(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 614 946**

(51) Int. CI.:

H04L 9/08 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **16.05.2011 PCT/EP2011/057842**

(87) Fecha y número de publicación internacional: **24.11.2011 WO11144554**

(96) Fecha de presentación y número de la solicitud europea: **16.05.2011 E 11719826 (7)**

(97) Fecha y número de publicación de la concesión europea: **09.11.2016 EP 2572470**

(54) Título: **Procedimiento de obtención de claves de cifrado, terminal, servidor y productos de programas de ordenador correspondientes**

(30) Prioridad:

31.05.2010 FR 1054217
20.05.2010 FR 1053945

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
02.06.2017

(73) Titular/es:

INGENICO GROUP (100.0%)
28-32 Boulevard de Grenelle
75015 Paris, FR

(72) Inventor/es:

BRIER, ERIC y
PEYRIN, THOMAS

(74) Agente/Representante:

CARPINTERO LÓPEZ, Mario

DESCRIPCIÓN

Procedimiento de obtención de claves de cifrado, terminal, servidor y productos de programas de ordenador correspondientes

1 Campo de la invención

La presente invención se refiere al campo de la protección de intercambio de datos entre dos dispositivos.

La presente invención se refiere más particularmente a la protección con la ayuda de claves criptográficas que permiten un cifrado de los datos intercambiados. Los datos intercambiados pueden ser por ejemplo unos datos de autenticación de un usuario tal como un identificador, una contraseña o incluso un código confidencial.

2 Soluciones de la técnica anterior

En un contexto de cifrado de códigos confidenciales o de cifrado de los datos que se refieren a un poseedor de tarjeta (número acceso personal, fecha de caducidad, etc.), el "*American National Standards Institute*" (ANSI) requiere el uso de un esquema de gestión de claves criptográficas de acuerdo con la norma ANSI X9.24. Este documento propone tres procedimientos posibles para el cifrado, cuyo procedimiento "Derive Unique Key Per Transaction" (DUKPT, literalmente en español "Deducción de clave única por transacción"). Este procedimiento es unánimemente reconocido como el más desarrollado en términos de seguridad.

Esta técnica DUKPT de gestión de claves criptográficas está cada vez más implementada y se impone naturalmente como la norma para cifrar unos datos que se refieren a un poseedor de tarjeta (código confidencial, número de acceso personal, fecha de caducidad, etc.).

Según esta técnica DUKPT, en cada transacción, se deduce una nueva clave en un terminal para cifrar los datos sensibles, datos que se enviarán al mismo tiempo que un contador que permite a continuación al servidor encontrar la clave utilizada por el terminal (por supuesto, se considera que se comparte un secreto entre el terminal y el servidor una vez inicializado el sistema).

Para más precisiones a propósito de la técnica DUKPT, se podrá hacer referencia al documento ANSI X9.24.

Este procedimiento presenta dos ventajas en términos de seguridad. En primer lugar, todas las claves criptográficas deducidas serán diferentes para cada transacción. Esto permite limitar grandemente los ataques físicos sobre las claves de transacción puesto que cada una de ellas no se utilizará más que una sola y única vez. Además, DUKPT es "forward secure": en cualquier momento, si una parte o la integridad de la información secreta contenida en el terminal queda comprometida, el atacante no podrá encontrar las claves criptográficas utilizadas por las transacciones anteriores. Esto permite limitar grandemente el impacto que tendría cualquier transgresión de un terminal (de pago por ejemplo).

La técnica DUKPT padece sin embargo de algunos inconvenientes que ralentizan su generalización.

De manera práctica, el esquema DUKPT descrito en el documento ANSI X9.24 permite generar algo más de un millón de claves (y por tanto gestionar otras tantas transacciones). Estas ventajas en términos de seguridad implican unos inconvenientes. Inicialmente, un terminal que implemente DUKPT debe reservarse en la memoria protegida *veintiún* registros de claves durante todo el ciclo de la vida del DUKPT. Estos registros se utilizan para calcular las claves deducidas. La memoria protegida está basten limitada en la práctica y esta limitación se convierte muy rápidamente en problemática cuando se desean utilizar varios DUKPT en un mismo terminal (lo que es frecuentemente el caso). Un registro es un emplazamiento específico de la memoria dedicado a un uso particular, en el marco de la presente divulgación.

