INTRODUCCIÓN AL HASH COMO TÉCNICA DE SEUDONIMIZACIÓN DE DATOS PERSONALES

v. octubre de 2019

RESUMEN EJECUTIVO

El presente estudio se dirige a que aquellos responsables que quieran utilizar técnicas de hash en sus tratamientos como garantía de seudonimización de datos personales. A lo largo del texto, se introducen los fundamentos de las técnicas de hash y sus propiedades. En la aplicación de estas técnicas, en algunos casos, existe un riesgo elevado de identificar el mensaje que generó el hash. En el documento se analizan las fuentes de riesgo de reidentificación en la aplicación de técnicas de hash y se establece la necesidad de realizar un análisis objetivo de estos riesgos para determinar la adecuación de este como técnica de seudonimización o incluso anonimización. Este análisis implica tanto al proceso empleado, como los restantes elementos que conforman el sistema de hash, con particular atención a la entropía de los mensajes y a la información vinculada o vinculable al propio valor representado por el hash.

ÍNDICE

I. INTRODUCCIÓN Y OBJETIVO DEL ESTUDIO

El valor de los datos es actualmente indiscutible. Los datos se han convertido en factor clave para la investigación científica, la administración pública y una creciente economía digital. El desarrollo tecnologías tan prometedoras como Big Data o Machine Learning depende en gran medida de la cantidad de datos que se les pueda suministrar.

Esta demanda creciente de datos personales ha supuesto un renovado interés en los procesos y técnicas de anonimización. Las funciones hash llevan mucho tiempo utilizándose para proporcionar una protección adicional en el tratamiento de datos personales. Sin embargo, existen dudas de hasta qué punto el hash es una técnica efectiva de seudonimización y si, bajo ciertas circunstancias como que el mensaje original ha sido eliminado, se puede llegar a considerarse incluso que el valor del hash está anonimizado[1].

Esta decisión es de capital importancia para determinar, entre otras cosas, el cumplimiento efectivo de los derechos establecidos en el RGPD en determinado tipo de tratamientos, como pueden ser en investigación, análisis de datos de tráfico o geolocalización, blockchain y otros. A la hora de tomar dicha decisión intervienen consideraciones jurídicas, técnicas y de gestión de procesos, por lo que aquellos involucrados en la misma necesitan tener un conocimiento básico de las técnicas de hash y sus posibles riesgos.

Este estudio está dirigido a los responsables de tratamientos que quieran utilizar implementaciones basadas en el uso de funciones hash para seudonimizar o anonimizar datos personales. En éste se presentan de forma breve los fundamentos de las funciones hash, sus propiedades, las posibilidades de reidentificar el mensaje que generó el hash y se establecen ciertas guías para analizar la adecuación de un tratamiento que utilice funciones hash.

II. FUNCIÓN RESUMEN O HASH

Una función resumen o función hash es un proceso que transforma cualquier conjunto arbitrario de datos en una nueva serie de caracteres con una longitud fija, independientemente del tamaño de los datos de entrada.

El resultado obtenido se denomina hash, resumen, digest o imagen. Muchas veces, el término “hash” se utiliza tanto para referirse a la función hash como al valor resultado de ejecutar dicha función sobre un mensaje en particular. A los datos que van a ser procesados por la función hash se le denomina mensaje o preimagen. El conjunto de todos los posibles mensajes o preimágenes es el dominio o espacio de mensajes.

Por ejemplo, si se utiliza la función hash SHA256[2] para determinar el valor hash de “Hola” se obtendrá como resultado el siguiente resumen:

[1] Blockchain and the General Data Protection Regulation.
http://www.europarl.europa.eu/thinktank/es/document.html?reference=EPRS_STU%282019%29634445

[2] Un lista de simuladores de funciones hash se puede encontrar en https://hash.online-convert.com/

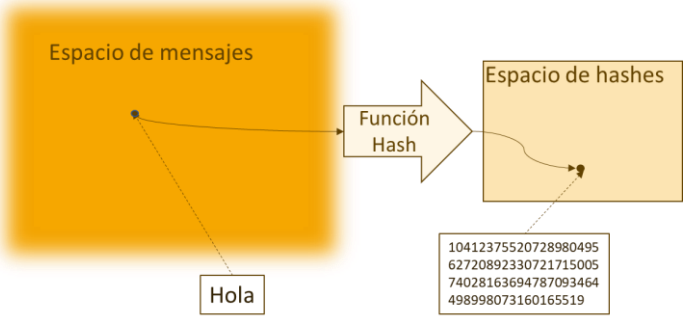
SHA256(Hola) =
10412375520728980495627208923307217150057402816369478709346449899807
3160165519[3]

En este caso, “Hola” se traduce en un conjunto de bits, a partir de los cuales y tras una serie de operaciones[4], se obtiene una cadena de 256 bits (representados aquí con su valor en decimal).

Si, por el contrario, introduzco en la función hash un mensaje más complejo, por ejemplo, el texto completo de El Quijote en formato pdf (471 páginas), el resultado de la función hash será distinto, pero el resultado será un resumen con el mismo tamaño en bits:

SHA256 (<<Texto de El Quijote en pdf>>) =
60306958082729627487200375730074782666835392077101919947310301380935
652572169[5]

En el primer caso, el mensaje “Hola” ocupaba 32 bits[6], mientras que, en el segundo caso, dicha edición del Quijote estaba en un documento que tenía un tamaño superior a 8 millones de bits. Por lo tanto, podríamos representar la función hash de la siguiente forma:



A la izquierda se representa el espacio de mensajes, o todos los conjuntos posibles de datos que se pueden crear y de los que se puede generar un hash. En la figura, este conjunto se representa con unos límites borrosos pues ese espacio puede ser infinito, ya que siempre se puede crear un mensaje de un tamaño mayor.

A la derecha de la imagen se representa el espacio de hashes como un caja de límites definidos y de un tamaño menor. El tamaño del espacio de hashes dependerá del número de bits utilizados como resultado del hash. Por ejemplo, en caso de SHA-256, al ser el resultado del hash de tamaño 256 bits, pero dependiendo del algoritmo podría ser de cualquier tamaño, siendo los más comunes de 32 a 512 bits. Para tener una idea de

[3] Normalmente se utiliza notación hexadecimal al haber una correspondencia directa entre dichos dígitos y los bits subyacentes, correspondencia que se pierde en la notación decimal. El correspondiente hexadecimal de la información anterior es:
E633F4FC79BADEA1DC5DB970CF397C8248BAC47CC3ACF9915BA60B5D76B0E88F
Hexadecimal: forma de representar los números que, en vez de utilizar la notación del 0 al 9, la extiende al 10 (A) 11 (B) 12 (C) 13 (D) 14 (E) y 15 (F). De esta forma, los números se representan del 0 al F. Su ventaja reside en que con un solo dígito se representan 4 bits. Con 4 bits hay 16 combinaciones distintas

[4] Una clarificadora descripción del proceso se puede encontrar en https://medium.com/biffures/part-5-hashing-with-sha-256-4c2afc191c40

[5] 60306958082729627487200375730074782666835392077101919947310301380935652572169

[6] Codificado en ASCII

cuántos valores de hash distintos se pueden obtener con un tamaño de hash de 256 bits el número total será superior a multiplicar un millón por sí mismo trece veces[7].

PROPIEDADES DESEABLES EN UNA FUNCIÓN HASH

Las propiedades ideales de una función hash son:

- Permite ejecutarse sobre contenido digital de cualquier tamaño y formato. Al final, todo contenido digital son números para el ordenador: textos, fotografías, videos, etc.
- Dada una entrada cualquiera, produce una salida numérica de tamaño fijo.
- El resultado es determinista, es decir, para el mismo mensaje o conjunto de datos de entrada siempre se obtiene el mismo resultado.
- Reconstruir el mensaje original a partir del resultado de la función hash debe ser extremadamente costoso, sino imposible.
- Una mínima variación en el mensaje original (un bit) ha de producir un hash totalmente distinto (difusión).
- Si se selecciona un mensaje de entrada, encontrar otro mensaje que tenga el mismo resumen ha de resultar extremadamente costoso (colisión débil).
- También ha de ser extremadamente costoso encontrar dos mensajes cualesquiera que tengan el mismo resumen (colisión fuerte).
- El algoritmo de hash deberá cubrir de forma uniforme todo el espacio de hash, los que significa que cualquier resultado de la función hash tiene, a priori, la misma probabilidad de ocurrencia que cualquier otro. Por lo tanto, todos los valores del espacio de hash pueden ser resultado de la función hash.

DESCRIPCIÓN DE UNA FUNCIÓN HASH

En general, una función hash funciona de la siguiente forma:

- El mensaje de entrada se divide en bloques.
- Un formula calcula el hash, un valor con un tamaño fijo, para el primer bloque.
- Se calcula el hash del siguiente bloque y suma al resultado anterior.
- Se realiza el mismo proceso sucesivamente hasta que se recorren todos los bloques.

Un ejemplo muy sencillo de función hash lo podemos esbozar a continuación. Diseñemos una función hash que a partir de un texto genera un hash de tamaño tres dígitos decimales (del 000 al 999). A su vez, como mensaje para calcular el hash sea el texto de El Quijote:

> *En un lugar de la Mancha de cuyo nombre no quiero acordarme no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero … etc.*

En primer lugar, se dividiría el texto en bloques de, para este ejemplo, veinte caracteres. De esta forma, en la siguiente tabla cada fila representaría un bloque:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
E	n		u	n		l	u	g	a	r		d	e		l	a		M	a
n	c	h	a		d	e		c	u	y	o		n	o	m	b	r	e	
n	o		q	u	i	e	r	o		a	c	o	r	d	a	r	m	e	

[7] Para calcular el valor aproximado de un conjunto de bits se puede tener en cuenta que con 10 bits se pueden codificar 1024 estados, por lo tanto con 20 algo más de un millón (1024x1024). En el caso de 256, dividimos dicho valor por 20, obteniendo como resultado 13.

n	o		h	a		m	u	c	h	o		t	i	e	m	p	o		q
u	e		v	i	v	í	a		u	n		h	i	d	a	l	g	o	
d	e		l	o	s		d	e		l	a	n	z	a		e	n		a
s	t	i	l	l	e	r	…												

A continuación, a cada carácter se le asignaría un valor numérico, por ejemplo: A – 1; B-2; C-3; …. Y finalmente cero para el espacio. Para el primer bloque del ejemplo se obtendría la siguiente codificación:

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
5	14	0	22	14	0	12	22	7	1	19	0	4	5	0	12	1	0	13	1

El valor de hash se podría calcular de muchas formas, por ejemplo, se puede multiplicar el valor asociado a un carácter con su posición en el bloque[8] y, a continuación, sumar todos los resultados. Para este bloque en concreto, dicho proceso daría como resultado el valor 1331. Como hemos comentado que esta función hash tendrá como resultado sólo tres dígitos decimales, podemos truncar el resultado directamente eliminando los dígitos por encima del tercero, por lo tanto, el valor obtenido del hash para el primer bloque sería 331.

