Lab - Getting Familiar with the Linux Shell

Instructions

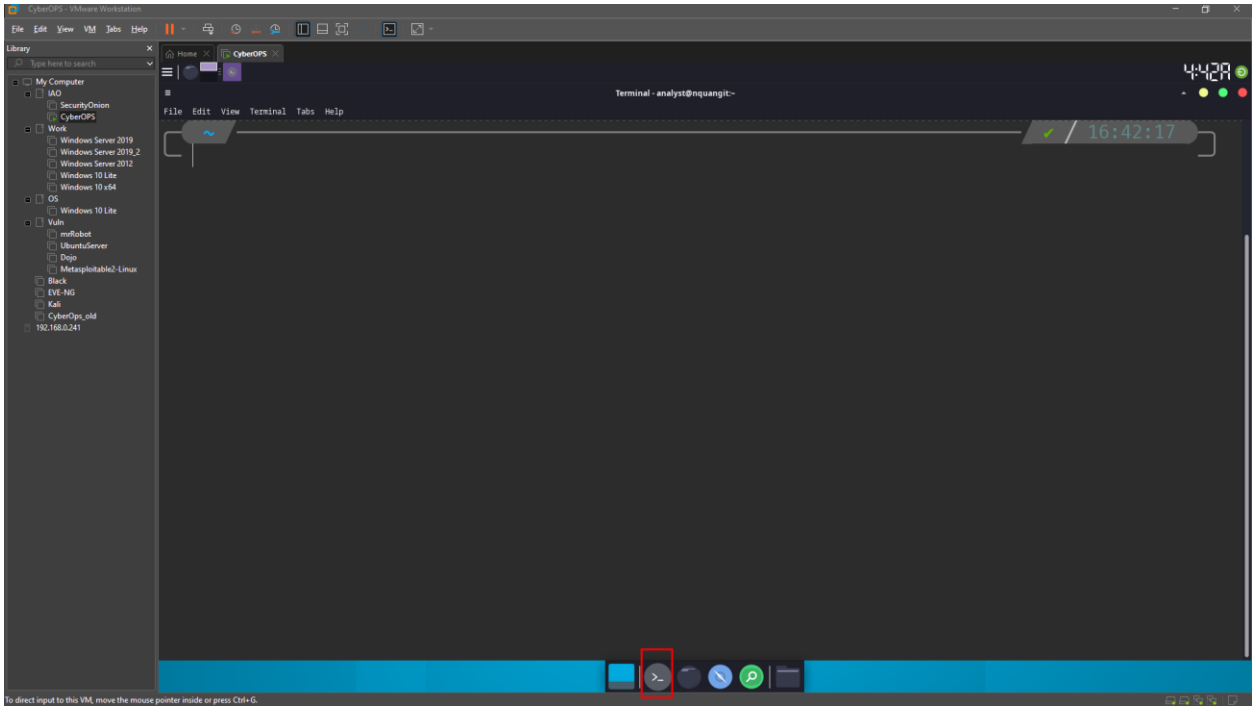
Shell Basics

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 very powerful way to interact with a Linux computer.

Access the Command Line

a. 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.

b. To access the command line, click the **terminal** icon located in the Dock, at the bottom of VM screen. The terminal emulator opens.

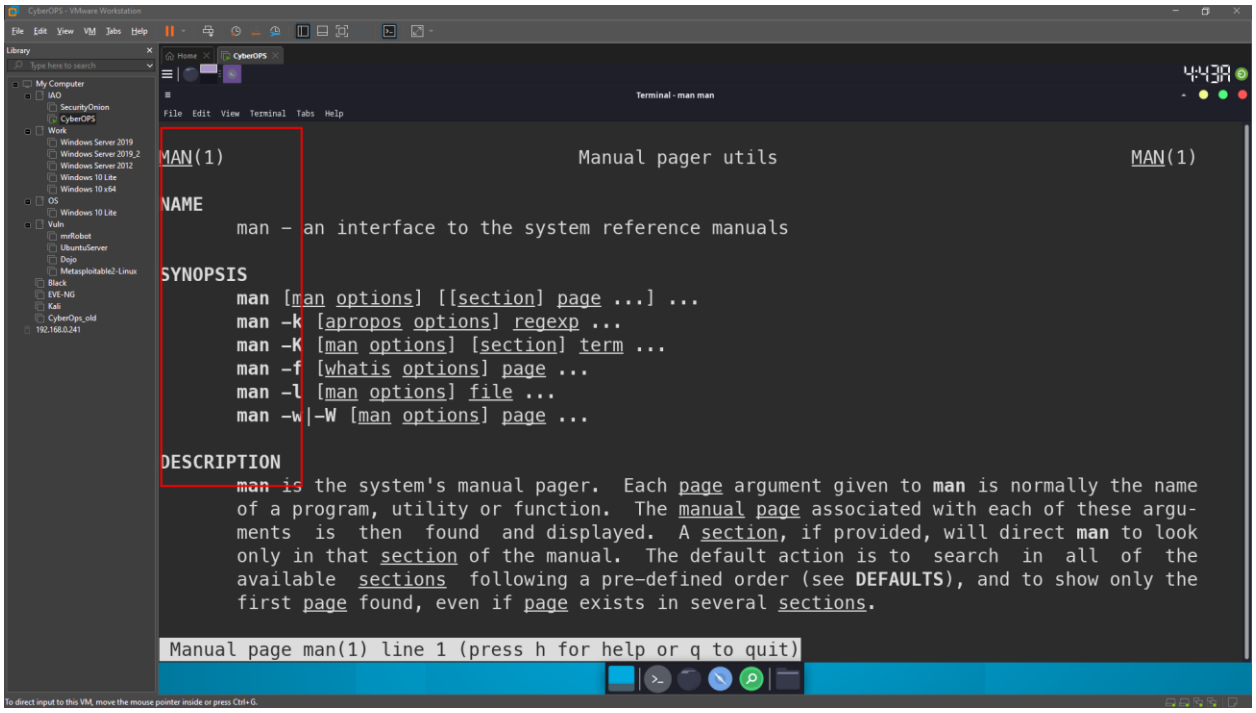


Display Manual Pages from the command line.

a. To learn more about the man page, type:

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

NAME, SYNOPSIS, DESCRIPTION, …

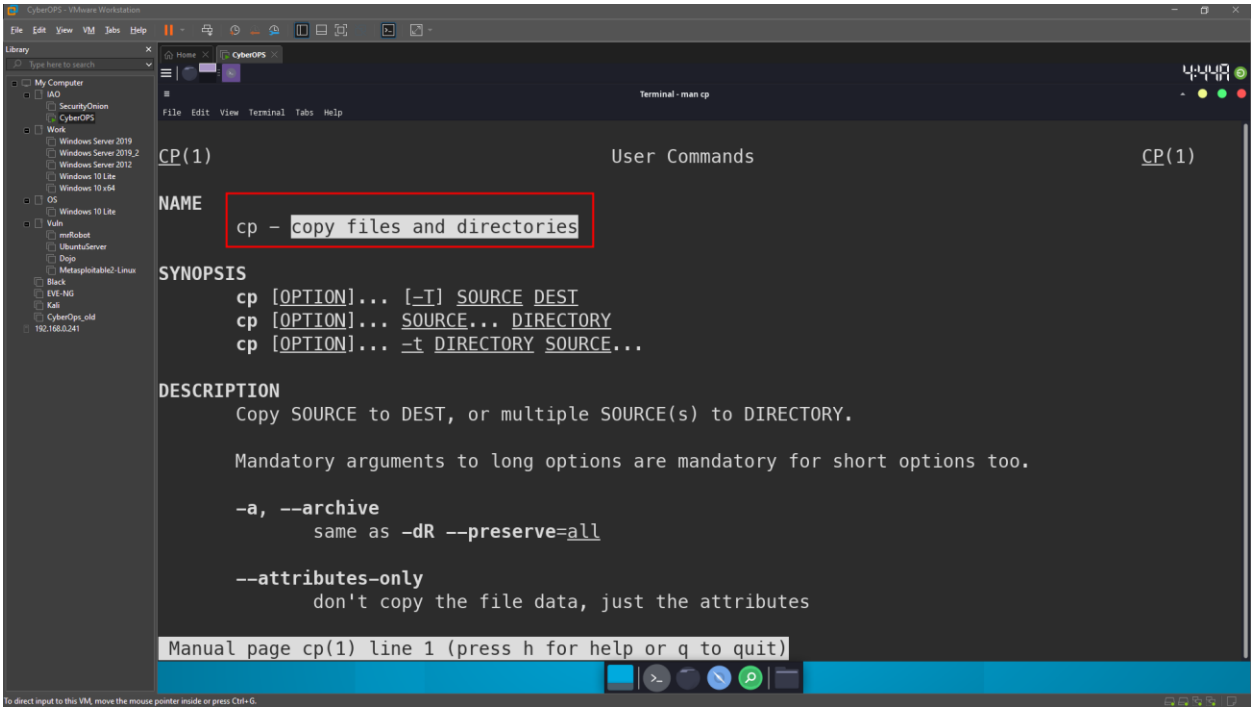


b. Type **q** to exit the man page.

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

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

copy files and directories.



