

Actividad 2 – Cálculo de hashes

Diego Mucci

02/06/2024







Certificado de Profesionalidad: Seguridad Informática

Módulo formativo: Sistemas seguros de acceso y transmisión de datos

ACTIVIDAD 2 – CÁLCULO DE HASHES

- En Windows, crea un fichero de texto y calcula su hash, md5, sha256, mediante PowerShell. Realiza cambios en el fichero y vuelve a calcular los hashes. Compara el resultado.

Creemos un archivo de texto llamado hash.txt:

Este equipo > Escritorio > Diego > Cyberseguridad Ironhack > Temario > Módulo 4			
Nombre		Fecha de modificación	T
	Apuntes.docx	30/05/2024 10:32	D
	Calculate-Hashes.ps1	28/05/2024 19:56	S
	Capítulo_1.-_Criptografía.pdf	25/05/2024 15:10	D
	Capítulo_2.-_Aplicación_de_una_infraestructura_...	25/05/2024 15:11	D
	Capítulo_3.-_Comunicaciones_seguras.pdf	25/05/2024 15:11	D
	hash.txt	30/05/2024 13:34	D

Abrimos Powershell, vamos al directorio donde se encuentra el fichero y ejecutamos el comando “Get-FileHash -Algorithm MD5 hash.txt” para calcular el hash md5 y el comando “Get-FileHash -Algorithm SHA256 hash.txt” para calcular el hash sha 256:

```
PS C:\WINDOWS\system32> cd 'C:\Users\dmucc\Desktop\Diego\Cyberseguridad Ironhack\Temario\Módulo 4\'
PS C:\Users\dmucc\Desktop\Diego\Cyberseguridad Ironhack\Temario\Módulo 4> Get-FileHash -Algorithm MD5 hash.txt

Algorithm      Hash                                          Path
-----
MD5            1A0ABBE8F8F9E54D3DD772B315B32B2          C:\Users\dmucc\Desktop\Diego\..

PS C:\Users\dmucc\Desktop\Diego\Cyberseguridad Ironhack\Temario\Módulo 4> Get-FileHash -Algorithm SHA256 hash.txt

Algorithm      Hash                                          Path
-----
SHA256        2FF500AF70F2F55F828C1A0FBAB4BDD2075F0E50459ABF5021D78A804AEFA480          C:\Users\dmucc\Desktop\Diego\..
```

Ahora realizamos cambios en el fichero y volvemos calcular el hash, como podemos observar, este cambia al aplicar cambios:

```
PS C:\Users\dmucc\Desktop\Diego\Cyberseguridad Ironhack\Temario\Módulo 4> Get-FileHash -Algorithm MD5 hash.txt

Algorithm      Hash                                          Path
-----
MD5            EDBE79D037B5F6F7C7234C5B26A3D004          C:\Users\dmucc\Desktop\Diego\..

PS C:\Users\dmucc\Desktop\Diego\Cyberseguridad Ironhack\Temario\Módulo 4> Get-FileHash -Algorithm SHA256 hash.txt

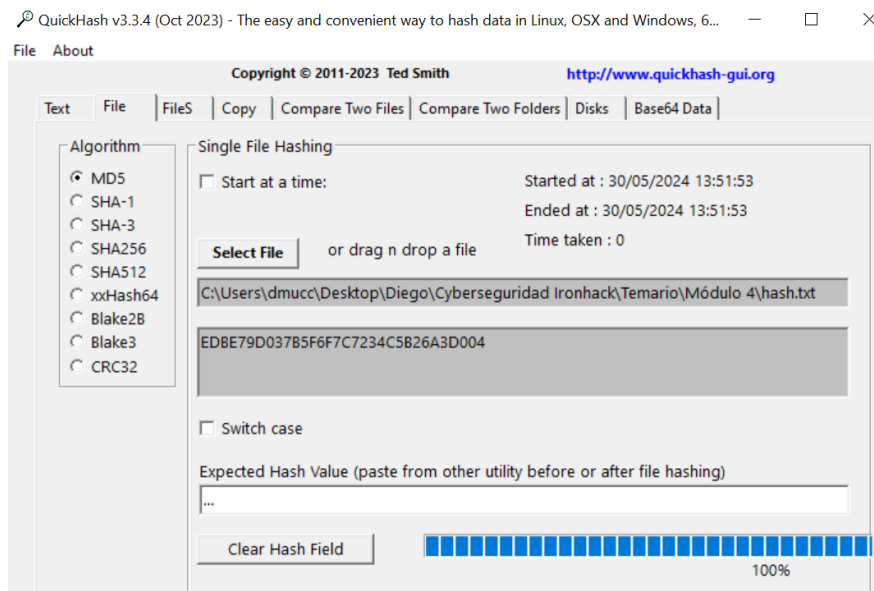
Algorithm      Hash                                          Path
-----
SHA256        0061763B8FFC5A429C27EC77D8E05F1896D3A8F6A8F61F8F478BDC68EE97BFD2          C:\Users\dmucc\Desktop\Diego\..
```

