PRÁCTICA : E3 - Asegurar la INTEGRIDAD de los datos en Windows / Linux **MF0486_3 : Seguridad en Equipos Informáticos**		**Fecha**	09 / 03 / 2022
		Página 1 de 3	
Curso	**7.1. MF0486_3 Seguridad en equipos informáticos**	**Plan de Formación**	FC-2021.1/II.000/1914256

Nombre y Apellidos:		**Firma del Alumno:**	
DNI:		**Firma del Profesor:**	

Apto: ☐ **No Apto:** ☐ **Calificación:** ☐

Instrucciones Generales

La puntuación máxima será de 10 puntos.
Esta prueba tendrá una duración máxima de 420 minutos
(Temporalizados durante la Unidad de Aprendizaje)

El alumno/a deberá acatar las siguientes normas durante la duración de la práctica :

- Rellene el encabezado con su nombre, apellidos y D.N.I.
- Firme en todas y cada una de las hojas entregadas, incluidas las que estén en blanco.
- Usar exclusivamente bolígrafo azul o negro
- Guardar los ficheros generados en una carpeta con nombre **MF0486_E3**
- El docente le indicará al final como entregar el contenido de dicha carpeta
- Al finalizar el ejercicio y antes de entregarlo **comprueba tus respuestas**, en caso de duda consulta al docente.

Equipo y material

- Bolígrafo azul.
- Folios.
- Ordenadores.
- Conexión a Internet. (Para buscar información a modo de ayuda)
- **SFC :** Sistema operativo Windows (Virtualizado)
- **Rootkit Hunter :** Sistema operativo Linux (virtualizado)
- Pendrive.

Instrucciones específicas

El objetivo de esta práctica guiada será como se puede asegurar la **integridad** de los datos en sistemas Windows y Linux.

Condiciones de realización:

La actividad se llevará a cabo en el aula y el alumnado contará en todo momento supervisión del docente.

El alumnado contará con una duración de 420 minutos para realizar la práctica.
Se podrá realizar en varias partes con una duración cada una de 60 minutos.

El alumno podrá hacer uso de internet para su realización, y se detallan a continuación algunas webs de ayuda.

Páginas webs :

SFC (System File Check)

https://en.wikipedia.org/wiki/System_File_Checker
https://neosmart.net/wiki/sfc/
https://support.microsoft.com/es-es/kb/929833

rootkit

https://es.wikipedia.org/wiki/Rootkit
https://es.wikipedia.org/wiki/Rkhunter
https://rootkit.nl/projects/rootkit_hunter.html

En ella se valorará la utilización de herramientas para la gestión del tiempo y secuenciación del uso de las aplicaciones necesarias. Y se observará especialmente la autonomía del alumnado a la hora de ejecutar y tomar decisiones. Como también la estructuración del ejercicio en donde se solicitará, orden, coherencia y limpieza.

Una vez terminado la práctica se le notificará al docente y pasará a su evaluación.

Descripción de la práctica

INTEGRIDAD / DE LA INFORMACIÓN.

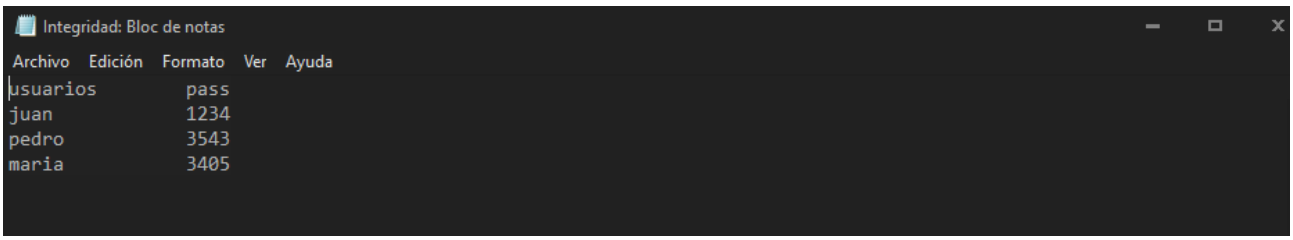
WINDOWS:

1: Mediante Consola.

