(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 655 818**

(51) Int. CI.:

H04L 9/32 (2006.01)
H04L 9/08 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **31.03.2015 E 15305475 (4)**

(97) Fecha y número de publicación de la concesión europea: **18.10.2017 EP 3076584**

(54) Título: **Método de recuperación de datos troceados**

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
21.02.2018

(73) Titular/es:

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE (50.0%)
Villa Douce 9 Boulevard de la Paix
51100 Reims, FR y
UNIVERSITÉ DE PICARDIE JULES VERNE (50.0%)

(72) Inventor/es:

DEQUEN, GILLES;
LEGENDRE, FLORIAN y
KRAJECKI, MICHAËL

(74) Agente/Representante:

LEHMANN NOVO, María Isabel

DESCRIPCIÓN

Método de recuperación de datos troceados.

La presente invención se refiere a métodos para recuperar datos troceados por una función de troceado criptográfica.

La democratización y el crecimiento de tecnologías digitales de alto rendimiento en expansión y de Internet han cambiado considerablemente el mundo de la comunicación. Las necesidades de preservar la protección comercial son, por consiguiente, numerosas. Ello puede lograrse comúnmente gracias a protocolos seguros y, por lo tanto, mediante el uso de mecanismos criptográficos.

Cuando un sistema necesita identificar un cliente como, por ejemplo, para una transacción bancaria, una conexión a un sitio web o una autenticación del sistema, la principal limitación se centra en el problema de la autenticación, debiendo el cliente confirmar su identidad al sistema con el fin de que este último esté seguro de la identidad del cliente. Para asegurar esta limitación, dichos sistemas requieren el uso de primitivas criptográficas.

De manera clásica, cuando se aplica un protocolo de autenticación, un cliente ha registrado previamente una cuenta en un servidor del sistema, preferiblemente información que comprende un ID de inicio de sesión y una contraseña, el ID de inicio de sesión siendo, por ejemplo, un nombre de cuenta, una dirección de correo electrónico o un identificador único, como se muestra en la Figura 1A. El servidor puede identificar al cliente dado que el servidor conoce la contraseña. La contraseña no se mantiene, en general, como un texto claro en el servidor. La etapa de registro incluye una computación de troceado criptográfica F de la contraseña con el fin de obtener una huella digital asociada, también llamada *digest*, mensaje *digest* o datos troceados. Gracias a las propiedades de las funciones de troceado criptográficas, dicho *digest* es una cadena de bits de tamaño fijo que permite identificar un dato sin acceder a su contenido. Dicho *digest* se almacena en el servidor y luego se usa para comprobar la integridad de los datos cada vez que se ingresa una contraseña, como se muestra en la Figura 1B.

Las funciones de troceado criptográficas conocidas son las funciones MD5 y SHA-0/1, respectivamente descritas en los documentos de referencia RFC-1321 y RFC-6194 ("*Request For Command*"). En los últimos años, se han descubierto algunas debilidades criptográficas teóricas y las funciones MD5 y SHA-0/1 ya no se usan con frecuencia. La función SHA-2 tiene un esquema de construcción muy cercano a una de las funciones MD5 y SHA-0/1 y, por consiguiente, tiene posibles debilidades de seguridad. Una nueva función de troceado, llamada SHA-3 y basada en un esquema de construcción muy diferente, se ha elegido por el organismo gubernamental NIST (Instituto Nacional de Normas y Tecnología) en octubre de 2012.

Sin embargo, los protocolos de autenticación dependen de un equilibrio entre la potencia de una contraseña y la capacidad de una persona para mantenerla en secreto, según se explica en el artículo de Robert Morris y Ken Thompson, "*Password security - a case history*" en *Communications of the ACM*, 22(11): 594-597, 1979. En general, o bien la contraseña es débil o el cliente corre el riesgo de olvidarla o perderla. Si ocurre esto último, se proponen dos soluciones al cliente: un sistema de Recuperación de Contraseña (PR, por sus siglas en inglés) para recuperar la contraseña original, o un Autoservicio de Restablecimiento de Contraseña (SSPR, por sus siglas en inglés) con el fin de obtener una nueva contraseña.

Ambos sistemas PR y SSPR pueden implementarse gracias a muchos enfoques como, por ejemplo, el uso de *tokens* de autenticación, como se describe en el artículo de Robert J. Zuccherato. "*Authentication token*" en *Encyclopedia of Cryptography and Security* (2da. Ed.), páginas 62-63, Springer, 2011, del método cliente-servidor, según se describe en el artículo de Lukasz y otros "*Client-server password recovery*", en *OTM Conferences 2*, páginas 861-878, 2009, de biométrica, como se explica en el artículo de Bernd Hohgrfe y Sebastian Jacobi "*Voice biometrics as a way to self-service password reset*", en *Norbert Pohlmann, Helmut Reimer, and Wolfgang Schneider, editors*, *ISSE 2009 Securing Electronic Business Processes*, páginas 137-144, Vieweg+Teubner, 2010, o mediante la respuesta a preguntas personales. Este último método, también llamado autenticación basada en el conocimiento, es con frecuencia privilegiado, sin embargo, presenta defectos de seguridad, en especial debido a las redes sociales que hacen mucho más fácil resolver preguntas personales de una persona, y debido a la piratería de bases de datos que puede llevar a la venta de información, como se muestra en los artículos de Lawrence O'Gorman, y otros "*Call center customer verification by query-directed passwords*" en *Financial Cryptography*, páginas 54-67, 2004, de Ariel Rabkin "*Personal knowledge questions for fallback authentication: security questions in the era of Facebook*", en SOUPS, páginas 13-23, 2008, de Markus Jakobsson y otros "*Love and authentication*", en *CHI 08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, páginas 197-200, ACM, 2008, y de Joseph Bonneau y otros "*What's in a name?*", en *Financial Cryptography*, páginas 98-113, 2010.

El documento "*Logical Reasoning to Detect Weaknesses about SHA-1 and MD4/5*" de Florian Legendre; Gilles Dequen; Michaël Krajecki publicado por la Asociación Internacional para la Investigación Criptológica, describe un criptoanálisis algebraico en un contexto binario gracias a la solución SAT. El presente documento trata dicho concepto aplicado a funciones de troceado criptográficas. En el presente documento, se lleva a cabo un criptoanálisis lógico de funciones de troceado mediante dos modos. El primero es usar un formalismo SAT para

proveer una comprensión original de las funciones de troceado y el segundo es abordar el problema de inversión que lleva a (segundas) preimágenes. Finalmente, pone en práctica los diferentes aportes para abordar la búsqueda de preimágenes sobre versiones reducidas de MD4, MD5, SHA-0 y SHA-1.

El documento US2010293600 describe una autenticación de recuperación de cuenta de seguridad de último recurso mediante el uso de una autenticación social. El titular de la cuenta solicita a administradores que se han identificado previamente que obtengan un código de recuperación de cuenta. El sistema de recuperación de cuenta envía una comunicación al administrador para solicitar información para verificar al administrador como uno de los administradores previamente identificados.

El documento US7809130 describe un método para recuperar una contraseña que incluye, para cada posible contraseña que debe verificarse, generar una unidad de periodicidad basada en un número de símbolos en la contraseña y un tamaño de un segmento usado por una función unidireccional para convertir la contraseña en un valor de control como, por ejemplo, un valor de troceado.

Una técnica conocida que permite restablecer o recuperar una contraseña perdida consiste en enviar al cliente por correo electrónico la contraseña. Dicho correo electrónico contiene información muy sensible y puede interceptarse por un ciberdelincuente si la conexión es insegura o si se el buzón de correo se rompe.

Otra técnica consiste en enviar al cliente por correo electrónico una contraseña temporal. Dicha contraseña temporal es tan sensible como la contraseña original salvo que se crea para un período corto. Sin embargo, si la conexión es insegura, un ciberdelincuente puede obtener dicha contraseña interceptando el correo electrónico, o un atacante puede ver la contraseña temporal en un buzón de correo roto y acceder al procedimiento de recuperación.

De manera alternativa, instrucciones sobre cómo restablecer la contraseña pueden enviarse por correo electrónico al cliente. Dichas instrucciones son tan sensibles como una contraseña temporal y un atacante puede interceptarlas a través de una conexión insegura, especialmente accediendo al buzón de correo del cliente.

Dichos métodos basados en el envío de información sensible por correo electrónico son también débiles porque los correos electrónicos se mantienen en el buzón de correo del cliente y, por consiguiente, la seguridad de los datos depende de la seguridad del buzón de correo.

Como ya se ha mencionado, visualizar la contraseña después de responder preguntas es riesgoso porque las preguntas son, con frecuencia, débiles y puede responderlas un atacante.

Visualizar la contraseña después de la verificación por teléfono del cliente, mediante el envío de un código en un texto de mensaje, por ejemplo, también se usa. Dicho código puede interceptarse por un atacante si la transmisión no es segura o si se corrompe el teléfono. Además, el código es, con frecuencia, corto y débil y, por consiguiente, es probable que un ataque de fuerza bruta lo aborde.

Otra técnica posible consiste en proveer una pista de la contraseña para ayudar al cliente. Ello también puede ayudar a un atacante y dicha técnica alienta las contraseñas débiles que son fáciles de recordar gracias a una pequeña indicación.

Un resumen del marco general para la recuperación de una contraseña perdida se presenta en la Figura 2. Durante una primera conexión a un servidor, en una etapa 51, un cliente se registra ingresando un identificador de inicio de sesión y una contraseña asociada. Esta última se encripta y almacena, en una etapa 52, en una memoria 61 a la cual el servidor tiene acceso. Si el cliente olvida su contraseña, puede solicitar una recuperación de contraseña al servidor, como se muestra en la etapa 53. Para ello, en una etapa 54, la contraseña regresa desde la memoria 61 con el fin de dársela al cliente después de un protocolo de recuperación de contraseña implementado en la etapa 55, como se ha explicado previamente, por ejemplo haciendo preguntas personales al cliente o enviando un código o correo electrónico que comprenden la contraseña original, una contraseña temporal o instrucciones para crear una nueva contraseña.

Dichas consideraciones pueden extenderse a cualquier dato que desea preservarse, diferente de una contraseña.

Existe, por consiguiente, una necesidad de mejorar la recuperación de datos que deben preservarse, con el fin de ofrecer mejores garantías de confidencialidad tanto a un cliente como a un servidor.