A continuación, se procedería con la segunda fila o bloque:

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
14	3	8	1	0	4	5	0	3	22	26	16	0	14	16	13	2	19	5	0

Operando la función hash sobre el mismo bloque se obtendría el valor 1947, trucando por encima del tercer dígito el resultado final sería 947.

A continuación, se encadenarían los dos bloques mediante una operación de suma del resultado de los dos bloques 331+947 = 1278, truncando de nuevo el valor de hash de los dos primeros bloques tendría como resultado 278. El proceso se repetiría con todas las filas o bloques hasta obtener el resultado final.

El sistema aquí presentado no tiene unas buenas propiedades como función hash: el valor del hash tiene una longitud demasiado pequeña (solo existen 1000 valores posibles), hay medios puede alterar el texto y preservar el hash[9], no es eficiente su implementación, etc. Por ello, se han desarrollado otros tipos de funciones hash más adecuadas como la familia SHA[10], MD5[11] u otros[12].

III. EL HASH COMO IDENTIFICADOR ÚNICO

El número de posibles resultados de una función hash es muy alto, pero no infinito. Dado que el espacio de mensajes puede ser infinito, existirán infinitos mensajes que pueden dar lugar a un mismo valor de hash. El conjunto de mensajes que dan como resultado el mismo valor de hash se denomina conjunto de preimágenes.

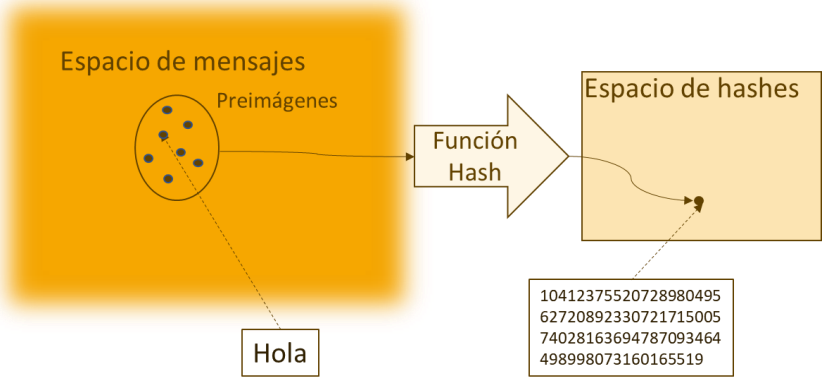
[8] Así se evitaría que se pudiera obtener el mismo hash moviendo las letras dentro del mismo texto, por ejemplo, sustituyendo "En un lugar de la Ma…" por "De un lugar en la Ma…".

[9] La letra con valor 5 en la posición 6 aporta igual que la letra con valor 6 en la posición 5. La utilización de pesos como factores primos impediría aprovechar estas equivalencias.

[10] FIPS PUB 180-4 Secure Hash Standard (SHS) https://ws680.nist.gov/publication/get_pdf.cfm?pub_id=910977

[11] RFC1321 The MD5 Message-Digest Algorithm https://www.ietf.org/rfc/rfc1321.txt

[12] https://en.wikipedia.org/wiki/List_of_hash_functions



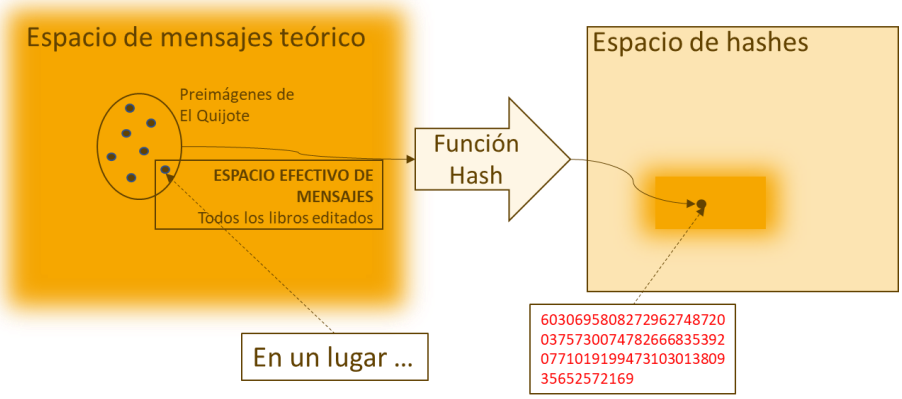
La existencia de los conjuntos de preimágenes, que podría crear dudas sobre la utilidad de la función hash como identificador único, se plantea únicamente cuando se considera un entorno teórico y no en el marco de un tratamiento concreto.

ANÁLISIS BASADO EN UN TRATAMIENTO ESPECÍFICO

Pongamos por caso que un tratamiento quiere asociar un valor de hash a cada uno de los libros que han sido publicados en el mundo, a partir de la digitalización de los contenidos de los mismos. Hace unos años Google publicaba que el número de libros publicados en el mundo era de 130 millones[13].

Esta es una cifra sensiblemente inferior al número de resultados hash posibles para una función como SHA-256. Aunque la multiplicásemos por 7.000 el número de libros, llegando al millón de millones, estaríamos muy lejos de poder saturar el espacio de valores de hash.

Si lo representásemos gráficamente, el conjunto de libros editados es un subconjunto de todos los mensajes posibles, y es lo suficientemente pequeño como para que las preimágenes de, por ejemplo, El Quijote, estén fuera del mismo.



Con una sombra sobre el espacio de hashes se representa el conjunto de valores hash que serían generados al obtener el hash de todos los libros editados, que cubriría una parte ínfima del conjunto de posibles valores de hash.

A pesar del número de libros editados, si en la figura anterior se dibujase el cuadro correspondiente a estos en la misma escala que el cuadro correspondiente al espacio de hashes, el primero sería prácticamente invisible por lo diminuto.

[13] http://booksearch.blogspot.com/2010/08/books-of-world-stand-up-and-be-counted.html

La diferencia entre el conjunto de libros editados y el resto de posibles mensajes que podrían ser una preimagen de El Quijote descansa en el concepto de "orden". Los libros no están formados por cualquier combinación de letras, sino que están formados por palabras de un idioma concreto, formando frases con una sintaxis particular, con una estructura narrativa y expresando un mensaje con sentido. Por lo tanto, si un mensaje sin sentido tuviese el mismo hash que el que corresponde a un libro concreto, lo descartaríamos como perteneciente a nuestro espacio efectivo de mensajes, es decir, como perteneciente a nuestro tratamiento.

Cuanto más estricto sea ese "orden", por ejemplo en el caso que solo se permitan libros en castellano y no en cualquier idioma, más pequeño será el conjunto de libros (espacio de mensajes de mi tratamiento) y se tendrá menos probabilidad de colisión.

Aun así, nada garantiza que dos valores de hash correspondientes a dos libros distintos no coincidan, aunque la probabilidad es realmente ínfima en un algoritmo bien construido y se puede determinar esa posibilidad analíticamente mediante una generalización de la paradoja del cumpleaños[14].Por ello, si representásemos dentro del espacio de hashes un cuadro que contenga todos los hashes correspondientes al espacio real de mensajes, este tendría un límite difuso, es decir, si hay un millón de millones de libros editados y se generan sus hashes, es altamente probable que se generen un millón de millones de hashes distintos, pero no está garantizado aunque, a efectos prácticos, la correspondencia entre libros y hashes es de 1 a 1, biunívoca.

IV. PROBLEMA DE LA REIDENTIFICACIÓN

Las funciones hash aspiran a ser irreversibles (ver sus propiedades deseables) y por ello el resultado de aplicar una función hash a un identificador directo debería de evitar la reidentificación del mismo. A pesar de ello, el mismo "orden" que está implícito en un tratamiento y que garantiza la efectividad del hash como identificador único también aumenta la probabilidad de identificar el mensaje original a partir del hash.

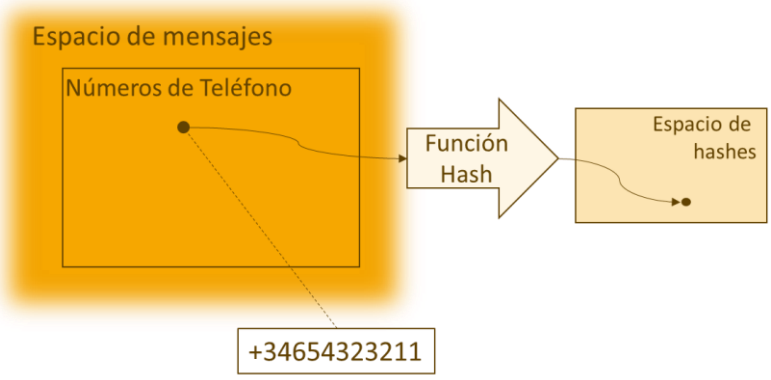
TRATAMIENTO DEL HASH DE NÚMEROS DE TELÉFONO

Como ejemplo, supongamos que en un tratamiento se evalúa el hash de un número de teléfono móvil de una compañía de telecomunicaciones. El diseñador del tratamiento utiliza una función hash que genera un valor de hash de tamaño 64 bits[15].

Un número de teléfono móvil está formado por 9 dígitos, más dos del código regional y precedido del símbolo "+", en total 12 símbolos. Si cada símbolo se almacena en un byte, el número total de bits es igual a: 12*8 = 96 bits.

[14] https://en.wikipedia.org/wiki/Birthday_problem#Generalizations

[15] Por ejemplo, Cityhash https://github.com/google/cityhash



En la figura anterior se representa cómo el espacio de números de teléfono es mayor que el espacio de hashes. Si se calculasen todos los posibles hashes para todas las combinaciones de 96 bits inevitablemente todo el espacio de hashes quedaría cubierto y se producirían más de cuatro mil millones de colisiones. En principio, la función hash tendría que "comprimir" los 96 bits del número en los 64 bits del hash.