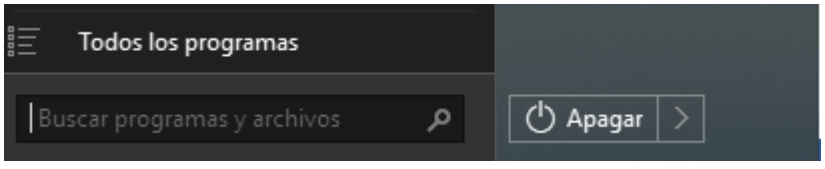
Como se ha visto en clase, una de las formas de comprobar la integridad de los datos sería utilizando la función criptográfica **HASH** que es un algoritmo matemático capaz de transformar un bloque de datos en una nueva serie de caracteres con una longitud fija independiente.

En esta ocasión para no consumir tantos recursos instalando una aplicación externa utilizaré **POWERSHELL** que es una interfaz de líneas de comandos muy útil donde podemos automatizar algunas tareas y **revisar la integridad de los datos.**

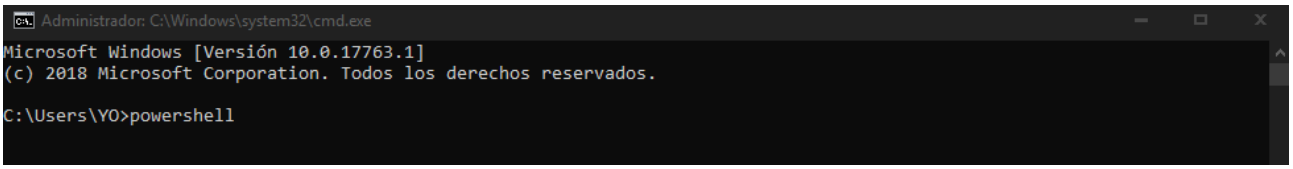
1. Creo un archivo de texto de pruebas llamado **Integridad** y coloco datos e información dentro de él.



2. Guardo la información y abro la consola de Windows, en el cuadro de buscar escribo CMD y ENTER



3. Se nos abre la consola d Windows y escribimos POWERSHELL



4. Entramos en la consola de POWERSHELL la diferenciamos porque vemos el PS al inicio de la línea,

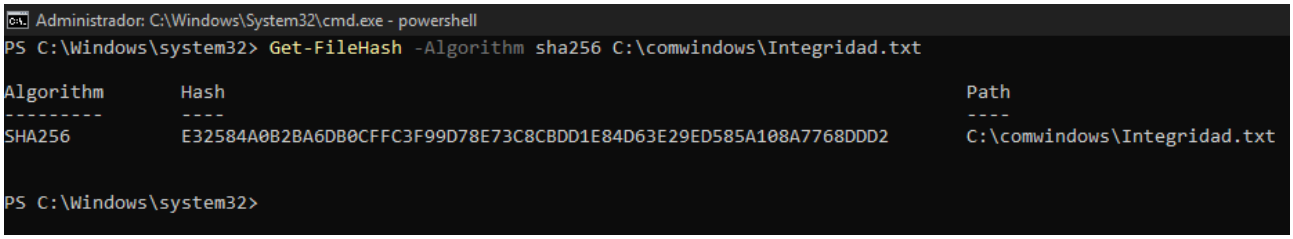
```
C:\Users\YO>powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

PS C:\Users\YO>
```