Además, en el servidor, una vez recibido el contador de parte del terminal, son necesarios un cierto número de cálculos para permitir al servidor deducir la clave utilizada por el terminal. DUKPT garantiza que la clave se deduce como máximo al cabo de *diez* iteraciones. Es necesario tomar nota de que estos cálculos son relativamente largos, y representan una parte predominen en la carga del procesador del servidor. En otros términos, la técnica DUKPT:

- requiere mucha memoria protegida en el terminal;
- requiere muchos cálculos en el servidor de descifrado;
- es una solución compleja de implantar y no es modulable.

De ese modo, para promover la solución propuesta por DUKPT, existe por tanto una necesidad de proporcionar una técnica de deducción de claves que resuelva los inconvenientes previamente descritos.

3 Sumario de la invención

La invención no comprende estos inconvenientes de la técnica anterior. En efecto, la invención se refiere a un procedimiento de obtención de al menos una clave de cifrado de al menos unos datos transmitidos desde un cliente hacia un servidor caracterizado porque comprende:

- una etapa de determinación de un número *R* de registros disponibles dentro de dicho cliente para realizar una pluralidad de cálculos de claves de cifrado;
- una etapa de determinación de un número *N* máximo de iteraciones necesarias en una obtención de al menos una clave de cifrado en dicho servidor;
- una etapa de obtención de una estructura de datos representativos del estado de cálculo de la clave realizado dentro de dichos *R* registros disponibles;
- una etapa de cálculo de dicha al menos una clave de cifrado en función:
  - de dicho número de registros disponibles R de dicho cliente,
  - efectuando al menos N llamadas a una función *F* pseudoaleatoria y
  - *de dicha estructura de datos*;

de manera que dicha al menos una clave de cifrado pueda obtenerse entre una combinación de como máximo $T = C_{R+N}^{N} - 1$ claves de cifrado a partir de un secreto previamente compartido entre dicho servidor y dicho cliente.

De ese modo, en función de las limitaciones, se determinan los parámetros R, N y T con el fin de determinar unas claves de cifrado que corresponden a unas situaciones diferentes. En efecto, esta técnica puede adaptarse completamente a una situación precisa (mientras que la DUKPT es completamente fija en sus parámetros a causa de su complejidad). De ese modo, es fácil, gracias a la invención, gestionar con la misma implantación de software y hardware dos situaciones opuestas y adaptarse en consecuencia para maximizar los rendimientos.

Según un modo de realización particular, el número R de registros disponibles y el número N máximo de iteraciones se definen en función de las limitaciones del servidor y de las limitaciones del cliente y de las limitaciones de seguridad.

De ese modo, es posible establecer, para un número de claves que se va a poder generar constante T (denominado “nivel de seguridad T”), un compromiso entre la carga del servidor, de la que se deduce el parámetro N y las capacidades del cliente, principalmente en términos de capacidad de memoria protegida, representada por el parámetro R, que define el número de los registros disponibles dentro del cliente. Es igualmente posible hacer variar el nivel de seguridad T para determinar un compromiso entre el nivel de seguridad T y los parámetros del cliente y/o del servidor.

Las etapas de determinaciones consisten en establecer (obtener) un consenso sobre la elección de los valores de los parámetros N, R, T entre el servidor y el cliente.

La etapa de establecimiento de los parámetros N, R, T puede ser dinámica: el servidor y el cliente intercambian unas informaciones para modificar al menos uno de los parámetros N, R, T, es decir:

- una única vez durante el establecimiento de la comunicación entre el servidor y el cliente: antes de la primera transacción (o la primera comunicación entre el cliente y el servidor), el cliente y el servidor se ponen de acuerdo sobre la elección de los parámetros N, R, T
- “en el transcurso de la vida” del cliente, se cambia de secreto compartido inicial y se inicializan los parámetros N, R, T: en efecto, habiéndose borrado la clave IK en el cliente, se determina una nueva clave IK entre el servidor y el cliente (en función de las claves presentes en los registros de los clientes o en función de una nueva clave intercambiada entre el cliente y el servidor).

La etapa de establecimiento de los parámetros N, R, T puede ser estática: se trata del caso en el que todo es fijo previamente como en la técnica DUKPT usual.

Finalmente, las limitaciones de seguridad llevan al número de claves que se pueden deducir.

Según un modo de realización particular, dicho procedimiento comprende además, cuando se implementa dentro de dicho cliente, una etapa de emisión de un dato que representa dicha estructura de datos representativos del estado de cálculo de la clave realizado dentro de dichos R registros disponibles, registros denotados por R_R, …R_1.