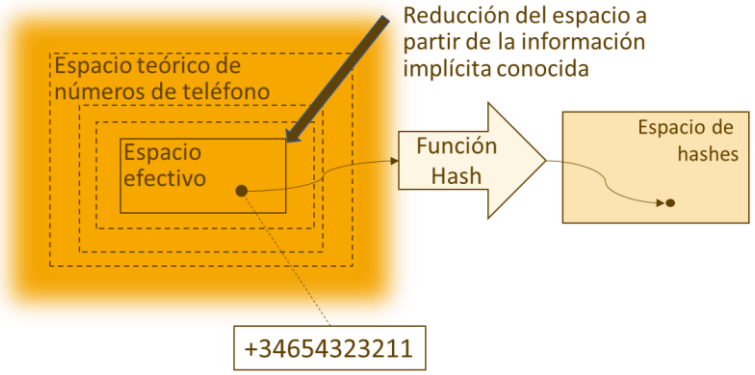
Por un lado, si los 96 bits del número contuviesen información, el convertir dicho número a un hash de 64 bits necesariamente perdería información y haría el hash irreversible. Por otro lado, cabría preguntarse si tal diseño del tratamiento cumple con el requisito de disponer de un identificador unívoco.

ANÁLISIS DEL TRATAMIENTO

Un análisis más detallado daría respuesta a estas cuestiones:

- Por un lado, once de los dígitos son numéricos, lo que supone 100.000 millones de combinaciones. Si suponemos que sólo hay 20 símbolos posibles en un teclado, la codificación de un símbolo adicional para poder registrar el "+" inicial aumentaría el número de combinaciones a 2 billones[16]. Parece un número muy elevado, pero si lo traducimos a bits, hemos reducido la cantidad de datos de 96 bits a aproximadamente 41 bits. Es decir, mucho menos que los 96 iniciales y ya por debajo de los 64 bits el tamaño del hash, por lo tanto, sería útil como identificador único.
- Como el marco del tratamiento supone que todos los números son de abonados españoles, se conoce que el prefijo +34 es común, por lo tanto, dichos datos son fijos y con el resto de los 9 dígitos quedan mil millones de combinaciones (aproximadamente 30 bits).
- Si se están tratando números españoles de teléfonos móviles, estos comenzarán por 6 o por 7. Por lo tanto, al estar fijado el primer número, existen solo 200 millones de combinaciones (aproximadamente 28 bits).
- Pero no hay 200 millones de abonados en España. El número de líneas móviles actuales realmente no llega a 60 millones (26 bits).
- El operador con más líneas móviles no llega a 20 millones (20 bits de información), por lo tanto, si se conoce el operador móvil al que corresponde el hash del número, y se tiene acceso a lista de los números, el número de combinaciones decrece enormemente, y la información real que contienen los originales 96 de datos es de sólo 20 bits de información.

[16] Billón en el sentido de un millón de millones



Como muestra el ejemplo, en la definición del tratamiento hay implícito un orden que limita el conjunto de mensajes posibles en el marco de ese tratamiento (su espacio de mensajes). El número de mensajes válidos posibles (2^{20}) es muy inferior al de hashes posibles (2^{64}), de donde la posibilidad de colisión es muy baja[17] y el hash se comportaría como un identificador único de forma práctica.

ANÁLISIS DE LA REIDENTIFICACIÓN

En cuanto a la posibilidad de dado un hash determinar el número a que corresponde, teniendo en cuenta que un ordenador de sobremesa puede calcular más de 1 millón de hashes por segundo, se puede crear un diccionario para todos los hashes posibles de los teléfonos de un operador dado en menos de 20 segundos, prácticamente en tiempo real[18]. Es decir, se puede recuperar la información referenciada por el hash.

En este caso, la cantidad de información era pequeña, pero incluso para espacios de mensajes mucho mayores, con más información, existen técnicas conocidas como Rainbow Tables[19] que permiten la reversibilidad del hash.

ORDEN, DESORDEN E INFORMACIÓN

Cuando los datos que se utilizan en un tratamiento tienen un orden implícito, el conjunto de mensajes posibles (espacio de mensajes) se reduce enormemente, lo que facilita la reversibilidad de los mensajes (reidentificación).

Cuanto mayor es el orden implícito en un conjunto de datos, el mensaje está más determinado y menos información real contienen. El conocimiento deriva de la propia estructura de los datos (p. e. los números de teléfono móvil españoles empiezan por +346 o +347) o conocimiento sobre el entorno del tratamiento (p. e. los números de teléfono que gestiona un operador son conocidos). Por todo ello, es necesario distinguir entre los datos de un mensaje (96 bits en el ejemplo anterior) y la información que contiene dichos datos (20 bits en el ejemplo anterior).

---

[17] Muy baja pero no nula. Si lo analizamos como un sistema sin memoria, la probabilidad de que dos o más hashes coincidan con 20 millones de ocurrencias en este caso se calcularía como: $P=1-[2^{64}! / ((2^{64*20.000.000})*(2^{64-20.000.000})!)]$. Por la complejidad de cálculo de grandes números, se podría garantizar que la probabilidad es muy inferior al 0,002 %

[18] https://automationrhapsody.com/md5-sha-1-sha-256-sha-512-speed-performance/

[19] Ataque de Tablas Rainbow: Método de ataque de datos que utiliza una tabla computarizada previa de cadenas de hash (resumen de mensaje de longitud fija) para identificar la fuente de datos original. CCN-STIC 401

El grado de orden, más bien de desorden, de un conjunto de datos se conoce como entropía[20]. A mayor entropía, mayor información contendrá un conjunto de datos. Por el contrario, menos desorden (menos entropía) implica la existencia de menos alternativas y por tanto los mismos datos contendrán menos información.

Cuanto menor sea el espacio de mensajes, y menor la entropía, existirá un menor riesgo de colisión en el tratamiento del hash, pero la reidentificación será más probable. Por el contrario, a mayor entropía, la posibilidad de una colisión será mayor, pero el riesgo de reidentificación será menor.

La medida de la cantidad de información, que es muy distinta del número de bits que se está empleando para registrar un mensaje, es uno de los análisis más importantes que hay que realizar siempre que se pretende proteger un mensaje, ya sea mediante hashes o utilizando otras técnicas como cifrado, y requiere de un análisis riguroso[21].

V. VINCULACIÓN DE INFORMACIÓN AL HASH

En el caso del capítulo anterior, cuanta más información se disponía del potencial espacio de mensajes (su estructura, la localización geográfica de los usuarios, la compañía a la que pertenecía), mayor era el "orden" implícito en el mensaje y menor era la información que contenía el propio hash, lo que hacía su reidentificación más probable.

IDENTIFICADORES VINCULADOS AL HASH

Un fichero con datos personales puede contener "identificadores" que por sí solos están asociados de forma unívoca a un sujeto (v.g. el DNI, el nombre completo o el pasaporte).

Si existen identificadores vinculados a un hash, por ejemplo, se almacena el DNI y el hash de un número de teléfono asociado a ese DNI, es evidente que la información pertenece a un sujeto determinado. En cuanto a la confidencialidad de la información representada en el hash, el hecho de tener un identificador vinculado añadirá una vulnerabilidad adicional a la debilidad del hash en cuestión ya que a partir de la información del DNI se podría obtener información que reduzca el espacio efectivo de mensajes para hash en concreto.

Es decir, cuanta más información personal se vincule con el hash existirá un mayor riesgo de identificar el contenido del hash.

SEUDOIDENTIFICADORES VINCULADOS AL HASH

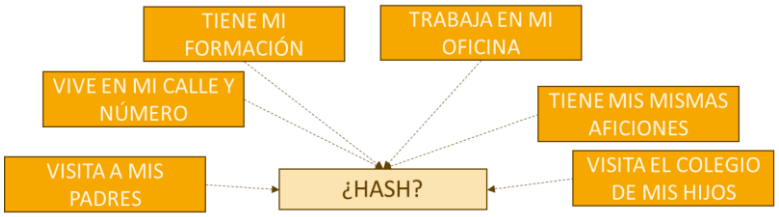
Los ficheros con datos personales también pueden contener otros datos que, convenientemente agrupados y cruzados con otras fuentes de información, pueden llegar a identificar a un individuo. Estos datos se denominan "seudoidentificadores",

[20] La entropía es un principio de la termodinámica que establece que el desorden de un sistema crece con el tiempo. Para ver un ejemplo de su relación con la información veamos el caso de una pared perfectamente recién pintada de blanco, que se podría describir con muy pocas palabras "tres metros de alto, cinco de largo, blanco puro". Según transcurre el tiempo, zonas de la pared se vuelven más grises, aparecen grietas, manchas, etc: aumenta su entropía. Esa pared no podríamos describirla con pocas palabras, sino que necesitaríamos muchas para detallar todos sus accidentes: hay más información.

[21] Se puede realizar un análisis de la información vinculada a un tratamiento mediante del análisis de la entropía, identificando el conjunto de todos los estados posibles x(i) y su probabilidad asociada P(xi) y calculando: Información = $- \Sigma_{(1..n)} P(x_i) \log_2 P(x_i)$

"cuasi-identificadores" o identificadores indirectos[22]. La relación entre estos y el valor de hash se puede establecer de dos formas.

La primera es que el hash se pueda vincular con seudoidentificadores como un efecto secundario que no es el objeto del tratamiento. El segundo caso se plantea cuando el propósito en el tratamiento es vincular seudoidentificadores entre sí mediante un valor de hash.



De una forma u otra, la información que estos proporcionan disminuye la eficacia de las funciones hash al proporcionar indicios sobre la información contenida en el valor de hash y que permitirán identificar la sujeto. El riesgo de reidentificación dependerá no sólo del tipo de seudoidentificadores sino también de la correcta aplicación de técnicas aleatorización o generalización[23] sobre los identificadores indirectos. Para determinar hasta qué punto ese conjunto de información es anónimo, sería necesario hacer un análisis de, por ejemplo, k-anonimidad[24].

VINCULACIÓN DE OTRO TIPO DE INFORMACIÓN

Pueden existir condicionantes adicionales que surgen cuando el diseño teórico de un tratamiento se concreta en una implementación sobre un sistema TIC y con unos procedimientos de trabajo. La operativa real puede generar circunstancias que permiten vincular información adicional al hash, y que no estaba prevista en el concepto original del tratamiento, por ejemplo:

- La posición relativa del hash en una tabla o cadena de datos, permite establecer relaciones entre la información previa o posteriormente almacenada.
- La fecha de registro del hash en una tabla permite vincularlo con información almacenada en otras tablas o cadenas que se haya realizado en la misma fecha.
- De igual forma, ficheros de log asociados con la trasmisión, uso de servicios de cómputo, accesos a sistemas de almacenamiento, etc., son fuentes de información que permiten cruzar datos.