5. Para ver el Hash de cualquier fichero, en POWERSHEL existe una función llamada Get-FileHash

```
Administrador: C:\Windows\System32\cmd.exe - powershell
PS C:\Windows\system32> get-help get-Filehash

NOMBRE
    Get-FileHash

SINOPSIS
    Computes the hash value for a file by using a specified hash algorithm.


SINTAXIS
    Get-FileHash [-Algorithm {SHA1 | SHA256 | SHA384 | SHA512 | MACTripleDES | MD5 | RIPEMD160}] -InputStream
    <System.IO.Stream> [<CommonParameters>]

    Get-FileHash [-Algorithm {SHA1 | SHA256 | SHA384 | SHA512 | MACTripleDES | MD5 | RIPEMD160}] -LiteralPath
    <System.String[]> [<CommonParameters>]

    Get-FileHash [-Path] <System.String[]> [-Algorithm {SHA1 | SHA256 | SHA384 | SHA512 | MACTripleDES | MD5 |
    RIPEMD160}] [<CommonParameters>]


DESCRIPCIÓN
    The `Get-FileHash` cmdlet computes the hash value for a file by using a specified hash algorithm. A hash value is
    a unique value that corresponds to the content of the file. Rather than identifying the contents of a file by its
    file name, extension, or other designation, a hash assigns a unique value to the contents of a file. File names
    and extensions can be changed without altering the content of the file, and without changing the hash value.
    Similarly, the file's content can be changed without changing the name or extension. However, changing even a
    single character in the contents of a file changes the hash value of the file.

    The purpose of hash values is to provide a cryptographically-secure way to verify that the contents of a file have
    not been changed. While some hash algorithms, including MD5 and SHA1, are no longer considered secure against
    attack, the goal of a secure hash algorithm is to render it impossible to change the contents of a file -- either
    by accident, or by malicious or unauthorized attempt -- and maintain the same hash value. You can also use hash
    values to determine if two different files have exactly the same content. If the hash values of two files are
    identical, the contents of the files are also identical.

    By default, the `Get-FileHash` cmdlet uses the SHA256 algorithm, although any hash algorithm that is supported by
    the target operating system can be used.


VÍNCULOS RELACIONADOS
    Online Version: https://docs.microsoft.com/powershell/module/microsoft.powershell.utility/get-filehash?view=powersh
    ell-5.1&WT.mc_id=ps-gethelp
    Format-List
```

6. Aplicamos la ;ínea de código al fichero Integridad.txt y nos dá el siguiente resultado

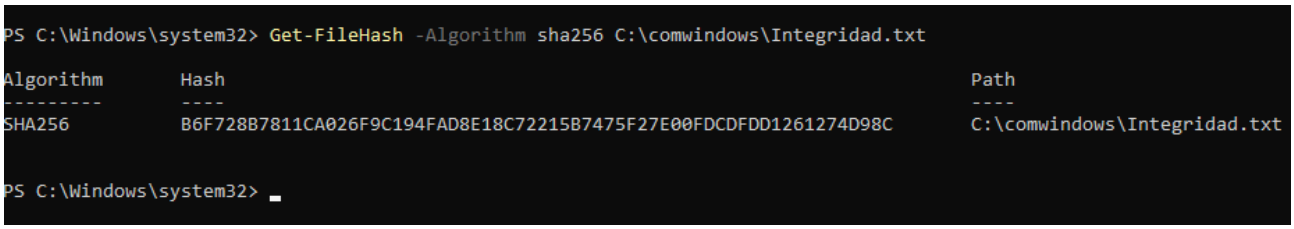
Get-FileHash -Algorithm sha256 C:\comwindows\Integridad.txt