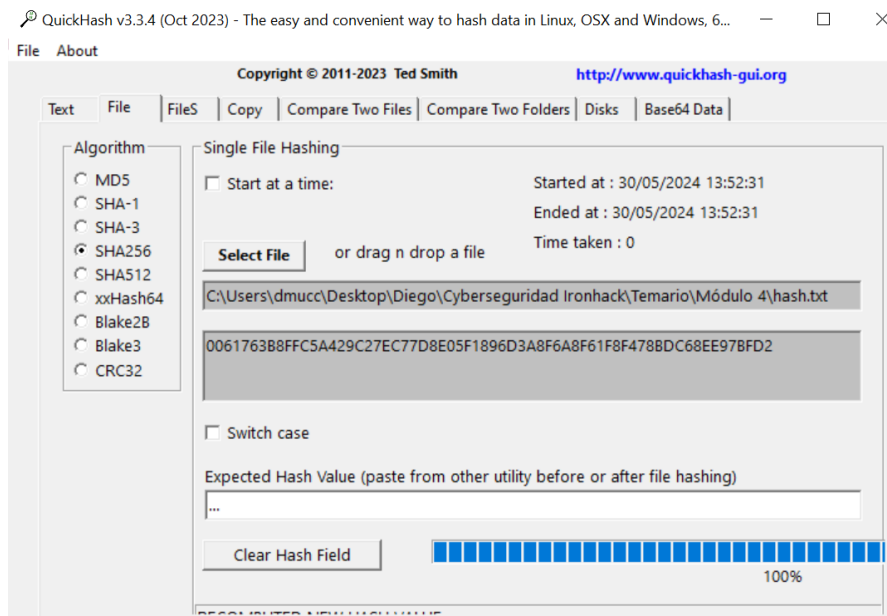
- **Adicionalmente, busca herramientas para el cálculo de hashes mediante interface gráfica y repite el ejercicio.**

Ahora calculamos el hash de este fichero modificado, mediante una calculadora de hash online, la cual nos da el resultado de todos los algoritmos. Como podemos observar, el valor para el algoritmo “md5” y “sha256” es el mismo que en el ejercicio anterior:

es.infobyip.com/hashcalculator.php			
md5	edbe79d037b5f6f7c7234c5b26a3d004	128	32
murmur3a	7c95f1ba	32	8
murmur3c	7fa0137249bf0e62feb82f323c615ca3	128	32
murmur3f	faf1701767c3f1dbad6a748b68d15e62	128	32
ripemd128	f4d65b5de8bdfc4dc6f600cb5ad28b2e	128	32
ripemd160	f302d394c48bda5cbf3f5791b8d791c99ce9882e	160	40
ripemd256	4f716e7f5817db7dfbbb4a452e418b109a5656dfaf9ebf450b53fe2ca25b513b	256	64
ripemd320	45339a4fd667ea8e393da33b24988f866adfb3dc7a925504e52e18bc3bdaff135300eb42795299c5	320	80
sha1	a8b0f13bbcf1b0123bc6f32239de3ce302773aeb	160	40
sha224	e0ec688f8410041199a05cc1e3aa361f37fe67a8d773248dc0e036cd	224	56
sha256	0061763b8ffc5a429c27ec77d8e05f1896d3a8f6a8f61f8f478bdc68ee97bfd2	256	64

