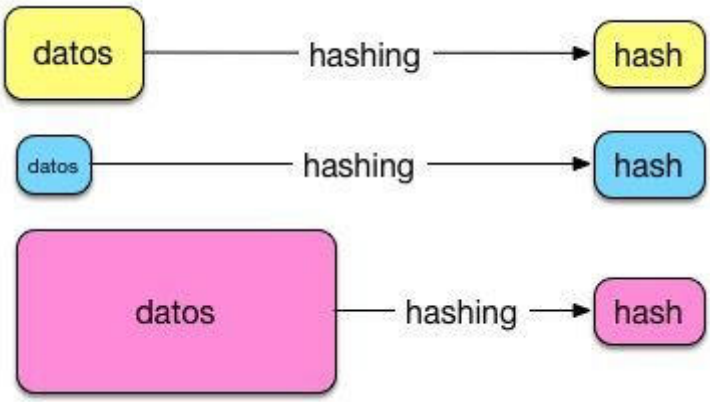
USO DE HASHING EN CRIPTOGRAFÍA

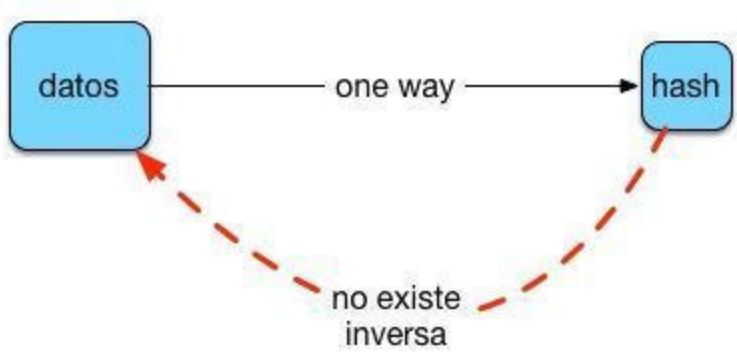
¿Qué es un algoritmo de hashing y para que sirven?. Simplificando mucho , un algoritmo de hash es un algoritmo que ante un texto de entrada de tamaño cualesquiera genera otra cadena resultado **con una longitud fija a la que se denomina hash.**



Esta es la definición muy sencilla. Ahora bien los algoritmos de hashing suelen cumplir con algunas otras características.

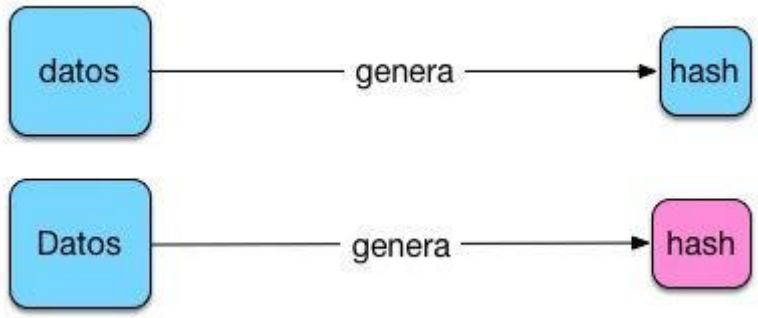
1. Hashing Algorithm One Way

Una de las características más habituales es que son **"one-way"** , es decir **solo tienen una dirección**, a partir del texto inicial se usa una función para generar el hash , **pero no hay forma de a partir del hash generar el texto inicial**. No existe función inversa.



2. Hashing Algorithm y la dispersion

Otra de las características que suelen cumplir los algoritmos de hash es que ante una pequeña variación del texto inicial e**l hash que se genera es completamente diferente**. Es lo que se conoce **como dispersión.**



3. Hashing y Cryptografía

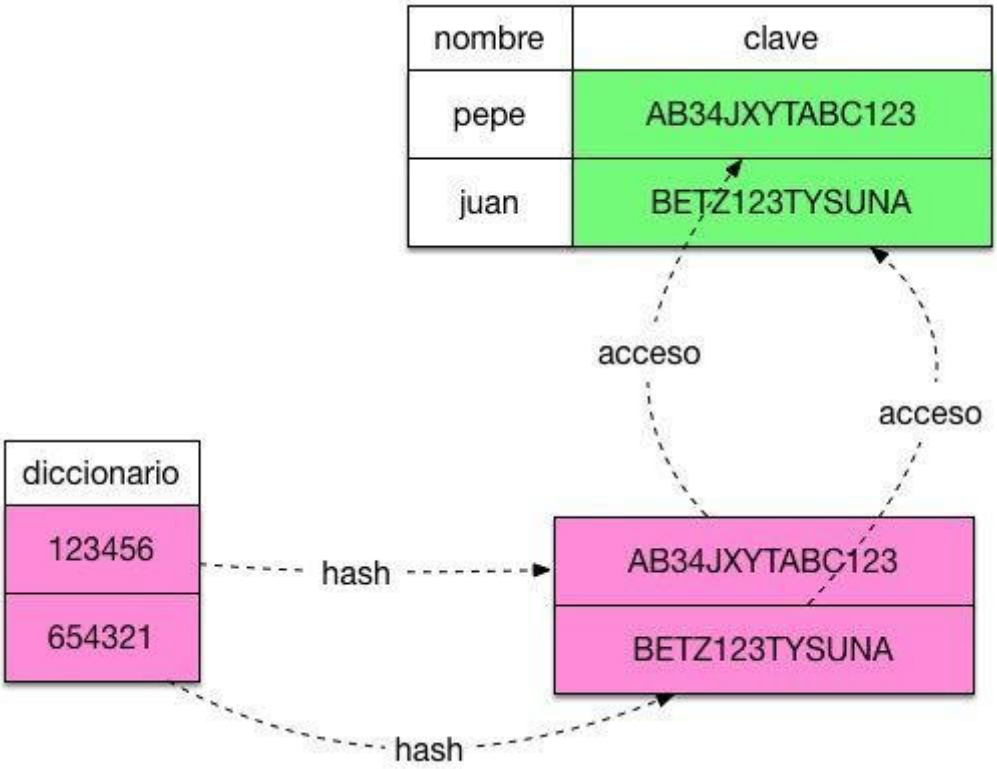
Todas estas características les hacen ideales **para la gestión del almacenamiento de contraseñas.** Si nosotros tenemos un sistema en el que los usuarios se logean , **deberemos de almacenar sus claves de alguna forma** . El problema es evidente , **si alguien hackea el sistema podrá acceder a las claves de todos nuestros usuarios ya que simplemente están almacenadas de forma transparente en la base de datos.**