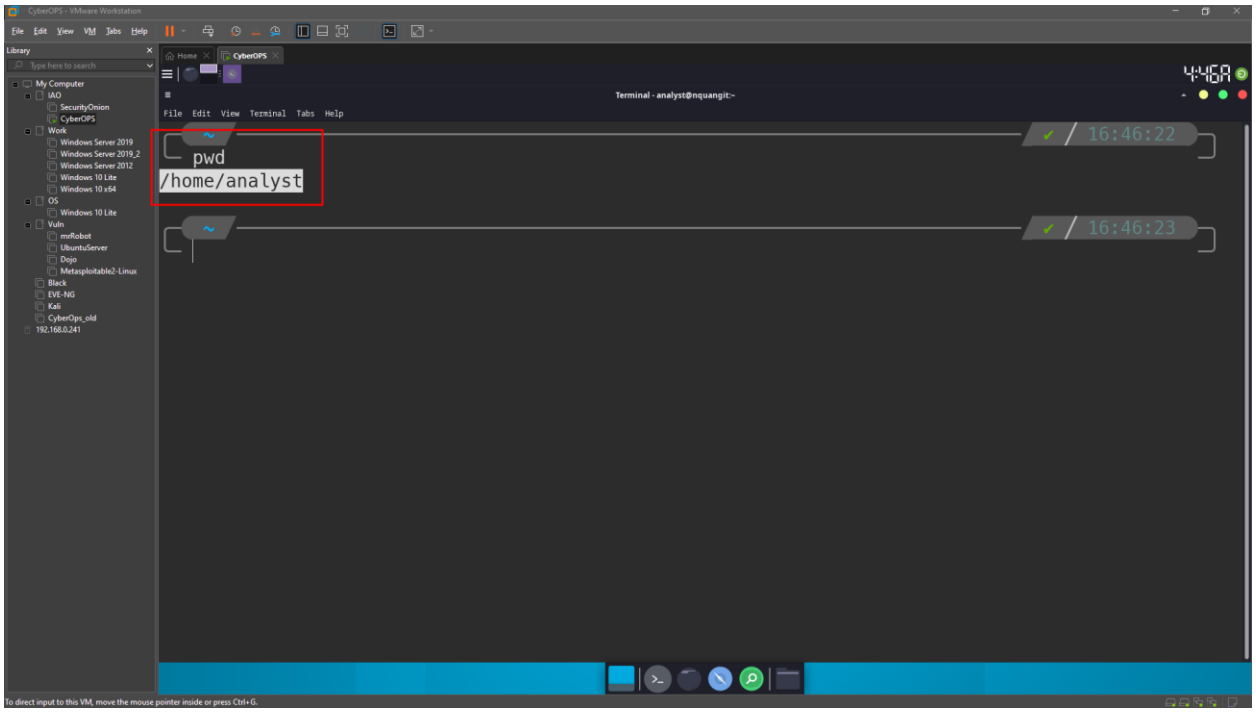
What command would you use to find out more information about the **pwd** command? What is the function of the **pwd** command?

Use “man pwd” command to find out more information.

Function of the **pwd** command: pwd - print name of current/working directory.

Create and change directories.

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

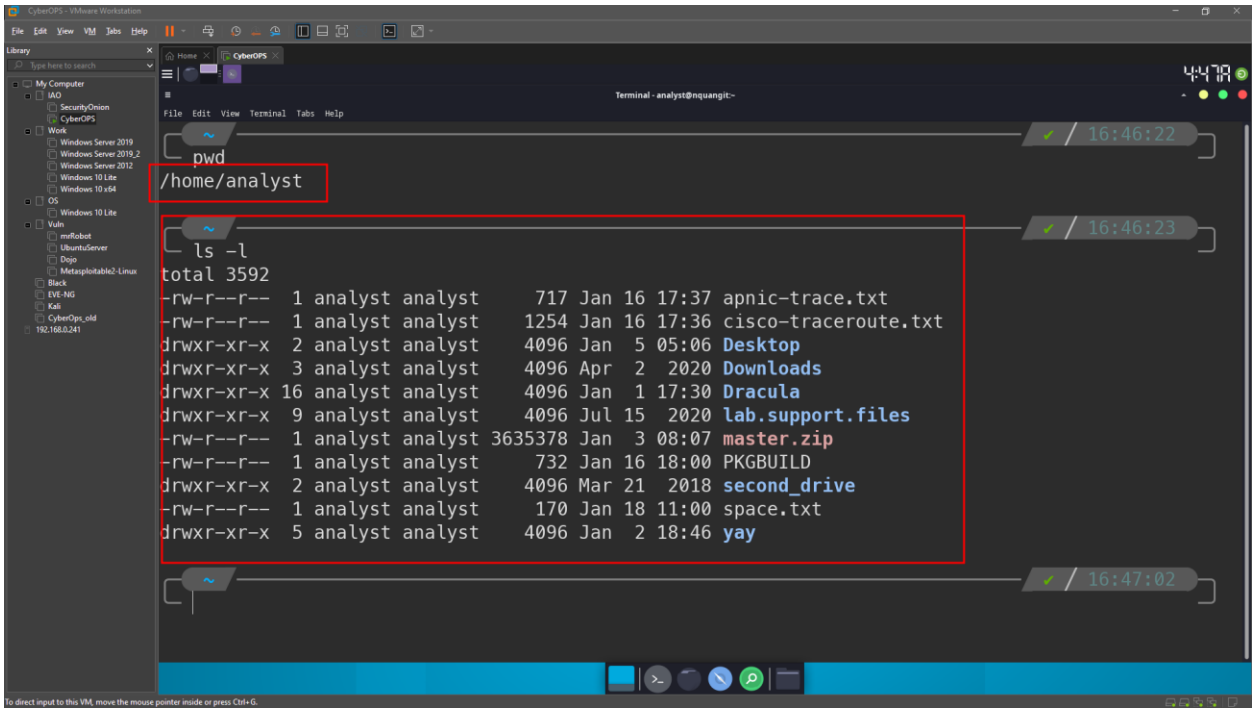


What is the current directory?

~ or /home/analyst

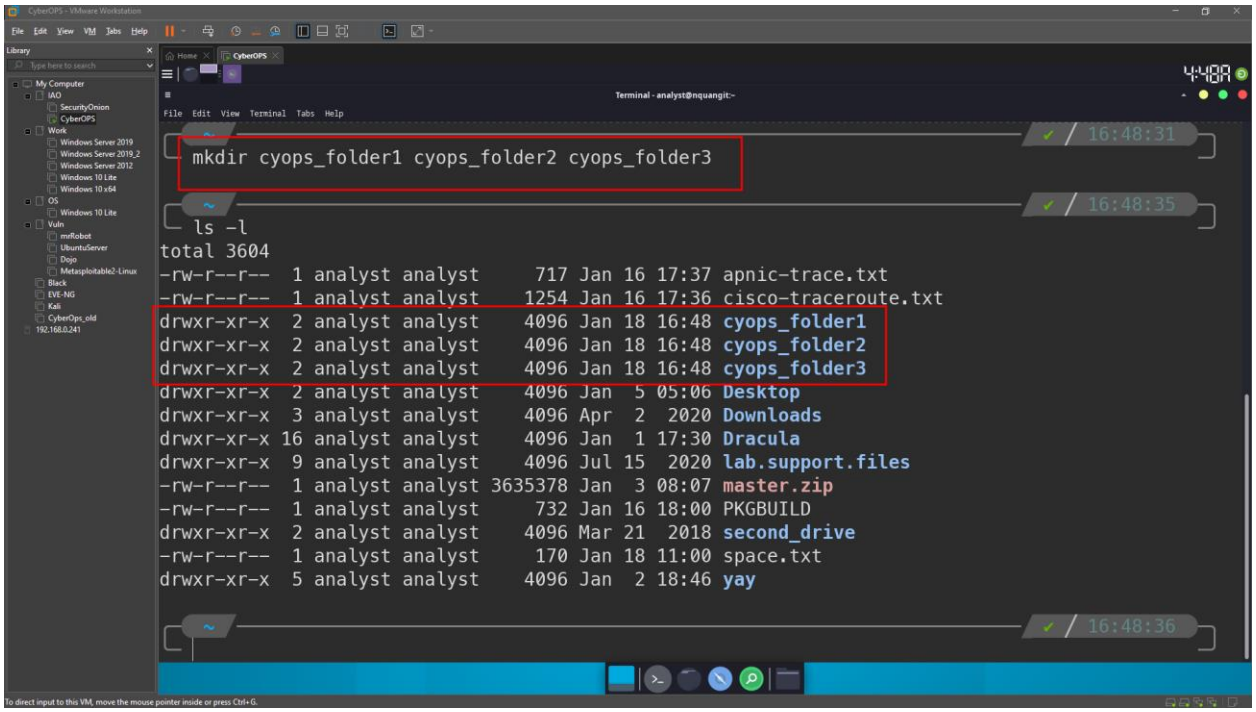
b. Navigate to the **/home/analyst** directory if it is not your current directory. Type **cd /home/analyst**

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.

