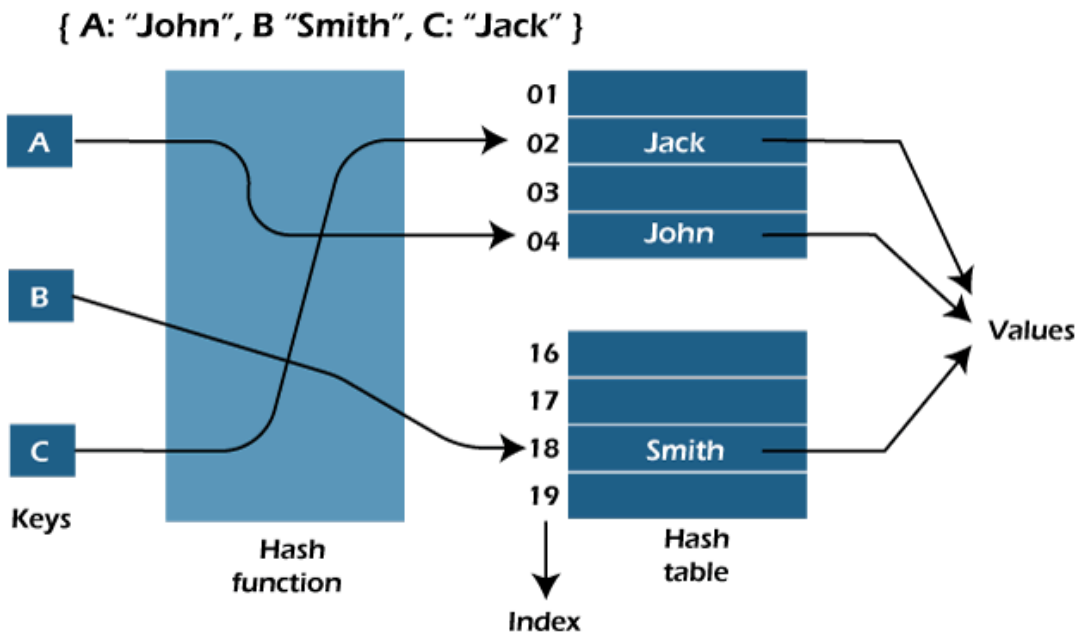
Verificación de integridad de ficheros mediante funciones hash



AUTOR: Pedro Antonio Giménez Meroño

Índice

1. Introducción

1.1 Objetivos

El objetivo de esta tarea es que los estudiantes comprendan el uso de las funciones hash para garantizar la integridad de los archivos, tanto en sistemas Windows como Linux. Aprenderán a generar y verificar huellas digitales utilizando distintas herramientas y algoritmos de hash, reforzando el concepto de seguridad e integridad en los sistemas.