Todas estas posibilidades se han de tener en cuenta a la hora de determinar la dificultad de reidentificación del hash.

VI. ESTRATEGIAS PARA DIFICULTAR LA REIDENTIFICACIÓN

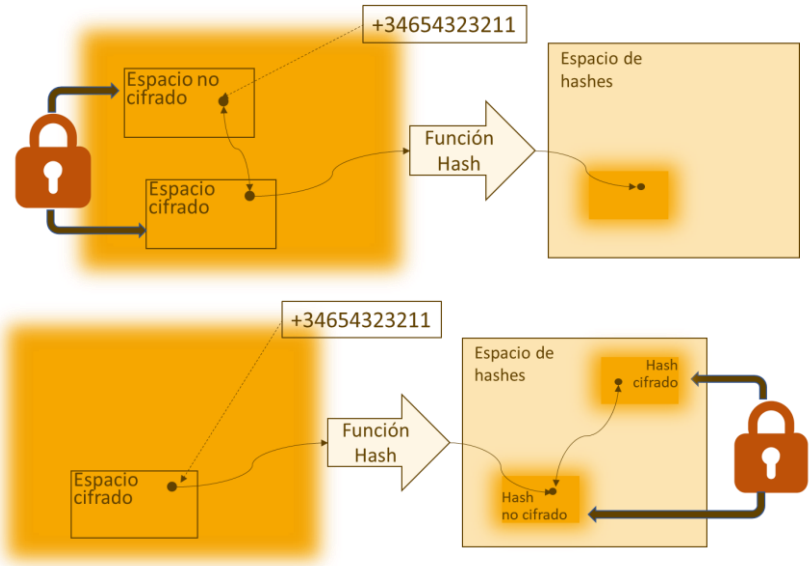
REALIZAR UN CIFRADO CON REUTILIZACIÓN DE CLAVE

Una estrategia para dificultar la reidentificación del valor del hash es utilizar un algoritmo de cifrado[25] con una clave que almacena de forma confidencial el responsable

[22] AEPD: La K-anonimidad como medida de la privacidad.
[23] WP29: Dictamen 05/2014 sobre técnicas de anonimización
[24] AEPD: La K-anonimidad como medida de la privacidad.
[25] Sería indiferente si es un sistema simétrico o asimétrico de cifrado.

o intervinientes en el tratamiento, que cifre bien el mensaje antes de realizar el hash, o bien el hash una vez calculado. El proceso de cifrado obtiene un mensaje nuevo (texto cifrado) a partir del original (texto claro) existiendo un proceso eficiente para obtener uno de otro mediante el uso de claves.

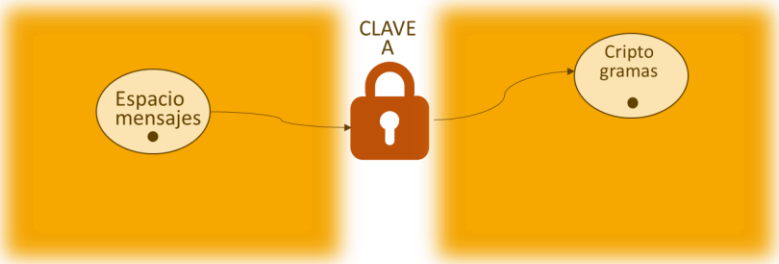
Ambas estrategias se representan en la figura a continuación:



Como se muestra en la figura anterior, el cifrado realiza una correspondencia entre distintos espacios de mensajes, en el primer caso, o cuando se cifra el hash, entre dos espacios de hashes.

El proceso de cifrado, al contrario que el proceso de calcular el hash, es un proceso inherentemente reversible, que por su propia naturaleza conserva la información cifrada. Es decir, la información está ahí, ni aumenta ni disminuye con el proceso de cifrado y los espacios cifrados y no cifrados están vinculados por la clave.

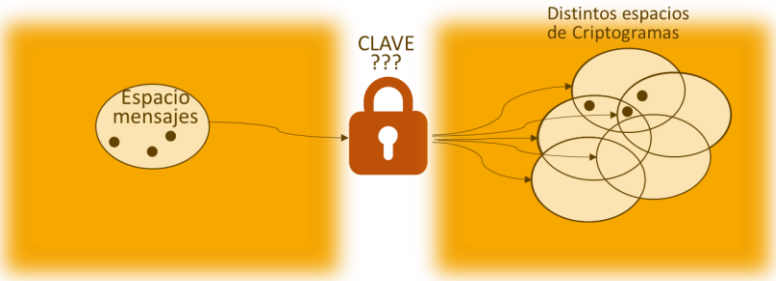
En ese caso, podríamos representar la correspondencia entre espacios de mensajes y espacios cifrados (criptograma) para una clave dada de la siguiente forma:



En la figura el punto en los dos espacios representa al mismo mensaje sin cifrar (izquierda) y cifrado (derecha).

Si la clave no es conocida, un observador tan solo tendrá la referencia de que existe un mensaje cifrado, no conocería la clave y este mensaje podría corresponder a casi tantas claves como pudieran idearse:

Pero si se dispone de múltiples textos cifrados de múltiples mensajes, llegará un momento que todos ellos sólo podrán corresponder a un único espacio de criptogramas que se puede generar con una única clave:



Este principio se conoce con el nombre de Distancia de Unicidad[26] y establece que, por encima de un umbral de texto cifrado, tanto el texto claro como la clave están determinados. Esto supone que, aunque la clave pudiera ser borrada por el responsable del tratamiento, la clave se encuentra implícita en el texto cifrado, por lo que ni la clave ni la información que esta protege desaparecen y, por tanto, son susceptibles de ser recuperadas[27].

Por lo tanto, la reidentificación estará protegida con un determinado nivel de garantía que dependerá de las debilidades inherentes a cualquier sistema de cifrado:

- El compromiso de la confidencialidad de la clave. El trabajar entornos distribuidos podría aumentar dicho riesgo[28].
- La fortaleza en la generación de la clave.
- La vulnerabilidad a ataques tipo texto claro conocido o texto claro escogido[29].
- El volumen de información cifrada, cuanta más información más facilidad de criptoanalizar.
- El principio de distancia de unicidad.
- El desarrollo de la potencia de computación y de nuevos algoritmos de criptoanálisis.
- La existencia de debilidades no conocidas en el sistema de cifrado.

[26] Establecido por Shannon en el informe "Communication Theory of Secrecy Systems", se enuncia de la siguiente forma: en función de la entropía del sistema de cifrados, longitud de mensaje a partir de la cual, dado un criptograma y un algoritmo de cifrado determinado, tanto la clave como el mensaje claro quedan totalmente determinados.

[27] Esta distancia puede ser muy corta, si Distancia de Unicidad = Log_2(nº claves) / Redundancia = Log_2(nº claves) / (log_2(símbolos)-entropía) en el caso de un texto castellano cifrado con AES256 la distancia de unicidad es igual a $log_2(2^{256})/(log_2(27)-2)$ aproximadamente 95 caracteres. Es decir, con un sólo texto cifrado de 100 caracteres se ha superado dicha distancia.

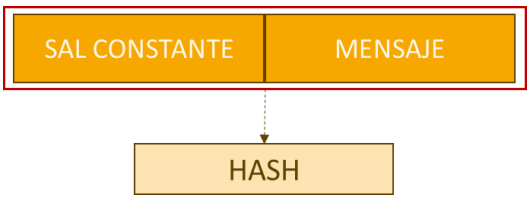
[28] Siempre se puede incrementar esa seguridad con criptosistemas asimétricos y estrategias de gestión de la relación de claves.

[29] El primero hace referencia a que existan casos en los que la información que se ha cifrado sea pública, el segundo que un tercero pueda provocar que se cifre un mensaje escogido.

ANEXIÓN DE UNA CABECERA AL MENSAJE O SAL CON REUTILIZACIÓN

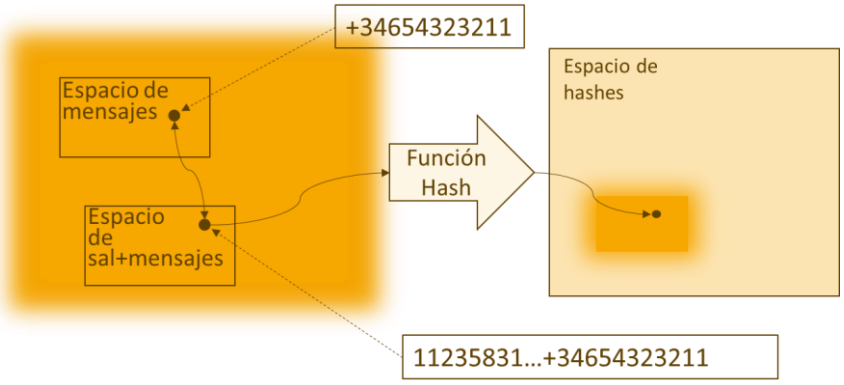
Otra estrategia para dificultar la reidentificación es añadir un valor constante o "sal" a todos los mensajes antes de evaluar el hash. Una sal es un valor aleatorio[30] que se añade al mensaje original. Si es aleatorio ha de ser independiente del mismo mensaje o de cualquier otra información.

El formato del mensaje cambia, ya que, al mensaje original, hay que añadirle el campo de sal. El nuevo mensaje ampliado tendrá la siguiente forma:



Supongamos que estamos en el caso del tratamiento de hashes de números de teléfono. Con esta nueva estrategia, el espacio de mensajes del tratamiento no estaría formado por únicamente por números de teléfono, sino por el par "sal+número de teléfono". Por ejemplo, antes de calcular el hash del número "+34654323211" se añadiría un valor aleatorio y precalculado al mensaje: "112358314…+3465323211". Una vez que se ha creado el mensaje ampliado se calcula el hash. Para comprobar el hash para cualquier mensaje, solo hay que añadir la sal que está almacenada en el sistema al mensaje original. Si, por causa de una colisión, el hash de un número de teléfono coincide con un hash almacenado, dicha coincidencia no valida el número, ya que números aislados no forman parte del espacio de mensajes ampliado.