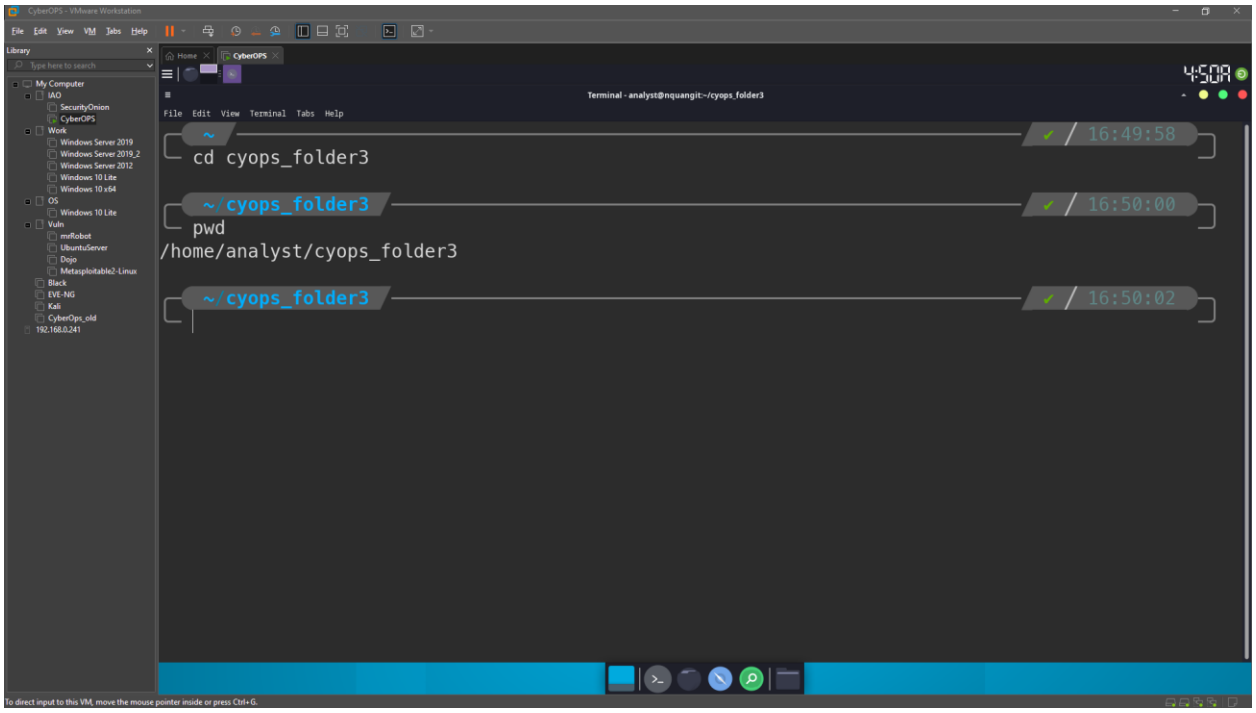


d. In the current directory, use the **mkdir** command to create three new folders: **cyops_folder1**, **cyops_folder2**, and **cyops_folder3**. Type **mkdir cyops_folder1** and press **Enter**. Repeat these steps to create **cyops_folder2** and **cyops_folder3**.

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



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



Which folder are you in now?

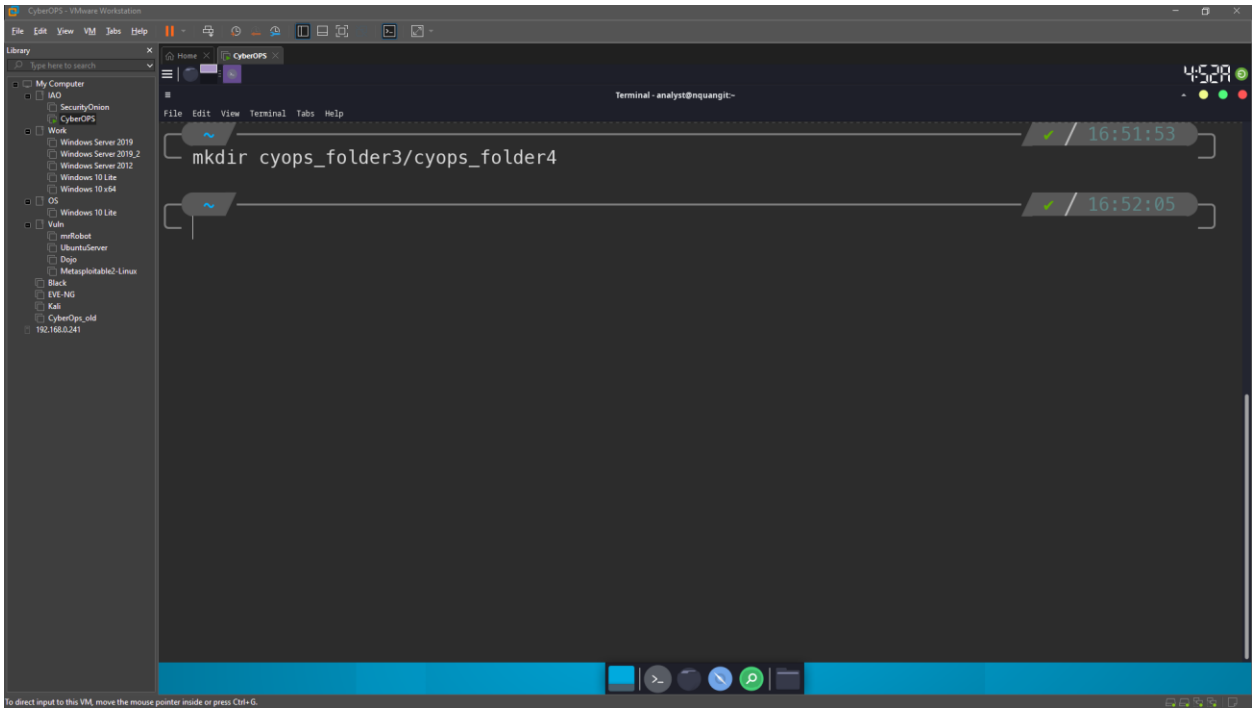
/home/analyst/cyops_folder3 or ~/cyops_folder3

Challenge: Type the command cd ~ and describe what happens.

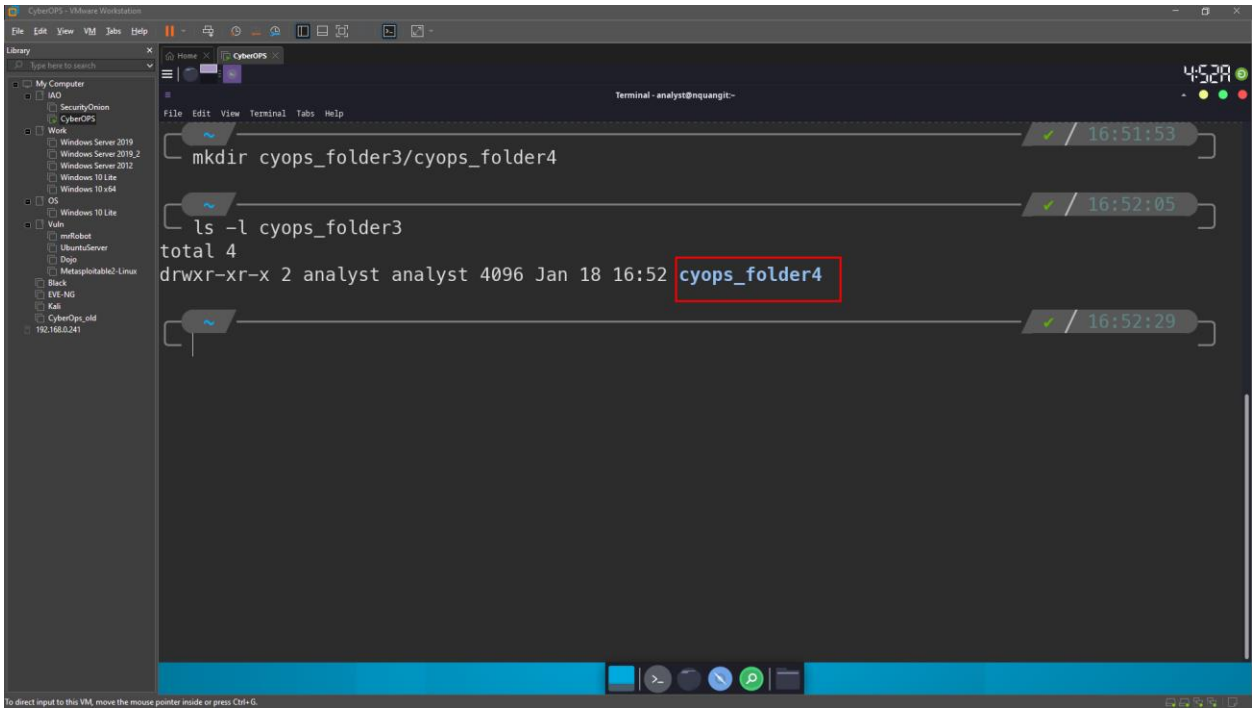
Why did this happen?