El dato correspondiente a dicha estructura de datos representativos de un estado de cálculo de la clave realizado dentro de dichos R registros disponibles es o bien un contador que corresponde al número de la transacción en curso o bien a dicha estructura de datos en tanto que tal.

Según una característica particular, caracterizado porque esa estructura de datos representativos de un estado de cálculo de la clave realizado dentro de dichos *R* registros disponibles es una concatenación de *R* números enteros, $st = D = D_R, \ldots D_1$ representando cada entero D_i una distancia entre dicho secreto previamente compartido entre dicho servidor y dicho cliente, y una clave almacenada en un registro R_i de dicho cliente, correspondiendo dicha distancia a un número de iteraciones de dicha función *F* pseudoaleatoria.

Según un modo de realización particular, dicho procedimiento comprende:

- una etapa de determinación, partiendo del registro R_1 hasta como máximo el registro R_R, de una posición *p* del

registro más pequeño R_p, para el que una distancia D_p asociada es estrictamente más pequeña que el número N+1, comprendiendo dicho registro R_p una clave K de cifrado, utilizada para una transacción dada;

- una etapa de actualización de dicha estructura de datos que comprende:

  cuando $D_p < N$:

  ◦ una etapa de actualización de los p registros R_p, …R_1 calculando a partir de dicha clave K de registro, de dicha función F pseudoaleatoria, de dicha estructura de datos D y de un número entero i, tal que $R_i = F(K, D, i)$ para $1 <= i <= p$;
  ◦ una etapa de actualización de dicha estructura de datos de manera que $D_i = D_p + 1$ siendo $1 <= i <= p$;

  cuando $D_p = N$, una etapa de borrado del contenido de dicho registro R_p y una etapa de actualización del número $D_p = D_p + 1 = N + 1$.

Según un modo de realización particular, cuando se implementa dentro de dicho servidor, dicha etapa de obtención comprende:

- una etapa de recepción de un dato que representa dicha estructura de datos representativos de un estado de cálculo de la clave realizado dentro de dichos R registros disponibles;
- una etapa de lectura de dicho dato recibido que suministra dicha estructura de datos representativos;
- una etapa de cálculo de dicha estructura de datos representativos de un estado de cálculo de la clave realizado dentro de dichos R registros disponibles cuando dicha estructura de datos representativos no se lee directamente en dicho dato recibido.

Según un modo de realización particular, dicha estructura de datos se define como una concatenación de R números enteros $st = D = D_R, \ldots D_1$ en la que cada número entero D_i representa una distancia entre dicho secreto previamente compartido entre dicho servidor y dicho cliente, y una clave almacenada en un registro R_i de dicho cliente, y porque la etapa de cálculo de dicha al menos una clave de cifrado comprende:

- una etapa de inicialización de una instancia local d a 1 y una posición de registro p tal que $p = p'$, siendo p' un número que corresponde a la posición de mayor peso, partiendo del registro R_R hasta el registro R_1, tal que $D_{p'} > 1$;
- una primera etapa de cálculo para obtener una clave $K = F(\textit{secreto previamente compartido}, 0^R, p')$ en la que 0^R corresponde a R veces 0,
- y al menos una iteración:

  • de un cálculo para obtener una clave $K = F(K, g_{p,d}(D), p)$ en la que $g_{p,v}(D) = D'_R, \ldots D'_1$ para la que $D'_i = N + 1$ si i está comprendido entre 1 y p–1, $D'_i = d$ si $i = p$ y $D'_i = D_i$ si i está comprendido entre p+1 y R y de una etapa para actualizar la variable d tal que $d = d$+1 en tanto que se verifique la condición siguiente: $d < D_p - 1$;
  • si el número $p = 1$, una etapa de cálculo que consiste en obtener una clave $K = F(K, g_{p,i}(D), p)$ que corresponde entonces a la clave compartida temporalmente con el cliente y detener la iteración de las etapas;
  • determinar una posición p' tal que $D_{p'} > D_p$;
  • Si D_p! = N+1, una etapa que consiste en calcular una clave $K = F(K, g_{p,d}(D), p')$ y una etapa de actualización de las variables $p = p'$ y $d = d$+1.
  • Si no, efectuar una etapa que consiste en calcular una clave $K = F(K, g_{p,d}(D), p'+1)$ que corresponde entonces a la clave compartida temporalmente con el cliente y detener la iteración de las etapas.