Como en el caso de la utilización de clave, el valor de la sal ha de permanecer secreto y la cantidad de información del espacio de mensajes permanece constante, ni se aumenta ni disminuye. El proceso se puede representar gráficamente de la siguiente forma:



Si el valor de la sal es eliminado por el responsable del tratamiento la información no desaparece, sino que estará protegido su acceso con un determinado nivel de garantía, ya que se cumple una aproximación al principio de distancia de unicidad como en el caso del cifrado. Aunque el algoritmo de hash no es inherentemente reversible, la situación en

[30] No todos los generadores de números pseudoaleatorios serán adecuados para ese propósito, sino que han de tener unas características particulares. Estos se denominan CPRNG o CSPRNG o cryptographically secure pseudo-random number generators. Se puede encontrar una documentación de cómo comprobar la bondad de un CSPRNG en: http://csrc.nist.gov/groups/ST/toolkit/rng/documentation_software.html

la que se encuentra el espacio de mensajes en relación con el espacio de sal+mensajes es parecida a lo que ocurre con el modelo de cifrado. Es más, la sal también se encuentra implícita en el espacio de hash generado[31].

Además, la reidentificación estará protegida con un determinado nivel de garantía que dependerá de las debilidades del sistema, entre otras:

- El compromiso de la confidencialidad de la única sal utilizada, cuyo control es más difícil en entornos distribuidos.
- Las propiedades de la sal, que dependerán de su longitud y la aleatoriedad del sistema de generación de sales. Una sal de longitud más corta será vulnerable a ataques de fuerza bruta y por Rainbow Tables. Un sistema de generación del valor de la sal con debilidades puede inferir que esta toma un valor dentro de un conjunto limitado de opciones.
- El nivel de desarrollo de la potencia de computación y de nuevos algoritmos de ruptura de hashes, que hacen necesario la generación de sales cada vez más largas[32].
- La existencia de debilidades no conocidas en el algoritmo de hash o en el sistema de tratamiento.

MODELOS DE SAL DE UN SOLO USO

El empleo de sal de un solo uso permite el desarrollo de modelos en los que se reduce el riesgo de reidentificación del hash. En el caso de que se elimine dicha sal, el mensaje original y siempre que se cumplan ciertas garantías[33], el identificador sobre el que se aplica un modelo de sal de un solo uso podría considerarse anonimizado.

La sal de un solo uso supone generar un elemento aleatorio distinto para cada uno de los mensajes. Dicho elemento aleatorio debe ser completamente independiente de cualquier mensaje y de cualquier otra sal generada para otro mensaje.

Un ejemplo didáctico de este modelo pudiera ser el siguiente:

En primer lugar, se expande el formato del mensaje original a un mensaje-ampliado formado por tres campos:

- El mensaje en sí.
- Una sal adicional que llamaremos nonce[34] cuyo propósito es aumentar la entropía del mensaje a un valor aproximado al número de bits de salida de la función hash. Por supuesto, este valor de nonce ha de generarse de forma aleatoria, ha de ser independiente del mensaje e independiente de cualquier otro valor de nonce, es decir, también de un solo uso.
- Un valor de sal de un solo uso, independiente de los dos anteriores y de cualquier otro valor de sal. Es decir, también de un solo uso.

El formato de mensaje ampliado para el tratamiento sería el siguiente:

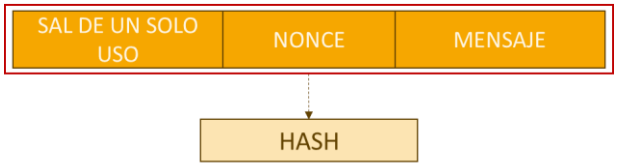
---

[31] Un ejemplo muy simplificado, que no tiene en cuenta el encadenamiento de bloques: supuesto un algoritmo de hash A que trabaja con bloques de tamaño k y utiliza una sal S, de tamaño n=k. El atacante conoce o consigue que se calcule el hash H de un mensaje M de tamaño k: A(S || M) =H. Al trabajar por bloques, sé que A(S)+A(M) = H. Por tanto, A(S) = H-A(M). Conocido el hash de S puedo desarrollar un ataque por Rainbow Table u otras estrategias para conseguir una sal equivalente.

[32] En particular, y con motivo del minado en Bitcoin que consiste en obtener valores de hash con unas determinadas características, se está realizando un gran desarrollo en la optimización de velocidad del algoritmo SHA256.

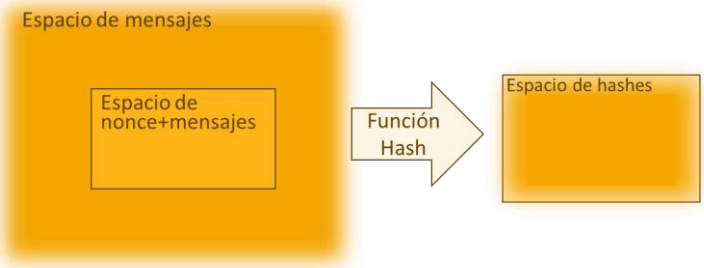
[33] Ver el apartado relativo al análisis del hash como sistema de seudonimización de este documento.

[34] Se aproxima al concepto de sal. Nonce es un número arbitrario, normalmente generado aleatoriamente, que solo puede ser utilizado en el esquema de cifrado una única vez.

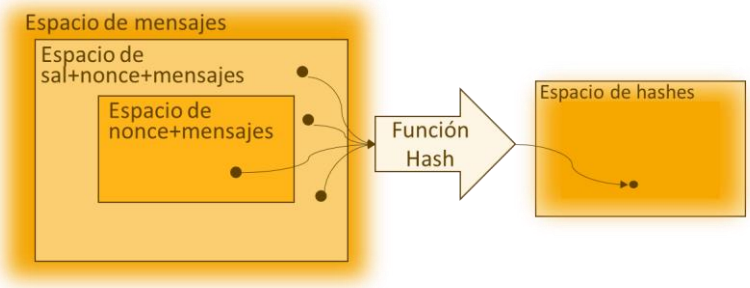


Un mensaje válido en el espacio de mensajes del tratamiento debería tener la estructura anterior, que denominaremos mensaje ampliado. El mensaje ampliado es almacenado por el responsable y los campos sal y nonce mantenidos de forma confidencial para cada uno de esos mensajes. Dicha estrategia tendría los siguientes efectos:

- Por un lado, al añadir un nonce a cada posible mensaje que amplíe la entropía del espacio de mensajes, se obtendría un espacio de mensajes del orden del espacio de hashes, lo que haría altamente probable una cobertura homogénea de este último. Es decir, existe una alta posibilidad que todo posible valor de hash pueda corresponder a un mensaje válido.



- Por otro lado, al ser la sal de un solo uso, y no ser compartida, el espacio de posibles (sal+nonce+mensajes) crece de tal forma que cada valor de hash tiene una alta probabilidad de corresponder a tantos tríos (sal+nonce+mensaje) como valores de hash existen. Es decir, se garantiza la posibilidad de colisión.



Las características que tendría este modelo serían las siguientes:

- Para comprometer la reidentificación hay que tener acceso al valor sal y nonce asociado a cada mensaje.
- La posibilidad de colisión en el espacio de mensajes ampliado está garantizada, ya que para un mismo mensaje existe una alta probabilidad de que todos los mensajes del espacio original pudieran tener los mismos hashes si se selecciona la sal y el nonce adecuados.
- El valor del hash no identifica un único mensaje ampliado dentro del espacio de posibles mensajes ampliados, aunque el estado actual de la tecnología hace que sea muy difícil construir un mensaje ampliado (sal+nonce+ mensaje falso)

que tenga el mismo hash que el generado a partir del mensaje ampliado (sal+nonce+mensaje verdadero). Por lo tanto, cumple con su función de identificador único. Para ello, la longitud de la sal+nonce tiene que ser lo suficientemente grande.

- Por otro lado, la entidad que posea el trio (sal+nonce+mensaje) puede validar el hash.

En el caso de que dicho mensaje ampliado sea borrado por el responsable, es decir, se elimina el mensaje, pero también la sal y el nonce asociado a ese mensaje en concreto, no es posible validar el hash en una primera instancia. Al utilizar información aleatoria de un solo uso, y si no hay vinculación entre la sal de los distintos hashes, se dificulta la reidentificación. Si la sal es lo suficientemente grande, y la sal original se ha eliminado, para cualquier mensaje se podría construir una sal que validase el hash (siempre que no hubiera información adicional vinculada) lo que debilitaría asociar el hash al mensaje original.

MODELOS DIFERENCIALES

En la misma línea que el modelo anterior, y con las mismas prevenciones y conclusiones, está la aplicación de modelos de privacidad diferencial.

En este caso, la estrategia consiste en añadir al mensaje un valor de ruido que, al contrario que una sal que se incluye como cabecera del mensaje, se incorpora al mensaje en sí. Este valor de ruido se puede aplicar de muchas formas tanto sobre información gráfica, sonora u otro tipo de datos de datos escalares, utilizando técnicas de privacidad diferencial para diseñarlas y para implementarlas técnicas como las empleadas en las marcas de agua digitales[35].

Para que el valor de ruido se considerase aceptable sería necesario que cumpliese con ciertas condiciones. En primer lugar, es preciso analizar su naturaleza aleatoria y su desvinculación con el contenido del mensaje. Por otro lado, habría que garantizar la no correlación del ruido introducido entre distintos mensajes, además de asegurar que añade una entropía muy por encima del número de bits del resultado del hash y, finalmente, hay comprobar que el estado de la tecnología no permita modelos de tratamiento que permitan la reidentificación[36].

Una de las ventajas de estos modelos sobre el modelo anterior de sal de un solo uso es la eliminación de la vulnerabilidad asociada al procesamiento por bloques de la implementación de la mayor parte de los algoritmos de hash.

