



Lab 3.1.2.7 – Getting Familiar with the Linux Shell



This lab has been updated for use on NETLAB+.
www.netdevgroup.com

Introduction

In this lab, you will use the Linux command line to manage files and folders and perform some basic administrative tasks.

Part 1: Shell Basics

The shell is the term used to refer to the command interpreter in Linux. Also known as Terminal, Command Line and Command Prompt, the shell is a very powerful way to interact with a Linux computer.

Step 1: Access the Command Line

- Log on to the **CyberOps Workstation** VM as the **analyst** using the password **cyberops**. The account **analyst** is used as the example user account throughout this lab.
- To access the command line, click the **terminal** icon located in the **Dock**, at the bottom of VM screen. The terminal emulator opens.



Step 2: Display Manual Pages from the command line.

You can display command line help using the **man** command. A man page, short for manual page, is a built-in documentation of the Linux commands. A **man** page provides detailed information about a given command and all its available options.

- To learn more about the man page, type:

```
[analyst@secOps ~]$ man man
```

Name a few sections that are included in a man page.

There are many in the page some of those Name, Synopsis, Description, Warnings, Sections, Conditions, Errors, Return Values, Strings and some more.

- Type **q** to exit the man page.

- c. Use the **man** command to learn more about the **cp** command:

```
[analyst@secOps ~]$ man cp
```

What is the function of the **cp** command?

Copy the files and directories from one to other location in the local file system.

What command would you use to find out more information about the *pwd* command? What is the function of the *pwd* command?

Which give the information about current directory of the file.

Step 3: Create and change directories.

In this step, you will use the change directory (*cd*), make directory (*mkdir*), and list directory (*ls*) commands.

Note: A directory is another word for folder. The terms directory and folder are used interchangeably throughout this lab.

- a. Type **pwd** at the prompt.

```
[analyst@secOps ~]$ pwd
```

What is the current directory?

The current directory of the file is /home/analyst.

- b. Navigate to the **/home/analyst** directory if it is not your current directory. Type **cd /home/analyst**

```
[analyst@secOps ~]$ cd /home/analyst
```

- c. Type **ls -l** at the command prompt to list the files and folders that are in the current folder. Standing for list, the **-l** option displays file size, permissions, ownership, date of creation and more.

```
[analyst@secOps ~]$ ls -l
```

```
total
20
drwxr-xr-x 2 analyst analyst 4096 Sep 26 2014 Desktop
drwx----- 3 analyst analyst 4096 Jul 14 11:28 Downloads
drwxr-xr-x 8 analyst analyst 4096 Jul 25 16:27 lab.support.files
drwxr-xr-x 2 analyst analyst 4096 Mar 3 15:56 second_drive
```

- d. In the current directory, use the **mkdir** command to create three new folders: **cyops_folder1**, **cyops_folder2**, and **cyops_folder3**. Type **mkdir cyops_folder1** and press **Enter**. Repeat these steps to create **cyops_folder2** and **cyops_folder3**.

```
[analyst@secOps ~]$ mkdir cyops_folder1
```

```
[analyst@secOps ~]$ mkdir cyops_folder2
```

```
[analyst@secOps ~]$ mkdir cyops_folder3
```

```
[analyst@secOps ~]$
```

- e. Type **ls -l** to verify that the folders have been created:

```
[analyst@secOps ~]$ ls -l
total
32
drwxr-xr-x 2 analyst analyst 4096 Aug 16 15:01 cyops_folder1
drwxr-xr-x 2 analyst analyst 4096 Aug 16 15:02 cyops_folder2
drwxr-xr-x 2 analyst analyst 4096 Aug 16 15:02 cyops_folder3
drwxr-xr-x 2 analyst analyst 4096 Sep 26 2014 Desktop
drwx----- 3 analyst analyst 4096 Jul 14 11:28 Downloads
drwxr-xr-x 8 analyst analyst 4096 Jul 25 16:27 lab.support.files
drwxr-xr-x 2 analyst analyst 4096 Mar  3 15:56 second_drive
-rw-r--r-- 1 analyst analyst 254 Aug 16 13:38 space.txt
```

- f. Type **cd /home/analyst/cyops_folder3** at the command prompt and press **Enter**.

```
[analyst@secOps ~]$ cd /home/analyst/cyops_folder3
[analyst@secOps cyops_folder3]$
```

Which folder are you in now?

The current file directory is `cyops_folder3`.

Note: In the `[analyst@secOps ~]$` prompt above: The tilde symbol `~` represents the current user's home directory. In this example, the current user's home directory is `/home/analyst`. After the `cd /home/analyst/cyops_folder3` command, the current user's home directory is now `/home/analyst/cyops_folder3`.