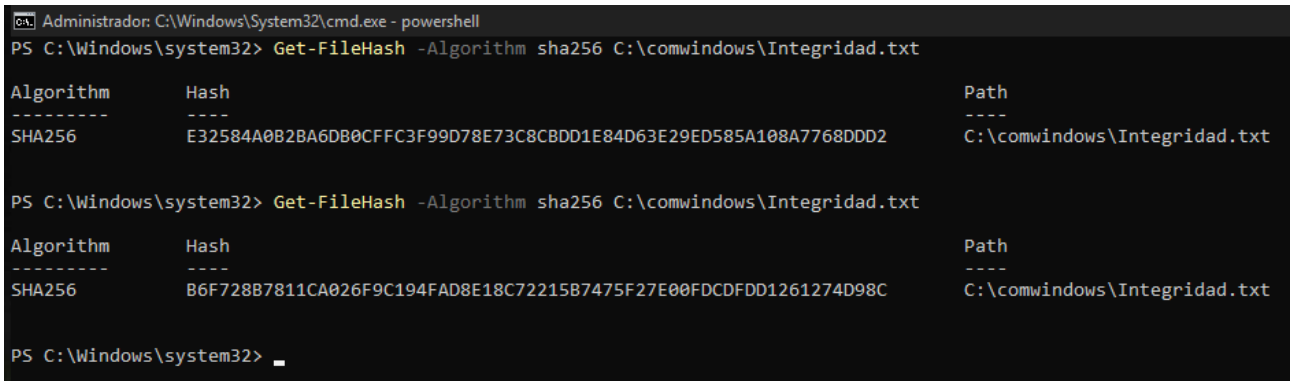
7. Abrimos el documento lo modificamos y guardamos

8.Volvemos aplicar el comando

Get-FileHash -Algorithm sha256 C:\comwindows\Integridad.txt

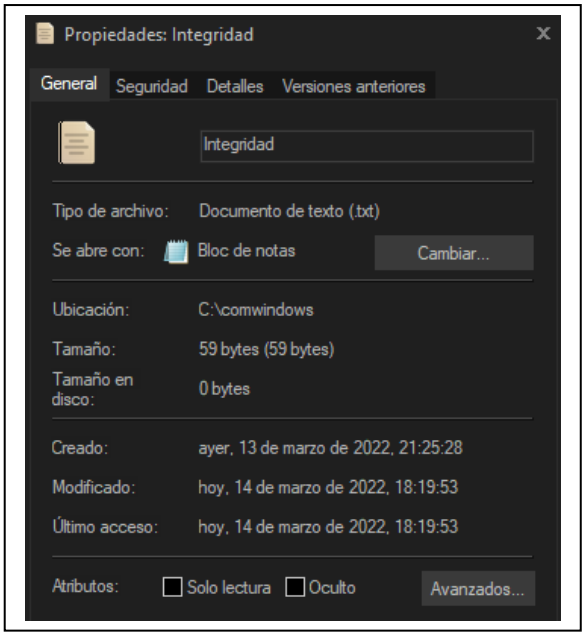


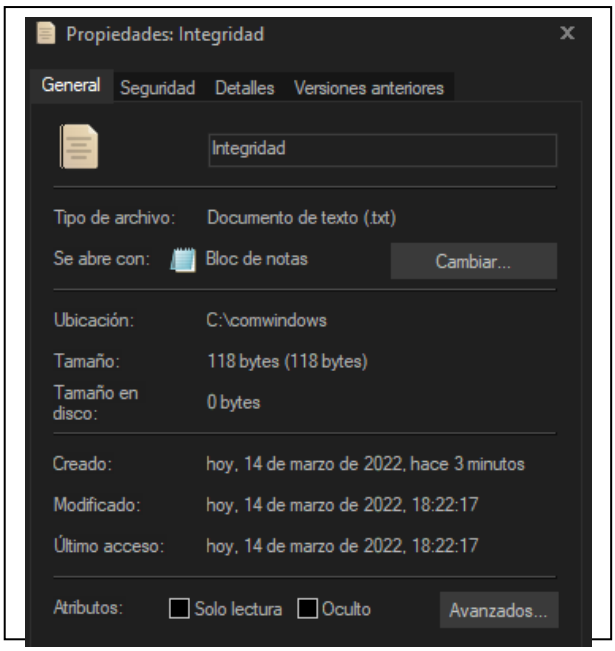
8. Lo vemos de una forma comparativa



El Hash del fichero ha cambiado lo que nos indica que el fichero original ha sido modificado y por consecuencia su integridad podria estar comprometida dependiendo de que fuente provenga el cambio. En este caso lo he modificado yo, asi que el fichero sigue siendo seguro

También se puede ver que el tamaño del fichero original es distinto al ha sido modificado





En dado caso de que sea una tarea muy recurrente para una empresa podemos automatizar este proceso desarrollando un pequeño script en PS y otro en bash y juntarlos para faciliar el trabajo diario.