VII. ANALISIS DEL HASH COMO SISTEMA DE SEUDONIMIZACIÓN O DE ANONIMIZACIÓN DE LA INFORMACIÓN DE CARÁCTER PERSONAL

Los apartados anteriores describen la forma de operación de las técnicas de hash desde un punto de vista teórico. Sin embargo, para evaluar su adecuación para proteger datos de carácter personal se han de tener en cuenta los diversos elementos que

[35] La marca de agua digital es un código de identificación que se inserta directamente en el contenido de un archivo multimedia (imagen, audio, video), de manera que sea difícil de apreciar por el sistema perceptual humano, pero fácil de detectar usando un algoritmo dado y una clave, en un ordenador. http://digital.csic.es/bitstream/10261/8864/1/Marcas_de_agua_en_el_mundo_real.pdf

[36] Por ejemplo, ya se están desarrollando modelos para evitar el ruido introducido en el proceso de escaneo de imágenes afecte al hash para la comparación entre copia y original: Ahmad, Fawad and Lee-Ming Cheng. “Paper Document Authentication Using Print-Scan Resistant Image Hashing and Public-Key Cryptography.” SpaCCS (2019).

conforman un tratamiento real, algunos de los cuales ya se han expuesto y otros dependerán de la implementación, como pueden ser:

- El propio cálculo del hash, que incluye a su vez varios elementos, entre otros:
  - La selección de un algoritmo concreto[37] (e.g. SHA-512, BLAKE-256 or SWIFFT).
  - La implementación concreta dicho algoritmo en un código o circuito (e.g. OpenSSL, Bouncy Castle or Libgcrypt).
  - El tipo de sistema sobre que se implementa el tratamiento: local, en remoto, transaccional, algún servicio en la nube, etc.
- El espacio de mensajes del tratamiento, que tendrá aspectos a tener en cuenta como:
  - La entropía del mismo.
  - El preproceso del mensaje antes de ejecutar el hash, como el añadido de padding[38] o elementos aleatorios.
  - Y, relacionado con el anterior, la redundancia y los elementos repetitivos de la estructura del mensaje.
- La vinculación del hash con otra información del entorno del tratamiento, en particular:
  - Información directamente vinculada a los registros en los que se incluye el hash, tanto identificadores como seudoidentificadores.
  - Información indirectamente vinculada, como la correlación que se establece entre registros anteriores y posteriores, correspondencia de fechas de actualización de registros de distintas tablas, ficheros de log de los servicios que conforman el tratamiento, etc., y cualquier otro elemento que se deriva de los procedimientos de implementación del tratamiento.
- Las contraseñas y elementos aleatorios introducidos, ya sea sal u otras técnicas:
  - Mecanismos de generación, almacenamiento, distribución y eliminación.
  - Tamaño y entropía de estos y su intervención en el preproceso del mensaje, como ya se ha indicado antes.
- La gestión y auditoría continua de los elementos anteriores, incluyendo la seguridad física y el factor humano, que se ven afectados por la evolución tecnológica y la modificación de los tratamientos.

Todos los elementos anteriores incluyen una serie de debilidades e introducen distintos elementos de riesgo cuya materialización tiene una probabilidad que crece en el tiempo debido a la acumulación de información, vulnerabilidades descubiertas, desarrollo tecnológico, etc.

Estas debilidades afectan, de igual manera, a los sistemas criptográficos. Estableciendo un paralelismo, la adecuación de un sistema criptográfico para un tratamiento específico se determina en función del "tiempo de vida" del mensaje que se está cifrando. El tiempo de vida de un mensaje está definido por el momento en el que el mensaje deja de tener valor[39] o importancia. Un sistema de cifrado será adecuado para un tratamiento si hay una expectativa razonable de proteger el mensaje durante todo su tiempo de vida.

[37] https://en.wikipedia.org/wiki/List_of_hash_functions

[38] Padding son técnicas para añadir datos al final de un mensaje para que tenga el tamaño suficiente para formar un bloque de entrada en el algoritmos de hash

[39] La información de la salida de unos valores a bolsa o una noticia pierde su valor en horas o días, un secreto comercial puede tener valor durante unos años, mientras que los secretos diplomáticos deben mantenerse confidenciales durnate décadas.

Un dato de carácter personal, de forma general, tiene una expectativa de vida tan larga como la del sujeto al que pertenece, especialmente si estamos tratando categorías especiales de datos. Un sistema de cifrado que tuviera como requisito expectativas razonables de protección por encima de setenta años sería un sistema excepcionalmente robusto. Por lo tanto, los requisitos que se imponen a un sistema de hash para considerar que es una técnica adecuada de seudonimización, o incluso la anonimización, son muy altos.

Finalmente, recalcar que, para anonimizar un fichero, hay que tratar los datos "de tal manera que no puedan usarse para identificar a una persona física mediante «el conjunto de los medios que puedan ser razonablemente utilizados» por el responsable del tratamiento o por terceros."[40] De ahí que **los procesos de anonimización deban garantizar que tampoco el responsable del tratamiento pueda reidentificar a los individuos de un fichero anonimizado**. Si la función hash se ha implementado teniendo en cuenta los factores listados, cuando el responsable del tratamiento borra los elementos aleatorios introducidos o el ruido añadido, anonimiza los datos. Como consecuencia natural, el responsable del tratamiento perderá la capacidad de validar el hash.

VIII. CONCLUSIONES

El Dictamen 5/2014 sobre técnicas de anonimización incluye las funciones hash en el apartado de técnicas de seudonimización[41].

La utilización de las técnicas de hash para seudonimizar o anonimizar la información de carácter personal ha de estar acompañada de un análisis de los riesgos de reidentificación que tiene la técnica de hash concreta empleada en el tratamiento. En dicho análisis de riesgos se ha de analizar tanto el proceso de hash, como los restantes elementos que conforman el sistema de hash, con particular atención a la información vinculada o vinculable al propio valor representado por el hash. El análisis ha de resultar en una evaluación objetiva[42] de la probabilidad de reidentificación a largo plazo.

Independientemente del análisis de riesgos, los elementos básicos a tener en cuenta para la utilización de funciones hash para la protección de la información son, entre otros:

- Alta entropía de la información a realizar el hash.
- Utilización de valores de sal/aleatorios de un solo uso.
- En su caso, tamaño de un valor sal por encima del tamaño de bloque del hash, sin ser múltiplo del tamaño de bloque.
- Utilizar generadores de información aleatoria apropiados para técnicas criptográficas.
- Acceso seguro al proceso de ejecución del hash.
- Nula vinculación con identificadores, seudoidentificadores u otra información, en particular en el mismo registro y entre registros o tablas/cadenas paralelas.
- Realizar auditorías periódicas de los procesos de gestión del sistema de hash.

Para considerar la técnica de hash como una técnica de anonimización, dicho análisis de riesgos ha de evaluar, además:

[40] WP29: Dictamen 05/2014 sobre técnicas de anonimización. Sección 2.1

[41] En el anexo se incluye un extracto del Dictamen 5/2014

[42] RGPD – Cons. 76: …El riesgo ha de ponderarse sobre la base de una evaluación objetiva….

- Las medidas organizativas que garantizan una eliminación de la información que permita le reidentificación.
- Una garantía razonable de que el sistema será robusto más allá de la vida esperada de los datos de carácter personal.

En definitiva, la adopción de garantías para la aplicación de los principios establecidos en el RGPD requiere de un análisis cualitativo riguroso previo para determinar su adecuación de forma objetiva.

IX. BIBLIOGRAFÍA

- Reglamento (UE) 2016/679 del Parlamento Europeo y del Consejo, de 27 de abril de 2016, relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos y por el que se deroga la Directiva 95/46/CE
- WP29: Dictamen 05/2014 sobre técnicas de anonimización.
- AEPD: Orientaciones y Garantías en los Procedimientos de Anonimización de Datos Personales de la AEPD
- AEPD: La K-anonimidad como medida de la privacidad.
- ENISA: Recommendations on shaping technology according to GDPR provisions. An overview on data pseudonymisation
- FIPS PUB 180-4 Secure Hash Standard (SHS) Federal Information Processing Standards Publication National Institute of Standards and Technology Gaithersburg, MD 20899-8900
  RFC1321 The MD5 Message-Digest Algorithm
- Blockchain and the General Data Protection Regulation. Can distributed ledgers be squared with European data protection law? Parlamento Europeo. European Parliamentary Research Service. Julio 2019
- C.E. Shannon Communication Theory of Secrecy Systems
- B. Schneier: Applied Cryptography, 2º edition. Wiley

X. ANEXOS

EXTRACTOS DEL RGPD

En los considerandos 26, 28, 29, 75, 78, 85 y 156 se contempla la seudonimización, los más relevantes para el tema tratado en el presente documento son:

> *26. Los principios de la protección de datos deben aplicarse a toda la información relativa a una persona física identificada o identificable. Los datos personales seudonimizados, que cabría atribuir a una persona física mediante la utilización de información adicional, deben considerarse información sobre una persona física identificable. Para determinar si una persona física es identificable, deben tenerse en cuenta todos los medios, como la singularización, que razonablemente pueda utilizar el responsable del tratamiento o cualquier otra persona para identificar directa o indirectamente a la persona física. Para determinar si existe una probabilidad razonable de que se utilicen medios para identificar a una persona física, deben tenerse en cuenta todos los factores objetivos, como los costes y el tiempo necesarios para la identificación, teniendo en cuenta tanto la*

tecnología disponible en el momento del tratamiento como los avances tecnológicos. Por lo tanto los principios de protección de datos no deben aplicarse a la información anónima, es decir información que no guarda relación con una persona física identificada o identificable, ni a los datos convertidos en anónimos de forma que el interesado no sea identificable, o deje de serlo. En consecuencia, el presente Reglamento no afecta al tratamiento de dicha información anónima, inclusive con fines estadísticos o de investigación.

28. La aplicación de la seudonimización a los datos personales puede reducir los riesgos para los interesados afectados y ayudar a los responsables y a los encargados del tratamiento a cumplir sus obligaciones de protección de los datos. Así pues, la introducción explícita de la «seudonimización» en el presente Reglamento no pretende excluir ninguna otra medida relativa a la protección de los datos.

29. Para incentivar la aplicación de la seudonimización en el tratamiento de datos personales, debe ser posible establecer medidas de seudonimización, permitiendo al mismo tiempo un análisis general, por parte del mismo responsable del tratamiento, cuando este haya adoptado las medidas técnicas y organizativas necesarias para garantizar que se aplique el presente Reglamento al tratamiento correspondiente y que se mantenga por separado la información adicional para la atribución de los datos personales a una persona concreta. El responsable que trate datos personales debe indicar cuáles son sus personas autorizadas.