2. Práctica en Windows

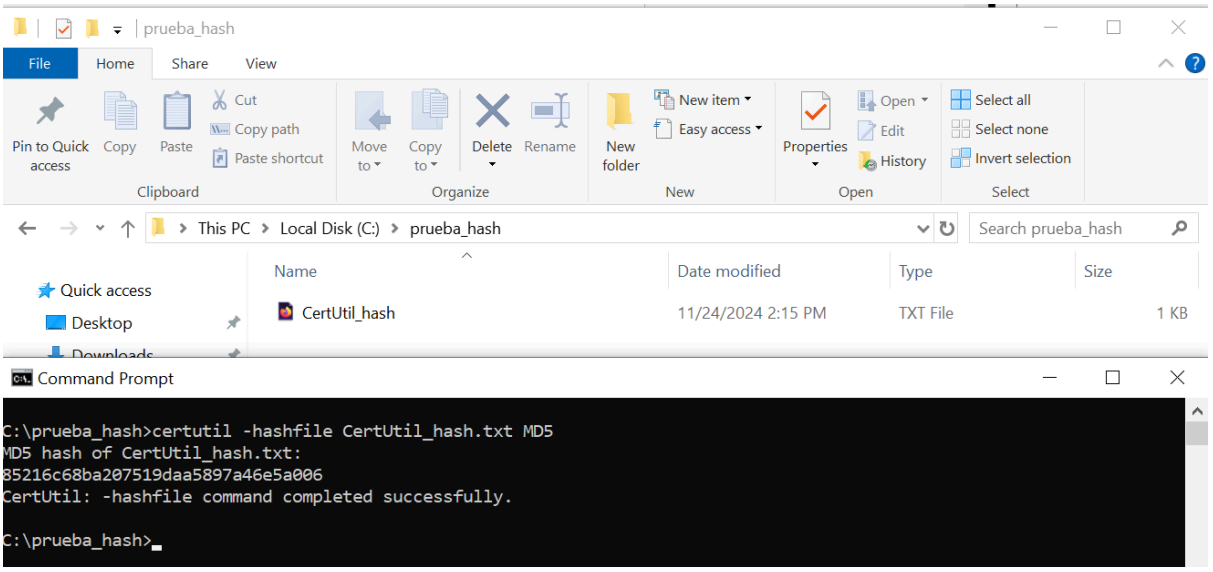
2.1 Utilizando la herramienta CertUtil

Certutil.exe es un programa de línea de comandos que se instala como parte de los servicios de certificados en Windows. Puede usarse para mostrar la información de configuración de la entidad de certificación (CA), configurar los servicios de certificados, realizar copias de seguridad y restaurar los componentes de la CA. El programa también comprueba los certificados, los pares de claves y las cadenas de certificados.

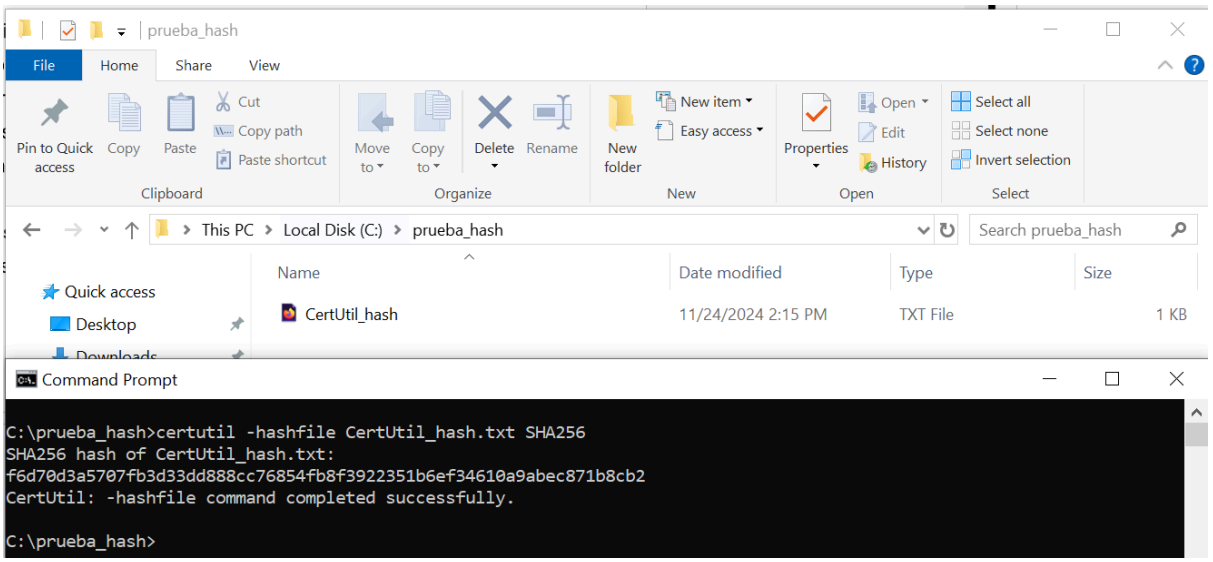
Lo primero que hemos hecho es crear un archivo de prueba llamado “CertUtil_hash.txt” para hacer las pruebas de generar la función hash en Windows.

Después hemos procedido a abrir una terminal cmd y ejecutar el comando correspondiente para usar CertUtil primero probando en la primera imagen el algoritmo MD5 y en la segunda imagen el algoritmo SHA256.

```
certutil -hashfile CertUtil_hash.txt MD5
```




```
certutil -hashfile CertUtil_hash.txt SHA256
```