scriptintegridad: Bloc de notas

Archivo Edición Formato Ver Ayuda

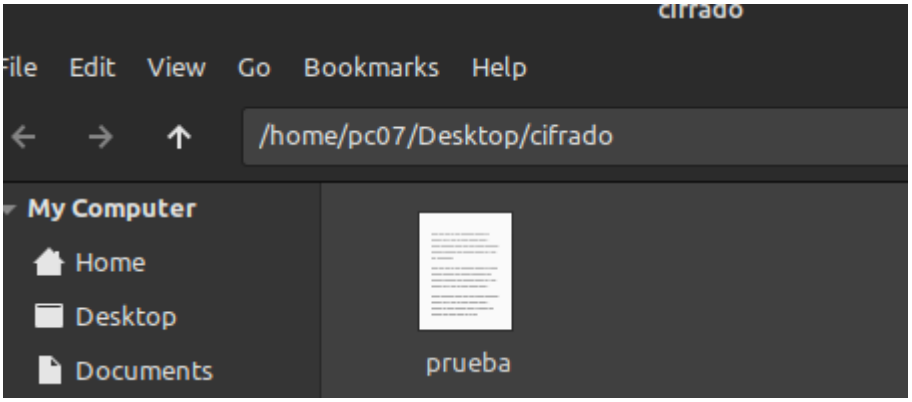
```
@echo off
echo ================================
echo VEMOS EL CONTENIDO DEL FICHERO 1
echo ================================
type Integridad.txt
echo.
pause
echo.
echo --------------------------------
echo VEMOS EL CONTENIDO DEL FICHERO 2
echo --------------------------------
type Integridad-copia.txt
echo.
pause
echo.
echo ============================================
echo COMPARAMOS LA INTEGRIDAD DE LOS DOS FICHEROS
echo ============================================
powershell c:\comwindows\scriptintegridad.ps1
pause
echo.
echo ============================================
echo Agregamos 2 registros en el fichero 2
echo ============================================
echo Carlos             5674 Nuevo Registro >> Integridad-copia.txt
echo Carolina   1111 Nuevo Registro >> Integridad-copia.txt
echo.
type Integridad-copia.txt
echo.
echo ============================================
echo VOLVEMOS A COMPARAR FICHERO 1 Y FICHERO 2
echo ============================================
powershell c:\comwindows\scriptintegridad.ps1
echo.
echo Vemos que al ser modificado el CÓDIGO HASH256 del fichero 2 HA CAMBIADO.
echo.
pause

del Integridad-copia.txt
type Integridad.txt >> Integridad-copia.txt
exit
```

Adjunto ficheros para ver esta práctica de una manera más sencilla, ejecutar "scriptintegridad.bat"

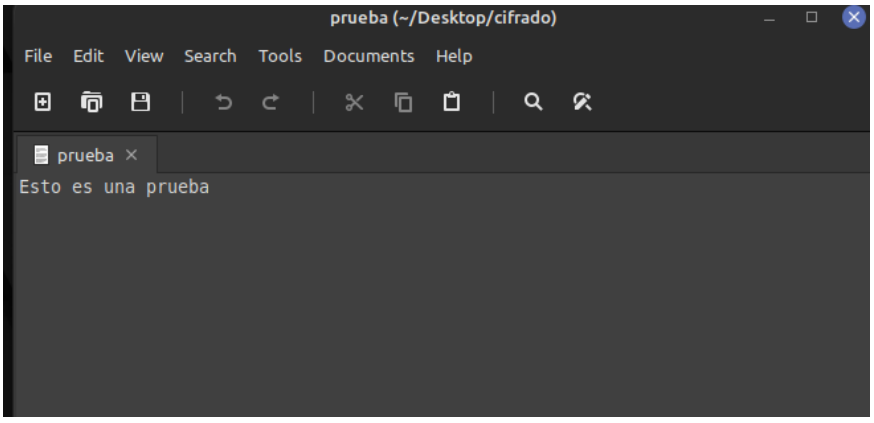
LINUX UBUNTU:

1: Mediante Terminal

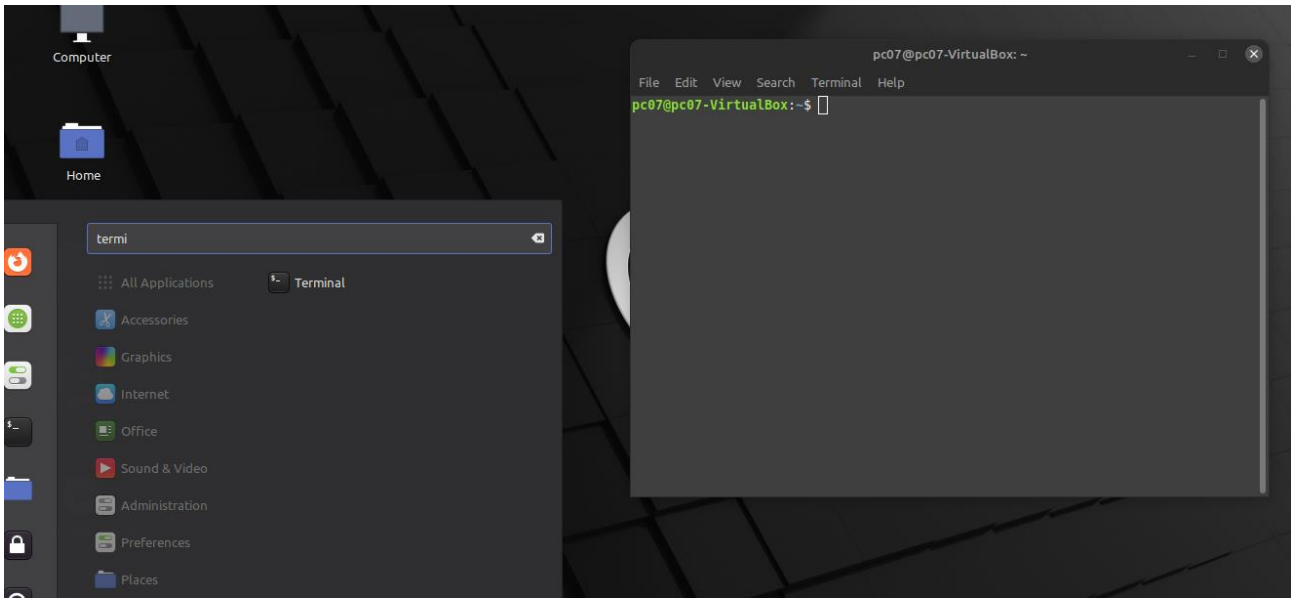
- Creo una carpeta nueva y un documen to de texto nuevo para hacer la comparación



- Abrimos el archivo llamado prueba escribimos un texto y luego guardamos..

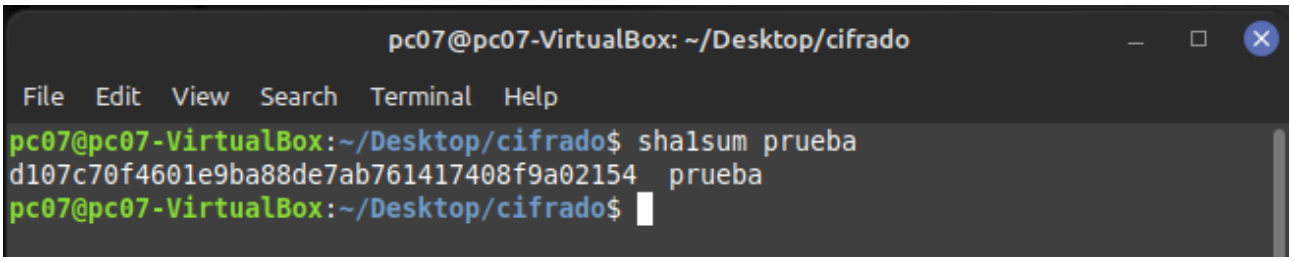


- **Abrimos Terminal de Mint:**



- Buscamos la carpeta donde se encuentra localizado el archivo que guardamos. Lo podemos buscar o escribir la ruta directamente y escribimos el siguiente comando