75. Los riesgos para los derechos y libertades de las personas físicas, de gravedad y probabilidad variables, pueden deberse al tratamiento de datos que pudieran provocar daños y perjuicios físicos, materiales o inmateriales, en particular en los casos en los que el tratamiento pueda dar lugar a problemas de discriminación, usurpación de identidad o fraude, pérdidas financieras, daño para la reputación, pérdida de confidencialidad de datos sujetos al secreto profesional, reversión no autorizada de la seudonimización o cualquier otro perjuicio económico o social significativo; en los casos en los que se prive a los interesados de sus derechos y libertades o se les impida ejercer el control sobre sus datos personales; en los casos en los que los datos personales tratados revelen el origen étnico o racial, las opiniones políticas, la religión o creencias filosóficas, la militancia en sindicatos y el tratamiento de datos genéticos, datos relativos a la salud o datos sobre la vida sexual, o las condenas e infracciones penales o medidas de seguridad conexas; en los casos en los que se evalúen aspectos personales, en particular el análisis o la predicción de aspectos referidos al rendimiento en el trabajo, situación económica, salud, preferencias o intereses personales, fiabilidad o comportamiento, situación o movimientos, con el fin de crear o utilizar perfiles personales; en los casos en los que se traten datos personales de personas vulnerables, en particular niños; o en los casos en los que el tratamiento implique una gran cantidad de datos personales y afecte a un gran número de interesados.

78. La protección de los derechos y libertades de las personas físicas con respecto al tratamiento de datos personales exige la adopción de medidas técnicas y organizativas apropiadas con el fin de garantizar el cumplimiento de los requisitos del presente Reglamento. A fin de poder demostrar la conformidad con el presente Reglamento, el responsable del tratamiento debe adoptar políticas internas y aplicar medidas que cumplan en particular los principios de protección de datos desde el diseño y por defecto. Dichas medidas podrían consistir, entre otras, en reducir al máximo el tratamiento de datos personales, seudonimizar lo antes posible los datos personales, dar transparencia a las funciones y el tratamiento de

datos personales, permitiendo a los interesados supervisar el tratamiento de datos y al responsable del tratamiento crear y mejorar elementos de seguridad. Al desarrollar, diseñar, seleccionar y usar aplicaciones, servicios y productos que están basados en el tratamiento de datos personales o que tratan datos personales para cumplir su función, ha de alentarse a los productores de los productos, servicios y aplicaciones a que tengan en cuenta el derecho a la protección de datos cuando desarrollan y diseñen estos productos, servicios y aplicaciones, y que se aseguren, con la debida atención al estado de la técnica, de que los responsables y los encargados del tratamiento están en condiciones de cumplir sus obligaciones en materia de protección de datos. Los principios de la protección de datos desde el diseño y por defecto también deben tenerse en cuenta en el contexto de los contratos públicos.

En el artículo 4.5 del RGPD se establece:

«seudonimización»: el tratamiento de datos personales de manera tal que ya no puedan atribuirse a un interesado sin utilizar información adicional, siempre que dicha información adicional figure por separado y esté sujeta a medidas técnicas y organizativas destinadas a garantizar que los datos personales no se atribuyan a una persona física identificada o identificable

Asimismo, se incluyen las siguientes referencias en el resto del articulado:

- Artículo 6 Licitud de Tratamiento en el apartado 4 sobre el tratamiento para un fin distinto, en su letra e):

e) la existencia de garantías adecuadas, que podrán incluir el cifrado o la seudonimización.

- Artículo 25 Protección de datos desde el diseño y por defecto

1. *Teniendo en cuenta el estado de la técnica, el coste de la aplicación y la naturaleza, ámbito, contexto y fines del tratamiento, así como los riesgos de diversa probabilidad y gravedad que entraña el tratamiento para los derechos y libertades de las personas físicas, el responsable del tratamiento aplicará, tanto en el momento de determinar los medios de tratamiento como en el momento del propio tratamiento, medidas técnicas y organizativas apropiadas, como la seudonimización, concebidas para aplicar de forma efectiva los principios de protección de datos, como la minimización de datos, e integrar las garantías necesarias en el tratamiento, a fin de cumplir los requisitos del presente Reglamento y proteger los derechos de los interesados.*

- Artículo 32 Seguridad del tratamiento

1. *Teniendo en cuenta el estado de la técnica, los costes de aplicación, y la naturaleza, el alcance, el contexto y los fines del tratamiento, así como riesgos de probabilidad y gravedad variables para los derechos y libertades de las personas físicas, el responsable y el encargado del tratamiento aplicarán medidas técnicas y organizativas apropiadas para garantizar un nivel de seguridad adecuado al riesgo, que en su caso incluya, entre otros:*

   a) la seudonimización y el cifrado de datos personales;

- Artículo 40 Códigos de conducta

2. *Las asociaciones y otros organismos representativos de categorías de responsables o encargados del tratamiento podrán elaborar códigos de conducta o modificar o ampliar dichos códigos con objeto de especificar la aplicación del presente Reglamento, como en lo que respecta a:*

   d) la seudonimización de datos personales

- Artículo 89 Garantías y excepciones aplicables al tratamiento con fines de archivo en interés público, fines de investigación científica o histórica o fines estadísticos

1. *El tratamiento con fines de archivo en interés público, fines de investigación científica o histórica o fines estadísticos estará sujeto a las garantías adecuadas, con arreglo al presente Reglamento, para los derechos y las libertades de los interesados. Dichas garantías harán que se disponga de medidas técnicas y organizativas, en particular para garantizar el respeto del principio de minimización de los datos personales. Tales medidas podrán incluir la seudonimización, siempre que de esa forma puedan alcanzarse dichos fines. Siempre que esos fines pueden alcanzarse mediante un tratamiento ulterior que no permita o ya no permita la identificación de los interesados, esos fines se alcanzarán de ese modo.*

EXTRACTOS DEL DICTAMEN 5/2014 SOBRE TÉCNICAS DE ANONIMIZACIÓN

El Dictamen 5/2014 sobre técnicas de anonimización establece las funciones hash como una de las técnicas de seudonimización:

> ***Función hash:*** *Se trata de una función que devuelve un resultado de tamaño fijo a partir de un valor de entrada de cualquier tamaño (esta entrada puede estar formada por un solo atributo o por un conjunto de atributos). Esta función no es reversible, es decir, no existe el riesgo de revertir el resultado, como en el caso del cifrado. Sin embargo, si se conoce el rango de los valores de entrada de la función hash, se pueden pasar estos valores por la función a fin de obtener el valor real de un registro determinado.*
>
> *Por ejemplo, si se aplica la función hash al número de identificación nacional para seudonimizar un conjunto de datos, dicho atributo se puede obtener simplemente ejecutando la función con todos los posibles valores de entrada y comparando los resultados con los valores del conjunto de datos. Habitualmente, las funciones hash se diseñan para poder ejecutarse de manera relativamente rápida, por lo que están sujetas a ataques de fuerza bruta (Estos ataques consisten en probar todas las posibles entradas para crear tablas de correspondencia). También se pueden crear tablas precalculadas para lograr una reversión masiva de un gran número de valores hash.*
>
> *El uso de una función hash «con sal» (en la que se añade un valor aleatorio, conocido como «sal», al atributo al que se aplica la función hash) puede reducir la probabilidad de obtener el valor de entrada.*
>
> *No obstante, usando medios razonables, todavía existe la posibilidad de calcular el valor original del atributo que se oculta tras el resultado de una función hash con sal. Especialmente si se conoce el tipo de atributo (nombre, número de seguridad social, fecha de nacimiento, etc.). Para añadir dificultad computacional, se podría recurrir a una función hash de derivación de clave, en la que al valor computado se le aplica varias veces la función hash con poca sal.*

Función con clave almacenada: *Se trata de un tipo de función hash que hace uso de una clave secreta a modo de valor de entrada suplementario (lo cual la diferencia de una función hash con sal, ya que, normalmente, la sal no es secreta.) El responsable del tratamiento puede reproducir la ejecución de la función con el atributo y la clave secreta. Sin embargo, los atacantes, que no conocen la clave, lo tendrían mucho más difícil: el número de combinaciones que habría que probar sería tan grande, que convertiría este procedimiento en impracticable.*

Cifrado determinista o función hash con clave con borrado de clave: *Esta técnica equivale a generar un número aleatorio a modo de seudónimo para cada atributo de la base de datos y, posteriormente, borrar la tabla de correspondencia. Esta solución reduce el riesgo de vinculabilidad entre los datos personales contenidos en el conjunto de datos y los datos personales relativos a la misma persona contenidos en otro conjunto de datos en el que se usa un seudónimo diferente. Si se ejecutan los algoritmos más avanzados, el esfuerzo de cálculo que debería realizar un atacante para descifrar o reproducir la ejecución de la función sería muy grande, ya que tendría que probar cada posible clave, puesto que esta se desconoce.*

…

Una de las ideas falsas más extendidas sobre la anonimización es la de que esta equivale al cifrado o a la codificación con clave. Esta idea falsa se basa en dos suposiciones, a saber, a) que una vez que se aplica el cifrado a algunos atributos de un registro en una base de datos (p. ej., nombre, dirección, fecha de nacimiento), o si estos atributos se sustituyen con una cadena de caracteres supuestamente aleatorizada mediante una operación de codificación con clave (como una función hash con clave), entonces ese registro se ha anonimizado;

…

Es engañoso fiarse exclusivamente de la solidez del mecanismo de cifrado como medida del grado de anonimización de un conjunto de datos, ya que existen otros muchos factores técnicos y organizativos que afectan a la seguridad global de un mecanismo de cifrado o una función hash.

EXTRACTOS DE LAS ORIENTACIONES Y GARANTÍAS EN LOS PROCEDIMIENTOS DE ANONIMIZACIÓN DE DATOS PERSONALES DE LA AEPD

3.8 SELECCIÓN DE LAS TÉCNICAS DE ANONIMIZACIÓN: CLAVES

1. ALGORITMOS DE HASH: es incuestionable la utilidad que tienen los algoritmos de cifrado cuando necesitamos anonimizar microdatos, resultando especialmente útiles los algoritmos de "hash". Un algoritmo de hash es un mecanismo que, aplicado a un dato concreto, genera una clave única o casi única que puede utilizarse para representar un dato. Por ejemplo, disponemos de un dato que queremos ocultar o anonimizar y para ello utilizamos un algoritmo de hash, como por ejemplo SHA1 o MD5. De la aplicación del algoritmo a un determinado dato obtenemos una clave o huella digital que puede utilizarse para reemplazar el dato real. El algoritmo de hash genera una huella digital y hace imposible reconstruir el dato original partiendo de la huella y por otra parte cualquier variación en el dato original dará lugar a una huella digital diferente, lo que expresado en términos computacionales podría decirse que la modificación de un solo bit en la información original almacenada en un ordenador daría lugar a una clave distinta o una huella digital distinta.