Según otro aspecto, la invención se refiere a un dispositivo de obtención de al menos una clave de cifrado de al menos unos datos transmitidos desde un cliente hacia un servidor. Según la invención, un dispositivo de ese tipo comprende:

- unos medios de determinación de un número R de registros disponibles dentro de dicho cliente para realizar una pluralidad de cálculos de claves de cifrado;
- unos medios de determinación de un número N máximo de iteraciones necesarias en una obtención de al menos una clave de cifrado en dicho servidor;
- unos medios de obtención de una estructura de datos representativos del estado de cálculo de la claves realizado dentro de dichos R registros disponibles;
- unos medios de cálculo de dicha al menos una clave de cifrado en función:

  - de dicho número de registros disponibles R de dicho cliente,
  - efectuando como máximo N llamadas a la función F pseudoaleatoria y
  - de dicha estructura de datos;

  de manera que dicha al menos una clave de cifrado pueda obtenerse entre una combinación $T = C_{R+N}^{N} - 1$ de claves de cifrado a partir de un secreto previamente compartido entre dicho servidor y dicho cliente.

Según otro aspecto, la invención se refiere a un producto de programa de ordenador que puede telecargarse desde una red de comunicación y/o almacenado en un soporte legible por ordenador y/o ejecutable por un

microprocesador. Según la invención, un programa de ese tipo comprende unas instrucciones de código de programa para la ejecución del procedimiento de obtención tal como se ha descrito anteriormente.

Según otro aspecto, la invención comprende igualmente una señal de obtención de al menos una clave de cifrado de al menos unos datos transmitidos desde un cliente hacia un servidor. Según la invención, una señal de ese tipo comprende una estructura de datos representativos del estado de cálculo de la claves de cifrado realizado dentro de dicho cliente, siendo un contenido de dicha estructura función de un número *R* de registros disponibles dentro de dicho cliente para realizar una pluralidad de cálculos de claves de cifrado y de un número *N* máximo de iteraciones necesarias para una obtención de al menos una clave de cifrado en dicho servidor, de manera que dicha al menos una clave de cifrado pueda obtenerse entre una combinación de un máximo de $T = C_{R+N}^{N} - 1$ claves de cifrado a partir de un secreto previamente compartido entre dicho servidor y dicho cliente.

4 Lista de las figuras

Surgirán más claramente otras características y ventajas de la invención con la lectura de la descripción que sigue de un modo de realización preferido, dado a título de simple ejemplo ilustrativo y no limitativo, y de los dibujos adjuntos, entre los que:

- la figura 1 presenta un sinóptico de la técnica de la invención;
- la figura 2 ilustra la técnica de la invención en un cliente;
- la figura 3 ilustra la técnica de la invención en un servidor;
- la figura 4 ilustra la estructura de un cliente;
- la figura 5 ilustra la estructura de servidor.

5 Descripción detallada de la invención

5.1 Recordatorio del principio de la invención

La invención propone un nuevo procedimiento de implementación del DUKPT, procedimiento en el que la información transmitida desde el terminal al servidor permite reducir por un lado la cantidad de memoria protegida utilizada en el terminal y por otro parte reducir, en el servidor, el número de iteraciones necesarias para el descubrimiento de la clave generada por el terminal. El procedimiento de la invención permite por tanto:

- disminuir la cantidad de memoria protegida utilizada en el terminal;
- reducir el número de cálculos en el servidor de descifrado.

De manera complementaria e inesperada, los presentes inventores han constatado igualmente que el procedimiento propuesto incrementa el nivel de seguridad del DUKPT. En efecto, cuando el procedimiento propuesto se implementa en un terminal que comprende una memoria protegida suficiente y un servidor dedicado que efectúa los cálculos de manera rápida, es posible calcular no un millón de claves (como es este el caso con DUKPT que utiliza veintiún registros y diez iteraciones), sino cuarenta millones, siempre utilizando veintiún registros y diez iteraciones.

De ese modo, se describe, en la presente solicitud una nueva técnica que mejora del algoritmo DUKPT original. Las garantías de seguridad son exactamente las mismas, a saber una clave única por transacción así como el hecho de que una clave que sea comprometida no compromete las claves anteriores y siguientes (“forward security” en inglés). Por el contrario, se mejoran los rendimientos y la memoria requerida.

Por ejemplo, la utilización clásica consiste en poder generar al menos un millón de claves con una garantía de que cada clave de transacción se deducirá en el servidor con un máximo de *diez* iteraciones. En esta situación, el procedimiento de la invención permite no utilizar más que *trece* registros en el terminal en lugar de los *veintiún* registros requeridos para DUKPT. Este tipo de mejora facilita la implantación de este tipo de gestión de claves criptográficas.