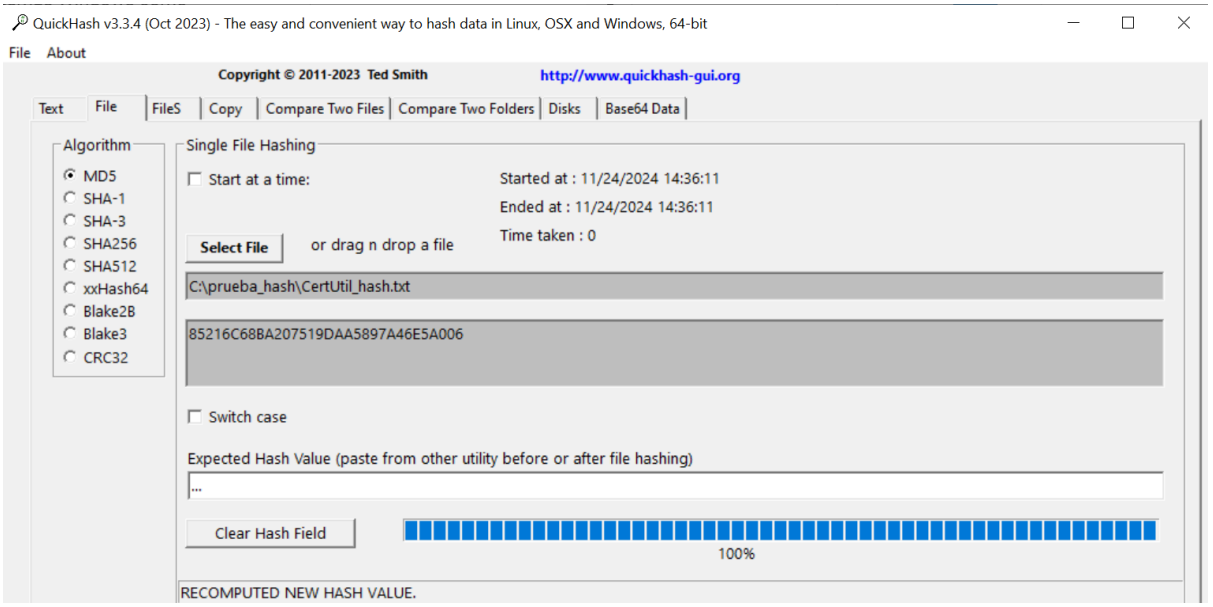


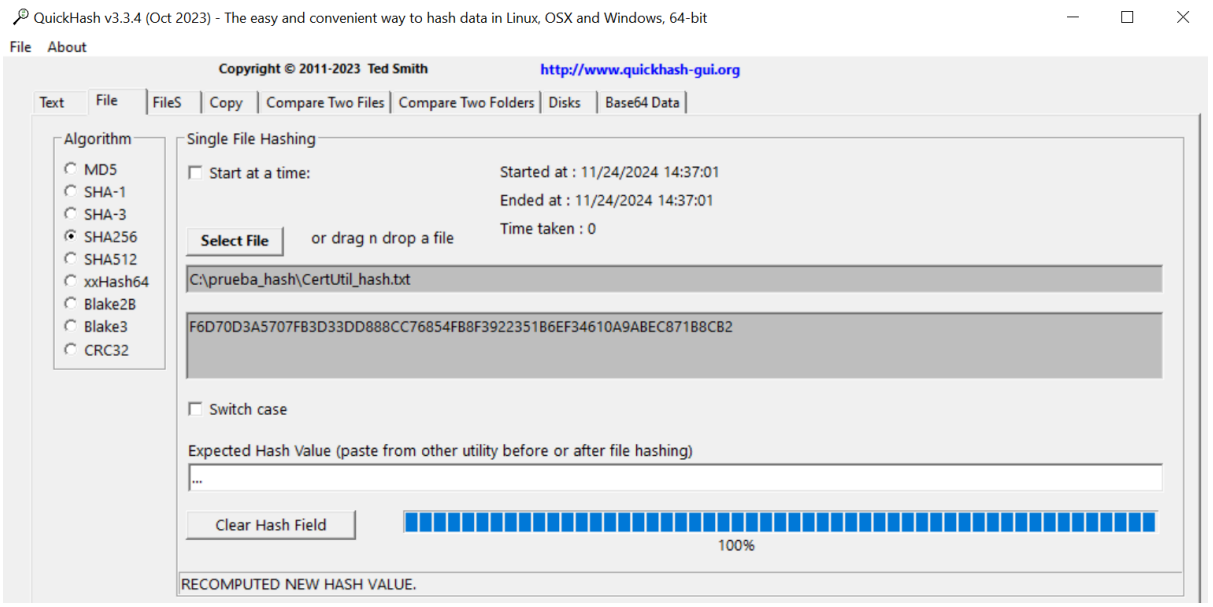
2.2 Utilizando la herramienta QuickHash GUI

QuickHash GUI es una aplicación con interfaz gráfica para crear el hash de cualquier archivo que también nos permite exportar esta información en formato .html o .csv para almacenarlo. Además, también nos muestra información sobre el contenido del archivo como el número de archivos, el espacio total, ruta y mucho más.