Note: `$` (dollar sign) symbol indicates regular user privilege. If a `#` (hashtag or pound sign) symbol is displayed at the prompt, it indicates elevated privilege (root user).

Note: While these symbols, conventions and main concepts remain the same, the prompt of a terminal window is highly customizable in Linux. Therefore, the prompt structure seen in the *CyberOps Workstation* VM will likely differ from the prompt in other Linux installations.

Challenge: Type the command `cd ~` and describe what happens. Why did this happen?

The directory is changed to the home directory. `cd` changes to the current user's home.

- g. Use the **mkdir** command to create a new folder named **cyops_folder4** inside the **cyops_folder3** folder:

```
[analyst@secOps ~]$ mkdir /home/analyst/cyops_folder3/cyops_folder4
[analyst@secOps ~]$
```

- h. Use the **ls -l** command to verify the folder creation.

```
analyst@secOps ~]$ ls -l /home/analyst/cyops_folder3
total 4
drwxr-xr-x 2 analyst analyst 4096 Aug 16 15:04 cyops_folder4
```

- i. Up to this point, we have been using *full paths*. Full path is the term used when referring to paths that always start at the root (`/`) directory. It is also possible to work with *relative paths*. Relative paths reduce

the amount of text to be typed. To understand relative paths, we must understand the `.` and `..` (dot and double dot) directories. From the `cyops_folder3` directory, issue a `ls -la`:

```
analyst@secOps ~]$ ls -la /home/analyst/cyops_folder3
total 12
drwxr-xr-x  3 analyst analyst 4096 Aug 16 15:04 .
drwxr-xr-x 20 analyst analyst 4096 Aug 16 15:02 ..
..
drwxr-xr-x  2 analyst analyst 4096 Aug 16 15:04 cyops_folder4
```

The `-a` option tells `ls` to show all files. Notice the `.` and `..` listings shown by `ls`. These listings are used by the operating system to track the current directory (`.`) and the parent directory (`..`). You can see the use of the `.` and `..` when using the `cd` command to change directories. Using the `cd` command to change the directory to the `.` directory incurs no visible directory change as the `.` points to the current directory itself.

- j. Change the current directory to `/home/analyst/cyops_folder3`:

```
[analyst@secOps ~]$ cd /home/analyst/cyops_folder3
[analyst@secOps cyops_folder3]$
```

- k. Type `cd .`

```
[analyst@secOps cyops_folder3]$ cd .
[analyst@secOps cyops_folder3]$
```

What happens?

When we enter `cd` then it will change to the current directory.

- l. Changing the directory to the `..` directory, will change to the directory that is one level up. This directory is also known as *parent directory*. Type `cd ..`

```
[analyst@secOps cyops_folder3]$ cd ..
[analyst@secOps ~]$
```

What happens?

The directory was changed to `/home/analyst/cyops_folder3` is the main directory.

What would be the current directory if you issued the `cd ..` command at `[analyst@secOps ~]$`?
/home

What would be the current directory if you issued the `cd ..` command at `[analyst@secOps home]$`?
It is moving to root of the file system.

What would be the current directory if you issued the `cd ..` command at `[analyst@secOps /]$`?
/, It is moving to root of the file system.

Step 4: Redirect Outputs.

Another powerful command line operator in Linux is known as *redirect*. Represented by the **>** symbol, this operator allows the output of a command to be redirected to some location other than the current terminal window (the default).

- a. Use the **cd** command to change to the **/home/analyst/ (~)** directory:

```
[analyst@secOps ~]$ cd /home/analyst/
[analyst@secOps ~]$
```

- b. Use the **echo** command to echo a message. Because no output was defined, *echo* will output to the current terminal window:

```
analyst@secOps ~]$ echo This is a message echoed to the terminal by echo.
This is a message echoed to the terminal by echo.
```

- c. Use the **>** operator to redirect the output of *echo* to a text file instead of to the screen:

```
analyst@secOps ~]$ echo This is a message echoed to the terminal by echo. >
some_text_file.txt
No output was shown. Is that expected?
Yes, Output was some_text_file.
```

- d. Notice that even though the *some_text_file.txt* file did not exist, it was automatically created to receive the output generated by *echo*. Use the **ls -l** command to verify if the file was really created:

```
[analyst@secOps ~]$ ls -l some_text_file.txt
-rw-r--r-- 1 analyst analyst 50 Feb 24 16:11 some_text_file.txt
```

- e. Use the **cat** command to display the contents of the *some_text_file.txt* text file:

```
[analyst@secOps ~]$ cat some_text_file.txt
This is a message echoed to the terminal by echo.
```

- f. Use the **>** operator again to redirect a different *echo* output of *echo* to the *some_text_file.txt* text file:

```
analyst@secOps ~]$ echo This is a DIFFERENT message, once again echoed to the
terminal by echo. > some_text_file.txt
```

- g. Once again, use the **cat** command to display the contents of the *some_text_file.txt* text file:

```
[analyst@secOps ~]$ cat some_text_file.txt
This is a DIFFERENT message, once again echoed to the terminal by echo.
What happened to the text file? Explain.
```

When we enter **>** it will kill content of the text file and present text file will be replaced with the new message.