Un objeto de la invención, según un primer aspecto de sus aspectos, es un método para recuperar un dato que debe preservarse, ingresado por un cliente durante una conexión previa a un servidor, el servidor teniendo acceso a una memoria que comprende una palabra troceada generada mediante la aplicación de una función de troceado a un primer dato de entrada de una capacidad predefinida, el primer dato de entrada correspondiendo a los datos que deben preservarse modificados por una función de procesamiento, la capacidad de la palabra troceada siendo más baja que dicha capacidad predefinida, una clave de seguridad del cliente habiendo sido generada mediante la aplicación de una función de troceado a un segundo dato de entrada de una capacidad predefinida, el segundo dato

de entrada correspondiendo a los datos que deben preservarse modificados por una función de procesamiento, la capacidad de la clave de seguridad siendo igual a la diferencia entre dicha capacidad predefinida y la capacidad de la palabra troceada, la clave de seguridad no almacenándose en la memoria a la cual el servidor tiene acceso, el método en donde:

- después de una solicitud del cliente para recuperar los datos que deben preservarse, la palabra troceada y la clave de seguridad se concatenan con el fin de alcanzar dicha capacidad predefinida, y

- una función de troceado inversa, mediante el uso de una solución algebraica de la función de troceado habiendo generado la palabra troceada, se aplica a la concatenación de dicha palabra troceada y clave de seguridad, con el fin de recuperar los datos que deben preservarse.

La invención provee un método simple para recuperar un dato que debe preservarse, el cual no necesita almacenar de forma alguna el dato de entrada o revelar información sensible. Ello permite tanto al cliente como al servidor confiar solamente en ellos mismos.

La concatenación de la palabra troceada y la clave de seguridad es esencial para recuperar los datos que deben preservarse. Por consiguiente, un atacante malicioso necesita juntar dichas dos informaciones.

La memoria del servidor solo almacena la palabra troceada correspondiente a los datos que deben preservarse ingresados por el cliente. El servidor no tiene acceso a un seguimiento claro o codificado de los datos. Los datos que deben preservarse por el cliente no pueden, por consiguiente, robarse del servidor.

Además, de manera similar, el cliente no almacena los datos o información sensible. Ninguna malignidad directa hacia el cliente puede llevar a la corrupción de los datos que deben preservarse. El cliente solo tiene acceso a la clave de seguridad, también llamada « *información de troceado inverso* », que es del mismo tipo que la palabra troceada mantenida por el servidor. Es casi imposible obtener los datos a partir de esta información solamente.

Los datos perdidos no se envían nunca al cliente de forma directa, en un correo electrónico por ejemplo, como en los métodos conocidos. Ello permite evitar fallos de seguridad debido a la piratería de buzones de correo.

En el caso donde el dato que debe preservarse es una contraseña para la autenticación del cliente en el servidor, este último puede autenticar al cliente gracias a la palabra troceada. La clave de seguridad es, además, inútil para la autenticación.

En la presente invención, "capacidad" debe comprenderse como el número de bits de una cadena de bits.

Funciones de troceado criptográficas

De manera conocida, una función de troceado criptográfica F computa una palabra troceada *h* a partir de un dato de entrada *m*: *h* = F(*m*). Para un dato de entrada *m*, solo corresponde una palabra troceada *h*.

No existe enlace reconocible entre el dato de entrada *m* y la palabra troceada *h*. Las funciones de troceado criptográficas son ventajosamente no biyectivas. Encontrar un dato *m* por medio del conocimiento de *h* y llevar a cabo la función inversa $F^{-1}(h)$ es casi imposible. Ello garantiza la alta seguridad de las funciones de troceado.

Un protocolo de registro/autenticación mediante el uso de funciones de troceado puede comprender una primera etapa de registro: la contraseña *Mclient* del cliente se trocea y solo la palabra troceada *Hclient* = F(*Mclient*) se almacena por el servidor. Luego, cuando el cliente necesita autenticarse en el servidor, ingresa una secuencia *M'client.* El servidor computa *H'client* = F(*M'client*). Si *H'client* es igual a *Hclient,* el servidor autentifica al cliente y, de lo contrario, lo rechaza.

Para garantizar que las funciones de troceado son seguras, deben ser resistentes, teórica y computacionalmente, a colisiones, preimagen y segunda preimagen.

Una colisión ocurre cuando uno puede encontrar dos mensajes *m* y *m'* como, por ejemplo, F(*m*) = F(*m'*). Dicho ataque es la manera más fácil de debilitar una función de troceado y proveer muchos resultados tremendos, como se explica en los artículos de Xiaoyun Wang "*Collisions for hash functions MD4, MD5, haval-128 and ripeMD*", en *Crypto'04*, página 199, 1997, de Xiaoyun Wang y Hongbo Yu "*How to break MD5 and other hash functions*", en *EUROCRYPT*, páginas 19-35, 2005, de Hongbo Yu y Xiaoyun Wang, "*Multi-collision attack on the compression functions of MD4 and 3-pass haval*", en ICISC, páginas 206-226, 2007, de Christophe De Cannière y otros "*Collisions for 70-step SHA-1: On the full cost of collision search*", en *Selected Areas in Cryptography*, páginas 56-73, 2007, de Somitra Kumar Sanadhya y Palash Sarkar, "*New collision attacks against up to 24-step SHA-2*", en *INDOCRYPT*, páginas 91-103, 2008, y de Marc Stevens y otros "*Chosen-prefix collisions for MD5 and applications*", IJACT, 2(4):322-359, 2012.

Un ataque de preimagen consiste en, dada una función de trocado F y una palabra troceada *h,* encontrar un mensaje *m* como, por ejemplo, F(*m*) = *h.*

SHA-3

La función de troceado es, preferiblemente, una función de troceado SHA-3, especialmente mediante el uso del algoritmo Keccak. La invención puede, sin embargo, adaptarse a cualquier función de troceado criptográfica.

La función de troceado SHA-3, mediante el uso del algoritmo Keccak, pertenece a la familia de funciones esponja, es decir funciones que toman como entrada un dato de cualquier tamaño y entregan una palabra de tamaño fijo, según se explica en los artículos de Guido Bertoni y otros "*Sponge functions*", en *Ecrypt Hash Workshop 2007*, "*The keccak reference*", enero 2011, y "*Keccak*", en *EUROCRYPT*, páginas 313-314, 2013. La cadena de bits usada para concatenarse con los datos de entrada con el fin de alcanzar la capacidad de la permutación SHA-3 que se necesita para computar una palabra troceada con la capacidad final deseada se llama esponja.

El algoritmo Keccak puede implementarse de 12 a 24 rondas, con una capacidad de estado interno igual a 200, 400, 800 o 1600.

En SHA-3, un equilibrio entre los valores de la velocidad binaria r_b y la capacidad *c* de la esponja determina la seguridad de la función de troceado contra los ataques de preimagen y colisión. La capacidad de estado interno de la permutación SHA-3 se define por la suma de la velocidad binaria r_b y la capacidad *c* de la esponja. La permutación SHA-3 tiene, por ejemplo, una capacidad de estado interno de 1600 bits, incluidas palabras de 64 bits para estados internos, correspondiente a la capacidad predefinida $C_p = r_b + c$, *con* $r_b = 576$ y $c = 1024$, como se ilustra en la Figura 3 para una secuencia de datos, concatenada con uno o varios bits, llamada palabra "*de relleno*", con el fin de alcanzar la velocidad binaria r_b. La función de permutación completa consiste, de manera ventajosa, en 24 rondas de 5 subfunciones, que contienen solamente funciones limitadas a XOR a nivel de bit, AND a nivel de bit, operador NOT y Modulo. Una descripción detallada de una sola ronda, con palabras de 64 bits para estados internos, puede ser:

Requerir:

- palabras de 64 bits para estados internos
- 25 estados internos en el comienzo de la Ronda (a saber (texto llano I I relleno I I ISC) en la primera ronda)

- para xx en {00, ..., 24} y para i en {0, ..., 63}. Denotado $M_{xx}[i]$
- 25 estados internos al final de la ronda (a saber [digest || FSC] en la ronda final)
  para xx en {00, ..., 24} y para i en {0, ..., 63}. Denotado $M^{+}_{xx}[i]$
- 25 estados intermedios de la ronda
  para xx en {00, ..., 24} y para i en {0, ..., 63}. Denotado $T_{xx}[i]$
- 24 rondas como máximo (una ronda aquí descrita)
- palabras de 64 bits de 24 Constantes lota (denotado $X[r]$ donde 'r' es el número de ronda):
  (nota : notación de extremidad grande)

```
X[00]: 0x0000000000000001, X[01]: 0x0000000000008082, X[02]: 0x800000000000808A, X[03]: 0x8000000080008000,
X[04]: 0x000000000000808B, X[05]: 0x0000000080000001, X[06]: 0x8000000080008081, X[07]: 0x8000000000008009,
X[08]: 0x000000000000008A, X[09]: 0x0000000000000088, X[10]: 0x0000000080008009, X[11]: 0x000000008000000A,
X[12]: 0x000000008000808B, X[13]: 0x800000000000008B, X[14]: 0x8000000000008089, X[15]: 0x8000000000008003,
X[16]: 0x8000000000008002, X[17]: 0x8000000000000080, X[18]: 0x000000000000800A, X[19]: 0x800000008000000A,
X[20]: 0x8000000080008081, X[21]: 0x8000000000008080, X[22]: 0x0000000080000001, X[23]: 0x8000000080008008
```

- La puerta XOR es $\oplus$
- No x es $\overline{x}$
- La puerta AND es $\wedge$
- La puerta OR es $\vee$
- Módulo es $\%$