The terminal change the current working folder to the user's home directory (~ or /home/analyst)

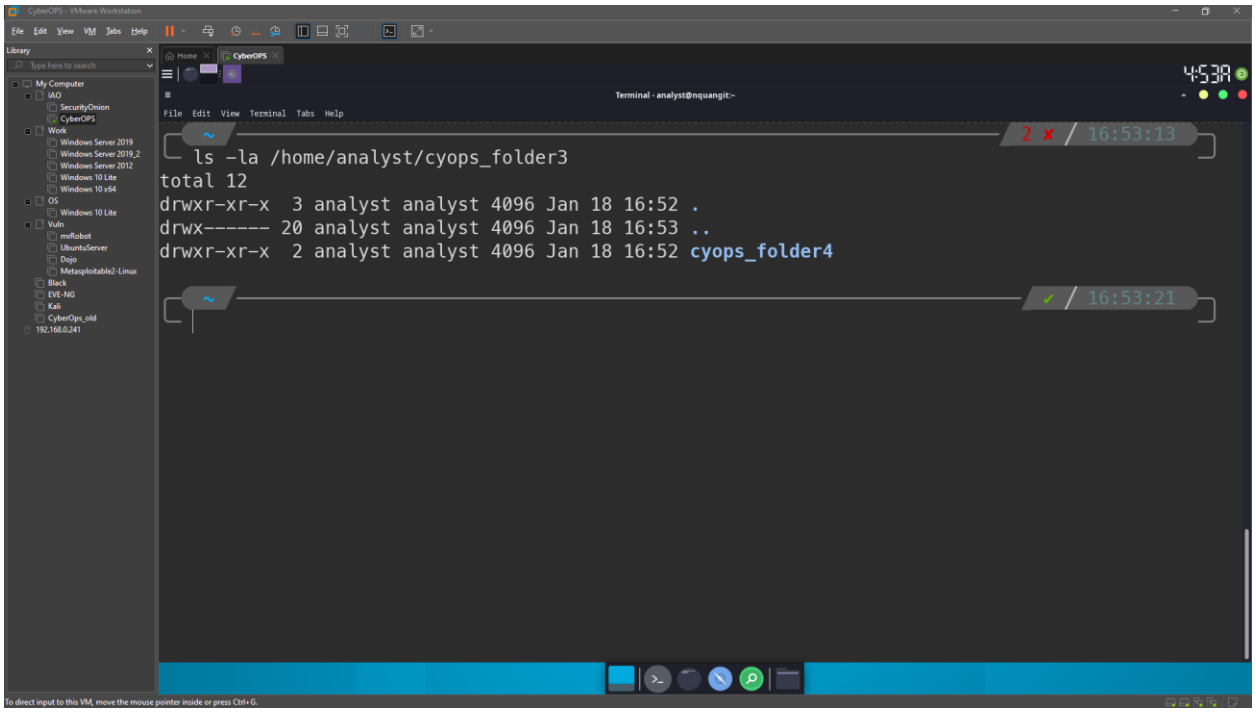
g. Use the **mkdir** command to create a new folder named **cyops_folder4** inside the **cyops_folder3** folder:



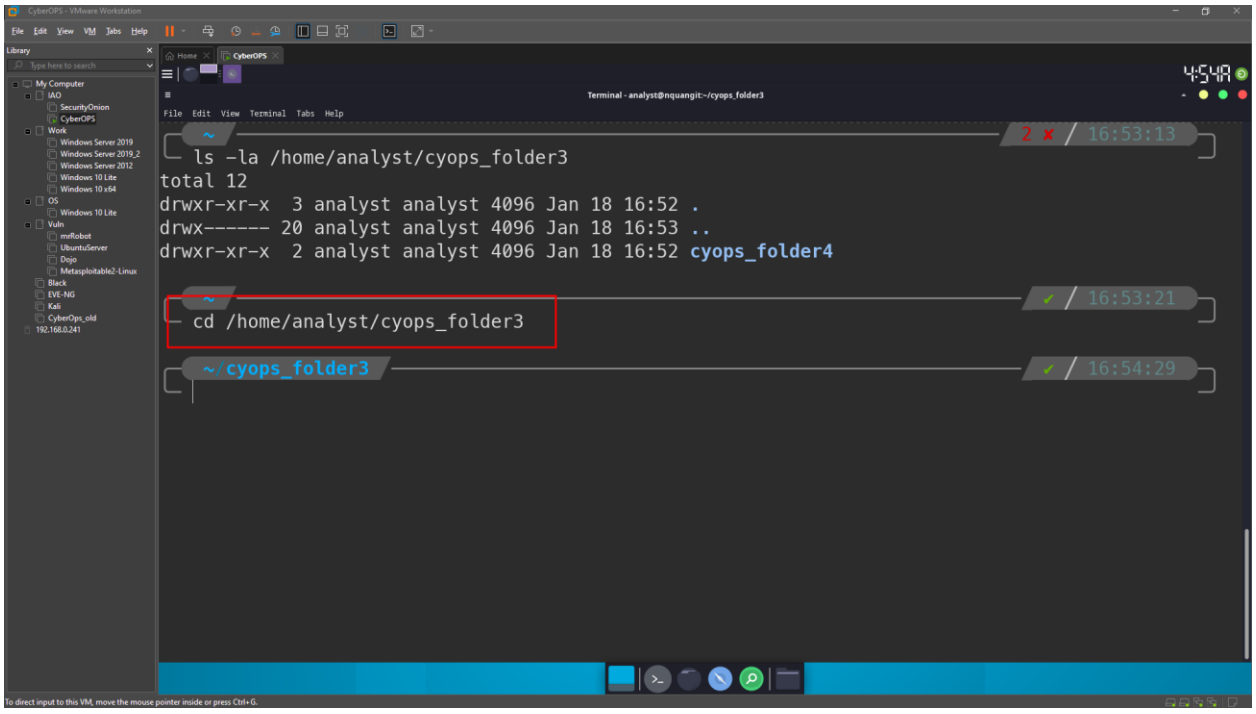
h. Use the **ls -l** command to verify the folder creation.



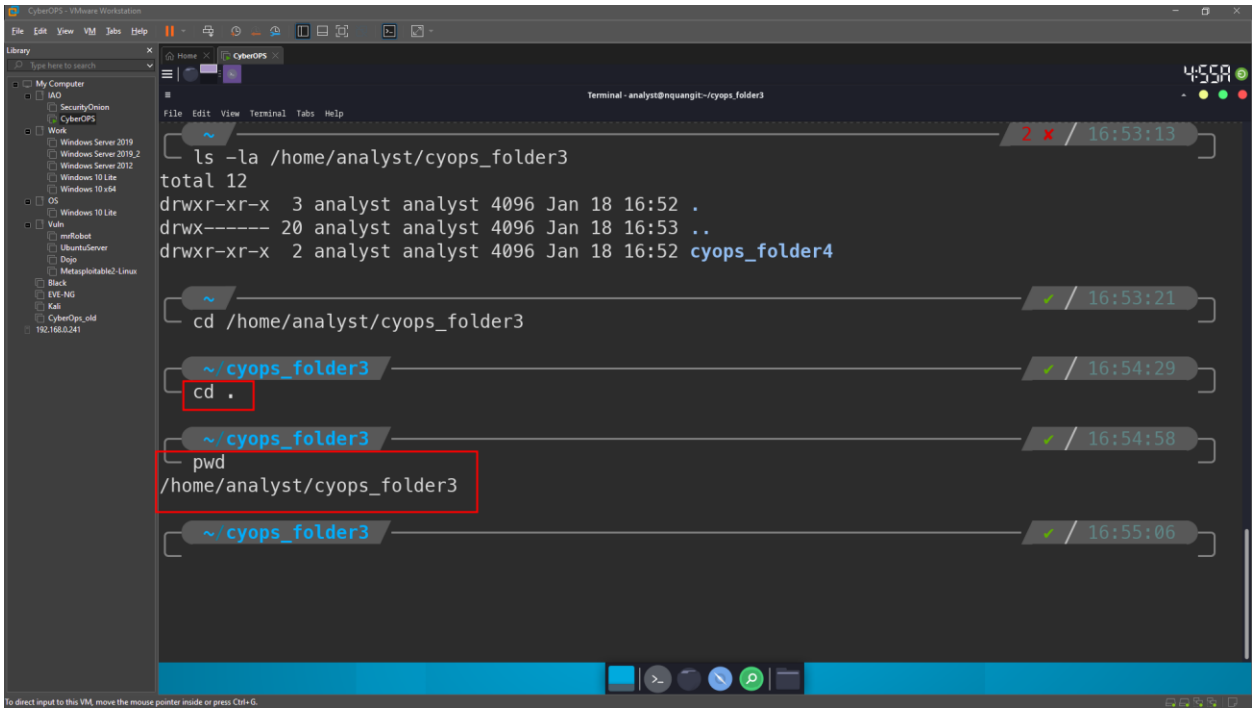
i. Up to this point, we have been using *full or absolute paths.* Absolute path is the term used when referring to paths that always start at the root (/) directory. It is also possible to work with *relative paths.* Relative paths reduce the amount of text to be typed. To understand relative paths, we must understand the **.** and **..** (dot and double dot) directories. From the **cyops_folder3** directory, issue a **ls –la**:



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



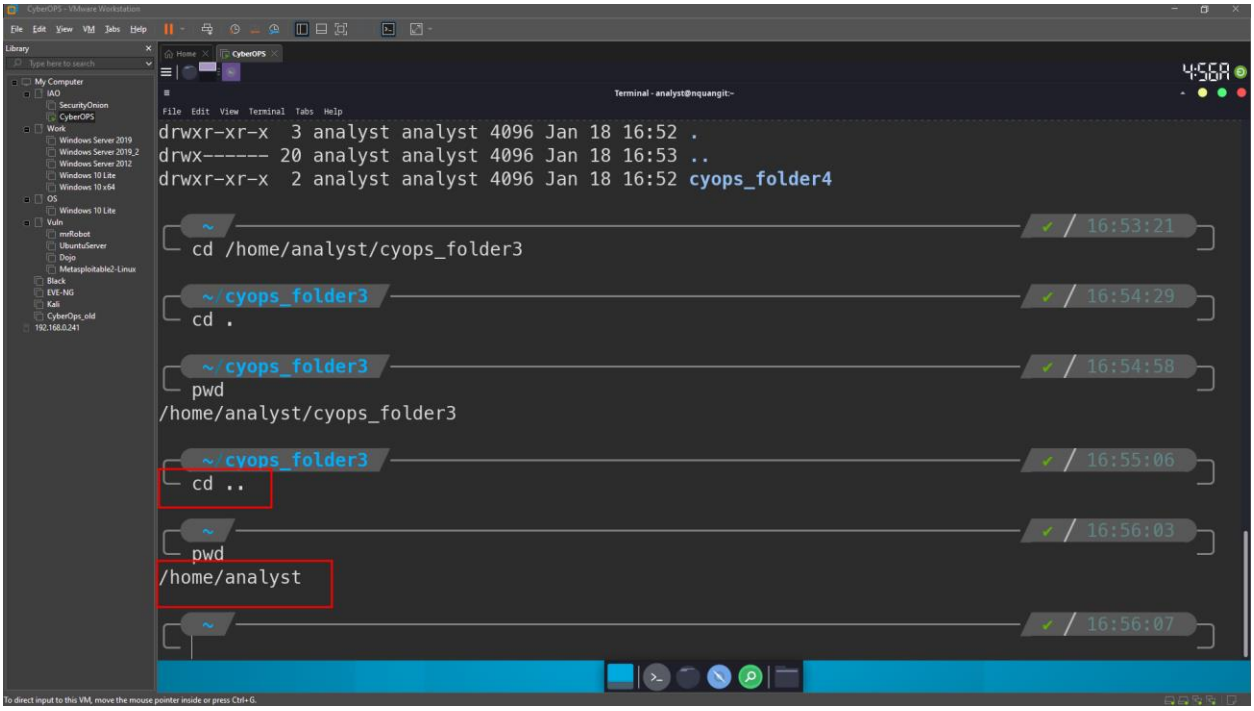
k. Type **cd .**



What happens?

It still in the **cyops_folder3** directory and with change to another.

I. 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 ..`**



What happens?

The directory changed to the **cyops_folder3** 's parent: **/home/analyst**

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

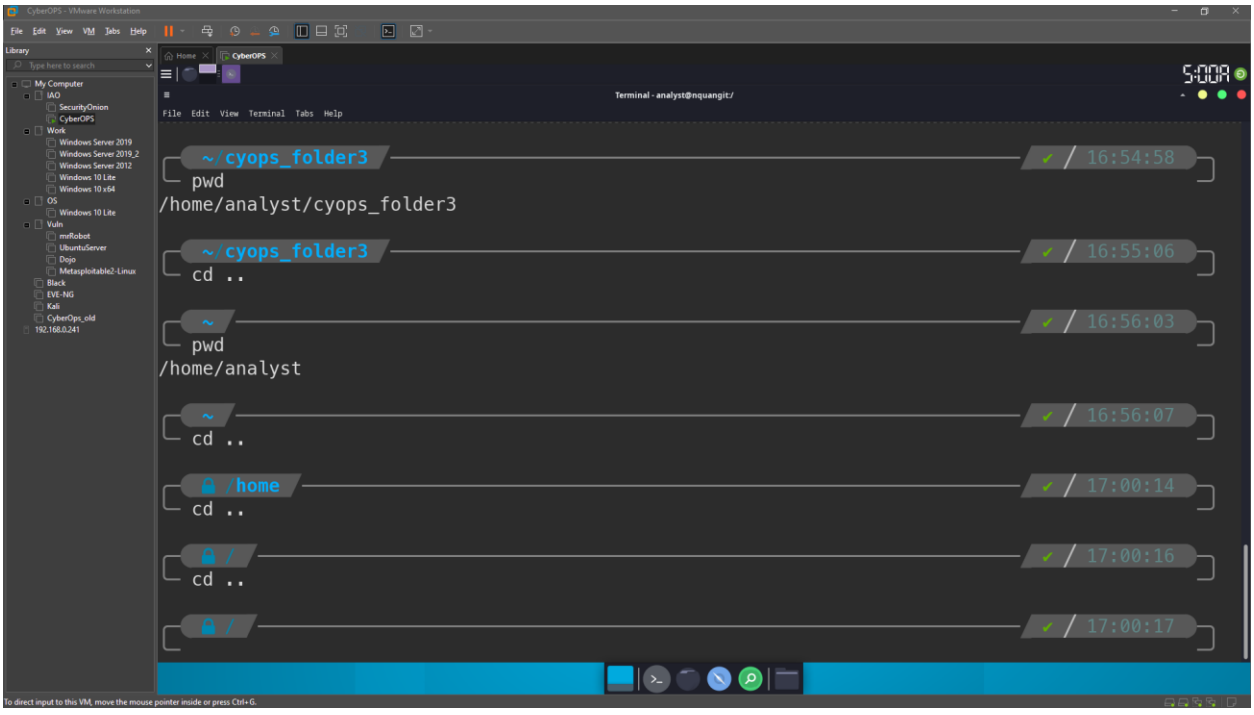
/home

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

/ (the root of the filesystem)

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

/ (the root of the file system. This is the highest level. It cannot go up)



Redirect Outputs.

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

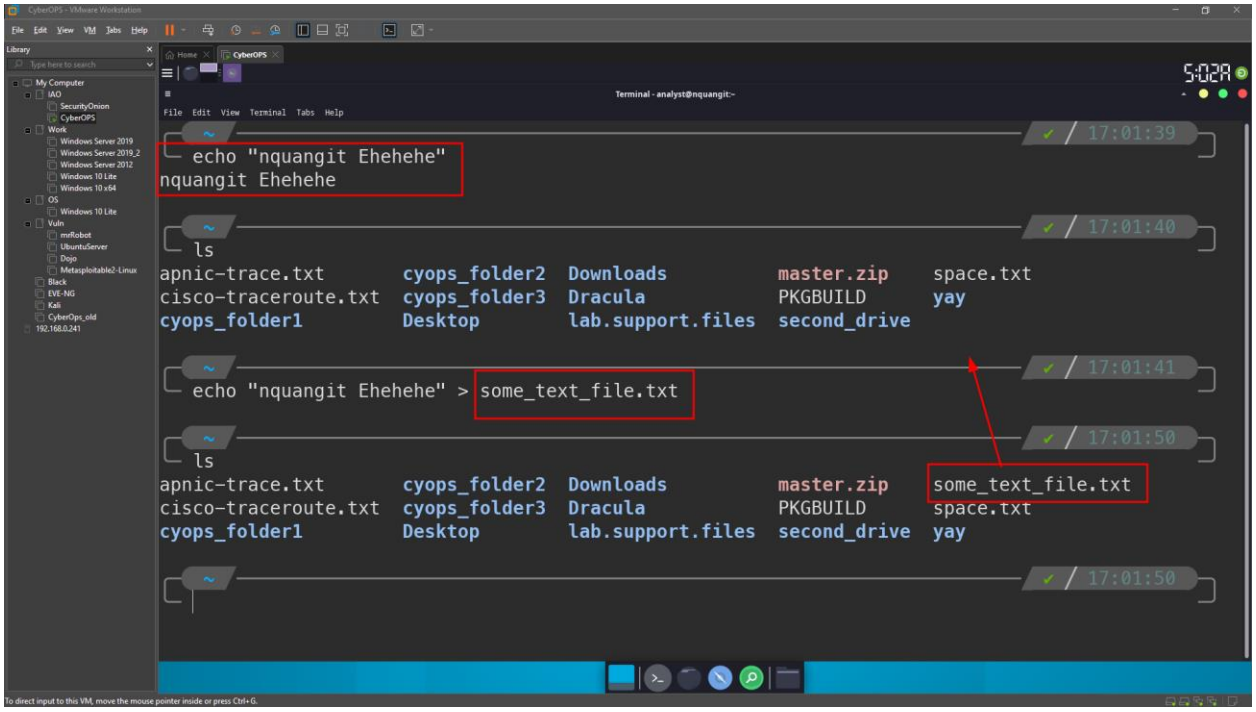
b. Use the **echo** command to echo a message. Because no output was defined, echo will output to the current terminal window:

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

Is that expected? Explain.

Yes. The output was redirected to the some_text_file.txt file.