Para probar esta herramienta utilizaremos el mismo archivo que utilizamos con CertUtil. Al Iniciar el programa veremos una interfaz como la de la imagen, donde para seleccionar el archivo sobre el que comprobaremos el hash, nos dirigimos a la pestaña File y una vez ahí clicamos en Select File para seleccionar CertUtil_hash.txt.

Por último solo tenemos que escoger en la columna de la izquierda el algoritmo que deseamos utilizar y ya se ejecutará automáticamente sobre el fichero seleccionado. Debajo de la ruta de fichero nos dará la clave hash del archivo, MD5 en la imagen 1 y SHA256 en la imagen 2.





2.3 Opinión con herramientas en Windows

Puedo decir de forma general que ambas herramientas cumplen su función, si bien es cierto que para mi gusto, QuickHash GUI puede ser una mejor opción para el perfil de un usuario que no suele utilizar este tipo de herramientas. Ya sea por su interfaz gráfica que permite gran intuitividad o por las opciones avanzadas que ofrece frente a CertUtil.

A continuación también he redactado las principales diferencias de ambas aplicaciones:

2.3.1 CertUtil

Ventajas:

- CertUtil es una herramienta muy rápida y conveniente que funciona directamente desde el símbolo del sistema, sin necesidad de instalar software adicional.
- Es altamente flexible, ya que es compatible con casi todos los algoritmos populares.

Desventajas:

- No cuenta con una interfaz gráfica de usuario, por lo que no es muy práctica para personas con poca experiencia técnica.

2.3.2 QuickHash GUI

Ventajas:

- QuickHash GUI es compatible con una amplia gama de algoritmos y operaciones en archivos, texto y directorios.
- Es muy útil para comparar un gran número de hashes al mismo tiempo, lo que facilita la gestión de muchos archivos.
- No requiere instalación y es una herramienta portátil, lo que hace que sea muy fácil de usar.

Desventajas:

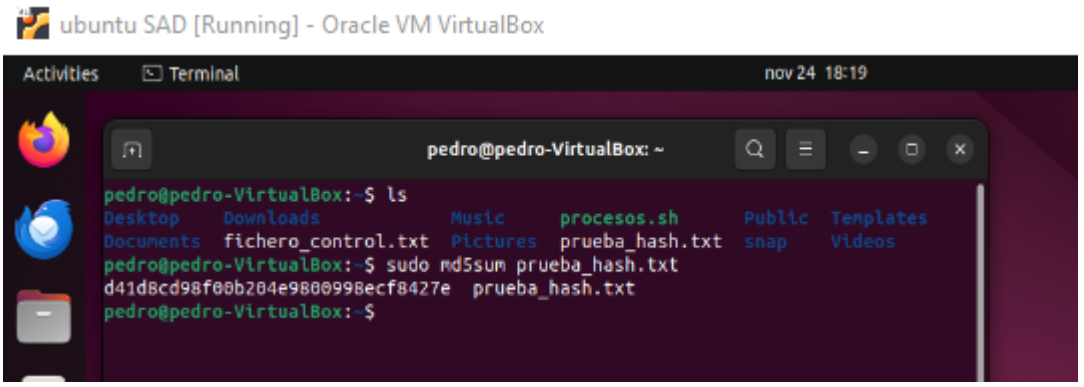
- Puede resultar un poco más difícil para usuarios ocasionales realizar estas tareas en Windows.

3. Práctica en Linux

3.1 Utilizando md5sum

Para realizar la función hash de un fichero en Linux utilizaremos la terminal. He creado un archivo para hacer estas pruebas “prueba_hash.txt”. Solo tenemos que ejecutar para el algoritmo MD5 el siguiente comando que nos debe dar la función hash correspondiente como se ve en la captura de pantalla.