Índices de Matriz de Estado Interno (ISM denotado)

```
[00]  [ 0, 4, 9, 14, 19, 24, 1, 6, 11, 16, 21]
[01]  [ 1, 0, 5, 10, 15, 20, 2, 7, 12, 17, 22]
[02]  [ 2, 1, 6, 11, 16, 21, 3, 8, 13, 18, 23]
[03]  [ 3, 2, 7, 12, 17, 22, 4, 9, 14, 19, 24]
[04]  [ 4, 3, 8, 13, 18, 23, 0, 5, 10, 15, 20]
[05]  [ 5, 4, 9, 14, 19, 24, 1, 6, 11, 16, 21]
[06]  [ 6, 0, 5, 10, 15, 20, 2, 7, 12, 17, 22]
[07]  [ 7, 1, 6, 11, 16, 21, 3, 8, 13, 18, 23]
[08]  [ 8, 2, 7, 12, 17, 22, 4, 9, 14, 19, 24]
[09]  [ 9, 3, 8, 13, 18, 23, 0, 5, 10, 15, 20]
[10]  [10, 4, 9, 14, 19, 24, 1, 6, 11, 16, 21]
[11]  [11, 0, 5, 10, 15, 20, 2, 7, 12, 17, 22]
[12]  [12, 1, 6, 11, 16, 21, 3, 8, 13, 18, 23]
[13]  [13, 2, 7, 12, 17, 22, 4, 9, 14, 19, 24]
[14]  [14, 3, 8, 13, 18, 23, 0, 5, 10, 15, 20]
[15]  [15, 4, 9, 14, 19, 24, 1, 6, 11, 16, 21]
[16]  [16, 0, 5, 10, 15, 20, 2, 7, 12, 17, 22]
[17]  [17, 1, 6, 11, 16, 21, 3, 8, 13, 18, 23]
[18]  [18, 2, 7, 12, 17, 22, 4, 9, 14, 19, 24]
[19]  [19, 3, 8, 13, 18, 23, 0, 5, 10, 15, 20]
[20]  [20, 4, 9, 14, 19, 24, 1, 6, 11, 16, 21]
[21]  [21, 0, 5, 10, 15, 20, 2, 7, 12, 17, 22]
[22]  [22, 1, 6, 11, 16, 21, 3, 8, 13, 18, 23]
[23]  [23, 2, 7, 12, 17, 22, 4, 9, 14, 19, 24]
[24]  [24, 3, 8, 13, 18, 23, 0, 5, 10, 15, 20]
```

Índices de Matriz de Estado Medio (MSM denotado)

```
[  0,  6, 12 ]
[  6, 12, 18 ]
[ 12, 18, 24 ]
[ 18, 24,  0 ]
[ 24,  0,  6 ]
[  3,  9, 10 ]
[  9, 10, 16 ]
[ 10, 16, 22 ]
[ 16, 22,  3 ]
[ 22,  3,  9 ]
[  1,  7, 13 ]
[  7, 13, 19 ]
[ 13, 19, 20 ]
[ 19, 20,  1 ]
[ 20,  1,  7 ]
[  4,  5, 11 ]
[  5, 11, 17 ]
[ 11, 17, 23 ]
[ 17, 23,  4 ]
[ 23,  4,  5 ]
[  2,  8, 14 ]
[  8, 14, 15 ]
[ 14, 15, 21 ]
[ 15, 21,  2 ]
[ 21,  2,  8 ]
```

Desplazamientos de Cambio de Estado Medio (MSS denotado)

```
[  0, 44, 43 ]
[ 44, 43, 21 ]
[ 43, 21, 14 ]
[ 21, 14,  0 ]
[ 14,  0, 44 ]
[ 28, 20,  3 ]
[ 20,  3, 45 ]
[  3, 45, 61 ]
[ 45, 61, 28 ]
[ 61, 28, 20 ]
[  1,  6, 25 ]
[  6, 25,  8 ]
[ 25,  8, 18 ]
[  8, 18,  1 ]
[ 18,  1,  6 ]
[ 27, 36, 10 ]
[ 36, 10, 15 ]
[ 10, 15, 56 ]
[ 15, 56, 27 ]
[ 56, 27, 36 ]
[ 62, 55, 39 ]
[ 55, 39, 41 ]
[ 39, 41,  2 ]
[ 41,  2, 62 ]
[  2, 62, 55 ]
```

- ETAPA 1: computar estado interno intermedio $T_{xx}[i]$

$$\forall i \in [0,63],\ \forall xx \in [0,24],\ T_{xx}[i] = \bigoplus_{j=0}^{5} M_{ISM[xx][j]}[i] \bigoplus_{j=6}^{10} M_{ISM[xx][j]}[(i-1)\%64]$$

- ETAPA 2: computar estados internos al final de la ronda $M_{xx}+[i]$

$$\forall i \in [0,63],\ M^{+}_{00}[i] = T_0[i] \oplus (\overline{T_6[(i-44)\%64]} \wedge T_{12}[(i-43)\%64]) \oplus X_r$$

$$\forall i \in [0,63],\ \forall xx \in [1,24]\ M^{+}_{xx}[i] = T_{A_0}[(i-B_0)\%64] \oplus (\overline{T_{A_1}[(i-B_1)\%64]} \wedge T_{A_2}[(i-B_2)\%64])$$

donde $A_y = MSM[xx][y]$ y $B_y = MSS[xx][y]$

Al final de la ronda final, solo los primeros *n* bits del estado interno se consideran *digest*, dicho número de bits *n* depende de la velocidad binaria r_b y de la capacidad *c* de la esponja, *n* siendo igual, por ejemplo, a 512 en el caso donde r_b = 576 y *c* = 1024. Una particularidad de SHA-3 es que la función de troceado puede invertirse fácilmente a partir de un estado interno si se conocen todos los bits, gracias a un procedimiento de complejidad polinomial.

Solución algebraica de funciones de troceado criptográficas

La solución algebraica de la función de troceado que ha generado la palabra troceada permite invertir dicha función de troceado y recuperar los datos originales. Ello puede llevarse a cabo gracias a una codificación Booleana de la primitiva de función de troceado y un resolutor algebraico dedicado o genérico.

La solución algebraica de las funciones de troceado es ventajosamente una solución de SATisfacibilidad (SAT) booleana. Este tipo de solución algebraica es un problema NP-completo conocido, según se describe en los artículos de A. Biere y otros "*Handbook of Satisfiability*", volumen 185 de *Frontiers in Artificial Intelligence and Applications*, IOS Press, febrero 2009, y de Stephen A. Cook "*The complexity of theorem proving procedures*", en *ACM Symposium on Theory of Computing*, páginas 151-158, 1971.

La solución de SATisfacibilidad consiste en determinar si una expresión booleana *F* tiene al menos una asignación de valor de verdad {VERDADERO, FALSO}, también llamada una interpretación, para su variable de modo que es verdadera. *F* se considera, preferiblemente, una fórmula CNF ("Forma Normal Conjuntiva") que puede definirse como un conjunto de cláusulas, interpretadas como un conjunto, donde una cláusula es un conjunto de literales, interpretados como una disyunción.

Más precisamente, dejemos que v = {v_1, ..., v_n} sea un conjunto de *n* variables booleanas. Una variable booleana firmada se llama un *literal.* Uno puede denotar v_i y $\overline{v_i}$ los literales positivos y negativos que se refieren a la variable v_i respectivamente. El literal v_i, respectivamente $\overline{v_i}$, es VERDADERO, también dicho "*satisfecho*", si la variable v_i correspondiente se asigna a VERDADERO, respectivamente FALSO. Los literales se asocian comúnmente a operadores lógicos AND y OR, respectivamente denotados por ∧ y ∨. Una disyunción de literales se denota, por ejemplo, por $v_1 \vee \overline{v_2} \vee v_3 \vee v_4$.

Una cláusula se satisface, en general, si al menos uno de sus literales se satisface, la expresión *F* satisfaciéndose si todas sus cláusulas se satisfacen. En otras palabras, si existe una asignación de *V* en {VERDADERO, FALSO} como, por ejemplo, para llevar a cabo la expresión *F* VERDADERO, *F* es dicha SAT, y de lo contrario UNSAT.

El criptoanálisis lógico consiste en un proceso de dos etapas que usa un modelado asociado a una solución algebraica para el modelo *and*. Ello puede llevar al ataque de un criptosistema, como se explica en los artículos de Fabio Massacci "*Using walk-SAT and rel-sat for cryptographic key search*", en IJCAI, páginas 290-295, 1999, y de Fabio Massacci y Laura Marraro "*Logical cryptanalysis as a SAT problem*", *J.Autom.Reasoning*, páginas 165-203, 2000, en los tres artículos de Florian Legendre y otros "*Encoding hash functions as a SAT problem*", en ICTAI, páginas 916-921, 2012, "*Inverting thanks to SAT solving - an application on reduced-step MD**", en SECRYPT, páginas 339-344, 2012, y "*From a logical approach to internal states of hash functions - how SAT problem can help to understand SHA-* and MD**"; en SECRYPT, 2013, y en la Tesis de Máster de Vegard Nossum "*SAT based preimage attacks on SHA-1*", 2012.

El artículo de Ilya Mironov y Lintao Zhang "*Applications of SAT solvers to cryptanalysis of hash functions*", en SAT, páginas 102-115, 2006, presenta un resultado interesante sobre la aplicación del criptoanálisis lógico a funciones de troceado criptográficas. En el presente artículo, los autores suponen que la ejecución de un ataque criptoanalítico debe mejorarse mediante el uso de un formalismo lógico para expresar funciones complejas. Estas modelan todo un trayecto diferencial para las funciones de troceado conocidas MD* y SHA-* en un circuito booleano y obtienen resultados concluyentes mediante el uso de algunos de los resolutores SAT conocidos.

Solución de SATisfacibilidad de SHA-3

Modelar una función de troceado como una fórmula SAT puede llevarse a cabo gracias a herramientas automáticas como, por ejemplo, CryptLogVer descrita en el artículo de Pawel Morawiecki y Marian Srebrny "*A SAT-based preimage analysis of reduced Keccak hash functions*", en *Inf. Process. Letters*, 113(10-11):392-397, 2013, o mediante un enfoque hecho a mano. El uso de un enfoque hecho a mano permite obtener un modelado resultante optimizado, en términos de número de cláusulas y variables implicadas.

La codificación de la función de troceado SHA-3 como una fórmula SAT requiere, de manera ventajosa, considerar cada bit de cada palabra implicada en la primitiva original como una variable. Cada función interna, también correspondiente a un circuito lógico, se asocia a un conjunto de cláusulas.