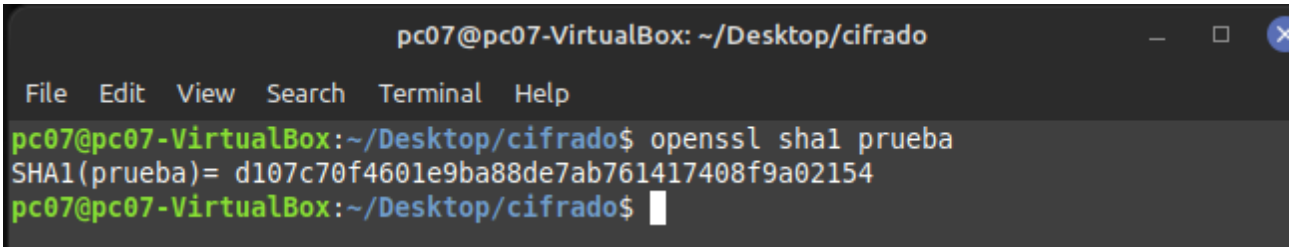
  Shaisum prueba



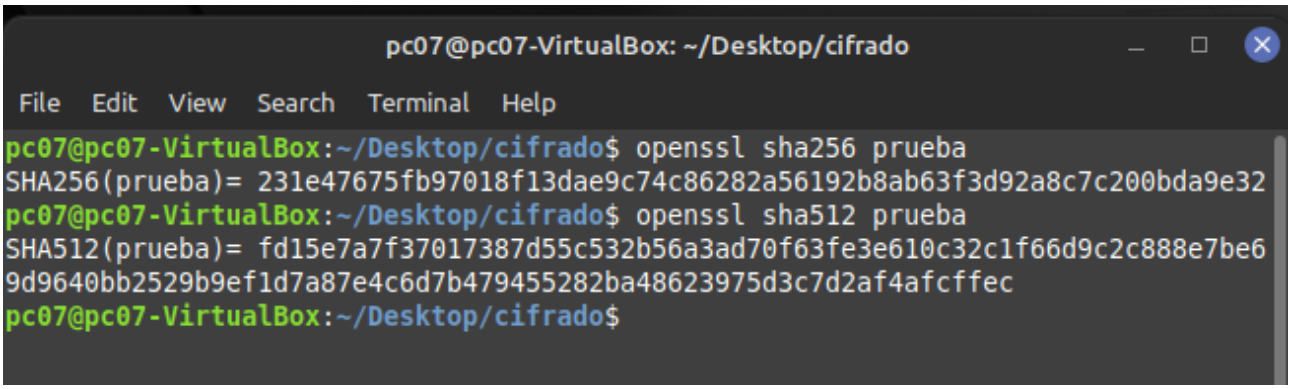
Automaticamente nos muestra el Hash del fichero.

