

Quiz: Creating, Modifying, and Removing Files and Folders in Linux

Introduction

Today, Linux is everywhere. Lots of the server systems behind your favorite websites are Linux-based. As an IT Support Specialist, you'll most likely be interacting with Linux on a regular basis, mainly through the command line. Actions like modifying configuration files and moving or copying them may become part of your everyday tasks. File management in Linux is a super important core skill to have as an IT Support Specialist. So, let's dive in!

What you'll do

You'll learn how to navigate a Linux file system from the command line and understand the basics of creating, modifying, copying, and deleting files and directories. Your main learning objective for this lab is to practice these commands in the Linux VM.

Learning tip

We encourage you to try and memorize all of these commands as best you can. With enough practice, using Linux commands will become second nature to you. If you have access to your own Linux

machine, try out the commands as you follow along in the next section.

If you don't have Linux available on your local machine, no worries! You can type these commands in a text editor, so you can refer back to them when you're doing the active lab exercises.

Linux overview

Before we kick things off, below is a brief intro into the Linux file system.

The **file system** controls how data is stored and retrieved in a computer. All files and folders in a Linux system are part of a bigger tree-like structure rooted at /. Files and folders are added to the file system by appending them to this tree structure, and deleted by removing them. **All file names are case sensitive.** When working with files and directories on the command line, special characters, like space and brackets, have to be **escaped** using a backslash.

Note: This section consists of basic commands to help you explore the Linux file system. Do not expect any interesting response back within this section, but they would be helpful in various stages of the lab.

Here are a few helpful navigation commands to help you explore the Linux file system:

You can check out the contents of the current directory using the `ls` command.

```
ls
```

Copied!

```
content_copy
```

You can view more details about the files, like ownership and permissions, by adding the flag `-l` to the `ls` command.

```
ls -l
```

Copied!

```
content_copy
```

You can see hidden files in the current directory by passing flag `a` to the `ls` command.

```
ls -a
```

Copied!

```
content_copy
```

You can find out where you are in relation to the rest of the file system using the `pwd` command.

```
pwd
```

Copied!

content_copy

You can navigate to different directories using the *cd* command.

```
cd /path/to/other/directory
```

Copied!

content_copy

You can check out the contents of a file using the *cat* command.

```
cat /path/to/file/file_name
```

Copied!

content_copy

For large input files, the *less* command allows movement within the files. The syntax is similar to that of the *cat* command, but you can move.

Reading large file using *less*

```
less /path/to/file/file_name
```

Copied!

content_copy

The command will provide you with a scrollable view of the content within the file, up to the end of the file content. Scroll down using "Enter", and exit the view by pressing "q".

The rest of this lab will teach you how to create, copy, modify, and remove files and folders.

Start the lab

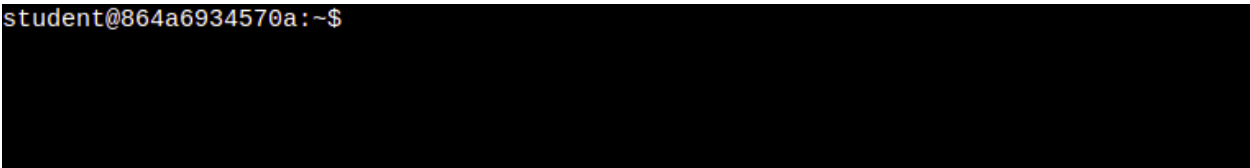
You'll need to start the lab before you can access the materials. To do this, click the green "Start Lab" button at the top of the screen.



Start Lab

After you click the "Start Lab" button, you will see a shell, where you will be performing further steps in the lab. You should have a shell that looks like this:

```
student@864a6934570a:~$
```



Creating directories (folders)

Directories (folders) in Linux are created using the *mkdir* command.

The command takes the directory name as the argument.

Note: The example commands are for reference only.

Example 1

```
mkdir dir_name
```

Copied!

content_copy

Multiple directories can be supplied as arguments, and *mkdir* will create all of them.

Example 2

```
mkdir dir1 dir2 dir3
```

Copied!

content_copy

Parameters

mkdir can take three options:

- **-p:** allow *mkdir* to create parent directories if they don't exist

- -m: (mode) used to set permissions of directories during creation
- -v: run command in verbose mode

Let's take a look at how to use mkdir by going through an example.

In this Linux virtual machine, directory */home/user/Desktop* contains a file called "colors". We'll open the file, and for every line listed in it, we'll create a new folder with that name in the directory

/home/user/Documents

Step 1: Change into working directory.

```
cd /home/user/Documents
```

Copied!

content_copy

```
student@ca721599e80e:~$ cd /home/user/Documents
```

```
student@ca721599e80e:/home/user/Documents$
```

Step 2: Show the contents of the file "colors" within the "Desktop" directory.

```
cat /home/user/Desktop/colors
```

Copied!

content_copy

```
red
```

blue

green

yellow

magenta

Step 3: Create the directories.

```
mkdir red blue green yellow magenta
```

Copied!

content_copy

```
student@ca721599e80e:/home/user/Documents$ mkdir red blue green yellow  
magenta
```

```
student@ca721599e80e:/home/user/Documents$ ls
```

```
Hidden blue green magenta red yellow
```

Click *Check my progress* to verify the objective.

