Tecnológico de Costa Rica

Escuela de Computadores

Curso: Algoritmos y Estructuras de Datos II

Anteproyecto:

Detección de manipulación en imágenes y documentos digitales mediante funciones Hash SHA-256

Elaborado por:

Bryan Sibaja Garro 2022207842

María Pamela Chacón Barrantes 2023110706

Jose Julián Duarte Astua 2022437580

Profesor: Luiz Barboza

Fecha: Octubre, 2024

Descripción General del Proyecto

Este proyecto se centra en desarrollar una solución de software en C++ que permita verificar la integridad de archivos y detectar manipulaciones en documentos digitales mediante el uso de funciones hash, como SHA-256. Esta herramienta de verificación de integridad está diseñada para identificar cualquier alteración en los archivos mediante la comparación de los hashes antes y después de modificaciones simuladas. Además, se incluye un módulo de cifrado básico para proteger los datos confidenciales almacenados en los sistemas de grandes empresas. La necesidad de contar con métodos confiables para la verificación de integridad y la protección de datos surge ante el incremento de ataques a la integridad de datos, que ponen en riesgo la confidencialidad y la confiabilidad de la información.

Justificación

El proyecto aborda una necesidad crítica de las grandes empresas: proteger la integridad y confidencialidad de sus datos. La verificación mediante funciones hash es una solución accesible y eficiente para garantizar que los archivos no han sido manipulados o corrompidos. Asimismo, implementar un módulo de cifrado básico ayuda a resguardar archivos confidenciales, reforzando la seguridad de la información. Este proyecto se alinea con las políticas nacionales de seguridad de la información y con la tendencia global de asegurar los datos en el entorno empresarial, lo cual tiene beneficios económicos y sociales, ya que protege la confidencialidad y la integridad de datos sensibles.

Marco Teórico

1. Funciones Hash y SHA-256: Descripción y Relevancia en la Verificación de Integridad

Las funciones hash son herramientas matemáticas que transforman un mensaje o dato en una cadena de longitud fija, jugando un papel clave en la verificación de integridad y autenticidad en la seguridad informática. SHA-256 (Secure Hash Algorithm 256-bit), desarrollado por la Agencia de Seguridad Nacional de los EE. UU., es una función hash de la familia SHA-2 y es ampliamente utilizada en sectores como banca, criptomonedas y seguridad en la nube. Su resistencia a colisiones, la baja probabilidad de que dos mensajes diferentes generen el mismo hash, la convierte en un método confiable y seguro para asegurar la integridad de los archivos al comparar los hash antes y después de la trasmisión. Esta función resulta especialmente relevante en aplicaciones donde la autenticidad de los datos debe ser preservada y verificada automáticamente.

M. Stamp (2011) destaca que SHA-256 es esencial en sistemas que requieren alta seguridad sin comprometer la velocidad de procesamiento, motivo por el cual ha sido adoptada como estándar en diversas plataformas de encriptación y verificación de datos.

Manipulación de Archivos Digitales: Estudios de Casos y Técnicas de Detección de Alteraciones

La manipulación de archivos digitales, como documentos y contenido multimedia, ha sido objeto de estudios significativos debido a su impacto en la autenticidad de la información.En 2018 se crearon técnicas de detección de alteraciones en archivos PDF y multimedia mediante herramientas hash y métodos de análisis forense para identificar cambios no autorizados. Estas técnicas incluyen no sólo la comparación de hashes, sino también la detección de patrones y firmas digitales alteradas, que permiten determinar si un archivo ha sido manipulado.

En el campo de la ciberseguridad, se profundiza en cómo los algoritmos hash, incluyendo SHA-256, pueden actuar como prueba de autenticidad para prevenir fraudes y confirmar el origen de documentos. Se resalta la combinación de SHA-256 con análisis forense avanzado como una práctica robusta para la detección de alteraciones en datos, especialmente en contextos corporativos y gubernamentales donde la autenticidad y el historial de archivos son críticos.

Bibliotecas de Criptografía en C++: OpenSSL y otras Herramientas para la Implementación de SHA-256

Para la implementación de SHA-256 en aplicaciones de C++, OpenSSL es una de las bibliotecas de código abierto más utilizadas. OpenSSL permite generar hashes SHA-256, además de soportar un amplio rango de algoritmos criptográficos y funciones de cifrado y descifrado de datos. Hay estudios que resaltan que OpenSSL ofrece una estructura robusta y fácilmente integrable para desarrolladores en entornos de alta exigencia de seguridad.