Una solución directa de SATisfacibilidad de la función de troceado Keccak para una sola ronda, con palabras de 64 bits para estados internos, puede expresarse como:

$$\forall i \in [0,63] \bigwedge_{xx=0}^{24} \left(\bigoplus_{j=0}^{5} M_{ISM[xx][j]}[i] \bigoplus_{j=6}^{10} M_{ISM[xx][j]}[(i-1)\%64] \bigoplus \overline{T_{xx}[i]} \right)$$

$$\forall i \in [0,63] \bigwedge \left(T_{00}[i] \bigoplus E_{00}[i] \bigoplus \overline{M_{00}^{+}[i]} \bigoplus X_r[i] \right)$$

$$\forall i \in [0,63] \bigwedge_{xx=1}^{24} \left(T_{MSM[xx][0]}[(i - MSS[xx][0])\%64] \bigoplus E_{xx}[i] \bigoplus \overline{M_{xx}^{+}[i]} \right)$$

$$\forall i \in [0,63] \bigwedge_{xx=0}^{24} \left(T_{MSM[xx][1]}[(i - MSS[xx][1])\%64] \bigvee \overline{T_{MSM[xx][2]}[(i - MSS[xx][2])\%64]} \bigvee E_{xx}[i] \right)$$

$$\forall i \in [0,63] \bigwedge_{xx=0}^{24} \left(\overline{T_{MSM[xx][1]}[(i - MSS[xx][1])\%64]} \bigvee \overline{E_{xx}[i]} \right)$$

$$\forall i \in [0,63] \bigwedge_{xx=0}^{24} \left(T_{MSM[xx][2]}[(i - MSS[xx][2])\%64] \bigvee \overline{E_{xx}[i]} \right)$$

con los 25 estados internos denotados $M_{xx}[i]$, $T_{xx}[i]$ una palabra intermedia de 64 bits llamada *"Theta"*, $E_{xx}[i]$ una palabra de 64 bits llamada "*equivalency*", y *r* el número de ronda.

La codificación SAT de la función de troceado SHA-3 según la invención puede comprender 869 120 cláusulas y 92 160 variables. Dichos valores pueden variar según la técnica de codificación implementada.

Recuperación de datos

Durante una conexión previa al servidor, el cliente ha registrado e ingresado un dato que se preservará. Dicho dato puede ser la contraseña que se necesita para la autenticación en el servidor, o cualquier dato que el cliente quiera preservar como, por ejemplo, documentos administrativos, recibos, contratos, fotografías, audios o vídeos, etc.

Una conexión segura se establece, preferiblemente, entre el cliente y el servidor como, por ejemplo, una conexión SSL o conexión TLS (*"Capa de Enchufe Seguro"* o *"Seguridad de la Capa de Transporte"*).

Una aplicación web puede permitir al cliente ingresar su información, preferiblemente su dirección de correo electrónico, nombre de cuenta y contraseña.

En caso de que los datos que deben preservarse sean diferentes de dicha contraseña, puede invitarse al cliente a que ingrese los datos en una ubicación dedicada de la aplicación web.

Los datos que deben preservarse se trocean, de manera ventajosa, por la función de troceado *HF*, siendo preferiblemente la función de troceado SHA-3 previamente descrita, con el fin de generar la palabra troceada almacenada en la memoria a la cual el servidor tiene acceso.

La memoria puede ser una memoria interna del servidor o una remota.

El almacenamiento de una huella digital de los datos permite comprobar su integridad sin conocerla, gracias a la función de troceado unidireccional. También puede usarse, de forma algebraica, para reconstruir los datos cuando se asocian a una clave de seguridad dada de la información de troceado inverso.

La función de procesamiento que modifica los datos que deben preservarse, con el fin de formar datos de entrada de la capacidad predefinida C_p, puede corresponder a una concatenación con al menos una esponja inicial *ISC*.

La esponja inicial puede muestrearse de forma aleatoria, y comprende, por ejemplo, solamente bits iguales a 0.

Además de la concatenación con dicha esponja inicial, los datos pueden concatenarse con una palabra de relleno, con el fin de alcanzar la velocidad binaria r_b previamente definida, correspondiente a la diferencia entre la capacidad predefinida C_p y la capacidad de la esponja inicial c.

La palabra troceada H_b puede expresarse como:

$$H_b = HF\,(Entrada1) = HF\,(\,Datos \;||\; Relleno \;||\; ISC).$$

Las funciones de procesamiento usadas para modificar los datos para formar datos de entrada y generar la palabra troceada H_b y la clave de seguridad H_c pueden ser idénticas, el primer dato de entrada *Entrada1* y el segundo dato de entrada *Entrada2* siendo idénticos. Los datos que deben preservarse se concatenan, por lo tanto, de forma ventajosa, con una palabra de relleno, para alcanzar la velocidad binaria r_b, y con una esponja inicial de una capacidad igual a c, con el fin de además generar la clave de seguridad H_c mediante el troceado del resultado de la concatenación, teniendo la capacidad predefinida C_p.

Con el fin de generar la clave de seguridad según la invención, la función de troceado *HF* se modifica, de manera ventajosa, para formar la función de troceado *HF*,* configurada para conservar todos los bits del último estado interno calculado a partir de los datos que deben preservarse como entrada de la función de troceado *HF,* y dividirla en dos partes, preferiblemente la palabra troceada correspondiente a un vector de 512 bits menos significativos de un estado interno de 1600 bits, y la clave de seguridad correspondiente a un vector de 1088 bits menos significativos de un estado interno de 1600 bits.

La función de troceado modificada *HF** se configura para preservar todas las especificaciones estándares de la función de troceado *HF,* pero se configura también para computar información adicional que puede llevar a una clave de seguridad que permite reconstruir los datos de texto claro cuando se combinan con la palabra troceada computada por la función de troceado *HF.*

La generación de la clave de seguridad H_c, ilustrada en la Figura 4 para una capacidad de estado interno de 1600 bits, puede expresarse como:

$$H_c = HF*\,(Entrada2) = HF*\,(\,Datos \;||\; Relleno \;||\; ISC).$$

La capacidad C_{hc} de la clave de seguridad H_c, también llamada la capacidad de la esponja final, es igual a la diferencia entre la capacidad predefinida C_p y la capacidad C_{hb} de la palabra troceada H_b : $C_{hc} = C_p - C_{hb}$.

El cliente puede generar la clave de seguridad.

En el presente caso, un software dedicado para recuperar los datos puede proponerse al cliente. El cliente puede descargarlo e instalarlo en la máquina electrónica que está usando. Mientras abre el software, una ventana puede visualizarse en la pantalla de la máquina electrónica que el cliente está usando, e invitar al cliente a ingresar los datos en una área de texto. El software puede configurarse para ejecutar la función de troceado modificada para generar la clave de seguridad. Ello ofrece una mejor seguridad, porque la clave de seguridad no necesita enviarse al cliente por el servidor y, por consiguiente, este último nunca la conoce.

En una realización diferente, la clave de seguridad puede generarse por el servidor y enviarse al cliente, y no almacenarse en la memoria a la cual el servidor tiene acceso.

Cuando el cliente ha perdido u olvidado sus datos, o quiere acceder a ellos, puede enviar una solicitud al servidor para recuperarlos. El cliente puede ingresar su nombre y/o cuenta de correo electrónico en una página de un sitio web que ayuda a recuperar los datos perdidos. Un correo electrónico que contiene un enlace a una aplicación web puede enviarse al cliente, y establecer una conexión segura entre el cliente y el servidor. Dicho correo electrónico no contiene información sensible y podría interceptarse sin comprometer la seguridad de los datos.

El servidor puede enviar al cliente la palabra troceada, por ejemplo en un correo electrónico, de modo que el cliente concatena dicha palabra troceada y la clave de seguridad almacenada en una memoria de la máquina electrónica que está usando. Al llevar esto a cabo, el cliente puede, de forma ventajosa, reconstruir un bloque de bits que tiene la capacidad predefinida, correspondiente al último estado interno computado a partir de los datos que deben preservarse como entrada de la función de troceado. El cliente puede entonces computar los datos perdidos. El software instalado en la máquina electrónica que el cliente está usando puede configurarse para ejecutar la función de troceado inversa.

En una realización diferente, el cliente envía la clave de seguridad al servidor, con el fin de concatenarla con la palabra troceada, de modo que el servidor computa los datos perdidos.

Gracias a la solución algebraica de la función de troceado, la función de troceado inversa HF_*^{-1} aplicada a dicha secuencia concatenada permite recuperar el bloque de entrada completo, incluida la capacidad de la esponja inicial y, por consiguiente, los datos perdidos:

$$Datos \,||\, Relleno \,||\, ISC = HF_*^{-1}(H_b \,||\, H_c).$$

Los datos recuperados pueden visualizarse en la pantalla de la máquina electrónica que el cliente está usando, o almacenarse en la memoria de dicha máquina, especialmente en caso de que los datos sean un archivo. Los datos recuperados no se almacenan, de forma ventajosa, en la memoria a la cual el servidor tiene acceso.

La máquina electrónica que el cliente está usando puede ser cualquier dispositivo dedicado, por ejemplo un ordenador personal, un teléfono inteligente, un reloj inteligente, una tablet digital o un dispositivo de acceso integrado para Internet o televisión.

Producto de programa de ordenador

Otro objeto de la invención es un producto de programa de ordenador que comprende instrucciones que pueden leerse tanto por un servidor como por un cliente, dichas instrucciones controlan el funcionamiento de dicho servidor y cliente, de modo que, para recuperar un dato que debe preservarse, ingresado por el cliente durante una conexión previa al servidor, el servidor tiene acceso a una memoria que comprende una palabra troceada generada mediante la aplicación de una función troceada a un primer dato de entrada de una capacidad predefinida, el primer dato de entrada correspondiendo al dato que debe preservarse modificado por una función de procesamiento, la capacidad de la palabra troceada siendo más baja que dicha capacidad predefinida, una clave de seguridad del cliente habiéndose generado mediante aplicación de una función de troceado a un segundo dato de entrada de una capacidad predefinida, el segundo dato de entrada correspondiendo al dato que debe preservarse modificado por una función de procesamiento, la capacidad de la clave de seguridad siendo igual a la diferencia entre dicha capacidad predefinida y la capacidad de la palabra troceada, la clave de seguridad no almacenándose en la memoria a la cual el servidor tiene acceso:

- después de una solicitud del cliente de recuperar los datos que deben preservarse, la palabra troceada y la clave de seguridad se concatenan con el fin de alcanzar dicha capacidad predefinida, y

- una función de troceado inversa, mediante el uso de una solución algebraica de la función de troceado habiendo generado la palabra troceada, se aplica a la concatenación de dicha palabra troceada y clave de seguridad, con el fin de recuperar los datos que deben preservarse.

Todas las características definidas para el método para recuperar un dato que debe preservarse se aplican al producto de programa de ordenador.