Además, es primordial hacer notar que la técnica de la invención es más simple que DUKPT y sobre todo modulable. En efecto, la técnica de la invención puede adaptarse completamente a una situación precisa (mientras que la DUKPT tal como está propuesta actualmente es completamente fija en sus parámetros a causa de su complejidad). De ese modo, es muy fácil, con la técnica de la invención, poder gestionar con la misma configuración dos situaciones diametralmente opuestas y adaptarse en consecuencia para maximizar los rendimientos. Esta modularidad se ópera con la ayuda de tres parámetros:

- el número de registros *R* disponibles en el terminal;
- el número máximo de iteraciones *N* en el servidor para encontrar la clave;
- el número total de claves distintas *T* que pueden ser generadas.

De ese modo, para un nivel de seguridad dado (por ejemplo *T es superior o igual a 1 millón*), es posible con la técnica de la invención, hacer variar a la vez *R* y *N* en función de la situación del terminal (el número de registros *R* de los que dispone) y del servidor (el número de iteraciones *N* que puede aceptar en función de su carga). De ese modo, para un primer terminal, *R* puede ser por ejemplo igual a 13 y *N* igual a 10 en el servidor, mientras que para

un segundo terminal, *R* puede ser por ejemplo igual a 14 y *N* igual a 9 en el servidor. El nivel de seguridad (por ejemplo el número *T es superior o igual a 1 millón*) es el mismo.

Esta modularidad, tal como se presenta, no es posible con la técnica anterior. Esta modularidad es posible, según la invención, implementando dos procedimientos: uno en el terminal (el cliente), el otro en el servidor. La ejecución de estos procedimientos y la transmisión de datos realizada por el terminal al servidor permiten resolver el problema de ausencia de modularidad de las técnicas anteriores y como consecuencia los problemas de tamaño de memoria y los problemas de carga de los servidores.

Se presenta, en relación con la figura 1, el principio general de la invención tal como se implementa a la vez en el lado cliente y en el lado servidor. Existen unas disparidades entre el cliente y el servidor. Se describirán a continuación. El procedimiento de la invención comprende, en su interpretación más amplia:

- una etapa (10) de determinación de un número *R* de registros disponibles dentro de dicho cliente (1) para realizar una pluralidad de cálculos de claves de cifrado;
- una etapa (20) de determinación de un número *N* máximo de iteraciones necesarias en una obtención de al menos una clave de cifrado en dicho servidor (2);
- una etapa (30) de obtención de una estructura de datos (*st*) representativos de un estado de cálculo de las claves realizado dentro de dichos *R* registros disponibles;
- una etapa (40) de cálculo de dicha al menos una clave (*K*) de cifrado en función:

    ◦ de dicho número de registros *R* disponibles de dicho cliente,
    ◦ efectuando como máximo *N* llamadas a una función *F* pseudoaleatoria y
    ◦ de dicha estructura de datos;

  de manera que dicha al menos una clave de cifrado pueda obtenerse entre una combinación de $T = C_{R+N}^{N} - 1$ claves de cifrado a partir de un secreto previamente compartido entre dicho servidor y dicho cliente.

La combinación $T = C_{R+N}^{N} - 1$ se denota igualmente como $T = \binom{R+N}{N} - 1$ en ciertos pasajes del documento.

En otros términos:

Sea *IK* una clave originalmente compartida entre el terminal y el servidor. Se supone que esta clave está compuesta de *k* bits y se desea que las claves de transacción sean del mismo tamaño. Es decir $F: \{0,1\}^* \rightarrow \{0,1\}^k$ una función pseudoaleatoria. Se indica por || la operación de concatenación.

Lado del terminal

El terminal se inicializa primero rellenando sus *R* registros como sigue: para el *j-ésimo* registro *Rj*, el valor contenido en este registro es $Rj = F(IK \,||\, j)$. Paralelamente, facilitar la comprensión, se mantiene actualizada una tabla *Tbl* de *R* contadores locales, todos inicializados a uno. El contador situado en la casilla *j* de la tabla *Tbl* corresponde al número de iteraciones necesarias a partir de la clave de origen *IK* para obtener la clave de transacción almacenada en el registro *Rj*. Por supuesto, se asegura que ninguno de estos contadores sobrepasa el valor *N*, puesto que este número representa el máximo de iteraciones en el lado del servidor. Finalmente, se inicializa a cero un contador de transacción *CT*.