Create directories

Check my progress

Removing empty directories

To remove empty directories, use the *rmdir* command. The name of the directory to be removed is passed as an argument.

Note: The example commands are for reference only.

Example 1

```
rmdir dir_name
```


Copied!

content_copy

Multiple directory names can be passed as arguments, and *rmdir* will remove all of them.

Example 2

```
rmdir dir1 dir2 dir3 dir4
```

Copied!

content_copy

Head's up: *rmdir* only removes empty directories. To remove a non-empty directory, the command *rm*, discussed later in this course, is used.

Options

rmdir takes only one option, which tells it to remove parent directories if they're also empty.

- **-p:** remove parent directories, if they're also empty

Creating files

By default, the touch command is used to change the modification and access times of a file. If the file doesn't exist, the touch command is used to create a file with default permissions.

Let's take a look at an example of how to use the touch command. In the current directory, we can create an empty file called "empty_file":

```
touch empty_file
```

Copied!

```
content_copy
```

The touch command can take the c option to prevent a new file from being created.

Options

- -c: do not create file if it doesn't exist

Copying, moving and deleting files and directories (folders)

cp

The `cp` command is used to make a copy of one or more directories or files. The command takes **at least** one source name and one target name. If the target is a file, then the source must also be a file. A copy of the source will be made with the new name supplied in target. If the target name isn't specified, a copy of source will be made in the target directory under the same name. If a file with the target name already exists in the target directory, it'll be replaced. If the target is an existing directory, then all sources (one or more) will be copied into the target directory. If the target is a directory that doesn't exist, then the source must also be a directory. A copy of the directory and its contents will be made in target under the same name.

Note: The example commands are for reference only.

Example 1

Copy the file "source_file" in the directory `/home/user/` to the directory "duplicates" as "target_file".

```
cp /home/user/source_file /home/user/duplicates/target_file
```

Copied!

content_copy

The duplicates directory now contains a copy of the original file.

mv

The move command is used to move one or more files or directories into a different location, or rename them to a new name. You're required to pass **at least** one source and target file names or directories. The *mv* command follows the rules for existing or non-existing directories or files, as does *cp*.

Example 2

Move the file "source_file" in /home/user/ to the directory "moved_files" and give it the name "target_name".

```
mv /home/user/source_file /home/user/moved_files/target_file
```

Copied!

content_copy

The original directory doesn't contain the file now. It's been moved to the new directory "moved_files".

rm

The *rm* command is used to remove one or more files. You need to supply **at least** one argument to remove.

Example 3

We can remove the duplicate file we created in the directory

"duplicates" using *rm*

```
rm /home/user/duplicates/target_file
```

Copied!

content_copy

Let's see how to copy, move, and rename files by going through a few examples.

In the directory */home/user/Pictures*, we'll take all the hidden files and move them into the directory */home/user/Documents/Hidden*

Step 1: Change into the Pictures directory.

```
cd /home/user/Pictures
```

Copied!

content_copy

```
student@ca721599e80e:~$ cd /home/user/Pictures
```

Step 2: Show the directory contents, including hidden files.

```
ls -a
```

Copied!

content_copy

```
student@ca721599e80e:~$ cd /home/user/Pictures
```

```
student@ca721599e80e:/home/user/Pictures$ ls -a
```

```
. .. .apple .banana .broccoli .milk chocolate egg
```

Step 3: Move the hidden files into the target directory.

```
mv .apple .banana .broccoli .milk /home/user/Documents/Hidden
```

Copied!

content_copy

Click *Check my progress* to verify the objective.

Move hidden files

Check my progress

In the directory `/home/user/Movies`, there's a folder called "Europe Pictures". We'll move this folder into the correct directory for pictures: `/home/user/Pictures`. Note the use of the backslash `"\"` to escape the space between "Europe" and "Pictures" in the directory name, "Europe Pictures".

```
mv /home/user/Movies/Europe\ Pictures /home/user/Pictures
```

Copied!

content_copy

You can also use a "dot" to copy or move files to the current directory.

In the directory `/home/user/Images`, we can move the file

"Vacation.JPG" into the Pictures directory. To do that, we change into

the Pictures directory, then add a "dot" to the *mv* command as the target.

```
cd /home/user/Pictures
```

Copied!

content_copy

```
mv /home/user/Images/Vacation.JPG .
```

Copied!

content_copy

```
student@ca721599e80e:~$ cd /home/user/Pictures
```

```
student@ca721599e80e:/home/user/Pictures$ mv  
/home/user/Images/Vacation.JPG .student@ca721599e80e:/home/user/Pictures$  
ls
```

```
'Europe Pictures' Vacation.JPG chocolate egg
```

Click *Check my progress* to verify the objective.

Move files and folders

Check my progress

Some files in the directory */home/user/Music* need to be cleaned up.