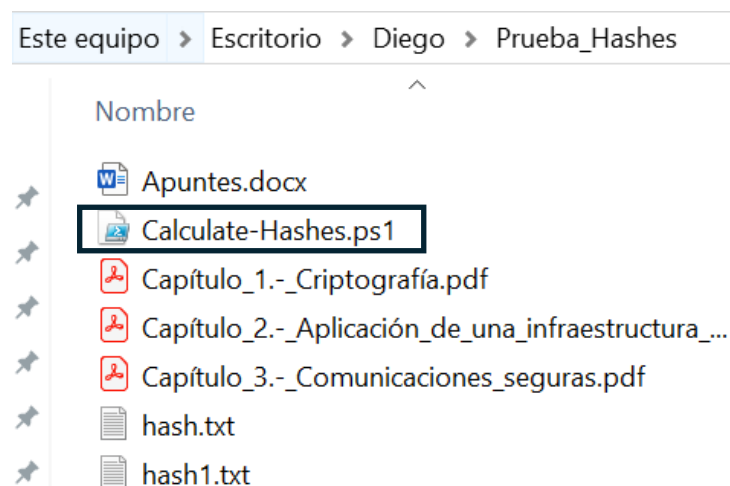
También podemos calcular los hashes con el software “Quickhash-GUI”, el valor es exactamente el mismo para ese fichero que fue modificado:





- **Mediante Powershell, automatiza el cálculo de hashes de todos los ficheros de un directorio.**

Creamos un archivo de texto que va a contener el script para calcular el hash de cada archivo que esté dentro de la carpeta objetivo. Al guardarlo le damos el formato “.ps1”, dentro del directorio recién creado para luego poder ejecutarlo como script.



- El contenido de este *script* es el siguiente:

\$directoryPath = "C:\Users\dmucc\Desktop\Diego\" → Ruta del directorio a analizar

\$outputFile = "C:\Users\dmucc\Desktop\Diego\Cyberseguridad Ironhack\Temario\Módulo 4\hashes_calculados" → archivo que se creará en ese directorio con todos los hashes.

**if (-Not (Test-Path -Path \$directoryPath)) {
Write-Output "El directorio \$directoryPath no existe."
exit
}** → En el caso de que el directorio no existiera o no lo encuentre por algún motivo, le decimos que finalice ahí el script y se imprima el mensaje "El directorio no existe".

\$outputFile → Inicializar el archivo de salida

**Get-ChildItem -Path \$directoryPath -File | ForEach-Object {
\$filePath = \$_.FullName
\$md5 = Get-FileHash -Path \$filePath -Algorithm MD5
\$sha256 = Get-FileHash -Path \$filePath -Algorithm SHA256
"\$filePath - MD5: \$(\$md5.Hash) - SHA-256: \$(\$sha256.Hash)" | Out-File -Append -FilePath \$outputFile
}** → *script* para recorrer todos los archivos del directorio y para calcular tanto el hash MD5, como el hash SHA-256.

Write-Output "Hashes calculados y guardados en \$outputFile" → Una vez finalice de recorrerlo todo, lanzará el mensaje de "Hashes calculados y guardados en -nombre del directorio-".

```

Calculate-Hashes.ps1: Bloc de notas
Archivo Edición Formato Ver Ayuda
# Ruta del directorio a analizar
$directoryPath = "C:\Users\dmucc\Desktop\Diego\Prueba_Hashes"

# Ruta del archivo de salida
$outputFile = "C:\Users\dmucc\Desktop\Diego\Prueba_Hashes\hashes_calculados"

# Verificar si el directorio existe
if (-Not (Test-Path -Path $directoryPath)) {
Write-Output "El directorio $directoryPath no existe."
exit
}

# Inicializar el archivo de salida
$outputFile

# Recorrer todos los archivos en el directorio y calcular los hashes
Get-ChildItem -Path $directoryPath -File | ForEach-Object {
$filePath = $_.FullName
$md5 = Get-FileHash -Path $filePath -Algorithm MD5
$sha256 = Get-FileHash -Path $filePath -Algorithm SHA256
"$filePath - MD5: $($md5.Hash) - SHA-256: $($sha256.Hash)" | Out-File -Append -FilePath $outputFile
}

Write-Output "Hashes calculados y guardados en $outputFile"