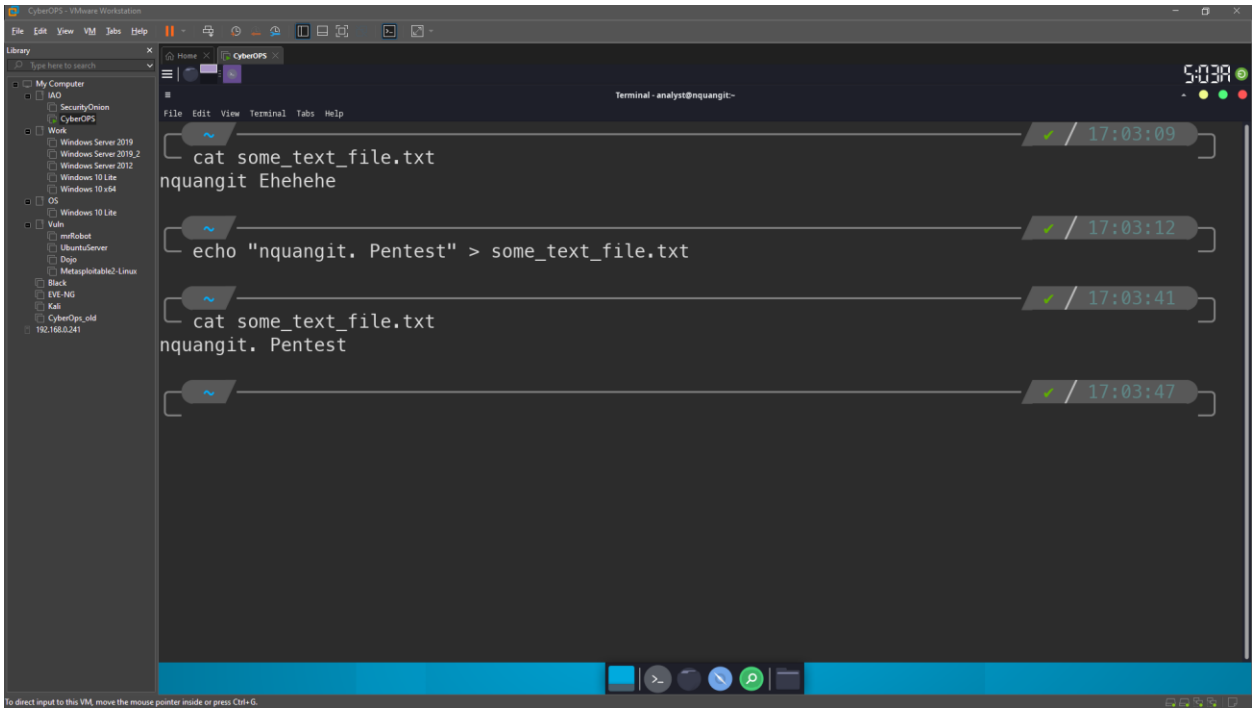
d. Notice, that even though the **some_text_file.txt** file did not exist, prior to the echo command, it was automatically created to receive the output generated by `echo`. Use the **ls -l** command to verify if the file was really created:



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

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

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



What happened to the text file? Explain.

The old text in the file was replaced by the new message. The > operator remove the file's content before writing the new data.

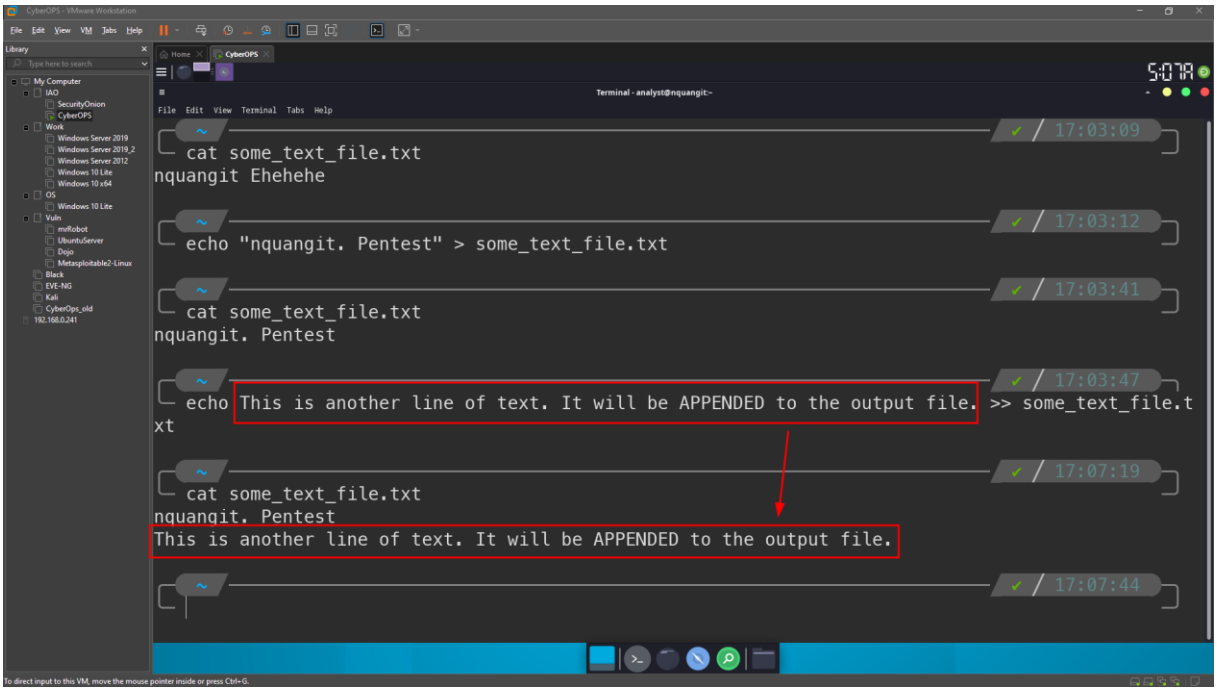
Redirect and Append to a Text File.