Step 5: Redirect and Append to a Text File.

- a. Similar to the **>** operator, the **>>** operator also allows for redirecting data to files. The difference is that **>>** appends data to the end of the referred file, keeping the current contents intact. To append a message to the *some_text_file.txt*, issue command below:

```
[analyst@secOps ~]$ echo This is another line of text. It will be APPENDED to
the output file. >> some_text_file.txt
```

- b. Use the `cat` command to display the contents of the `some_text_file.txt` text file yet again:

```
[analyst@secOps ~]$ cat some_text_file.txt
```

This is a DIFFERENT message, once again echoed to the terminal by echo.

This is another line of text. It will be APPENDED to the output file.

What happened to the text file? Explain.

When we `cat` command the new message was appended at the EOF.

Step 6: Work with hidden files in Linux.

- a. In Linux, files with names that begin with a '.' (single dot) are not shown by default. While dot-files have nothing else special about them, they are called hidden files because of this feature. Examples of hidden files are `.file5`, `.file6`, `.file7`.

Note: Do not confuse dot-files with the current directory indicator "." symbol. Hidden file names begin with a dot (period), followed by more characters while the dot directory is a hidden directory comprised of only a single dot.

- b. Use `ls -l` to display the files stored in the analyst home directory.

```
[analyst@secOps ~]$ ls -l
```

How many files are displayed?

I just see 4 files displayed.

- c. Use the `ls -la` command to display all files in the home directory of analyst, including the hidden files.

```
[analyst@secOps ~]$ ls -la
```

How many files are displayed now, more than before? Explain.

I can many more files are displayed.

Is it possible to hide entire directories by adding a dot before its name as well? Are there any directories in the output of `ls -la` above?

Yes, I can see that there are many hidden directories in the ouput.

Give three examples of hidden files shown in the output of `ls -la` above.

.atftp-history, bash-history, .bashrc, .dmrc, .fehbg and many more.

- d. Type the `man ls` command at the prompt to learn more about the `ls` command.

```
[analyst@secOps ~]$ man ls
```

- e. Use the down arrow key (one line at a time) or the space bar (one page at a time) to scroll down the page and locate the `-a` used above and read its description to familiarize yourself with the `ls -a` command.

Part 2: Copying, Deleting, and Moving Files

Step 1: Copying Files

- a. The `cp` command is used to copy files around the local file system. When using `cp`, a new copy of the file is created and placed in the specified location, leaving the original file intact. The first parameter is the source file and the second is the destination. Issue the command below to copy **some_text_file.txt** from the home directory to the **cyops_folder2** folder:

```
[analyst@secOps ~]$ cp some_text_file.txt cyops_folder2/
```

Identify the parameters in the `cp` command above. What are the source and destination files? (use full paths to represent the parameters)

Source file : /home/analyst/some_text_file.txt.

Destination file : home/analyst/cyops_folder2/some_text_file.txt.

- b. Use the `ls` command to verify that *some_text_file.txt* is now in *cyops_folder2*:

```
[analyst@secOps ~]$ ls cyops_folder2/
some_text_file.tx
t
```

- c. Use the `ls` command to verify that *some_text_file.txt* is also in the home directory:

```
[analyst@secOps ~]$ ls -l
total
36
drwxr-xr-x 2 analyst analyst 4096 Aug 16 15:01 cyops_folder1
drwxr-xr-x 2 analyst analyst 4096 Aug 16 15:11 cyops_folder2
drwxr-xr-x 3 analyst analyst 4096 Aug 16 15:04 cyops_folder3
drwxr-xr-x 2 analyst analyst 4096 Sep 26 2014 Desktop
drwx----- 3 analyst analyst 4096 Jul 14 11:28 Downloads
drwxr-xr-x 8 analyst analyst 4096 Jul 25 16:27 lab.support.files
drwxr-xr-x 2 analyst analyst 4096 Mar 3 15:56 second_drive
-rw-r--r-- 1 analyst analyst 142 Aug 16 15:09 some_text_file.txt
-rw-r--r-- 1 analyst analyst 254 Aug 16 13:38 space.txt
```