```

Ahora, abrimos Powershell y nos dirigimos al directorio especificado arriba. Como nunca se han ejecutado scripts en el sistema, se deberá cambiar la política de ejecución de scripts mediante el siguiente comando “Set-ExecutionPolicy Unrestricted” y posteriormente ejecutamos “.\Calculate-Hashes.ps1” que es el nombre que le hemos dado al *script* anterior:

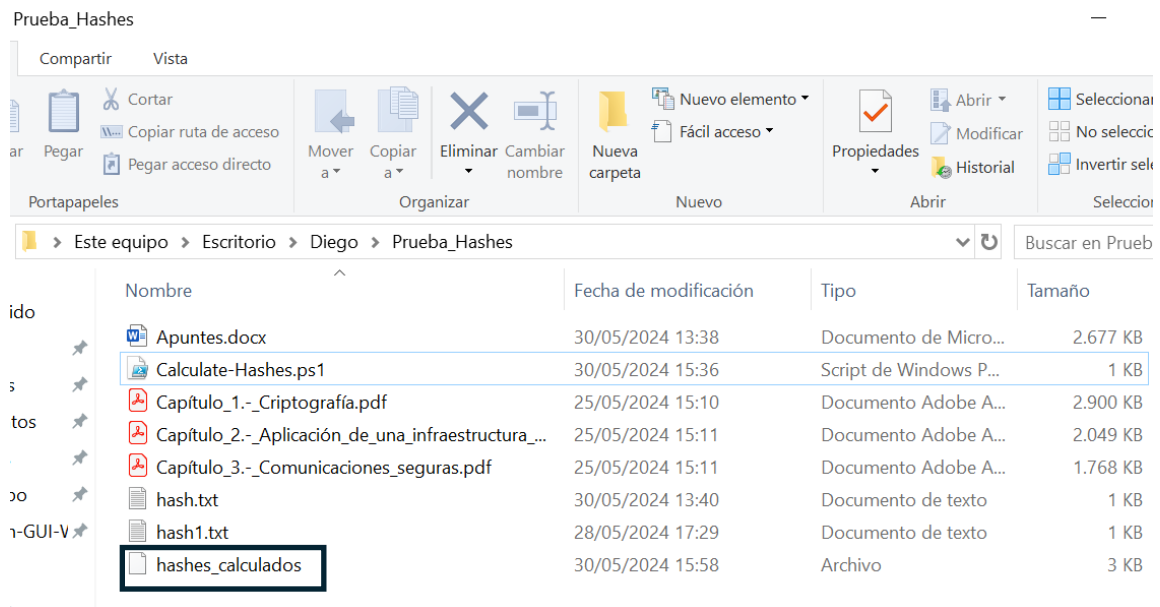
```

PS C:\Users\dmucc\Desktop\Diego\Prueba_Hashes> Set-ExecutionPolicy Unrestricted
>>

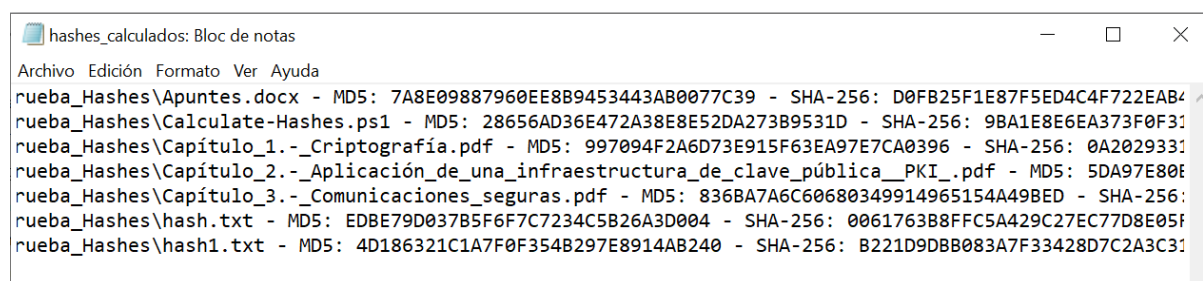
Cambio de directiva de ejecución
La directiva de ejecución te ayuda a protegerte de scripts en los que no confías. Si cambias dicha directiva, podrías exponerte a los
riesgos de seguridad descritos en el tema de la Ayuda about_Execution_Policies en https://go.microsoft.com/fwlink/?LinkID=135170.
¿Quieres cambiar la directiva de ejecución?
[S] Sí [O] Sí a todo [N] No [T] No a todo [U] Suspender [?] Ayuda (el valor predeterminado es "N"): s
PS C:\Users\dmucc\Desktop\Diego\Prueba_Hashes> .\Calculate-Hashes.ps1
C:\Users\dmucc\Desktop\Diego\Prueba_Hashes\hashes_calculados
Hashes calculados y guardados en C:\Users\dmucc\Desktop\Diego\Prueba_Hashes\hashes_calculados