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 the 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.
```



Question:

What happened to the text file? Explain.

The new text was append to a new line of the end of the text file.

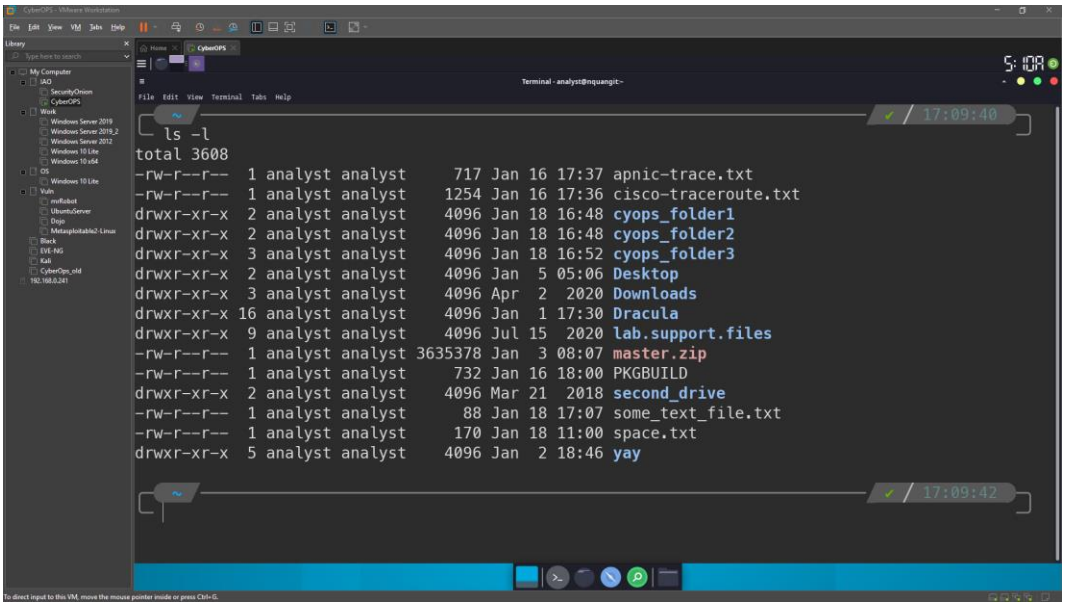
Work with hidden files in Linux.

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**.

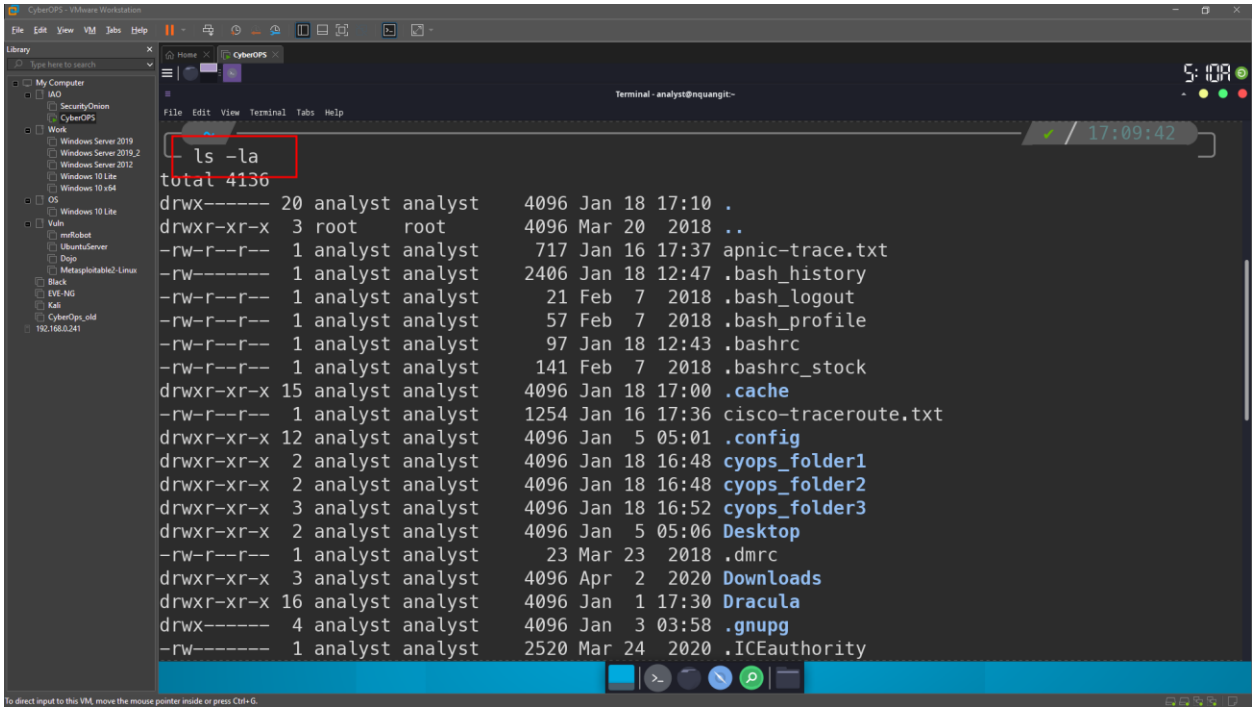
b. Use **ls -l** to display the files stored in the analyst home directory.

How many files are displayed?

15



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



How many more files are displayed than before? Explain.

51. The output included all the files and folder was hidden.

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, adding a dot before a folder's name will make it hide.

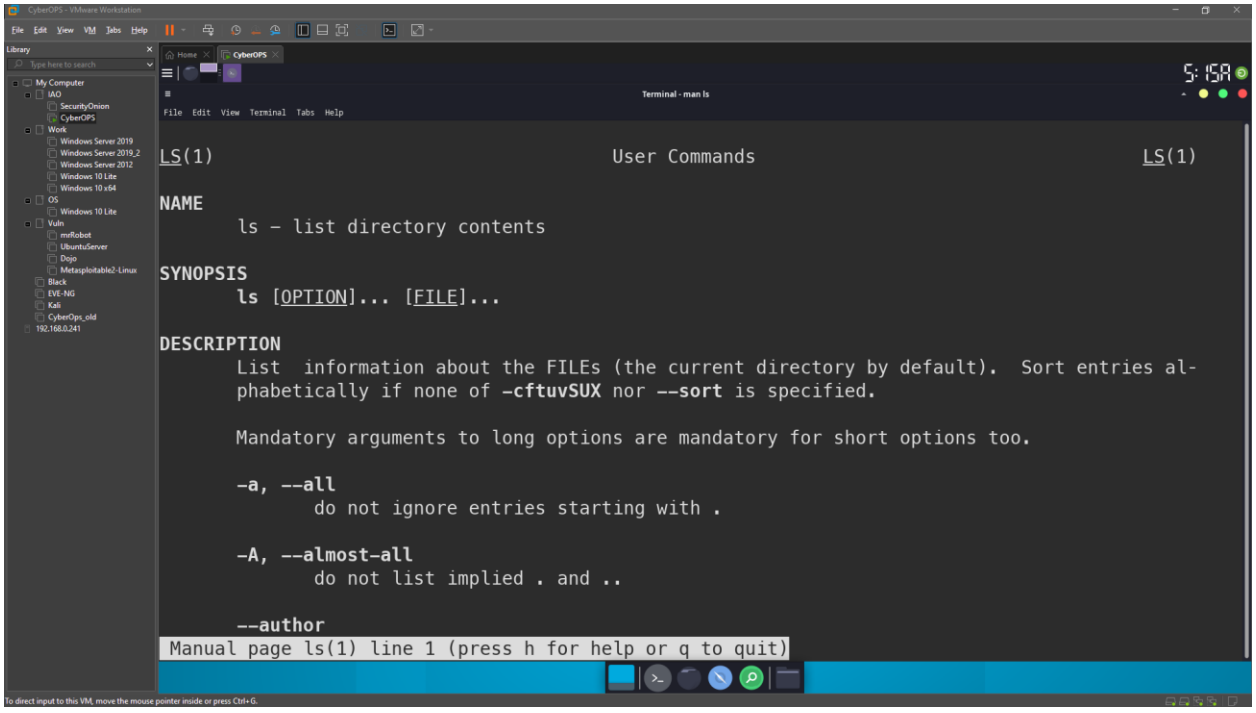
There are many hidden directories in the **ls -la** output

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

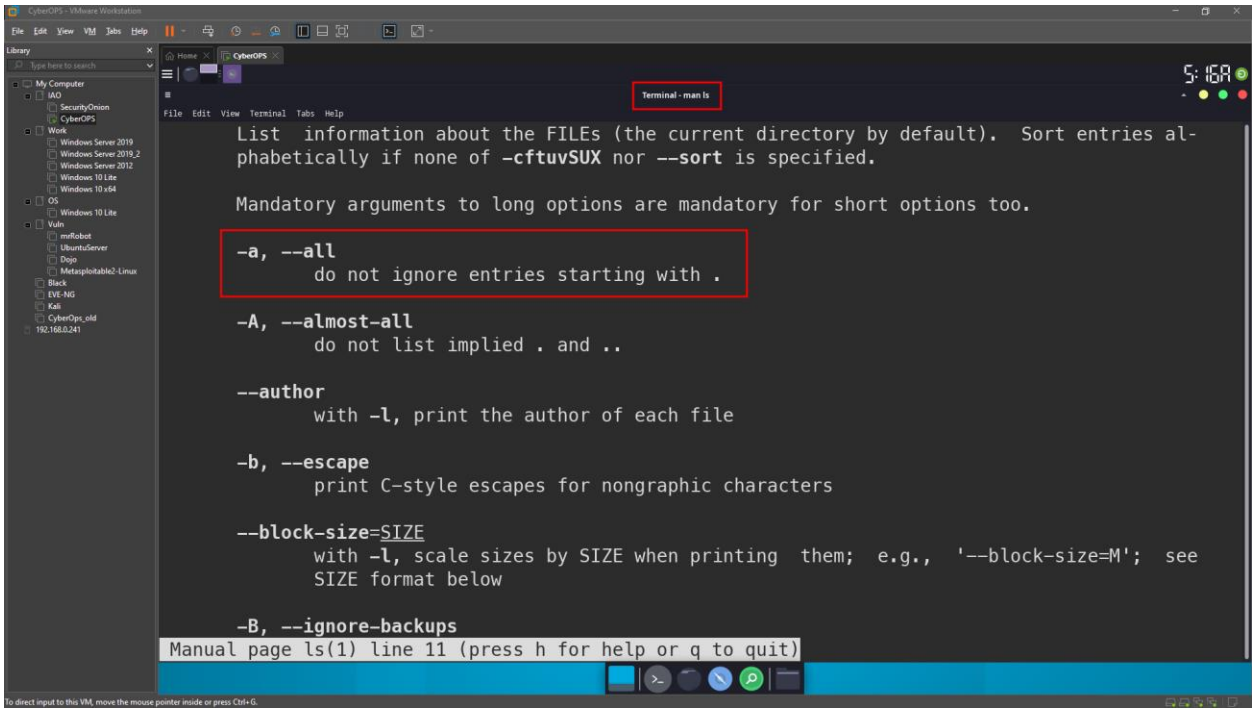
.bash_history, .bash_profile, .bashrc_stock

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** option used above and read its description to familiarize yourself with the **ls -a** command.



Copying, Deleting, and Moving 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.

Question:

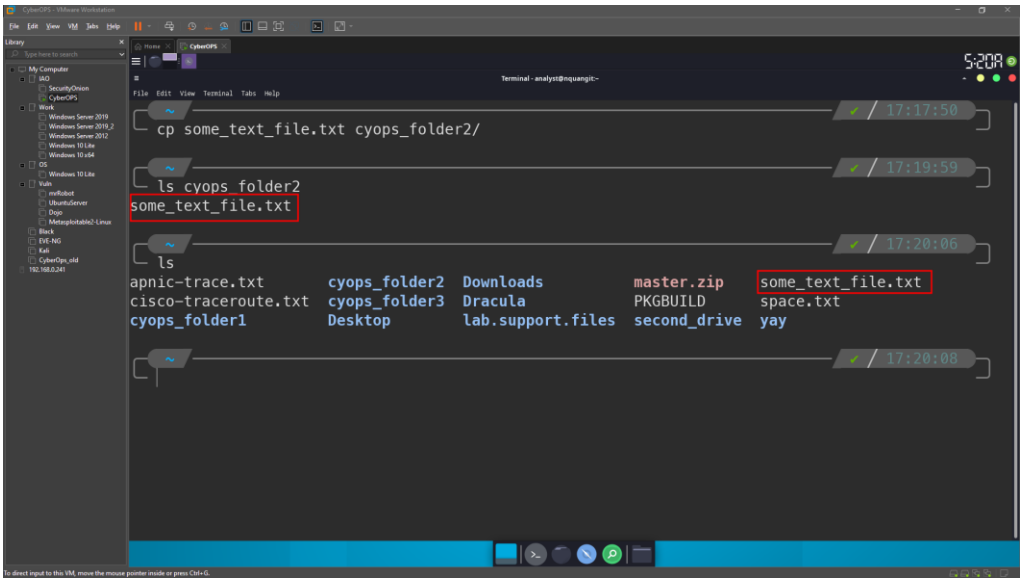
What are the source and destination files? (use full paths to represent the parameters)

Source: some_text_file.txt (absolute path: /home/analyst/some_text_file.txt)

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

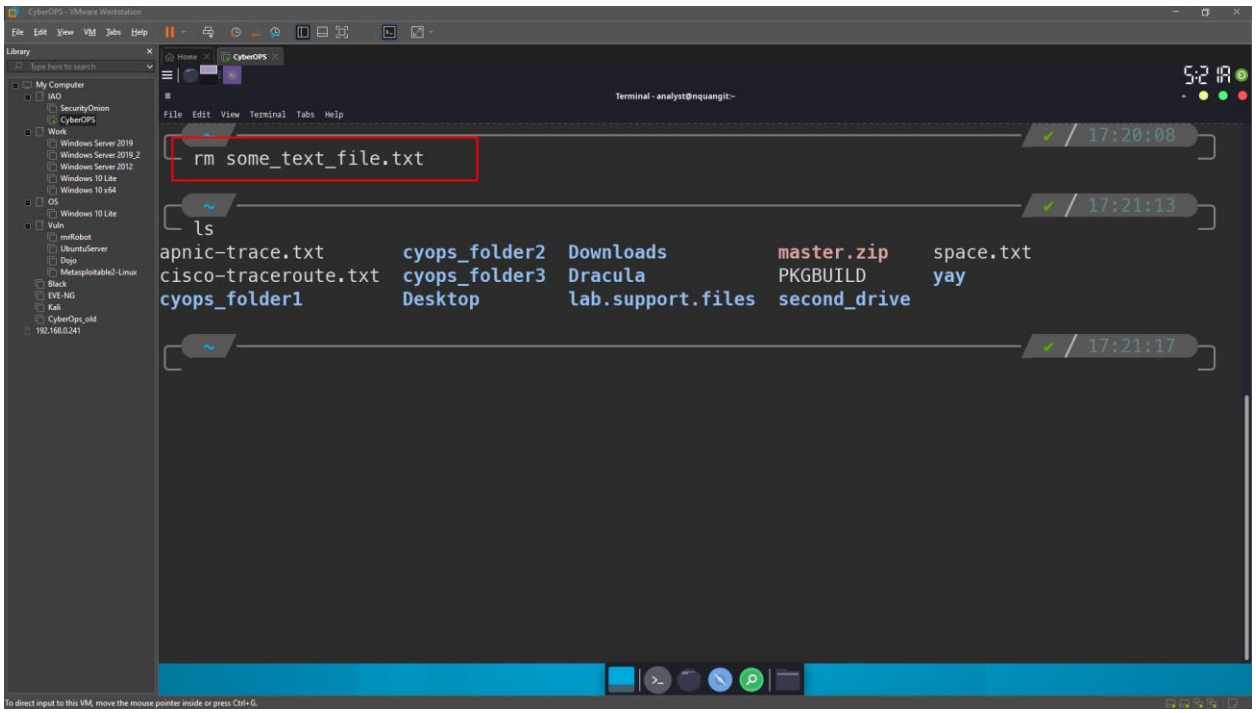
