Task 1. Generate hashes for files

The lab starts in your home directory, `/home/analyst`, as the current working directory. This directory contains two files `file1.txt` and `file2.txt`, which contain same data.

In this task, you need to display the contents of each of these files. You'll then generate a hash value for each of these files and send the values to new files, which you'll use to examine the differences in these values later.

1. Use the `ls` command to list the contents of the directory.

Two files, `file1.txt` and `file2.txt`, are listed.

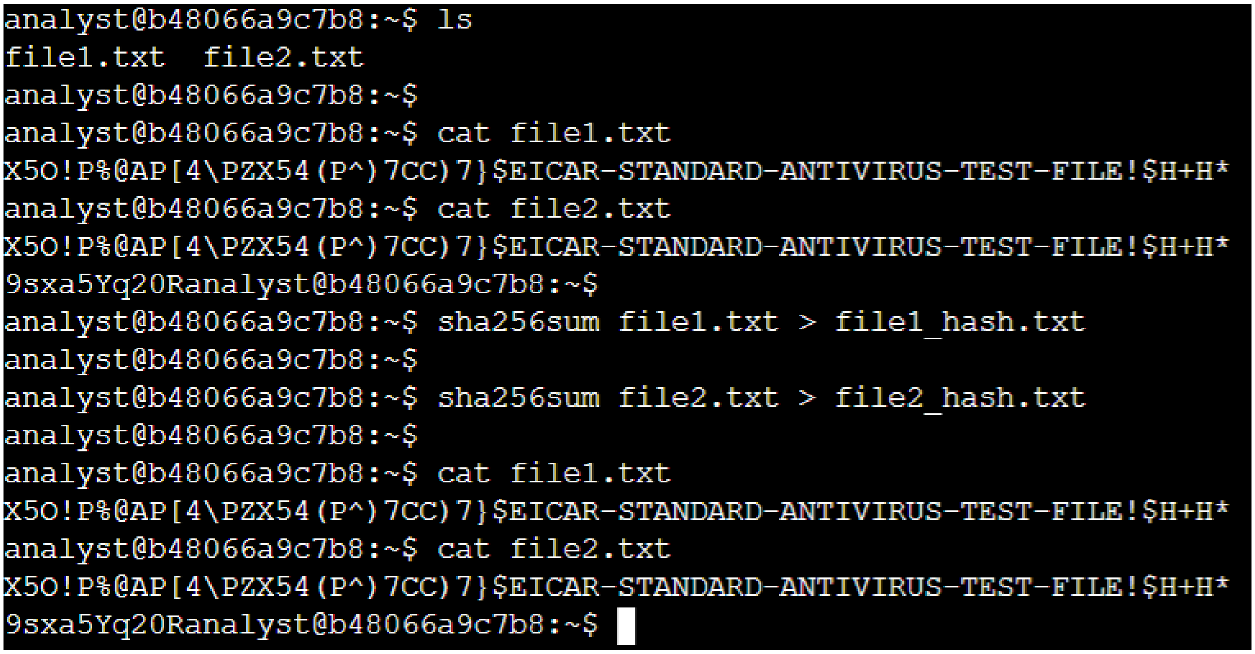
2. Use the `cat` command to display the contents of the `file1.txt` file:

```
cat file1.txt
```

***Note:** If you enter a command incorrectly and it fails to return to the command-line prompt, you can press **CTRL+C** to stop the process and force the shell to return to the command-line prompt.*

3. Use the `cat` command to display the contents of the `file2.txt` file:

```
cat file2.txt
```



```
analyst@b48066a9c7b8:~$ ls
file1.txt  file2.txt
analyst@b48066a9c7b8:~$
analyst@b48066a9c7b8:~$ cat file1.txt
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
analyst@b48066a9c7b8:~$ cat file2.txt
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
9sxa5Yq2ORanalyst@b48066a9c7b8:~$
analyst@b48066a9c7b8:~$ sha256sum file1.txt > file1_hash.txt
analyst@b48066a9c7b8:~$
analyst@b48066a9c7b8:~$ sha256sum file2.txt > file2_hash.txt
analyst@b48066a9c7b8:~$
analyst@b48066a9c7b8:~$ cat file1.txt
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
analyst@b48066a9c7b8:~$ cat file2.txt
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
9sxa5Yq2ORanalyst@b48066a9c7b8:~$
```

Although the contents of both files appear identical when you use the `cat` command, you need to generate the hash for each file to determine if the files are actually different.