La invención se comprenderá mejor al leer la siguiente descripción detallada de realizaciones a modo de ejemplo no restrictivas de aquella y al examinar los dibujos anexos en los cuales:

- la Figura 1A, previamente descrita, ilustra un marco general para el registro de un cliente en un servidor mediante el uso de un método de estado de la técnica;

- la Figura 1B, previamente descrita, ilustra un marco general para la autenticación de un cliente en un servidor;

- la Figura 2, previamente descrita, ilustra un marco general para la recuperación de una contraseña perdida según el estado de la técnica;

- la Figura 3, previamente descrita, es un proceso de troceado de una secuencia de entrada mediante el uso de la función SHA-3;

- la Figura 4, previamente descrita, ilustra la generación de la clave de seguridad según la invención;

- la Figura 5 ilustra el registro de un cliente en un servidor que implica una clave de seguridad según la invención;

- la Figura 6 ilustra una variante del registro de un cliente en un servidor según la invención; y

- la Figura 7 ilustra un marco general para la recuperación de un dato perdido según la invención.

El registro de un cliente en un servidor que implica una clave de seguridad según la invención se muestra en la Figura 5.

En una etapa 11, el cliente se conecta al servidor, mediante el uso, preferiblemente, de una conexión segura. El servidor reconoce al cliente en una etapa 12 y el cliente ingresa un ID de inicio de sesión y un *Dato* que debe preservarse, en las etapas 13 y 14. El *Dato* que debe preservarse puede ser una contraseña usada para la autenticación del cliente en el servidor, o cualquier otro dato que se quiera preservar.

En una etapa 15, el servidor genera una palabra troceada H_b mediante la aplicación de una función de troceado *HF* a un primer dato de entrada *Entrada1*, correspondiente al *Dato* que debe preservarse modificado por una función de procesamiento con el fin de alcanzar una capacidad predefinida C_p. Según se explica previamente, con el fin de formar los datos de entrada *Entrada1,* el *Dato* se concatena de forma ventajosa con una palabra de *Relleno* que comprende uno o varios bits, y con una esponja inicial *ISC,* que tiene una capacidad de esponja inicial *c*. El número de bits de la palabra de Relleno se elige como, por ejemplo, para alcanzar una velocidad binaria r_b correspondiente a la diferencia entre la capacidad predefinida C_p y la capacidad de la esponja inicial *c*.

$$H_b = HF\,(\,Datos\,||\quad Relleno\quad||\,ISC).$$

En una etapa 16, en el ejemplo ilustrado, el cliente genera una clave de seguridad H_c mediante la aplicación de la función de troceado modificada HF_* a un segundo dato de entrada *Entrada2,* correspondiente a los *Datos* que deben preservarse modificados por la función de procesamiento:

$$H_c = HF_*\,(\,Datos\,||\quad Relleno\quad||\,ISC).$$

El primer y segundo datos de entrada *Entrada1* y *Entrada2* son, de forma ventajosa, idénticos.

La capacidad C_{hc} de la clave de seguridad H_c es igual a la diferencia entre la capacidad predefinida C_p y la capacidad C_{hb} de la palabra troceada H_b : $C_{hc} = C_p - C_{hb}$.

En la realización ilustrada, la función de troceado HF_* es una función de troceado SHA-3, mediante el uso del algoritmo Keccak, y modificada como, por ejemplo, para conservar todos los bits del último estado interno computado a partir de los *Datos* que deben preservarse como entrada de la función de troceado SHA-3, y dividirla en dos partes.

La capacidad predefinida C_p es, por ejemplo, igual a 1600 bits, la capacidad de los datos siendo igual a 576 bits y, por consiguiente, la capacidad *c* de la esponja inicial siendo igual a 1024 bits, la capacidad C_{hb} de la palabra troceada es igual a 512 bits, y la única C_{hc} de la clave de seguridad es igual a 1088 bits.

En la realización diferente que se muestra en la Figura 6, la clave de seguridad H_c se genera en el servidor, en una etapa 23, y se envía al cliente, en una etapa 24, y no se almacena en la memoria 20 a la cual el servidor tiene acceso. La contraseña ingresada por un cliente en una etapa 21 de registro se trocea, como se describe previamente, y se almacena, en una etapa 22, en la memoria 20 a la cual el servidor tiene acceso.

La Figura 7 ilustra las principales etapas de un ejemplo para la recuperación de un dato perdido según la invención.

Después de una solicitud del cliente para recuperar los *Datos* que deben preservarse, en una etapa 31, y su identificación, en una etapa 32, mediante el envío de su ID de inicio de sesión, el servidor envía al cliente la palabra troceada H_b, en una etapa 33, de modo que el cliente concatena dicha palabra troceada y la clave de seguridad H_c almacenada en una memoria de la máquina electrónica que está usando con el fin de alcanzar dicha capacidad predefinida C_p.

Como se ha descrito previamente, en una etapa 34, una función de troceado inversa HF_*^{-1}, mediante el uso de una solución algebraica de la función de troceado HF_*, se aplica a la concatenación de dicha palabra troceada H_b y clave de seguridad H_c, con el fin de recuperar los *Datos* que deben preservarse:

$$Datos\,||\quad Relleno\quad||\,ISC = HF_*^{-1}(H_b\,||\,H_c).$$

En una realización diferente, que no se muestra, el cliente envía la clave de seguridad H_c al servidor, de modo que este último puede concatenarla con la palabra troceada H_b con el fin de recuperar los *Datos*.

Especialmente en la realización donde el cliente genera la clave de seguridad H_c y aplica la función de troceado inversa HF_*^{-1}, un software dedicado para recuperar los datos se propone, de forma ventajosa, al cliente, quien lo descarga e instala en la máquina electrónica que está usando, el software configurándose para ejecutar la función de troceado modificada HF_* para generar la clave seguridad H_c y la función de troceado inversa HF_*^{-1}.

Según se explica previamente, la solución algebraica de la función de troceado HF^* es, de forma ventajosa, una solución de SATisfacibilidad.

La ejecución para la solución de SATisfacibilidad de la parte de troceado SHA-3* del protocolo de recuperación de datos según la invención puede ser de entre 7 segundos a 56 segundos, mejor entre 9 segundos a 15 segundos, siendo igual, por ejemplo, a casi 10 segundos.

La invención no se limita a los ejemplos que se han descrito recién. En particular, las características de las realizaciones ilustradas pueden combinarse dentro de realizaciones que no se ilustran.

Otra solución algebraica diferente de la solución de SATisfacibilidad puede usarse como, por ejemplo, técnicas de razonamiento automático, metaheurística, técnicas de solución de álgebra finita o bases de Gröbner.

El método para recuperar un dato que debe preservarse según la invención y, según se define más arriba, puede usarse con el fin de evitar la circulación de contraseñas de texto claro en una red. Cuando un cliente inicia sesión, ingresa su contraseña, la cual se envía en texto claro al servidor del proveedor a través de la red. Ello podría evitarse mediante la computación, en el lado de cliente, de la clave de seguridad, también llamado *"troceado inverso"* o, en la presente solicitud, *"contraseña de sombra*", y su envío en la red al servidor para que pueda compararla con la contraseña troceada y comprobar la clave de seguridad.

La invención puede usarse también para reforzar la seguridad de la nube mediante la delegación de credenciales. Cuando un nodo, especialmente en la computación de la nube, inicia un trabajo en nombre de un cliente en otros nodos, lo hace sin credenciales. Para evitar dicho agujero de seguridad, cada nodo puede enviar claves de seguridad según la invención para iniciar trabajos en otros nodos y autenticar al cliente. La función de inicio de sesión ocurre solamente en el primer nodo y la lleva a cabo el cliente.

En aplicaciones militares, mediante el mantenimiento de todas las claves de seguridad en un servidor, los Servicios de Inteligencia (SI) pueden tener acceso a cada contraseña sin almacenar datos críticos: las contraseñas troceadas se mantienen solamente en los servidores de los proveedores. La generación de *"troceado inverso"* puede imponerse por ley a todos los proveedores para cada nueva cuenta. De esta manera, nadie, excepto los SI, puede conocer las contraseñas de texto claro de los individuos.

El método según la invención puede implementarse en un dispositivo de acceso integrado para Internet o televisión, en especial con el fin de recuperar contenido preservado en un descodificador de flujo de vídeo para canales de pago según consumo.

La clave de seguridad y/o la palabra troceada pueden también considerarse datos que deben preservarse, y pueden beneficiarse del método según la invención para su propia seguridad y, por consiguiente, almacenarse en varios servidores. Cuando más implique el consenso « *servidor(es) + cliente* » diferentes actores, mejor es la seguridad.

El método según la invención entre un servidor y un cliente para ofrecer una solución segura para recuperar un dato que debe preservarse puede extenderse a un consenso entre varios servidores y uno o varios clientes, además de reforzar la seguridad.

En el presente caso, el método según la invención tiene que reproducirse entre los diferentes servidores. Por ejemplo con dos servidores *srv1*, *srv2* y un cliente, el último tiene en el final {H_{c0}, *ID, IDsrv1*}, el primer servidor *srv1* tiene {$H_{c1}(H_{c0})$, *ID', IDsrv2*} y el segundo servidor *srv2* tiene {H_{b2}, *ID'*}, $H_{c1}(H_{c0})$ siendo una clave de seguridad intermedia generada mediante la aplicación de la función de troceado a la clave de seguridad H_{c0}.

Los datos pueden recuperarse si el primer servidor *srv1*, gracias a un consenso con el segundo servidor *srv2*, lleva a cabo:

$$H_{b1} \,||\, Relleno \,||\, ISC1 = HF^{*-1}(H_{b2} \,||\, H_{c1}(H_{c0})),$$

y si el cliente, gracias a un consenso con el primer servidor *srv1,* lleva a cabo:

$$Datos \,||\, Relleno \,||\, ISC0 = HF^{*-1}(H_{b1} \,||\, H_{c0}). Contraseña + Relleno + ISC = SHA-3^{-1}(HID + FSC1D)$$

Según ello, la invención puede permitir reforzar la seguridad de aplicaciones de « *seguridad digital* », correspondientes al almacenamiento e indexación de datos digitales sensibles, como documentos administrativos, recibos, contratos, fotos, etc. Gracias a la invención, con solamente una contraseña, el cliente puede recuperar

todos los datos de conexión que necesita para acceder a dichos datos sensibles, mediante el almacenamiento de dichos datos de conexión y sin necesidad de tener que confiar en un servicio en línea.