De la mismo modo, existen otras bibliotecas como Crypto++ y libsodium, analizadas en estudios comparativos debido a su facilidad de uso y soporte para múltiples algoritmos de hash, incluyendo SHA-256. Crypto++ destaca en aplicaciones de código cerrado gracias a su versatilidad y flexibilidad, permitiendo la adaptación a sistemas que requieren múltiples algoritmos hash y criptografía simétrica y asimétrica, lo que la hace atractiva en desarrollos corporativos y especializados.

Seguridad Informática y Protección de Datos: Métodos para la Protección de la Integridad de Datos Digitales

La integridad de los datos digitales es fundamental en la seguridad informática moderna. En el 2021 se presenta una revisión exhaustiva de los métodos y herramientas utilizados para proteger datos digitales, enfatizando el papel de los algoritmos hash, las firmas digitales y los certificados de autenticidad. Los autores argumentan que las estrategias combinadas de verificación hash, como

SHA-256, junto con cifrado y autenticaciones múltiples, son cruciales para construir sistemas robustos contra intentos de manipulación y accesos no autorizados.

La implementación de funciones hash como SHA-256, en conjunto con cifrado asimétrico, se ha demostrado efectiva en aplicaciones corporativas, ya que protege la integridad de la información en redes internas y en comunicaciones externas (Katz & Lindell, 2020). Estos algoritmos hash también ayudan a prevenir ataques de intermediarios y ataques de repetición, asegurando la confiabilidad de las transmisiones de datos.

Objetivos

1. Objetivo general:
    - Desarrollar un programa en C++ que permita detectar la manipulación de imágenes y documentos digitales utilizando funciones hash SHA-256.
2. Objetivos específicos:
    a. **Implementar el cálculo de hashes SHA-256:**
        i. Utilizar la biblioteca OpenSSL (o similar) para calcular el hash SHA-256 de un archivo dado.
        ii. Almacenar el hash generado para su posterior comparación.
    b. Desarrollar una interfaz de usuario:
        i. Crear una interfaz sencilla (consola o gráfica básica) que permita al usuario seleccionar el archivo a analizar.
        ii. Mostrar los resultados de la comparación de hashes de forma clara y concisa.
    c. Simular la manipulación de archivos:
        i. Permitir al usuario modificar el archivo original (imagen o documento) a través de un editor externo o una función básica de edición integrada en el programa.
    d. Comparar hashes para detectar modificaciones:

i. Implementar la lógica para comparar el hash inicial del archivo con el hash calculado después de la simulación de manipulación.
ii. Informar al usuario si el archivo ha sido modificado o no, basándose en la comparación de hashes.

Metodología

Esta sección describe los métodos y herramientas que se emplearán para llevar a cabo el proyecto de verificación de integridad de archivos.

Enfoques de solución:

1. **Cálculo de hashes:** Usar OpenSSL en C++ para calcular hashes SHA-256.
2. **Simulación de manipulación:** El usuario puede editar el archivo con un editor externo o una función básica de edición en el programa.
3. **Comparación de hashes:** Implementar una función para comparar el hash inicial con el nuevo y determinar si hubo cambios.

Herramientas:

- **Lenguaje de programación:** C++
- **Biblioteca:** OpenSSL
- **Sistema operativo:** Linux
- **Editor de código:** Visual Studio Code
- **Control de versiones:** Git y GitHub

Técnicas:

- **Programación orientada a objetos:** Para estructurar y modularizar el código.
- **Manejo de archivos:** Para lectura y escritura en el sistema.
- **Manejo de excepciones:** Para gestionar errores durante la ejecución.

Procedimientos:

1. **Configuración del entorno:** Instalar C++, OpenSSL, y configurar GitHub.
2. **Desarrollo:** Implementar el cálculo de hashes, la interacción con el usuario, y la comparación de hashes.
3. **Documentación:** Documentar el código y elaborar un informe del proyecto.

Extensiones Posibles:

- Guardar el hash inicial para futuras verificaciones.
- Verificar la integridad de múltiples archivos.
- Explorar otras funciones hash y firmas digitales para mayor seguridad.

Este proyecto ofrece una introducción práctica al uso de funciones hash para la verificación de la integridad de datos, con un nivel de dificultad accesible para estudiantes de nivel medio-bajo.

Plan de Acción

Para una mejor organización, se presenta el plan de acción en una tabla que incluye los objetivos específicos, los productos a obtener, las actividades a realizar, el período de realización y el responsable de cada actividad.