b. Use the **ls** command to verify that **some_text_file.txt** is now in **cyops_folder2:**

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



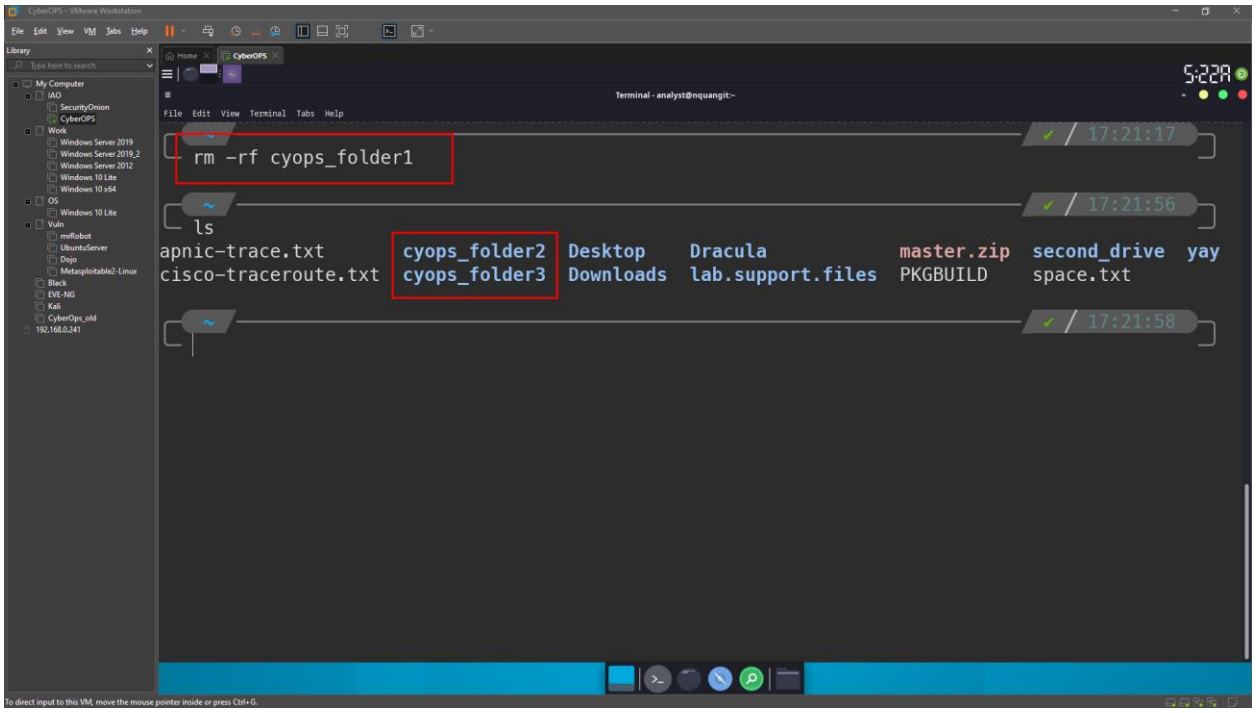
Deleting Files and Directories

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:



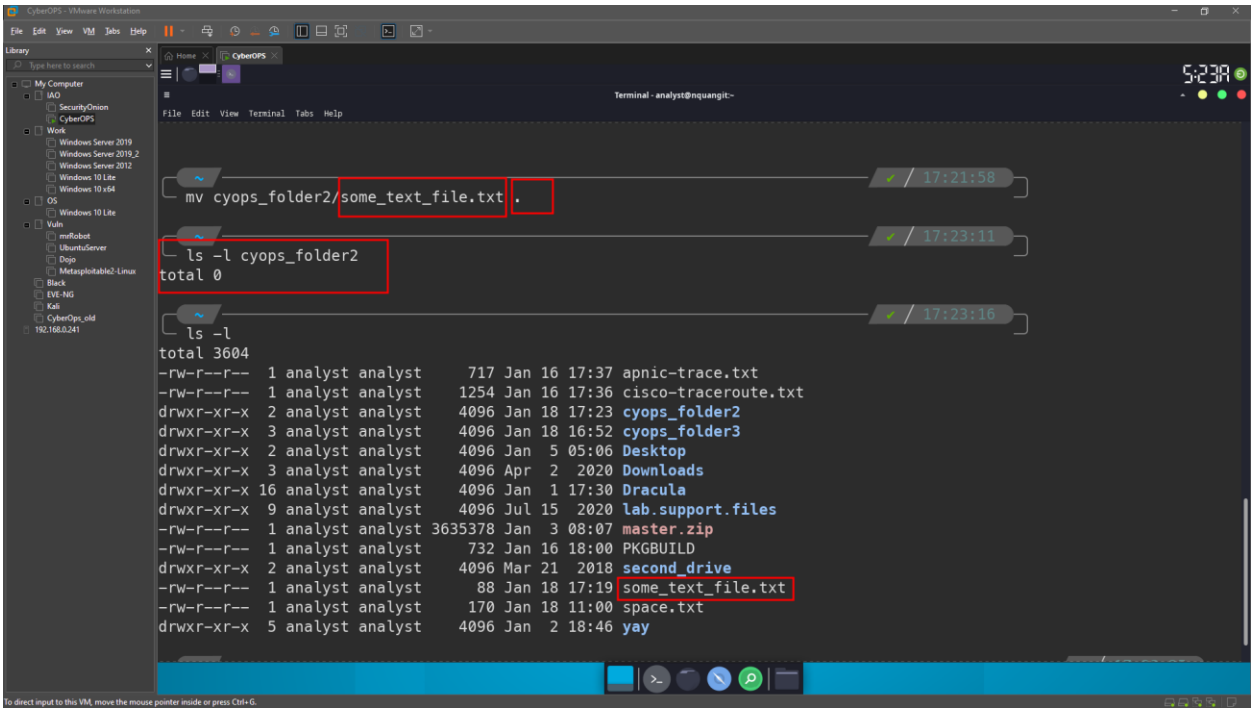
b. In Linux, directories are seen as a type of file. As such, the **rm** command is also used to delete directories but the **-r** (recursive) option must be used. Notice that all files and other directories

inside a given directory are also deleted when deleting a parent directory with the -r option. Issue the command below to delete the **cyops_folder1** folder and its contents:



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:



What command did you use to accomplish the task?

mv cyops_folder2/some_text_file.txt .

rm -r cyops_folder2/

Reflection

What are the advantages of using the Linux command line?

Advantages of Linux Command Line:

Efficiency: Faster and more precise execution of tasks.

Scripting: Powerful scripting capabilities for automation.

Resource Efficiency: Lower system resource usage compared to graphical interfaces.

Remote Access: Facilitates remote server management.

Customization: Highly customizable and scriptable for personalized workflows.

Learning Curve: Enhances understanding of system internals and commands.

Server Administration: Ideal for server environments with no graphical interfaces.

Resource Monitoring: Robust tools for monitoring system resources.

Text Processing: Powerful text processing tools for data manipulation.

Script Portability: Scripts are often portable across different Linux distributions.