La invención puede usarse para construir un servicio de autenticación totalmente centralizado, gestionado por solamente una contraseña, la información no sensible almacenándose en cualquier otro lugar.

Gracias a dicha técnica, la invención puede permitir ofrecer servicios de certificación de los diferentes sitios web que el cliente está usando, mediante la provisión de una solución contra ataques de *Phishing*, es decir ataques que pretenden robar la identidad de un cliente mediante la recolección de información personal.

Gracias a la invención, las contraseñas débiles como, por ejemplo « azerty », « 12345 » o « 00000 », pueden autorizarse y usarse sin riesgo alguno.

El método según la invención es seguro si el cliente usa solamente una máquina electrónica para acceder a los servicios mediante el uso de dicho método. En el caso donde el cliente usa diferentes máquinas electrónicas, un protocolo de transmisión puede usarse, especialmente mediante el copiado de la información local, es decir las claves de seguridad, de una máquina a otra.

Las expresiones "que comprende(n)" o "que incluye(n)" deben comprenderse como sinónimos de "que comprende(n) al menos un/una/uno" o "que incluye(n) al menos un/una/uno", a menos que se especifique lo contrario.

REIVINDICACIONES

1. El método para recuperar un dato que debe preservarse (*Datos*), ingresado por un cliente durante una conexión previa a un servidor, el servidor teniendo acceso a una memoria (20) que comprende una palabra troceada (H_b) generada mediante la aplicación de una función de troceado *(HF)* a un primer dato de entrada (*Entrada1*) de una capacidad predefinida (C_p), la capacidad siendo el número de bits de la cadena de bits de datos, el primer dato de entrada (*Entrada1*) correspondiendo a los datos que deben preservarse (*Datos*) modificados por una función de procesamiento, la capacidad (C_{hb}) de la palabra troceada (H_b) siendo más baja que dicha capacidad predefinida (C_p), una clave de seguridad (H_c) del cliente habiendo sido generada mediante la aplicación de una función de troceado (HF_*) a un segundo dato de entrada (*Entrada2*) de una capacidad predefinida (C_p), el segundo dato de entrada (*Entrada2*) correspondiendo a los datos que deben preservarse (*Datos*) modificados por una función de procesamiento, la función de troceado (*HF*) modificándose para formar la función de troceado (HF_*), la cual se configura para conservar todos los bits del último estado interno computado a partir de los datos que deben preservarse (*Datos*) como entrada de la función de troceado (*HF*), y dividirla en dos partes, la capacidad (C_{hc}) de la clave de seguridad (H_c) siendo igual a la diferencia entre dicha capacidad predefinida (C_p) y la capacidad (C_{hb}) de la palabra troceada (H_b), la clave de seguridad (H_c) no almacenándose en la memoria (20) a la cual el servidor tiene acceso, el método en donde:

- después de una solicitud del cliente para recuperar los datos que deben preservarse (*Datos*), la palabra troceada (H_b) y la clave de seguridad (H_c) se concatenan con el fin de alcanzar dicha capacidad predefinida (C_p), y

- una función de troceado inversa (HF_*^{-1}), mediante el uso de una solución algebraica de la función de troceado (*HF*) habiendo generado la palabra troceada (H_b), se aplica a la concatenación de dicha palabra troceada (H_b) y clave de seguridad (*Hc*), con el fin de recuperar los datos que deben preservarse (*Datos*).

2. El método según la reivindicación 1, en donde la función de troceado (*HF*) es una función de troceado SHA-3, especialmente mediante el uso del algoritmo Keccak.

3. El método según cualquiera de las reivindicaciones 1 o 2, en donde la solución algebraica de la función de troceado (*HF*) es una solución de SATisfacibilidad.

4. El método según cualquiera de las reivindicaciones precedentes, en donde la función de procesamiento que modifica los datos que deben preservarse (*Datos*), con el fin de formar datos de entrada (*Entrada1, Entrada2)* de la capacidad predefinida (C_p), corresponde a una concatenación con al menos una esponja inicial (*ISC*).

5. El método según la reivindicación precedente, en donde además de concatenarse con dicha esponja inicial *(ISC),* los datos que deben preservarse (*Datos*) se concatenan con una palabra de relleno (*Relleno*), con el fin de alcanzar una velocidad binaria (r_b) correspondiente a la diferencia entre la capacidad predefinida (C_p) y la capacidad (c) de la esponja inicial (*ISC*).

6. El método según cualquiera de las reivindicaciones precedentes, en donde las funciones de procesamiento usadas para modificar los datos (*Datos*) para formar los datos de entrada y generar la palabra troceada (H_b) y la clave de seguridad (H_c) son idénticas, el primer y segundo datos de entrada (*Entrada1, Entrada2*) siendo idénticos.

7. El método según cualquiera de las reivindicaciones 1 a 6, en donde el cliente genera la clave de seguridad (H_c).

8. El método según la reivindicación precedente, en donde el servidor envía al cliente la palabra troceada (H_b), de modo que el cliente concatena dicha palabra troceada (H_b) y la clave de seguridad (H_c) almacenada en una memoria de la máquina electrónica que está usando.

9. El método según la reivindicación precedente, en donde un software dedicado para recuperar los datos (*Datos*) se propone al cliente, quien lo descarga e instala en la máquina electrónica que está usando, el software configurándose para ejecutar la función de troceado modificada *(HF*)* usada para generar la clave de seguridad (H_c) y la función de troceado inversa *(*HF_*^{-1}*)*.

10. El método según cualquiera de las reivindicaciones 1 a 6, en donde la clave de seguridad (H_c) se genera en el servidor y se envía al cliente, y no se almacena en la memoria (20) a la cual el servidor tiene acceso.

11. El método según la reivindicación precedente, en donde el cliente envía la clave de seguridad (H_c) al servidor, con el fin de concatenarla con la palabra troceada (H_b).

12. El método según cualquiera de las reivindicaciones precedentes, en donde la capacidad predefinida (C_p) es igual a 1600 bits, la capacidad (c) de la esponja inicial (*ISC*) siendo igual a 1024 bits, la capacidad (C_{hb}) de la palabra troceada (H_b) es igual a 512 bits, y la única (C_{hc}) de la clave de seguridad (H_c) es igual a 1088 bits.

13. Un producto de programa de ordenador que comprende instrucciones que pueden leerse tanto por un servidor como por un cliente, dichas instrucciones controlan el funcionamiento de dicho servidor y cliente de modo que, para

recuperar un dato que debe preservarse (*Datos*), ingresado por el cliente durante una conexión previa al servidor, el servidor teniendo acceso a una memoria (20) que comprende una palabra troceada (H_b) generada mediante la aplicación de una función de troceado (*HF*) a un primer dato de entrada (*Entrada1*) de una capacidad predefinida (C_p), la capacidad siendo el número de bits de la cadena de bits de datos, el primer dato de entrada (*Entrada1*) correspondiendo a los datos que deben preservarse (*Datos*) modificados por una función de procesamiento, la capacidad (C_{hb}) de la palabra troceada (H_b) siendo más baja que dicha capacidad predefinida (C_p), una clave de seguridad (H_c) del cliente habiéndose generado mediante la aplicación de una función de troceado (HF_*) a un segundo dato de entrada (*Entrada2*) de una capacidad predefinida (C_p), el segundo dato de entrada (*Entrada2*) correspondiendo a los datos que deben preservarse (*Datos*) modificados por una función de procesamiento, la función de troceado (*HF*) modificándose para formar la función de troceado (HF_*), la cual se configura para conservar todos los bits del último estado interno computado a partir de los datos que deben preservarse (*Datos*) como entrada de la función de troceado (*HF*), y dividirla en dos partes, la capacidad (C_{hc}) de la clave de seguridad (H_c) siendo igual a la diferencia entre dicha capacidad predefinida (C_p) y la capacidad (C_{hb}) de la palabra troceada (H_b), la clave de seguridad (H_c) no almacenándose en la memoria (20) a la cual el servidor tiene acceso:

- después de una solicitud del cliente para recuperar los datos que deben preservarse (*Datos*), la palabra troceada (H_b) y la clave de seguridad (H_c) se concatenan con el fin de alcanzar dicha capacidad predefinida (C_p), y

- una función de troceado inversa (HF_*^{-1}), mediante el uso de la solución algebraica de la función de troceado (*HF*) habiendo generado la palabra troceada (H_b), se aplica a la concatenación de dicha palabra troceada (H_b) y clave de seguridad (*Hc*), con el fin de recuperar los datos que deben preservarse (*Datos*).

14. Un producto de programa de ordenador según la reivindicación precedente, en donde, el cliente habiendo generado la clave de seguridad (H_c), el servidor envía al cliente la palabra troceada (H_b), de modo que el cliente concatena dicha palabra troceada (H_b) y la clave de seguridad (H_c) almacenada en una memoria de la máquina electrónica que está usando.

15. Un producto de programa de ordenador según la reivindicación 13, en donde, la clave de seguridad (H_c) habiéndose generado en el servidor, enviada al cliente, y no almacenada en la memoria (20) a la cual el servidor tiene acceso, el cliente envía la clave de seguridad (H_c) al servidor, con el fin de concatenarla con la palabra troceada (H_b).