Objetivo específico	Producto a obtener	Actividades	Período	Responsable
Implementar el cálculo de hashes SHA-256	Función para calcular hashes	Investigar la biblioteca OpenSSL, implementar la función, realizar pruebas unitarias	Semana 16	Bryan Daniel Sibaja Garro
Desarrollar una interfaz de usuario	Interfaz de consola	Diseñar la interfaz, implementar la interacción con el usuario	Semana 16	Jose Julián Duarte Astua y María Pamela Chacón Barrantes
Simular la manipulación de archivos	Función para simular la manipulación	Investigar métodos de edición, implementar la función	Semanas 17-18	Todos los integrantes
Comparar hashes para detectar modificaciones	Función para comparar hashes	Implementar la lógica de comparación, mostrar resultados al usuario	Semanas 17-18	Todos los integrantes
Documentar el proyecto	Informe del proyecto	Documentar el código, redactar el informe	Semanas 18-19	Todos los integrantes

Definición del Cronograma

El cronograma del proyecto se define de la siguiente manera, teniendo en cuenta que la duración total del proyecto es de 4 semanas (de semana 16 a semana 19):

Semana 1:

- Investigación y familiarización con la biblioteca OpenSSL.
- Implementación de la función para calcular el hash SHA-256.
- Pruebas unitarias de la función de cálculo de hash.
- Inicio de la documentación del código.
- Diseño de la interfaz de consola.
- Implementación de la interacción con el usuario (selección de archivos, etc.).
- Pruebas de la interfaz de usuario.
- Continuación de la documentación.

Semana 2 y 3:

- Investigación de métodos para la simulación de manipulación de archivos.
- Implementación de la función para simular la manipulación.
- Pruebas de la función de simulación.
- Actualización de la documentación.
- Implementación de la lógica para comparar hashes.
- Integración de todas las funciones del programa.

Semana 4:

- Finalización de la documentación y redacción del informe.

Divulgación y Transferencia de Tecnología

Dado que este proyecto es principalmente académico, la divulgación y transferencia de tecnología se enfocará en la difusión del conocimiento adquirido y los resultados obtenidos. Por lo cual, se propone lo siguiente:

- **Presentación del proyecto:** Se realizará una presentación (o defensa) del proyecto al profesor, incluyendo una demostración del programa y la explicación de su funcionamiento.
- **Informe del proyecto:** Se elaborará un informe escrito que documente el proyecto en detalle, incluyendo la metodología, el código fuente y los resultados de las pruebas. Este informe se compartirá con el profesor.
- **Código fuente:** El código fuente del proyecto se publicará en un repositorio de GitHub.

Estas acciones de divulgación y transferencia de tecnología contribuirán a la difusión del conocimiento sobre la detección de manipulación en archivos digitales y al desarrollo de habilidades en programación y seguridad informática.

Presupuesto Detallado

Partida	Descripción	Cantidad	Precio Unitario	Costo Total	Justificación
Software					
Licencia OpenSSL	Uso de OpenSSL para el cálculo de hashes SHA-256 (licencia de soporte opcional, software gratuito)	1	$0 (gratis)	$0	OpenSSL es gratuito, pero la opción de soporte profesional facilita la resolución de problemas en empresas de gran escala.
Herramientas de IDE	IDE como Visual Studio Code para programación en C++ (gratuito)	1	$0 (gratis)	$0	Visual Studio Code es gratuito y proporciona el entorno necesario para desarrollar en C++.
Desarrollo y pruebas					
Horas de desarrollo	Programación, pruebas unitarias y funcionales del sistema (aproximado: 80 horas)	80 hrs	$15/hr	$1200	Incluye el tiempo de implementación y pruebas iniciales en varias configuraciones para garantizar el correcto funcionamiento.
Simulación de archivos	Creación de archivos de prueba para imágenes y documentos (falsificación de datos controlada)	10	$10	$100	Se utilizarán archivos variados para evaluar la detección de cambios en distintos tipos de contenido.
Computadora de desarrollo	Uso de equipo con suficiente capacidad para ejecutar y probar el programa (coste proporcional)	1	$200	$200	Se asigna un costo proporcional por el uso de un equipo adecuado durante la fase de desarrollo del proyecto.
Capacitación y Documentación					
Materiales de capacitación	Guías, tutoriales y artículos sobre hash y seguridad digital	1	$100	$100	El equipo debe contar con acceso a documentación clara y tutoriales, especialmente para capacitaciones futuras.
Informe final y tutorial	Redacción y diseño de documentación técnica y tutoriales para compartir resultados	1	$100	$100	Incluirá la metodología, uso de hash en seguridad y explicación de la implementación, ideal para futuras referencias.
Divulgación					
Publicación en línea	Distribución del informe y tutorial en plataformas educativas	1	$100	$100	Facilita el acceso al conocimiento y los resultados del proyecto a través de plataformas especializadas.
Presentación en eventos	Preparación de una charla o taller en eventos de ciberseguridad	1	$100	$100	La divulgación en eventos permite el intercambio de experiencias y promueve el uso de esta tecnología en distintos entornos.
Total:			$1800		