En cada transacción (por ejemplo bancaria), el contador de transacción *CT* se incrementa en 1. La clave de transacción (utilizada para esta transacción) es la clave situada en el registro de peso más reducido y cuyo contador local *j* correspondiente es inferior o igual a *N*. Se denotan este registro de peso más reducido por *Rk* y la clave de transacción correspondiente por *K*. El valor almacenado en el contador local *j* correspondiente a *Rk* se denota por *C*. Una vez utilizada la clave *K* (para la transacción), se actualizan los registros y los contadores locales:

- si $C < N$, todos los registros *Ri* de peso inferior o igual a *k* (es decir $i \leq k$) reciben el valor $Ri = F(K \,||\, k \,||\, i)$ mientras que los contadores correspondientes se modifican a $C + 1$ (se incrementan en el valor de 1);
- si $C = N$, se borra *K* del registro *Rk* y se incrementa el contador local correspondiente (que pasa por tanto a $N+1$).

Los datos de transacción se cifran entonces por el terminal con esta clave de transacción *K* para ser transmitidos al servidor.

Lado del servidor

Cuando se obtiene la clave de transacción *K* por el terminal (el procedimiento de obtención previamente descrito permite obtener esta clave), los datos sensibles de la transacción se cifran con esta clave de transacción *K* y se transmiten al servidor con el contador de transacción *CT*. El servidor recibe el contador *CT* y deduce de él el encaminamiento de la deducción de clave a partir de *IK*. Una vez obtenida la clave de transacción *K*, se pueden descifrar los datos cifrados por el terminal.

Se observa que todas las claves de transacciones serán diferentes. Además, el procedimiento es correctamente "forward secure" puesto que la transgresión de los *R* registros del terminal no permite encontrar información sobre las claves utilizadas durante las transacciones precedentes. Como para la técnica DUKPT, esto se debe al hecho de que la función *F* no es reversible.

En términos de rendimiento, el procedimiento de la invención garantiza correctamente un máximo de *N* iteraciones en el servidor para *R* registros almacenados en el terminal. El número de claves *T* que pueden ser generadas es una función de *R* y de *N*:

$$T = \sum_{i=1}^{N} \sum_{j=1}^{R} a(i, j)$$

siendo $a(i, 1) = 1$ para todo *i*, $a(1, j) = 1$ para todo *j*, y

$$a(n, r) = \sum_{i=1}^{r} a(n-1, i)$$

Se presenta, en relación con la figura 2, un modo de realización de la invención implementado dentro del cliente. El procedimiento de la invención, implementado dentro de un cliente, comprende:

- una etapa (10) de determinación de un número *R* de registros disponibles dentro de dicho cliente (1) para realizar una pluralidad de cálculos de claves de cifrado;
- una etapa (20) de determinación de un número *N* máximo de iteraciones necesarias en una obtención de al menos una clave de cifrado en dicho servidor (2);
- una etapa (30) de obtención de una estructura de datos (*st*) representativos de un estado de cálculo de las claves realizado dentro de dichos *R* registros disponibles;
- una etapa (40) de cálculo de dicha al menos una clave (*K*) de cifrado en función:
    - ◦ de dicho número de registros *R* disponibles de dicho cliente,
    - ◦ efectuando como máximo *N* llamadas a una función *F* pseudoaleatoria y
    - ◦ de dicha estructura de datos;

  de manera que dicha al menos una clave de cifrado pueda obtenerse entre una combinación de $T = C_{R+N}^{N} - 1$ claves de cifrado a partir de un secreto previamente compartido entre dicho servidor y dicho cliente;
- una etapa (50) de emisión de un dato que representa dicha estructura de datos representativos (*st*) de un estado de cálculo de la clave realizado dentro de dichos *R* registros disponibles, registros denotados por $R_R, \ldots R_1$.

Se presenta, en relación con la figura 3, un modo de realización de la invención implementado dentro del servidor. El procedimiento de la invención, implementado dentro de un servidor, comprende:

- una etapa (10) de determinación de un número *R* de registros disponibles dentro de dicho cliente (1) para realizar una pluralidad de cálculos de claves de cifrado;
- una etapa (20) de determinación de un número *N* máximo de iteraciones necesarias en una obtención de al menos una clave de cifrado en dicho servidor (2);
- una etapa (30) de obtención de una estructura de datos (*st*) representativos de un estado de cálculo de las claves realizado dentro de dichos *R* registros disponibles;
- una etapa (40) de cálculo de dicha al menos una clave (*K*) de cifrado en función:
    - ◦ de dicho número de registros *R* disponibles de dicho cliente,
    - ◦ efectuando como máximo *N* llamadas a una función *F* pseudoaleatoria y
    - ◦ de dicha estructura de datos;

  de manera que dicha al menos una clave de cifrado pueda obtenerse entre una combinación de $T = C_{R+N}^{N} - 1$ claves de cifrado a partir de un secreto previamente compartido entre dicho servidor y dicho cliente.