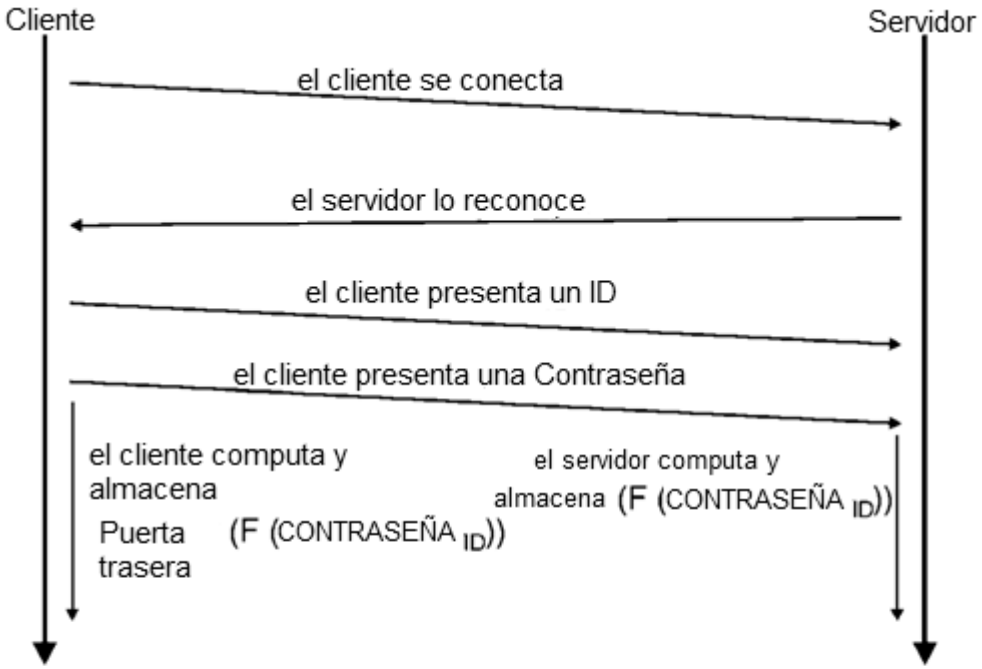

Fig. 1a

Estado de la técnica

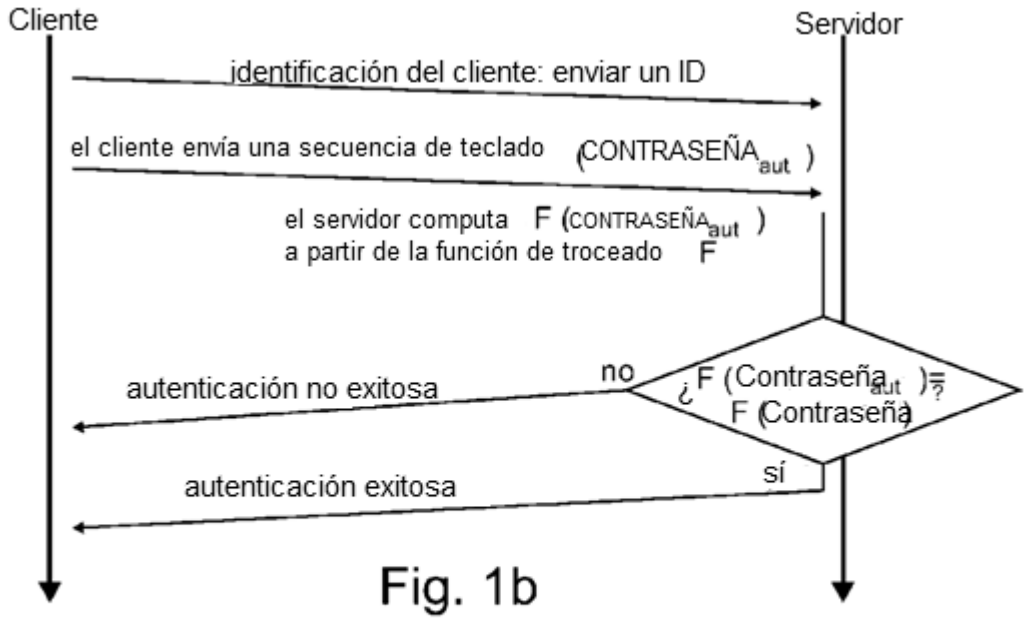

Fig. 1b

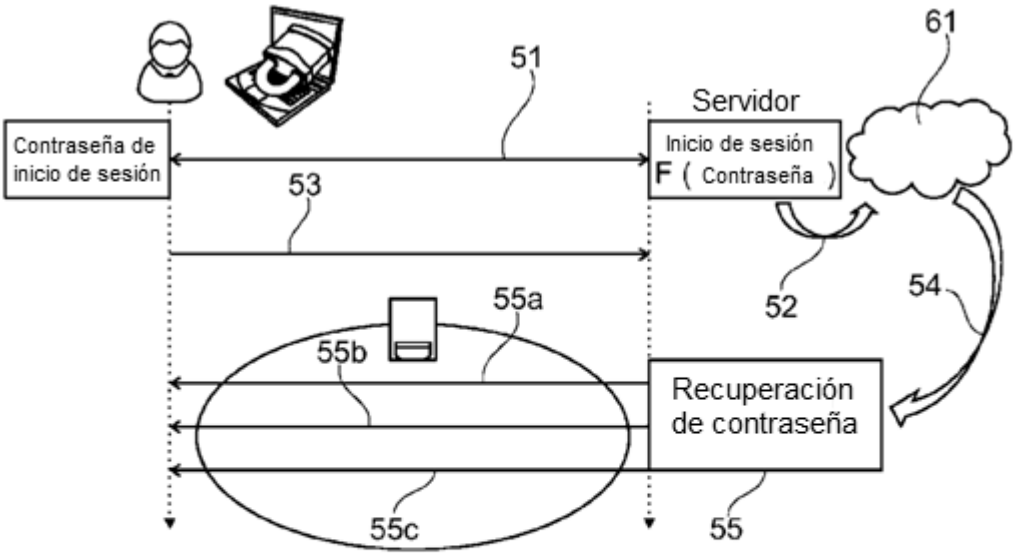

Fig. 2

Estado de la técnica

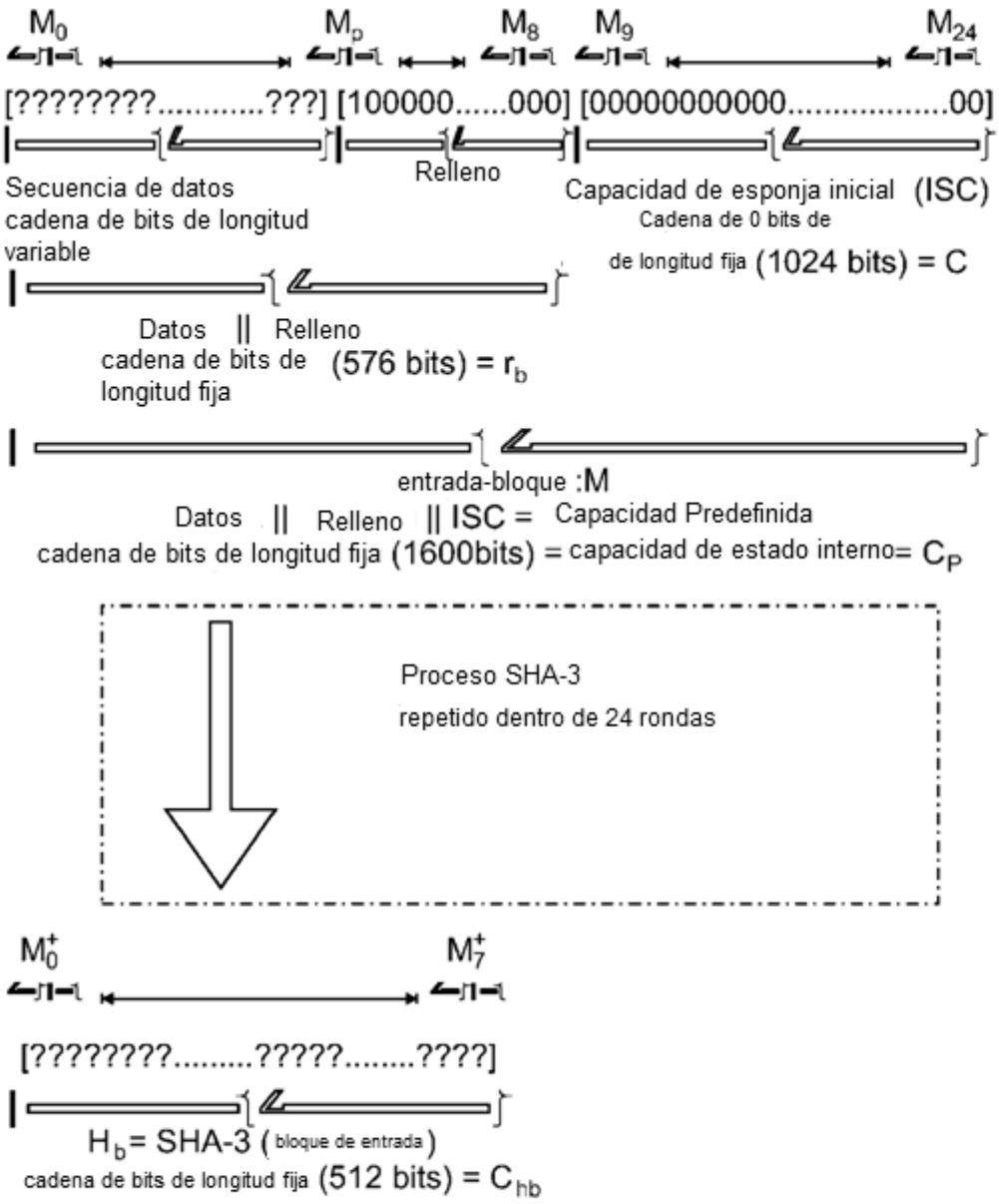
M0
Mp
M8
M9
M24
[????????.....???] [100000.....000] [00000000000.....00]
Relleno
Secuencia de datos
cadena de bits de longitud variable
Capacidad de esponja inicial (ISC)
Cadena de 0 bits de
de longitud fija (1024 bits) = C
Datos || Relleno
cadena de bits de longitud fija (576 bits) = rb
entrada-bloque :M
Datos || Relleno || ISC = Capacidad Predefinida
cadena de bits de longitud fija (1600bits) = capacidad de estado interno= CP
Proceso SHA-3
repetido dentro de 24 rondas
M0+
M7+
[????????.....?????.....????]
Hb= SHA-3 (bloque de entrada)
cadena de bits de longitud fija (512 bits) = Chb

