Documentación: Verificación de Integridad de Archivos ISO con SHA-256

Objetivo

Implementar un sistema de verificación de integridad para archivos ISO utilizando el algoritmo de hash SHA-256, comparando el hash calculado localmente con un hash oficial proporcionado.

¿Qué es SHA-256?

SHA-256 (Secure Hash Algorithm 256-bit) es una función hash criptográfica perteneciente a la familia SHA-2. Genera un valor hash de 256 bits (32 bytes) que representa de forma única los datos de entrada.

Características principales de SHA-256:

- **Tamaño de hash**: 256 bits (64 caracteres hexadecimales)
- **Resistencia a colisiones**: Extremadamente difícil encontrar dos entradas diferentes que produzcan el mismo hash
- **Propiedad de avalancha**: Pequeños cambios en la entrada producen cambios significativos en el hash
- **Algoritmo**: Basado en la compresión Merkle-Damgård
- **Seguridad**: Considerado seguro contra ataques conocidos

Importancia de la Verificación de ISOs

La verificación de hashes es crucial para:

- **Autenticidad**: Confirmar que el archivo proviene de la fuente oficial
- **Integridad**: Detectar corrupción durante la descarga o manipulación
- **Seguridad**: Prevenir la instalación de software malicioso modificado

Proceso de Verificación

1. Cálculo del Hash SHA-256

python

```
def calcular_hash_iso(ruta_iso):
    h = hashlib.new("sha256")
    with open(ruta_iso, "rb") as f:
        for bloque in iter(lambda: f.read(4096), b""):
            h.update(bloque)
    return h.hexdigest()
```

2. Lectura del Hash Oficial

python

```
def leer_hash_oficial(ruta_hash, nombre_iso):
    with open(ruta_hash, "r") as f:
        for linea in f:
            if nombre_iso in linea:
                return linea.split()[0]
    return None
```

3. Comparación y Verificación

python

```
def verificar_iso(ruta_iso, ruta_hash):
    nombre_iso = ruta_iso.split("/")[-1]
    hash_calculado = calcular_hash_iso(ruta_iso)
    hash_oficial = leer_hash_oficial(ruta_hash, nombre_iso)

    if hash_oficial is None:
        print("No se proporcionó un hash oficial para esta ISO.")
        return False

    if hash_calculado == hash_oficial:
        print("La ISO es auténtica. Los hashes coinciden. Proceda con la instalación.")
        return True
    else:
        print("La ISO no coincide con el hash oficial. OJO PUEDE SER MALICIOSA.")
        return False
```

Formato del Archivo de Hashes

El archivo de hashes oficial debe seguir el formato estándar:

text

```
hash_sha256  nombre_archivo.iso
```

hash_sha256 otro_archivo.iso

Ejemplo:

text

e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855 ubuntu-20.04.2-desktop-amd64.iso

Ejemplo de Uso

Configuración:

python

```
if __name__ == "__main__":
    ruta_iso = "descargas/ubuntu-20.04.2-desktop-amd64.iso"  # Ruta al archivo ISO
    ruta_hash = "hashes/SHA256SUMS"  # Ruta al archivo de hashes oficial
    verificar_iso(ruta_iso, ruta_hash)
```

Resultados Posibles:

Caso 1: Verificación Exitosa

text

La ISO es auténtica. Los hashes coinciden. Proceda con la instalación.

Caso 2: Verificación Fallida

text

La ISO no coincide con el hash oficial. OJO PUEDE SER MALICIOSA.

Caso 3: Hash No Encontrado

text

No se proporcionó un hash oficial para esta ISO.

Ventajas de SHA-256 sobre MD5/SHA-1

Algoritmo	Tamaño Hash	Seguridad	Recomendado
MD5	128 bits	Vulnerable	No
SHA-1	160 bits	Vulnerable	No

Algoritmo	Tamaño Hash	Seguridad	Recomendado
SHA-256	256 bits	Seguro	Sí

Consideraciones de Seguridad

Aspectos Positivos:

- **Lectura por bloques**: Maneja archivos grandes eficientemente sin consumir mucha memoria
- **Validación de existencia**: Verifica si el hash oficial existe para el archivo
- **Mensajes claros**: Informa al usuario sobre el estado de la verificación

Consideraciones:

- **Modo de operación**: El archivo de hashes debe ser de una fuente confiable
- **Protección del archivo de hashes**: Debe estar firmado digitalmente o descargado por canal seguro
- **Sistemas de archivos**: Funciona mejor con rutas absolutas para evitar confusiones

Instrucciones de Ejecución

Requisitos:

- Python 3.x
- Módulo hashlib (incluido en la librería estándar)

Ejecución:

bash

python verificar_iso.py

Estructura de Directorios Recomendada:

text

proyecto/

├── verificar_iso.py

├── descargas/

│ └── archivo.iso

└── hashes/

└── SHA256SUMS

Pruebas Recomendadas

Vector de Prueba 1: Archivo Válido

- **Archivo ISO**: Ubuntu 20.04 LTS
- **Hash oficial**: Consultar en el sitio oficial de Ubuntu
- **Resultado esperado**: Verificación exitosa

Vector de Prueba 2: Archivo Modificado

- **Archivo ISO**: Modificar 1 byte del archivo original
- **Resultado esperado**: Verificación fallida

Vector de Prueba 3: Archivo Inexistente

- **Archivo ISO**: Ruta incorrecta
- **Resultado esperado**: Error de archivo no encontrado

Vector de Prueba 4: Hash No Encontrado

- **Nombre de archivo**: No coincide con ninguna entrada en SHA256SUMS
- **Resultado esperado**: Hash oficial no proporcionado

Conclusiones

Ventajas de la Implementación:

- **Sencillez**: Código claro y fácil de entender
- **Eficiencia**: Lectura por bloques para archivos grandes
- **Robustez**: Manejo de casos edge (hash no encontrado)
- **Portabilidad**: Funciona en cualquier sistema con Python

Posibles Mejoras:

- Añadir verificación de firma digital del archivo de hashes
- Soporte para múltiples algoritmos hash (SHA-512, BLAKE2)
- Interfaz gráfica de usuario
- Verificación por lotes de múltiples archivos

Estado del Proyecto:

Funcionalidad: Completa
Seguridad: Adecuada para uso básico
Usabilidad: Sencilla de usar

Esta implementación proporciona una verificación confiable de integridad de archivos ISO usando SHA-256, ayudando a garantizar que los archivos descargados sean auténticos y no hayan sido modificados o corruptos.