5. Use the `sha256sum` command to generate the hash of the `file1.txt` file:

```
sha256sum file1.txt
```

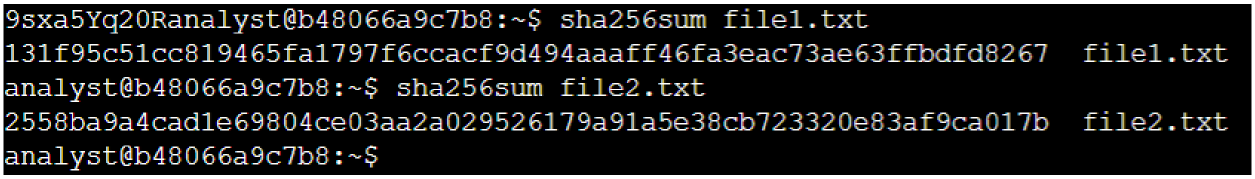
Copied!

content_copy

You now need to follow the same step for the `file2.txt` file.

6. Use the `sha256sum` command to generate the hash of the `file2.txt` file:

```
sha256sum file2.txt
```



```
9sxa5Yq2ORanalyst@b48066a9c7b8:~$ sha256sum file1.txt
131f95c51cc819465fa1797f6ccacf9d494aaaff46fa3eac73ae63ffbdfd8267  file1.txt
analyst@b48066a9c7b8:~$ sha256sum file2.txt
2558ba9a4cad1e69804ce03aa2a029526179a91a5e38cb723320e83af9ca017b  file2.txt
analyst@b48066a9c7b8:~$
```

Task 2. Compare hashes

In this task, you'll write the hashes to two separate files and then compare them to find the difference.

1. Use the `sha256sum` command to generate the hash of the `file1.txt` file, and send the output to a new file called `file1hash`:

```
sha256sum file1.txt >> file1_hash.txt
```

You now need to complete the same step for the `file2.txt` file.

2. Use the `sha256sum` command to generate the hash of the `file2.txt` file, and send the output to a new file called `file2hash`:

```
sha256sum file2.txt >> file2_hash.txt
```

Now, you should have two hashes written to separate files. The first hash was written to the `file1hash` file, and the second hash was written to the `file2hash` file.

You can manually display and compare the differences.

3. Use the `cat` command to display the hash values in the `file1_hash.txt` and `file2_hash.txt` files.
4. Inspect the output and note the difference in the hash values.

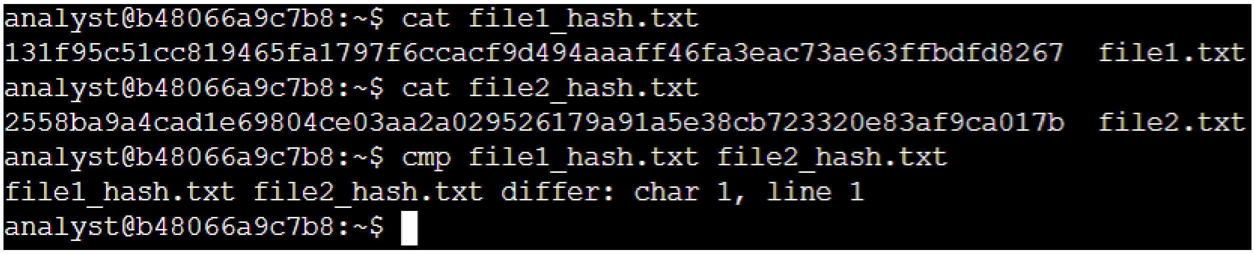
***Note:** Although the content in `file1.txt` and `file2.txt` previously appeared identical, the hashes written to the `file1hash` and `file2hash` files are **completely** different.*

Now, you can use the `cmp` command to compare the two files byte by byte. If a difference is found, the command reports the byte and line number where the first difference is found.

5. Use the `cmp` command to highlight the differences in the `file1hash` and `file2hash` files:

```
cmp file1_hash.txt file2_hash.txt
```

6. Review the output, which reports the first difference between the two files:



```
analyst@b48066a9c7b8:~$ cat file1_hash.txt
131f95c51cc819465fa1797f6ccacf9d494aaaff46fa3eac73ae63ffbdfd8267  file1.txt
analyst@b48066a9c7b8:~$ cat file2_hash.txt
2558ba9a4cad1e69804ce03aa2a029526179a91a5e38cb723320e83af9ca017b  file2.txt
analyst@b48066a9c7b8:~$ cmp file1_hash.txt file2_hash.txt
file1_hash.txt file2_hash.txt differ: char 1, line 1
analyst@b48066a9c7b8:~$ 
```

Conclusion

I practiced how to

- compute hashes using `sha256sum`,
- display hashes using the `cat` command, and
- compare hashes using the `cmp` command.

These are valuable tools you can use to validate data integrity as you contribute to the control of your organization's security.