En un modo de realización de la invención, la etapa (30) de obtención comprende:

- una etapa (31) de recepción de un dato que representa dicha estructura de datos representativos de un estado de cálculo de la clave realizado dentro de dichos *R* registros disponibles;
- una etapa (32) de lectura de dicho dato recibido que suministra dicha estructura de datos representativos;
- una etapa (33) de cálculo de dicha estructura de datos representativos del estado de cálculo de la clave realizado dentro de dichos *R* registros disponibles cuando dicha estructura de datos representativos no se lee directamente en dicho dato recibido.

De ese modo, conviene remarcar que la etapa (33) de cálculo no es siempre necesaria. En función del dato recibido enviado por el cliente, el servidor ve su carga de cálculo y/o almacenamiento más o menos sobrecargada.

A continuación, se presenta principalmente el caso de una implementación particular de la técnica de la invención. Es claro sin embargo que la invención no se limita a esta aplicación particular, sino que puede ponerse en práctica igualmente en otras configuraciones.

5.2 Descripción de un modo de realización

Se presenta en este modo de realización, la implementación del procedimiento de la invención en un procedimiento denominado "*Optimal-DUKPT*" en lo que tiene vocación de sustituir a la técnica DUKPT de la técnica anterior.

La idea básica para mejorar la DUKPT puede ser introducida de la simple manera siguiente: para toda primera operación *tc* = 1 de DUKPT, se utiliza la clave K_1 situada en el primer registro y se borra directamente.

Se observa que esta clave no tiene hijo en la jerarquía de claves y que su madre es la clave *IK* (que está a la distancia 1 de la clave *IK*). Dicho de otra manera, el servidor puede recuperar *K*1 de *IK* con una única aplicación de *F*. En lugar de borrar *K*1 directamente y puesto que se estaría lejos de alcanzar diez iteraciones de *F* en el lado del servidor, es posible deducir otra clave de *K*1 y colocarla en este primer registro. Profundizando en esta idea, los presentes inventores han tenido la idea de generar *nueve* claves suplementarias con el primer registro solamente.

Y esto puede generalizarse sobre los otros registros igualmente. En tanto que el primer registro contiene una clave situada a una distancia de diez de *IK*, no puede ser deducida de nuevo.

Entonces, es necesario utilizar una clave situada en el segundo registro, pero antes de borrarla de la memoria del cliente, se pueden deducir dos nuevas claves que se pueden colocar en el primer registro y posteriormente en el segundo registro.

Estas dos nuevas claves están a la distancia de *dos* de la clave *IK*. De nuevo, se pueden deducir varias claves utilizando solamente el primer registro, pero una menos que anteriormente puesto que se comienza con una clave a una distancia de *dos* (y no de *uno*) de la clave *IK*. Esta técnica se reitera finalmente en todos los registros.

5.2.1. Descripción

Con una preocupación por la preservación del escalonamiento del algoritmo, el DUKPT mejorado tal como se describe actualmente se define en tanto que familia de soluciones de gestión de claves. Cada miembro de la familia se identifica por la cantidad *R* de registros de claves disponibles por el lado del cliente y por el número *N* de iteraciones máximas necesarias para deducir una clave del servidor. Es fácil demostrar que cada miembro puede gestionar un número máximo de claves $T = \binom{R+N}{N} - 1$.

En cuanto al DUKPT de origen, se supone que la clave *IK* simétrica compartida se ha atribuido de manera protegida al cliente y al servidor. Para identificar la clave deducida, se transmite para cada operación una cadena pública *st*, que hace las veces de estructura de datos según la invención, desde el cliente hacia el servidor. Esta cadena comprende *R* enteros st_i, siendo $1 \leq st_i \leq N$ para $1 \leq i \leq R$. Un entero st_i representa la distancia de *IK* desde la clave almacenada en el registro *i* de la memoria del cliente antes de tratar la operación. Por ejemplo, la cadena empleada para cualquier primera iteración es 1, …1 1, 1 … 1 2 para la segunda, etc.