We'll see an example of removing files and directories by removing:

- Best_of_the_90s
- 80s_jams
- Classical

- Rock (folder)

Step 1: Navigate to the Music folder.

```
cd /home/user/Music
```

Copied!

content_copy

Step 2: Remove files.

```
rm Best_of_the_90s 80s_jams Classical
```

Copied!

content_copy

```
student@ca721599e80e:~$ cd /home/user/Music
```

```
student@ca721599e80e:/home/user/Music$ ls
```

```
80s_jams Best_of_the_90s Classical Electronic Rock Smooth_jazz
```

```
student@ca721599e80e:/home/user/Music$ rm Best_of_the_90s 80s_jams  
Classical
```

```
student@ca721599e80e:/home/user/Music$ ls
```

```
Electronic Rock Smooth_jazz
```

Step 3: Remove the directory.

```
rmdir Rock
```

Copied!

content_copy

To remove a directory with content, the *rm* command is used instead of *rmdir*. The option *-r* tells the command to remove the directory, along with its content recursively.

Click *Check my progress* to verify the objective.

Remove files and folders

Check my progress

Note: The example commands are for reference only.

Example 4

```
rm -r non_empty_dir
```

Copied!

content_copy

Searching in files

grep

grep is a super powerful Linux command used to search through files for the occurrence of a string of characters that matches a specified pattern. We can use the command in combination with a bunch of different options and flags for efficient searching.

Options and flags

- -r: search recursively
- -w: match the whole word
- -n: only in line number
- -e: match pattern
- --include and --exclude: include and exclude files in the search
- --include-dir and --exclude-dir: include or exclude directories in the search

Let's take a look at the *grep* command in action. In the directory */home/user/Downloads* of your virtual machine, a number of files exist. We'll find the files that have the word "vacation" in them, and move them to */home/user/Documents*

Step 1: Find files.

```
grep -rw /home/user/Downloads -e "vacation"
```

Copied!

content_copy

```
student@ca721599e80e:~$ grep -rw /home/user/Downloads -e "vacation"  
/home/user/Downloads/Japan:We enjoyed our vacation here.  
/home/user/Downloads/Iceland:We had a great vacation here.
```

Step 2: Move the directories that match into the target directory.

```
mv /home/user/Downloads/Iceland /home/user/Downloads/Japan  
/home/user/Documents
```

Copied!

content_copy

Click *Check my progress* to verify the objective.

Search the files and move them

Check my progress

Editing files

Lots of Linux distributions come with pre-installed text editors. The most popular ones are vi and nano, which will be included in nearly every distribution. Other text editors, like Emacs and Gedit, might also be present. In this lab, we'll modify files using the Nano editor.

You can use the *nano* command to open the Nano editor and modify an existing file, or create a new one. To edit an existing file, we'll first start with opening it.

Note: The example commands are for reference only.

Example:

```
nano /path/to/existing/file
```

Copied!

```
content_copy
```

The command will open the file in the terminal and display the current file contents. To modify, you can edit the content in the terminal, just like a normal editor. The editor is managed using various shortcuts.

To save modifications to the file, use Ctrl+O-:

```
CTRL-O
```

Copied!

```
content_copy
```

Once editing is done, we can close and exit the program using Ctrl+X

```
CTRL-X
```

Copied!

```
content_copy
```

NB: At any point in using the editor, you can get help using Ctrl+G

```
CTRL-G
```

Copied!

content_copy

To exit help mode, use Ctrl+X

CTRL-X

Copied!

content_copy

Alright, now let's **practice** how to edit files using *nano*.

In the current directory, create an empty file called *editor_test.txt*

```
touch editor_test.txt
```

Copied!

content_copy

Open the file with the Nano editor.

```
nano editor_test.txt
```

Copied!

content_copy

```
<br>
```

```
<br>
```

```
<br>
```

```
<br>
```

```
<br>
```

```
<br>
```

```
<br>
```

```
<br>
```

```
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell
```

Add content to the file. (In this case, we add five lines, each separated by an empty line.)

```
Knock Knock
```

```
<br>
```

```
Who's there ?
```

```
<br>
```

```
very long pause...
```

```
<br>
```

```
java.
```

```
<br>
```

```
: -)
```

```
<br>
```

```
<br>
```

```
<br>
```

```
<br>
```

```
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell
```

Save the file by hitting Ctrl+O

CTRL-O

Copied!

content_copy

```
Knock Knock
```

```
<br>
```

```
Who's there ?
```

```
<br>
```

```
very long pause...
```

```
<br>
```

```
java.
```

```
<br>
```

```
:~)
```

```
<br>
```

```
<br>
```

```
<br>
```

```
<br>
```

```
File Name to Write: editor_test.txt
```

```
^G Get Help      M-D DOS Format   M-A Append      M-B Backup File
```

```
^C Cancel        M-M Mac Format   M-P Prepend      ^T To Files
```

You'll need to confirm the file that you want to write the content to by hitting Enter. After this, exit the program by hitting Ctrl+X

CTRL-X

Copied!

content_copy

That's it! You've successfully created and modified a file.

Conclusion

In this lab, we've gone through the basics of creating, modifying, copying, and removing files and folders in Linux. As always, you can learn more about each of the commands we've covered by using the *man* command. Make sure to practice these commands so that you get comfortable using them.