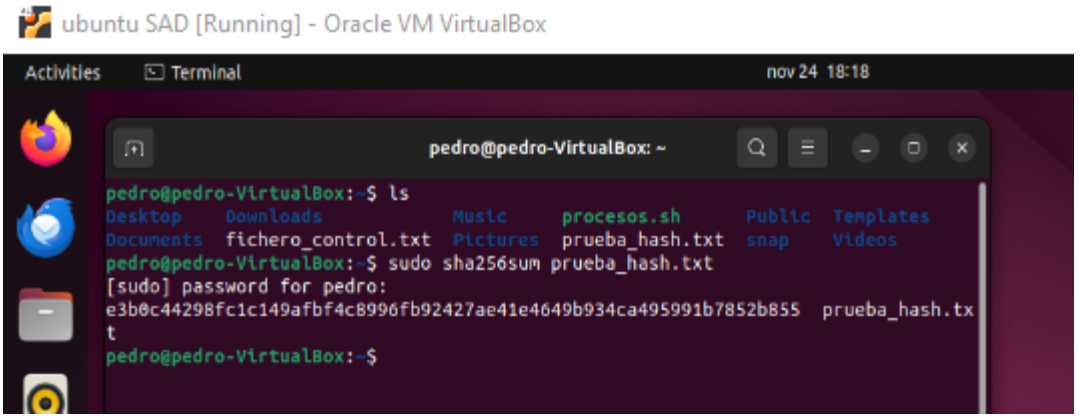
```
md5sum prueba_hash.txt
```



3.2 Utilizando sha256sum

El escenario es igual que el anterior tan solo cambiara el comando. Ejecutmaos para el algoritmo SHA256 el siguiente comando que nos debe dar la función hash correspondiente como se ve en la captura de pantalla.

```
sha256sum prueba_hash.txt
```



3.3 Verificar la integridad de un archivo descargado de internet

Esto consiste en simplemente ver si la función hash que proporciona la plataforma de descarga de tu archivo coincide con tu archivo una vez descargado. Esto es especialmente importante para prevenir la instalación de software modificado o comprometido.

3.3.1 Pasos para verificar la integridad

1. Obtener el hash oficial:

Antes de descargar el archivo, verifica si el sitio web del proveedor ofrece el hash correspondiente (generalmente publicado junto al enlace de descarga).

2. Calcular el hash del archivo descargado:

Usa el algoritmo que indique que utiliza el proveedor de descarga del fichero (md5, sha256,...) para generar el hash del archivo descargado. Por ejemplo si utiliza sha256:

```
sha256sum prueba_hash.txt
```

3. Comparar los hashes:

Compara el hash generado con el proporcionado por el sitio oficial. Si los hashes coinciden, el fichero está íntegro y no ha sido alterado. Si no coinciden, probablemente esté dañado o comprometido.

4. Cuestiones y conclusiones sobre algoritmos de verificación

¿Por qué los algoritmos MD5 y SHA-1 ya no son recomendados para aplicaciones críticas? Da un ejemplo de una situación en la que el uso de estos algoritmos podría representar un riesgo. Indica en qué situaciones podría ser aceptable utilizar MD5 en lugar de algoritmos más seguros como SHA-256 o SHA-512.

La razón por la que los algoritmos MD5 y SHA-1 se consideran inseguros es por los problemas con ataques de colisión. Una colisión se produce cuando dos entradas totalmente diferentes proporcionan el mismo hash, anulando así por completo el propósito de integridad y autenticidad en datos para el que se utilizan habitualmente las funciones hash.

Estas vulnerabilidades impiden el uso de MD5 y SHA-1 para aplicaciones como firmas digitales, hash de contraseñas o autenticación segura de sistemas. Un ejemplo seria, todo el peligro de utilizar MD5 o SHA-1 en la emisión de certificados digitales. Si un atacante puede crear una colisión en un certificado adecuado, se puede crear uno falso que colisione con la firma digital del certificado adecuado. Eso permite cometer diferentes ataques, como phishing o MITM.

En el caso de una aplicación en un entorno crítico, no se recomienda el uso de MD5, pero se podría considerar en situaciones en las que la velocidad de cálculo importa y donde el nivel de seguridad y la posibilidad de un ataque no son altos. Por ejemplo, se podría utilizar para confirmar la integridad de archivos en redes cerradas donde la probabilidad es extremadamente baja frente a manipulación maliciosa. Incluso en este caso, los administradores de sistemas eligen la mayoría de veces algoritmos como SHA-256 o SHA-512, porque ofrecen una seguridad muy superior sin afectar el rendimiento.