```

Podemos observar en el directorio, la creación de ese nuevo fichero, llamado “hashes_calculados”:



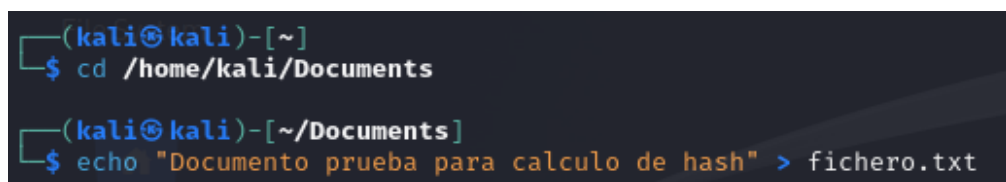
Si lo abrimos, podremos ver el hash MD5 y SHA 256 de cada uno de los archivos presentes en esa carpeta llamada “Prueba_Hashes”:



- En Linux, realiza las mismas operaciones de cálculo de hash mediante las herramientas propias del sistema.

Calcular el hash mediante las herramientas propias del sistema de Kali Linux es relativamente fácil.

Primero crearemos un fichero dentro de la carpeta *Documents*, mediante el comando: `echo "texto que queremos insertar" > nombredelfichero.txt`



Ahora, mediante el comando: `md5sum archivo.txt ; sha256sum fichero.txt` calcularemos el *hash* con algoritmo MD5 y SHA256:

```
(kali㉿kali)-[~/Documents]
$ md5sum fichero.txt ; sha256sum fichero.txt
9ee4486b720c79d6983ead028646c0f1 fichero.txt
4c4237d8ff70df6f24a2c339c11953cff7f4fdbd57abf7260fc508450491bb2e fichero.txt
```

Realizamos cambios en el fichero mediante el comando “nano fichero.txt”.

```
(kali㉿kali)-[~/Documents]
$ nano fichero.txt
```

```
File Actions Edit View Help
GNU nano 8.0
Documento prueba para calculo de hash

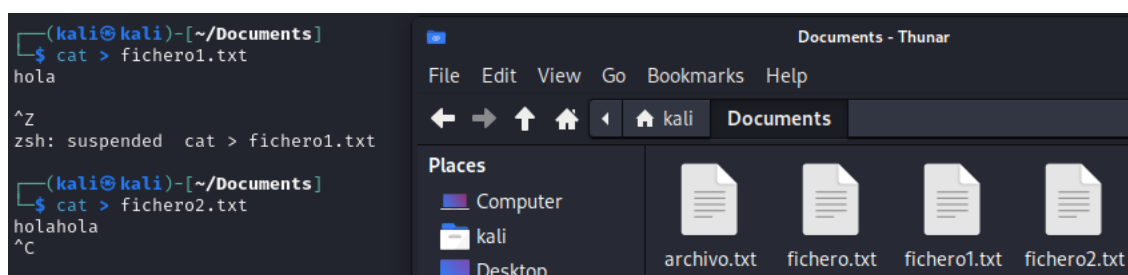
Cambios realizados.█
```

Ahora volvemos a calcular el hash con el mismo comando utilizado anteriormente. Tal y como podemos observar, al modificarse el contenido, el hash también cambia:

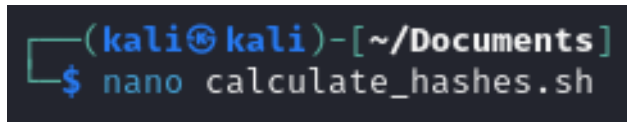
```
(kali㉿kali)-[~/Documents]
$ md5sum fichero.txt ; sha256sum fichero.txt
6406c6efc905192b58af2eeea150e547 fichero.txt
a6d5acf3a39659f09fddb2f156552308b34ed71780ed9c1019df196332e1db45 fichero.txt
```

- Por último, en Linux, automatiza mediante Shell script, la obtención de los hashes de todos los ficheros de un directorio

Primero creamos varios algunos ficheros de texto más en la carpeta *Documents* mediante el comando: `cat > nombredelfichero.txt`
“texto a introducir”



Ahora, creamos un archivo (mediante el comando *nano*) para albergar el script que luego ejecutaremos. Le damos el formato “.sh”, el cual es propio de los archivos bash script. Este archivo lo llamaremos “calculate_hashes.sh”:



Una vez dentro, escribiremos el siguiente script para calcular hashes de todos los ficheros en un directorio:

#!/bin/bash → Esto especifica que el script debe ejecutarse usando Bash

Directorio que quieres procesar

DIRECTORIO="/home/kali/Documents" → Define la ruta al directorio que deseamos procesar.

Archivo de salida

ARCHIVO_SALIDA="/home/kali/Documents/hashes_calculados" → Define el nombre del archivo de salida donde se guardarán los resultados.

Verifica que el directorio existe

if [! -d "\$DIRECTORIO"]; then → Verifica si el directorio existe. Si no existe, muestra el siguiente mensaje y termina el script.

echo "El directorio \$DIRECTORIO no existe."

exit 1

fi

Limpia el archivo de salida si ya existe

> "\$ARCHIVO_SALIDA" → Este comando limpia el archivo de salida si ya existe, asegurando que comience vacío.

Recorre todos los ficheros en el directorio y subdirectorios

find "\$DIRECTORIO" -type f → Utiliza el comando find para buscar todos los ficheros (-type f) en el directorio y sus subdirectorios.) | **while read -r FILE; do** → Recorre cada fichero encontrado

Calcula el hash MD5

MD5_HASH=\$(md5sum "\$FILE" | awk '{ print \$1 }') → Calcula el hash MD5 del fichero y extrae solo el hash usando la herramienta awk.

Calcula el hash SHA-256

SHA256_HASH=\$(sha256sum "\$FILE" | awk '{ print \$1 }') → Calcula el hash SHA-256 del fichero y extrae solo el hash usando herramienta awk.

Imprime el fichero y sus hashes en el archivo de salida

```
echo "$FILE : MD5=$MD5_HASH, SHA256=$SHA256_HASH" >>
"$ARCHIVO_SALIDA" → Imprime el nombre del fichero y sus hashes MD5 y SHA-
256.
```

done

```
File Actions Edit View Help
GNU nano 8.0
#!/bin/bash

# Directorio que quieres procesar
DIRECTORIO="/home/kali/Documents"

# Archivo de salida
ARCHIVO_SALIDA="/home/kali/Documents/hashes_calculados"

# Verifica que el directorio existe
if [ ! -d "$DIRECTORIO" ]; then
    echo "El directorio $DIRECTORIO no existe."
    exit 1
fi

# Limpia el archivo de salida si ya existe
> "$ARCHIVO_SALIDA"

# Recorre todos los ficheros en el directorio y subdirectorios
find "$DIRECTORIO" -type f | while read -r FILE; do
    # Calcula el hash MD5
    MD5_HASH=$(md5sum "$FILE" | awk '{ print $1 }')
    # Calcula el hash SHA-256
    SHA256_HASH=$(sha256sum "$FILE" | awk '{ print $1 }')
    # Imprime el fichero y sus hashes en el archivo de salida
    echo "$FILE : MD5=$MD5_HASH, SHA256=$SHA256_HASH" >> "$ARCHIVO_SALIDA"
done
```

Por último, para ejecutar el script y obtener los resultados en el directorio especificado, debemos ejecutar el comando “./calculate_hashes.sh”, pero aquí nos encontramos con un problema:

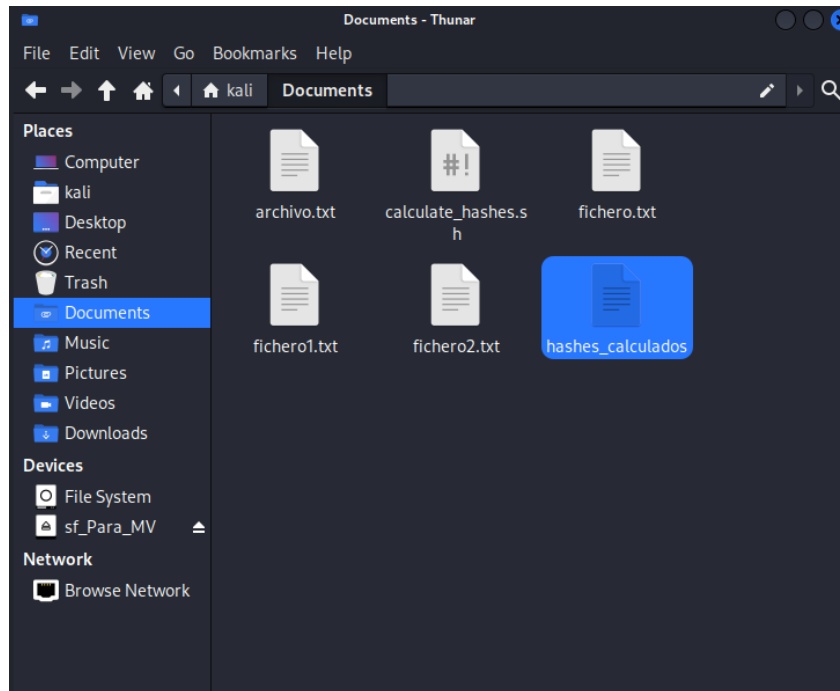
```
(kali@kali)-[~/Documents]
$ ./calculate_hashes.sh
zsh: permission denied: ./calculate_hashes.sh
```

Y es que, al igual que en Windows, debemos otorgarle los permisos necesarios para la ejecución de scripts. Esto lo haremos mediante el comando “chmod +x calculate_hashes.sh” y volvemos a ejecutar el script:

```
(kali@kali)-[~/Documents]
$ chmod +x calculate_hashes.sh

(kali@kali)-[~/Documents]
$ ./calculate_hashes.sh
```

Ahora, podemos ver como se ha creado correctamente el fichero, en el directorio especificado:



Si hacemos doble clic, se nos abrirá el fichero y podremos ver los hashes md5 y sha256 de todos los ficheros que están en la carpeta *Documents*:

```
1 /home/kali/Documents/fichero.txt : MD5=6406c6efc905192b58af2eeea150e547, SHA256=a6d5acf3a39659f09fddb2f156552308b34ed71780ed9c1019df196332e1db45
2 /home/kali/Documents/calculate_hashes.sh : MD5=00a92cf5eb3c238cd0153d457ae05004, SHA256=501725574f35f26b1db6457ad2ea2e5720d28fec6580679cef0274ab0c625262
3 /home/kali/Documents/fichero1.txt : MD5=b216ece280c40836e2090d74bb7963cd, SHA256=01e571f0db80365501c8b2feee0e9c02afba2252d0e158947496a12e4fe526d7
4 /home/kali/Documents/hashses_calculados : MD5=d794e57ac57ec4dc434175a223962d78, SHA256=53a5f98aaa25e75c81a8d86266e0146766b84879367b52ce3a372da353e5d2e2
5 /home/kali/Documents/archivo.txt : MD5=8ce79a694a2afcdcfcb30a6ef26b2d1, SHA256=b901d19b22ae9c3e017df06871eac09377d5cddb5e35db42ce712b94d90784b0
6 /home/kali/Documents/fichero2.txt : MD5=211695952b296f4dae62dbb188dda1d, SHA256=9c45c8e6c2054cd3fab22208bf95b06857a5ae7ede9d96f30dec53cc68a73db
7
```

Buen trabajo. 10/10