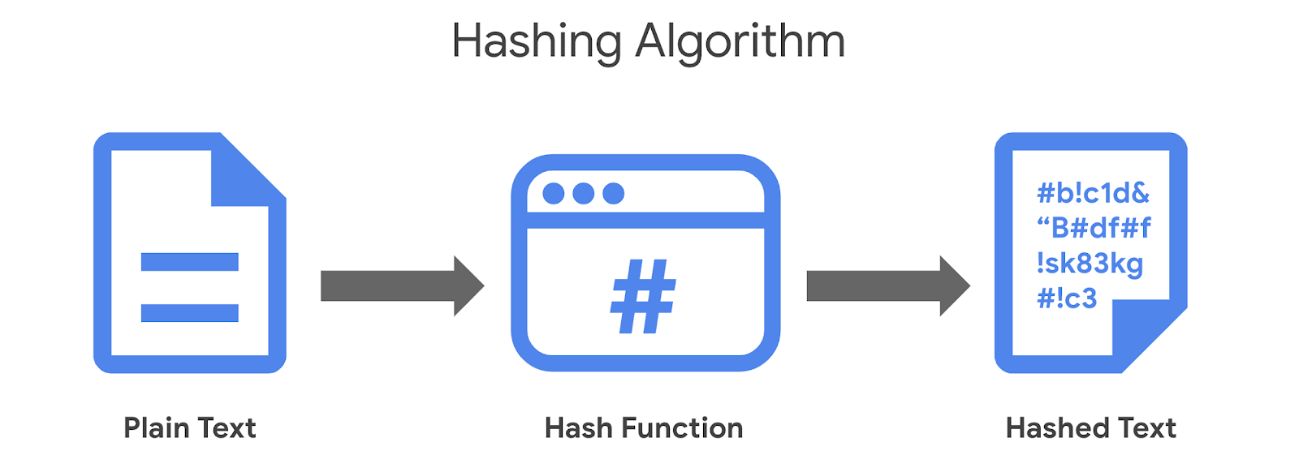
**La evolución de las funciones hash**

Las funciones hash son controles importantes que forman parte de la estrategia de seguridad de todas las empresas. El hashing se utiliza ampliamente para la autenticación y el **no repudio**, que es el concepto de que la autenticidad de la información no puede ser negada.

Anteriormente, aprendiste que **las funciones hash** son algoritmos que producen un código que no se puede descifrar**.** Las funciones hash convierten la información en un valor único que luego puede utilizarse para determinar su integridad. En esta lectura, aprenderás sobre los orígenes de las funciones hash y cómo han cambiado con el tiempo.



**Orígenes del hashing**

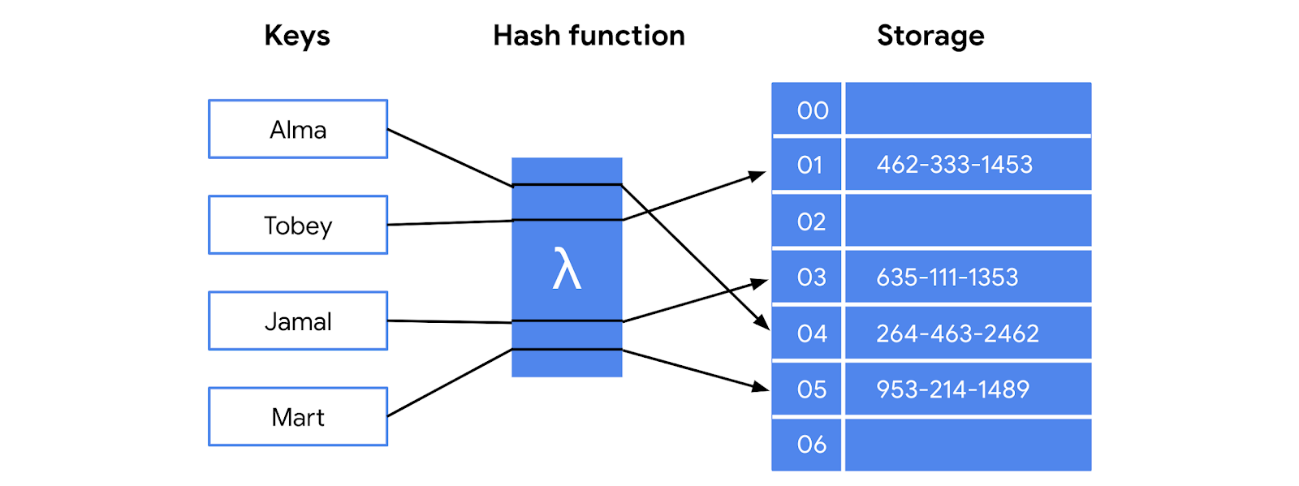
Las funciones hash existen desde los inicios de la informática. Originalmente, fueron creadas como una forma de buscar datos de manera rápida. Desde el principio, estos algoritmos han sido diseñados para representar datos de cualquier tamaño como valores pequeños de tamaño fijo o compendios. Mediante el uso de una tabla hash, que es una estructura de datos utilizada para almacenar y referenciar valores hash, estos pequeños valores se convirtieron en una forma más segura y eficiente para que las computadoras accedan a los datos.