nombre	clave
pepe	123456
juan	654321

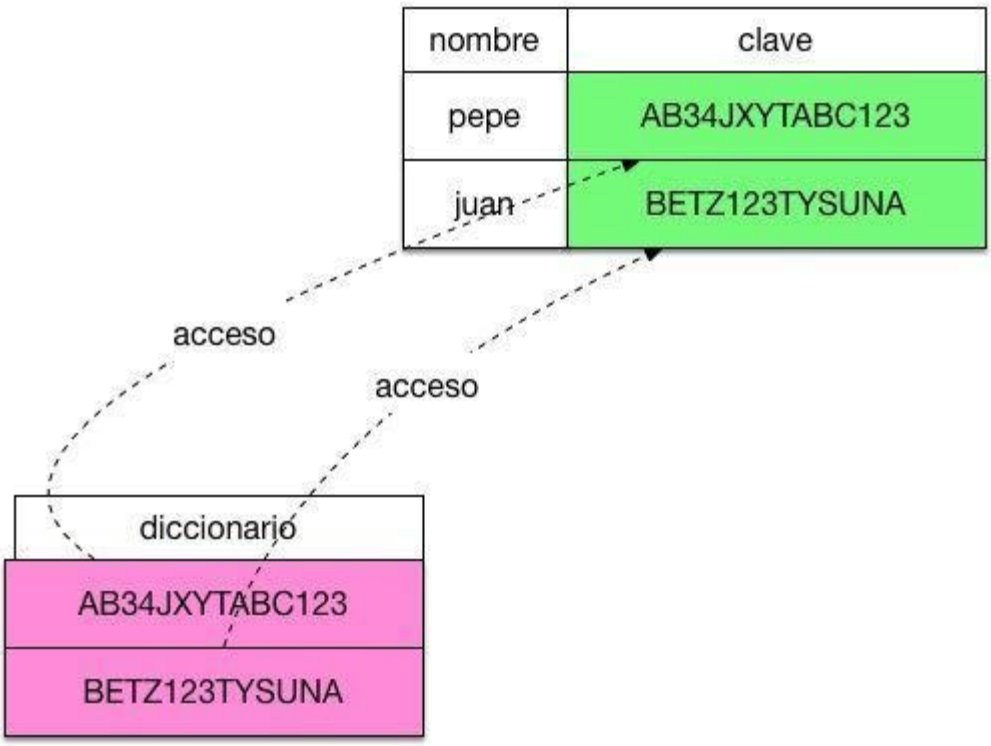
Para solventar este problema nos apoyamos **en los algoritmos de hash** . En vez de almacenar la contraseña **almacenaremos el hash** . Recordemos que ante un mismo texto origen siempre se genera el mismo hash de resultado**.**

nombre	clave
pepe	AB34JXYTABC123
juan	BETZ123TYSUNA

Esto suele **dejar muy tranquila a la gente ya que las contraseñas no se almacenan** y **no existe una función inversa** que nos permite obtener la contraseña. **¿Estamos a salvo? .. parece que sí.** La realidad es mucho más dura. Se pueden utilizar diccionarios de palabras para generar hash.



De tal forma que si alguien ha accedido a la lista de hash ,puede usar el diccionario con las claves más comunes para generar **el hash de la clave 123456** .Si este coincide con el hash almacenado para algún usuario el hacker habrá obtenido acceso a la clave del usuario. Es cierto que realizar un ataque con un diccionario lleva tiempo ya que hay que ir generando los hash. Sin embargo existen otras opciones como tener una base d**e datos de claves con sus hash ya generados.** Con lo cual la búsqueda es mucho más inmediata.



Así pues no estamos tan protegidos como pensamos. **¿Cómo podemos paliar esta situación ?**

4. Hashing Algorithm y Salt

Para solventar este problema debemos modificar la forma en la que convertimos la clave en un hash. Para ello añadiremos en el proceso **lo que se denomina un SALT**. Un SALT es información adicional que se agrega a la clave **antes de generar el hash**. De esta forma aunque nuestro sistema haya sido atacado y **el hacker acceda a todos los hash que tenemos almacenado**s , no podrá acceder a las claves sino conoce la función de SALT que hemos utilizado.