Tambien podemos usar la siguiente instrucción

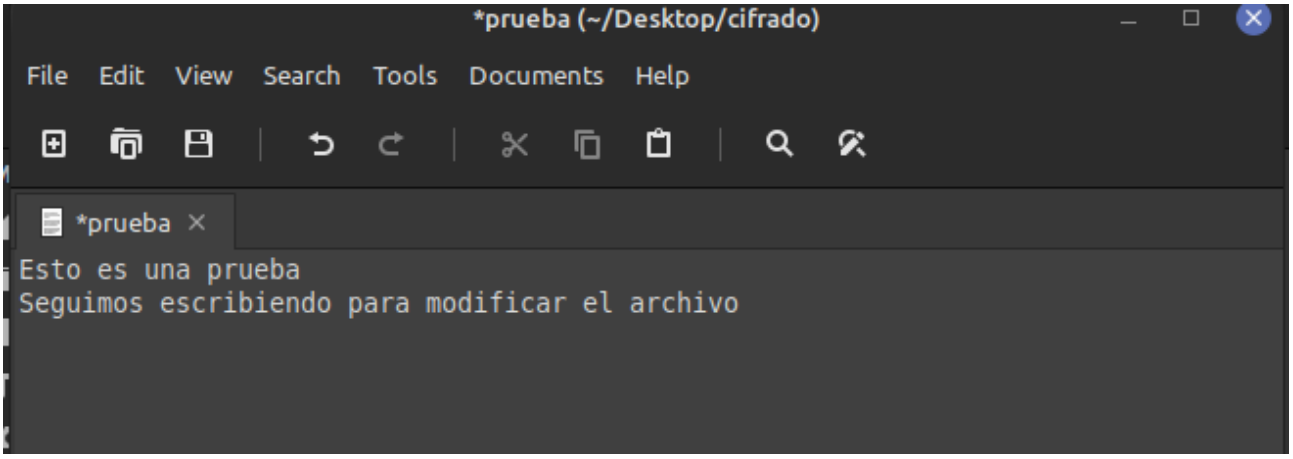
Openssl sha1 prueba



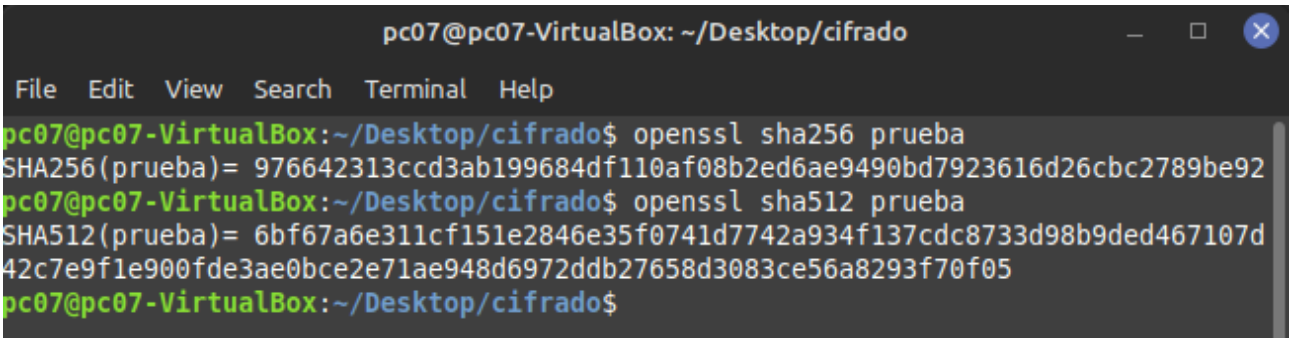
Con openssl podemos ver las distintas claves Hash como vemos en el ejemplo.



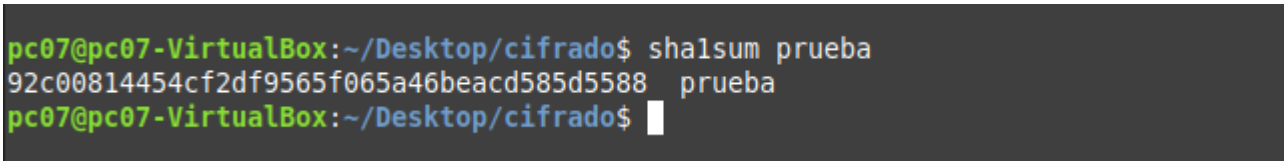
Ahora modificaremos el arhivo de texto para comprobar que la integridad no es la misma agragando texto.



Verificamos la integridad con openssl



Con shaisum



Al comparar vemos que la integridad ya no es la misma debido a que modificamos el archivo agregando mas texto.

Todo esto se puede aplicar también para carpetas y ver todos los archivos que se encuentran dentro de ellas.

Al igual como se hizo en windows tambien podriamos crear un script para hacer que el proceso sea automatizado en dado de ser un trabajo diario.