Fig. 3

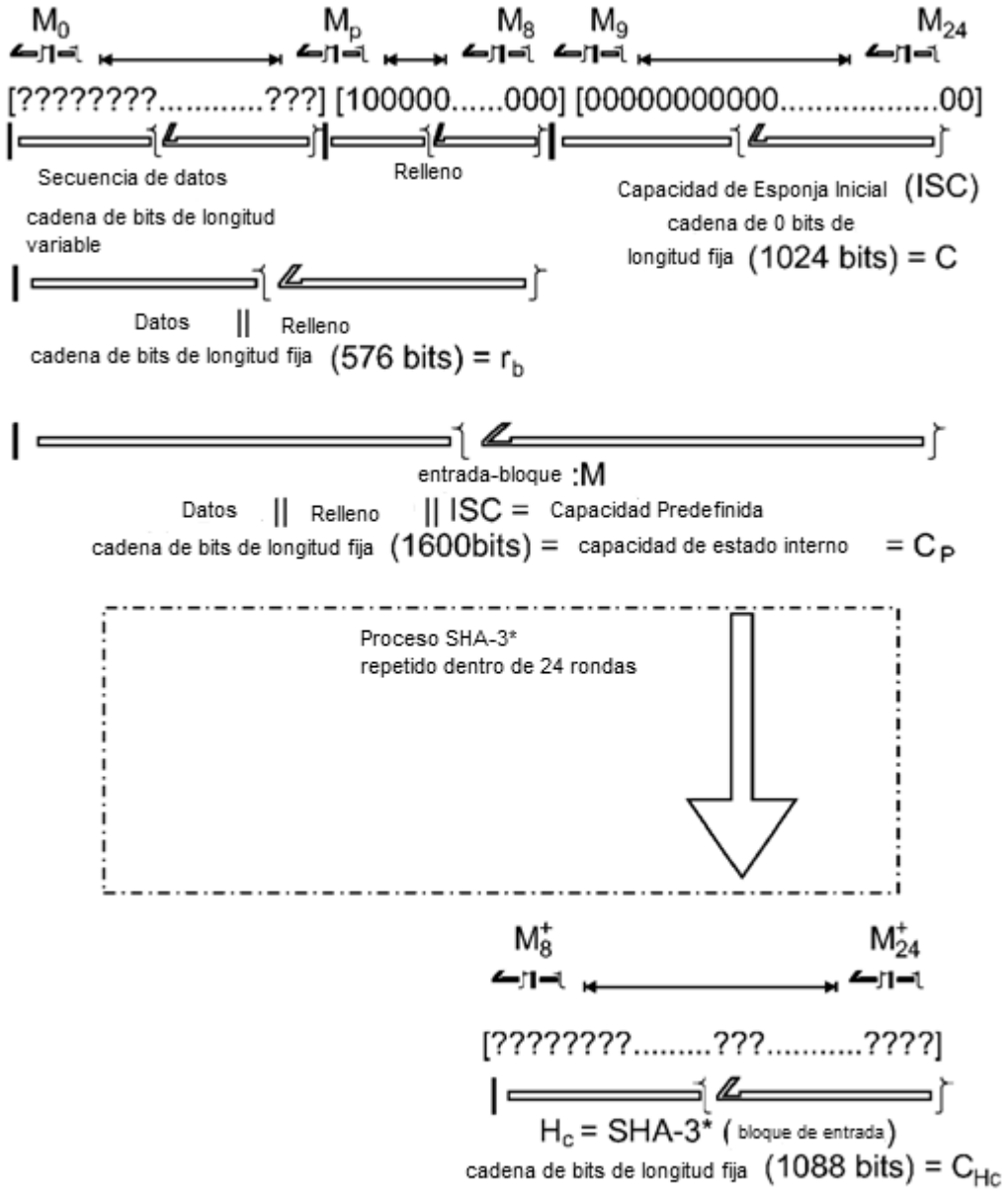
M0
Mp
M8
M9
M24
[????????.....???] [100000.....000] [00000000000.....00]
Secuencia de datos
cadena de bits de longitud variable
Relleno
Capacidad de Esponja Inicial (ISC)
cadena de 0 bits de longitud fija (1024 bits) = C
Datos || Relleno
cadena de bits de longitud fija (576 bits) = rb
entrada-bloque :M
Datos || Relleno || ISC = Capacidad Predefinida
cadena de bits de longitud fija (1600bits) = capacidad de estado interno = CP
Proceso SHA-3*
repetido dentro de 24 rondas
M8+
M24+
[????????.....???.....????]
Hc = SHA-3* (bloque de entrada)
cadena de bits de longitud fija (1088 bits) = CHc

Fig. 4

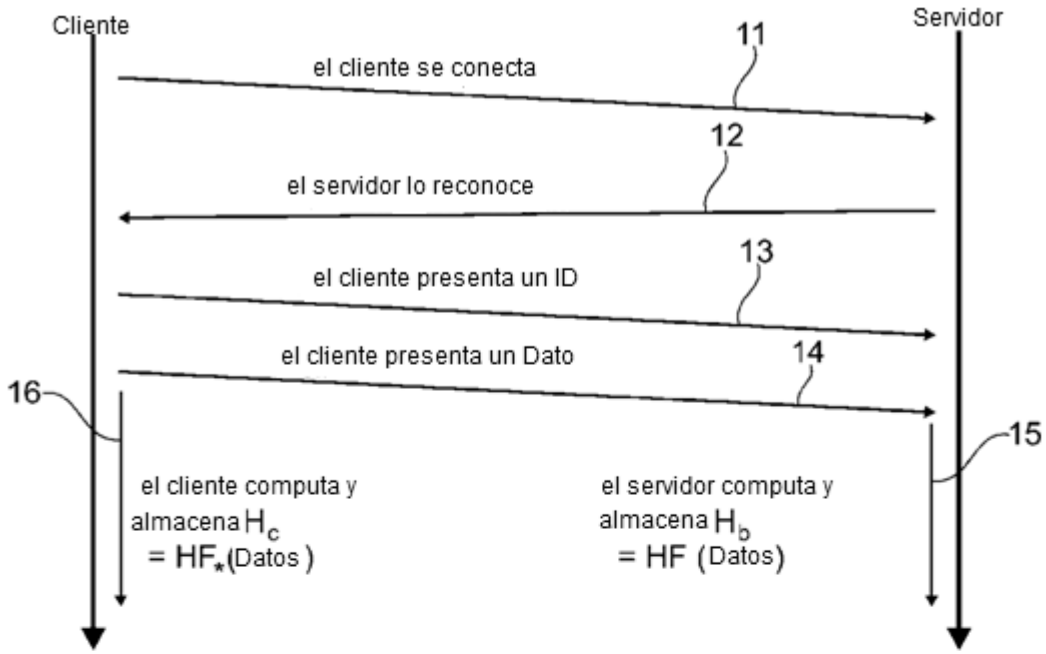
Cliente
Servidor
11
el cliente se conecta
12
el servidor lo reconoce
13
el cliente presenta un ID
14
el cliente presenta un Dato
16
15
el cliente computa y almacena $H_c = HF_*(Datos)$
el servidor computa y almacena $H_b = HF(Datos)$

Fig. 5

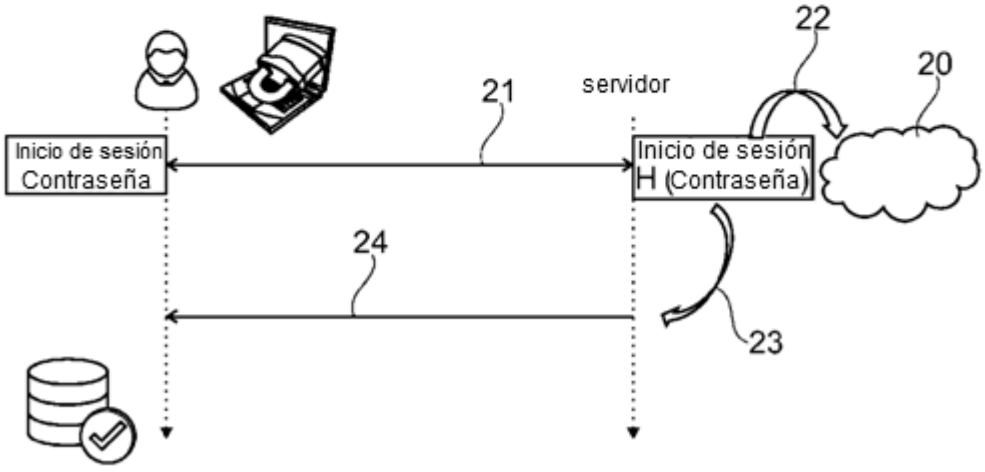
22
20
21
servidor
Inicio de sesión
Contraseña
Inicio de sesión
H (Contraseña)
24
23

Fig. 6

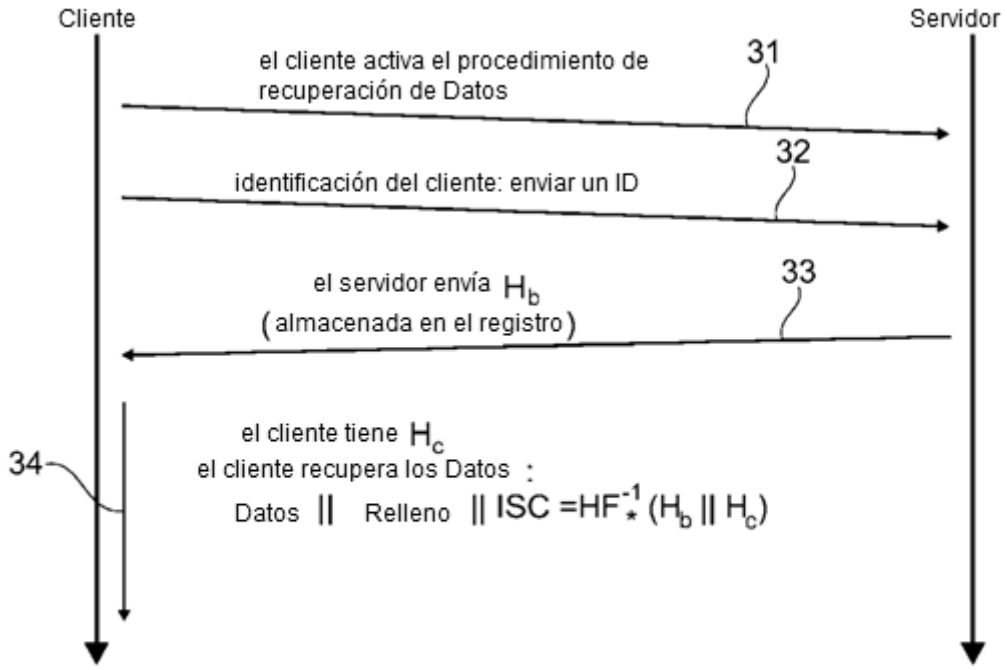
Cliente
Servidor
el cliente activa el procedimiento de recuperación de Datos
31
identificación del cliente: enviar un ID
32
el servidor envía H_b (almacenada en el registro)
33
el cliente tiene H_c
el cliente recupera los Datos :
Datos || Relleno || ISC $=HF_*^{-1}(H_b \| H_c)$
34

Fig. 7