Una de las primeras funciones hash es Message Digest 5, más conocida como MD5. El profesor Ronald Rivest, del Instituto Tecnológico de Massachusetts (MIT), desarrolló MD5 a principios de la década de 1990 como una forma de verificar que un archivo enviado a través de una red coincidiera con su archivo de origen.

Ya sea utilizado para convertir un solo correo electrónico o el código fuente de una aplicación, MD5 funciona convirtiendo datos en un valor de 128 bits. Recordarás que un **bit** es la unidad más pequeña de medición de datos en una computadora. Los bits pueden ser 0 o 1. En una computadora, los bits representan la entrada del usuario de una manera que las computadoras pueden interpretar. En una tabla hash, esto se muestra como una cadena de 32 caracteres. Cualquier alteración en el archivo de origen, genera un valor hash completamente nuevo.

Generalmente, cuanto más largo es el valor hash, más seguro es. Poco después de la creación de MD5, los expertos en seguridad descubrieron que los compendios de 128 bits daban lugar a una importante vulnerabilidad.

Este es un ejemplo de cómo el texto simple se convierte en valores hash:



**Colisiones de hash**

Una de las fallas de MD5 es, de hecho, una característica de todas las funciones hash. Los algoritmos hash asignan cualquier entrada, independientemente de su extensión, a un valor de tamaño fijo compuesto por letras y números. ¿Y cuál es el problema? Aunque existe una cantidad infinita de entradas posibles, ¡solo hay un conjunto finito de salidas disponibles!

Los valores MD5 están limitados a una extensión de 32 caracteres. Debido al tamaño de salida limitado, el algoritmo se considera vulnerable a la **colisión de hash**, una instancia en la que diferentes entradas producen el mismo valor hash. Dado que los hash se utilizan para la autenticación, una colisión de hash es similar a copiar la identidad de alguien. Los atacantes pueden llevar a cabo ataques de colisión para suplantar fraudulentamente datos auténticos.

**Hashing de próxima generación**

Para evitar el riesgo de colisiones de hash, se necesitaban funciones que generaran valores más largos. Las deficiencias de MD5 dieron paso a un nuevo grupo de funciones conocidas como algoritmos de hash seguro o SHA.

El Instituto Nacional de Estándares y Tecnología (NIST) aprueba cada uno de estos algoritmos. Los números junto a cada función SHA indican el tamaño de su valor hash en bits. Excepto SHA-1, que produce un compendio de 160 bits, se considera que estos algoritmos son resistentes a las colisiones. Sin embargo, eso no los hace invulnerables a otros exploits.

**Cinco funciones componen la familia de algoritmos SHA:**

* SHA-1
* SHA-224
* SHA-256
* SHA-384
* SHA-512

**Almacenamiento seguro de contraseñas**

Las contraseñas suelen almacenarse en una base de datos donde se asocian a un nombre de usuario. El servidor recibe una solicitud de autenticación que contiene las credenciales proporcionadas por el usuario. Luego, busca el nombre de usuario en la base de datos y lo compara con la contraseña brindada para verificar que coincida antes de otorgarle el acceso.

Se trata de un sistema seguro a menos que un atacante acceda a la base de datos de usuarios. Si las contraseñas se almacenan en texto sin cifrar, un atacante puede robar esa información y utilizarla para acceder a los recursos de la empresa. El hashing agrega una capa adicional de seguridad. Dado que los valores hash no pueden revertirse, un atacante no podría robar las credenciales de inicio de sesión de alguien incluso si lograra acceder a la base de datos.