Step 2: Deleting Files and Directories

- a. Use the `rm` command to remove files. Issue the command below to remove the file **some_text_file.txt** from the home directory. The `ls` command is then used to show that the file **some_text_file.txt** has been removed from the home directory:

```
[analyst@secOps ~]$ rm some_text_file.txt
[analyst@secOps ~]$ ls -l
```

```
total
32
drwxr-xr-x 2 analyst analyst 4096 Aug 16 15:01 cyops_folder1
drwxr-xr-x 2 analyst analyst 4096 Aug 16 15:11 cyops_folder2
drwxr-xr-x 3 analyst analyst 4096 Aug 16 15:04 cyops_folder3
drwxr-xr-x 2 analyst analyst 4096 Sep 26 2014 Desktop
drwx----- 3 analyst analyst 4096 Jul 14 11:28 Downloads
drwxr-xr-x 8 analyst analyst 4096 Jul 25 16:27 lab.support.files
drwxr-xr-x 2 analyst analyst 4096 Mar 3 15:56 second_drive
```

- b. In Linux, directories are seen as a type of file. As such, the `rm` command is also used to delete directories but the `-r` (recursive) option must be used. Notice that all files and other directories inside a given directory are also deleted when deleting a parent directory. Issue the command below to delete the **cyops_folder1** folder and its contents:

```
[analyst@secOps ~]$ rm -r cyops_folder1
[analyst@secOps ~]$ ls -l
total
28
drwxr-xr-x 2 analyst analyst 4096 Aug 16 15:11 cyops_folder2
drwxr-xr-x 3 analyst analyst 4096 Aug 16 15:04 cyops_folder3
drwxr-xr-x 2 analyst analyst 4096 Sep 26 2014 Desktop
drwx----- 3 analyst analyst 4096 Jul 14 11:28 Downloads
drwxr-xr-x 8 analyst analyst 4096 Jul 25 16:27 lab.support.files
drwxr-xr-x 2 analyst analyst 4096 Mar 3 15:56 second_drive
```

Step 3: Moving Files and Directories

- a. Moving files works similarly to copying files. The difference is that moving a file removes it from its original location. Use the `mv` commands to move files around the local filesystem. Like the `cp` commands, the `mv` command also requires source and destination parameters. Issue the command below to move the **some_text_file.txt** from **/home/analyst/cyops_folder2** back to the home directory:

```
[analyst@secOps ~]$ mv cyops_folder2/some_text_file.txt .
[analyst@secOps ~]$ ls -l cyops_folder2/
total
0
[analyst@secOps ~]$ ls -l /home/analyst/
total
32
drwxr-xr-x 2 analyst analyst 4096 Aug 16 15:13 cyops_folder2
drwxr-xr-x 3 analyst analyst 4096 Aug 16 15:04 cyops_folder3
drwxr-xr-x 2 analyst analyst 4096 Sep 26 2014 Desktop
drwx----- 3 analyst analyst 4096 Jul 14 11:28 Downloads
drwxr-xr-x 8 analyst analyst 4096 Jul 25 16:27 lab.support.files
drwxr-xr-x 2 analyst analyst 4096 Mar 3 15:56 second_drive
-rw-r--r-- 1 analyst analyst 142 Aug 16 15:11 some_text_file.txt
```

Why was the dot (".") used as the destination parameter for `mv`?

The current directory /home/analyst Here (. Dot) In the sense that it is the destination parameter `mv` should move the file to the current directory.

- b. The `mv` command can also be used to move entire directories and the files they contain. To move the **cyops_folder3** (and all the files and directories it contains) into **cyops_folder2**, use the command below:

```
[analyst@secOps ~]$ mv cyops_folder3/ cyops_folder2/
[analyst@secOps ~]$ ls -l /home/analyst/
total 28
```



```
drwxr-xr-x 3 analyst analyst 4096 Aug 16 15:15 cyops_folder2
drwxr-xr-x 2 analyst analyst 4096 Sep 26 2014 Desktop
drwx----- 3 analyst analyst 4096 Jul 14 11:28 Downloads
drwxr-xr-x 8 analyst analyst 4096 Jul 25 16:27 lab.support.files
drwxr-xr-x 2 analyst analyst 4096 Mar 3 15:56 second_drive
-rw-r--r-- 1 analyst analyst 142 Aug 16 15:11 some_text_file.txt
-rw-r--r-- 1 analyst analyst 254 Aug 16 13:38 space.txt
```

- c. Use the ls command to verify that the **cyops_folder3** directory was correctly moved to **cyops_folder2**.

```
[analyst@secOps ~]$ ls -l cyops_folder2/
total 4
drwxr-xr-x 3 analyst analyst 4096 Feb 27 11:47 cyops_folder3
```

Reflection

What are the advantages of using the Linux command line?

When we use these commands we can save a time and these helps us to configuration issue also has good flexibility and performance .
