirtualBox:~/projects$ cp tasklist ~/2022/Q1
user1@Ubuntu20VirtualBox:~/projects$ ls ~/2022/Q1
tasklist
user1@Ubuntu20VirtualBox:~/projects$
```

Figure 12: Copy files

Note that you can use the tilde with commands such as `cp` and `mv` to reduce the amount of typing involved in managing files.

9. Delete Files: The rm Command

To delete an existing file, use the `rm` command. Type `rm .secretfile` to delete the hidden file you created earlier. Use `ls -a` to confirm that the file is gone.

```
user1@Ubuntu20VirtualBox:~/projects$ ls -a
.  ..  labs  .secretfile  tasklist
user1@Ubuntu20VirtualBox:~/projects$ rm .secretfile
user1@Ubuntu20VirtualBox:~/projects$ ls -a
.  ..  labs  tasklist
user1@Ubuntu20VirtualBox:~/projects$
```

Figure 13: Delete files

Deleting a file is easy, but recall that there is no undo, so be careful. Some distributions will default to interactive mode by converting the `rm` command to `rm -i`, but not all of them do that.

It is possible to add the `-f` option, which forces the deletion without the interactive prompt. This is convenient if you have many files to delete, but dangerous in that you could inadvertently delete something you didn't want to.

10. Search for Content: The grep Command

Sometimes it's useful to be able to look inside files to find a specific piece of content. The `grep` command does exactly that. The `grep` command is a pattern-matching utility that can run against file content or the results of other commands. Grep is very powerful and can be used in conjunction with commands such as `ps` and `ls`, too.

Perhaps you're looking for a string of text in a particular file. The string is part of a word that you believe is in the file. You can use `grep` to check the file. To check for the phrase "Hello World" in the tasklist file, type `grep ll tasklist`. You're attempting to match the "ll" string (which is found in the word *Hello*):

```
user1@Ubuntu20VirtualBox:~/projects$ echo "Hello World" > tasklist
user1@Ubuntu20VirtualBox:~/projects$ grep ll tasklist
Hello World
user1@Ubuntu20VirtualBox:~/projects$
```

Figure 14: Match patterns in a file

If you combine `grep` with tools such as `ls`, you can display certain output from the command. What if you need to show all files in the /etc directory that contain the “net” character string? The following command would help: `ls /etc | grep -I net`

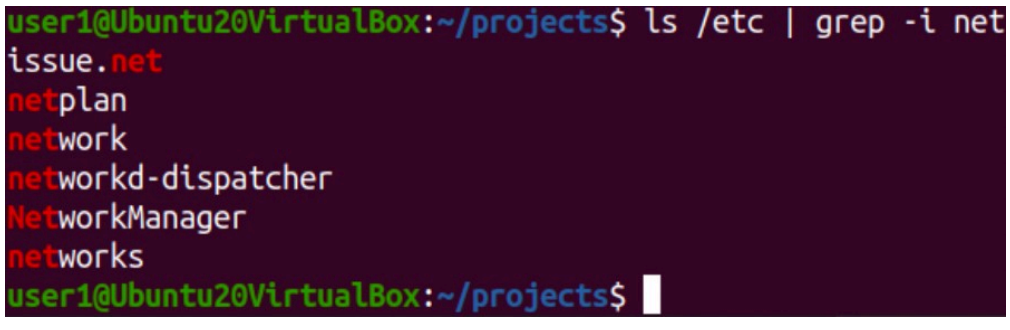

Figure 15: Match patterns in a command's output

Adding the `-i` option causes `grep` to ignore case. Notice that the output above contains the “net” string with some characters of each case.

Understanding Basic Linux File Management Commands

These common Linux commands help users move around the filesystem and manage files. These skills are useful for many day-to-day SysAdmin tasks. For example, before making changes to a configuration file, it is useful to copy the original file as a backup. Shortcuts such as the tilde (~) make administrative tasks quicker and more efficient.

Approaching these commands from the perspective of a tutorial can make them much easier to work with, especially when preparing for certification exams. These 10 commands should get you well on your way to basic navigation and file management.

TAGS: COMPTIA LINUX+, COMPUTER NETWORKS, IT CAREER CENTER, IT SKILLS, LINUX, OFFICIAL COMPTIA TRAINING, TOOLS, TRAINING

Email us at blogeditor@comptia.org for inquiries related to contributed articles, link building and other web content needs.

PREVIOUS POST
How to Become an (IT) Pro at Virtual Networking

NEXT POST
14 Tips for MSPs to Prepare for the Inevitable Cybersecurity Incident

Read More from the CompTIA Blog

How to grow to six-figure jobs in tech without a degree

2024-09-30

How upskilling benefits your organization

2024-09-30

CompTIA Cloud+ provides in-demand cloud skills

2024-09-27

Leave a Comment

Email

Name

I'm not a robot

reCAPTCHA
Privacy - Terms

SUBMIT COMMENT

Contact Us

Blog

Join Our Team

CERTIFICATION

CompTIA IT Certifications

Store

Account Login

MEMBERSHIP COMMUNITY

CompTIA – The IT Industry Association

SOCIAL IMPACT

CompTIA Spark

PARTNERS

CompTIA Authorized Partner Program

Change Language

 Sitemap Legal