El algoritmo de hash permite que, partiendo de un mismo dato o microdato, podamos generar siempre la misma huella digital pero partiendo de una determinada huella digital nunca podremos obtener el dato original, garantizando la confidencialidad al tratarse de una operación matemática de un solo sentido. Las claves resultantes de la aplicación de un algoritmo de hash son comúnmente conocidas como "huella digital" por entender que representan de forma unívoca a un dato o microdato concreto.

Sin embargo, un algoritmo de hash por sí solo no es suficiente para hacer irreversible la anonimización, ya que pequeñas cadenas de texto como, por ejemplo, los microdatos correspondientes al código postal de una persona, un número de teléfono, etc., pueden ser fácilmente reidentificables con un programa informático que genere cifras consecutivas y sus correspondientes huellas digitales. Si lo que queremos es garantizar la anonimización de un microdato es preciso utilizar un mecanismo criptográfico que nos garantice el secreto de la huella digital que hemos generado. Una buena opción es el algoritmo HMAC basado en RFC2014 HMAC puede utilizarse en combinación con varios algoritmos de hash como, por ejemplo, con MD5 y sobre la huella digital o clave resultante del algoritmo de hash aplica un algoritmo criptográfico que genera una nueva huella digital o clave en función de una clave secreta.

La utilización de HMAC en combinación con claves secretas no triviales y una política diligente de destrucción de claves puede servir para garantizar la irreversibilidad del proceso de anonimización. Cuando las claves utilizadas con HMAC se conservan pueden servir para generar datos seudonimizados que requieran una posterior reidentificación de los interesados. Mecanismos de hash con clave secreta pueden resultar útiles para enmascarar los datos. Sin embargo, deberá existir un procedimiento que permita la eliminación segura de las claves y la posibilidad de acreditar que el procedimiento se ha cumplido para garantizar la irreversibilidad del proceso.

EXTRACTOS DE DOCUMENTO DE ENISA. RECOMMENDATIONS ON SHAPING TECHNOLOGY ACCORDING TO GDPR PROVISIONS. AN OVERVIEW ON DATA PSEUDONYMISATION

Hashing without key

Hashing is a technique that can be used to derive pseudonyms, but, as will be shown later in this Section, has some serious drawbacks with regard to the design goals set in Section 3.1. Still, it is a starting point for understanding other stronger techniques in the field and this is why we present it first. Moreover, hashing can be a useful tool to support data accuracy. A cryptographic hash function h is a function with specific properties (as described next) which transforms any input message m of arbitrary length to a fixed-size output h(m) (e.g. of size 256 bits, that is 32 characters), being called hash value or message digest. The message digest satisfies the following properties [Menezes, 1996]:

i) given h(m), it is computationally infeasible to compute the unknown m, and this holds for any output h(m) - i.e. the function h is mathematically irreversible (pre-image resistance),

ii) for any given m, it is computationally infeasible to find another m'≠ m such that h(m')=h(m) (2nd pre-image resistance),

iii) it is computationally infeasible to find any two distinct inputs m, m' (free choice) such that h(m')=h(m) (collision resistance). Clearly, if a function is collision-resistant, then it is 2nd pre-image resistant too.

In other words, a cryptographic hash algorithm is one that generates a unique digest (which is also usually called fingerprint) of a fixed size for any single block of data of arbitrary size (e.g. an initial identifier of any kind). Note that for any given hash function, the same unique digest is always produced for the same input (same block of data). It is important to point out that state-of-the art hash functions should be chosen; therefore, commonly used hash functions such as MD5 and SHA-1 [Menezes,1996] with known vulnerabilities – with respect to the probability of finding collisions - should be avoided (see [Wang, 2005], [Dougherty,2008], [Stevens, 2017a], [Stevens,2017b]). Instead, cryptographically resistant hash functions should be preferable, e.g. SHA-2 and SHA-3 are currently considered as state-of-the-art [FIPS, 2012], [FIPS,2015].

The above properties of hash functions allow them to be used in several applications, including data integrity and entity authentication. For instance, once an app market has a hash server storing hash values of app source codes, any user can verify whether the source code has been modified or not via a simple validation of its hash value - since any modification of the code would lead to a different hash value (see, e.g., [Jeun, 2011]). Similarly, recalling the discussion in Section 3.1 on data accuracy, a pseudonym that is generated via hashing user's identifiers may be a convenient way for a data controller to verify a user's identity. However, when it comes to pseudonymisation, despite the aforementioned properties of a cryptographic hash function, simple hashing of data subjects' identifiers to provide pseudonyms has major drawbacks. More precisely, with regard to the aforementioned D1 and D2 design goals, we have the following:

- *The D2 property does not hold, since any third party that applies the same hash function to the same identifier gets the same pseudonym.*

- *In relation to the above observation, the D1 property also does not necessarily hold, since it is trivial for any third party to check, for a given identifier, whether a pseudonym corresponds to this identifier (i.e. though hashing the identifier).*

Therefore, a reversal of pseudonymisation is possible whenever such an approach is adopted, as having a list of the (possible) initial identifiers is adequate for any third party to associate these identifiers with the corresponding pseudonyms, with no any other association being in place . In fact, following the GDPR definition of pseudonymisation, one could argue that hashing is a weak pseudonymisation technique as it can be reversed without the use of additional information. Relevant examples are provided in [Demir, 2018] (and in references therein), where the researchers refer to the Gravatar service and describe how users' email addresses can be derived through their hash value, which is shown in the URL that corresponds to the gravatar of the user, without any additional information. Hence, hash functions are generally not recommended for pseudonymisation of personal data, although they can still contribute to enhancing security in specific contexts with negligible privacy risks and when the initial identifiers cannot be guessed or easily inferred by a third party. For the vast majority of the cases, such pseudonymisation technique does not seem to be sufficient as a data protection mechanism [Demir, 2018]. However, a simple hashing procedure may still have its own importance in terms of data accuracy, as stated previously.

Hashing with key or salt

A robust approach to generate pseudonyms is based on the use of keyed hash functions – i.e. hash functions whose output depends not only on the input but on a secret key too; in cryptography, such primitives are being called message authentication codes (see, e.g., [Menezes, 1996]). The main difference from the conventional hash functions is that, for the same input (a data subject's identifier), several different pseudonyms can be produced, according to the choice of the specific key – and, thus, the D2 property is ensured. Moreover, the D1 property also holds, as long as any third party, i.e. other than the controller or the processor, (e.g. an adversary) does not have knowledge of the key and, thus, is not in the position to verify whether a pseudonym corresponds to a specific known identifier. Apparently, if the data controller needs to assign the same pseudonym to the same individual, then the same secret key should be used. To ensure the aforementioned properties, a secure keyed-hash function, with properly chosen parameters, is needed. A known such standard is the HMAC [FIPS, 2008], whose strength is contingent on the strength of the underlying simple hash function (and, thus, incorporating SHA-2 or SHA-3 in HMAC is currently a right option). Moreover, the secret key needs to be unpredictable and of sufficient length, e.g. 256 bits, which could be considered as adequate even for the post-quantum era . If the secret key is disclosed to a third party, then the keyed hash function actually becomes a conventional hash function in terms of evaluating its pseudonymisation strength. Hence, recalling the definition of pseudonymisation in the GDPR, the data controller should keep the secret key securely stored separately from other data, as it constitutes the additional information, i.e. it provides the means for associating the individuals – i.e. the original identifiers – with the derived pseudonyms.

Keyed hash functions are especially applicable as pseudonymisation techniques in cases that a data controller needs - in specific data processing context - to track the individuals without, however, storing their initial identifiers (see also [Digital Summit, 2017]). Indeed, if the data controller applies - always with the same secret key - a keyed hash function on a data subject's identifier to produce a pseudonym, without though storing the initial user's identifier, then we have the following outcomes:

- *The same pseudonym will always be computed for each data subject (i.e. allowing tracking of the data subject).*
- *Associating a pseudonym to the initial identifier is practically not feasible (provided that the controller does not have knowledge of the initial identifiers).*

Therefore, if only tracking of data subjects is required, the controller needs to have access to the key but does not need to have access to the initial identifiers, after pseudonymisation has been performed. This is an important consideration that adheres to the principle of data minimization and should be considered by the controller as a data protection by design aspect. Moreover, a keyed hash function has also the following property: if the secret key is securely destroyed and the hash function is cryptographically strong, it is computationally hard, even for the data controller, to reverse the pseudonym to the initial identifier, even if the controller has knowledge of the initial identifiers. Therefore, the usage of a keyed hash function may allow for subsequent anonymisation of data, if necessary, since deleting the secret key actually deletes any association between the pseudonyms and the initial identifiers. More generally, using a keyed hash function to generate a pseudonym and subsequently deleting the secret key is somehow equivalent to generate random pseudonyms, without any connection with the initial identifiers.

Another approach that is often presented as an alternative to the keyed hash function is the usage of an unkeyed (i.e. conventional) hash function with a so-called "salt" – that is the input to the hash function is being augmented via adding auxiliary random-looking data that are being called "salt". Again, if such a technique is appropriately applied, for the same identifier, several different pseudonyms can be produced, according to the choice of the salt – and, thus, the D2 property is ensured, whilst the D1 property also holds with regard to third parties provided that they do not have knowledge of the salt. Of course, this conclusion is valid only as long as the salt is appropriately secured and separated from the hash. Note that, as in the case of keyed hash, the same salt should be used by the controller in cases that there is need to assign always the same pseudonym to the same individual. Moreover, salted hash functions can be utilized in cases where the controller does need to store the initial identifiers, while still being able to track the data subjects. Last, if the salt is securely destroyed by the controller, it is not trivial to restore the association between pseudonyms and identifiers. However, it should be stressed that in several typical cases employing salts for protecting hashes has some serious drawbacks:

On one hand, the salt does not share the same unpredictability properties as secret keys (e.g. a salt may consist of 8 characters, i.e. 64 bits, as in the cases of protecting users' passwords in some Linux systems). More generally, from a cryptographic point of view, a keyed hash function is considered as more powerful approach than a salted hash function. There exist though several cryptographically strong techniques for generating salted hashes, which in turn could be considered as appropriate candidates for generating pseudonyms – a notable example being the bcrypt [Provos, 1999].