**Tablas Arcoíris (Tabla Rainbow)**

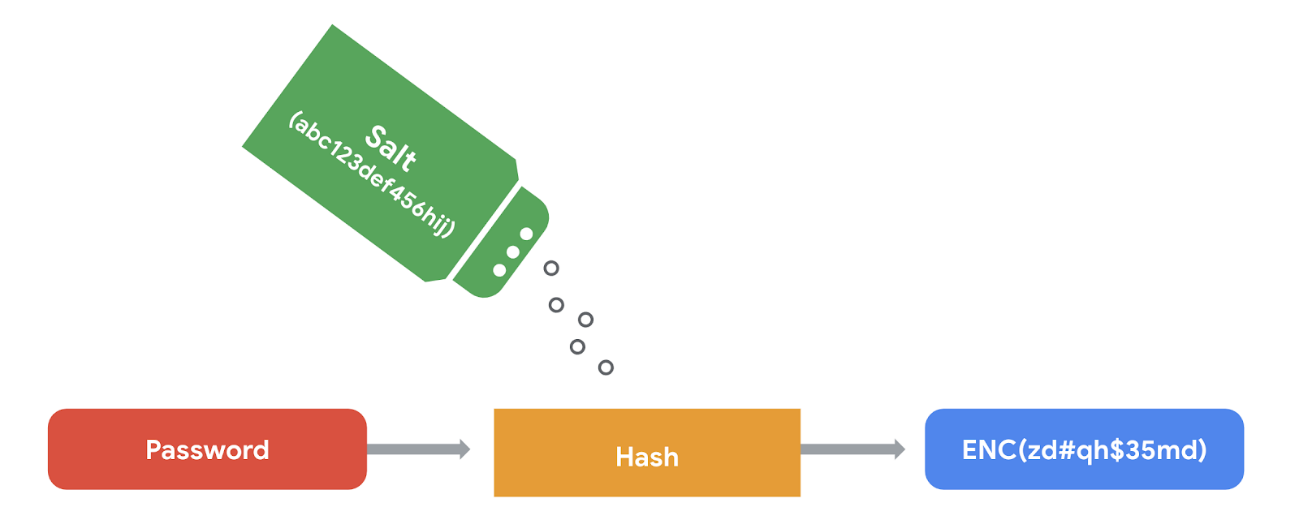
Una **tabla Arcoíris (Rainbow**) es un archivo de valores hash pregenerados y su texto sin cifrar asociado. Son como diccionarios de contraseñas poco seguras. Los atacantes capaces de obtener la base de datos de contraseñas de una organización pueden usar una tabla Arcoíris para compararlas con todos los valores posibles.

**Añadir un poco de “sal”**

Las funciones con compendios más grandes son menos vulnerables a los ataques de colisión y de tablas Arcoíris. Sin embargo, como ya sabes, ningún control de seguridad es perfecto.

El **salting** (salado) es una protección adicional que se utiliza para reforzar las funciones hash. Una “sal” es una cadena aleatoria de caracteres que se agrega a una entrada durante el proceso de hash. Por lo general, las “sales” se agregan al principio o al final de los datos mientras pasan por la función. Un uso cada vez más común del salting es el almacenamiento de contraseñas. Esta medida de seguridad adicional ayuda a proteger este tipo de información sin sobrecargar al usuario.

A continuación se muestra un ejemplo del proceso de salting:



**Conclusiones clave**

Los profesionales de la seguridad suelen usar el hashing como herramienta para validar archivos de programas, documentos y otros tipos de datos. Otra forma en que se utiliza es para reducir las posibilidades de que se produzca una filtración de datos. Como aprendiste, no todas las funciones hash brindan el mismo nivel de protección. Los ataques de tablas Arcoíris son más propensos a tener éxito contra algoritmos que generan claves más cortas, como MD5. Muchas pequeñas y medianas empresas todavía confían en MD5 para proteger sus datos confidenciales. Ahora conoces las alternativas disponibles, incluida la opción de agregar “sal” a las entradas. Con esta información, estás en mejores condiciones para hacer recomendaciones de seguridad con impacto.