5.2.1.1. Del lado del cliente.

El cliente mantiene dos tablas. En primer lugar, los registros clásicos de clave *R*, denotados *Ri* para $1 \leq i \leq R$. Se inicializan simplemente con $Ri = F(IK; 0^R; i)$ y cuando se finaliza la inicialización, la clave de origen *IK* se borra de la memoria del cliente.

En segundo lugar, el cliente mantiene una tabla *D* de enteros *R* denominados *Di*, en la que *Di* representa la distancia entre la clave *IK* y la clave almacenada en el registro *Ri*. Esta distancia representa el número de iteraciones necesarias para obtener la clave almacenada en el registro *Ri* a partir de la clave de origen *IK*. El contenido de *D* es exactamente lo que se envía al servidor en la cadena *st*. Naturalmente, se inicializa con $Ci = 1$ para $1 \leq i \leq R$.

Cuando recibe una solicitud de procesamiento de una nueva transacción, el cliente establece $st = D$ y busca el registro menos significativo que tenga una distancia correspondiente *Di* estrictamente inferior a $N + 1$.

Este registro, que se denomina *Rp*, contiene la clave de transacción *K* que se utilizará para la transacción. Posteriormente, una vez terminada la tasación:

- si $Dp < N$, el cliente actualiza los *p* registros *Rp* , *Rp*–1, ,,, *R*1 con $Ri = F(K; D; i)$ y actualiza la tabla de las distancias con $Di = Dp + 1$ con $1 \leq i \leq p$;
- si $Dp = N$, el cliente borra simplemente el contenido del registro *Rp* y actualiza $Dp = Dp + 1 = N + 1$. Este registro ya no se utilizará.

Se observa que en el proceso de deducción de clave, los datos utilizados en la entrada de *F* son siempre únicos. En efecto, *D* será diferente para cada transacción. Esto garantiza la seguridad del sistema. En efecto, la "forward secrecy" se mantiene siempre puesto de, después de que se haya utilizado una clave de transacción, se cuida que esta clave y sus predecesoras no estén ya presentes en la memoria del cliente.

Se da un ejemplo de la evolución del estado interno de los clientes en la tabla 2 a continuación.

5.2.1.2. Del lado del servidor.

El servidor recibe una cadena *st* que corresponde a la tabla *D* del cliente antes de procesar la transacción. Se observa que los valores memorizados en esta tabla de distancia se incrementan siempre del registro más significativo al registro menos significativo. Además, se recuerda que cuando el cliente extrae una clave de transacción a partir de un registro *Rp*, esto significa que la tabla de distancias es tal que $Di = N + 1$ para $1 \leq i \leq p-1$.

Se denota por $g_{p,v}(D)$ la transformación que mapea la tabla de distancia *D* a otra tabla de distancia *D'* siendo:

$$\begin{cases} D'_i = N+1, & \text{para } 1 \leq i \leq p-1 \\ D'_i = v, & \text{para } i = p \\ D'_i = D_i, & \text{para } p+1 \leq i \leq R \end{cases}$$

El servidor inicializa en primer lugar un valor local de distancia $d = 1$ y un valor de posición inscrito $p = p'$, siendo p' la posición más significativa con $Dp' > 1$. A continuación, calcula $K = F(IK; 0^R; p')$ y no cesa de repetir el proceso siguiente:

- En tanto que $d < Dp - 1$, se calcula $K = F(K, g_{p,d}(D), p)$ y $d = d + 1$;
- Si $p = 1$, entonces $K = F(K, g_{p,d}(D), p)$ es la clave compartida con el cliente, y la serie de iteraciones se detiene;
- El servidor busca la posición más significativa p' de manera que $Dp' > Dp$. Si Dp' es diferente de $N + 1$, entonces el servidor calcula $K = F(K, g_{p,d}(D), p0)$ y actualiza la variable local $p = p'$ y $d = d + 1$. Si no, $K = F(K, g_{p,d}(D), p'+1)$ es la clave compartida con el cliente y las iteraciones pueden detenerse.

Este algoritmo sigue exactamente el proceso implícito efectuado por el cliente para obtener la clave de transacción *K* a partir de la primera clave *IK*. Los procedimientos implementados en el cliente y en el servidor son por tanto los mismos, en sus etapas principales.

iteración	*st* enviado	clave de transacción usada y actualización de registros clave	evolución registros clave			tabla de distancia			
			R_3	R_2	R_1		D_3	D_2	D_1
inicial		$R_3 = F(IK, 000, 3)$ $R_2 = F(IK, 000, 2)$, $R_1 = F(IK, 000, 1)$	TK_{10}	TK_4	TK_1	entrada salida	0 1	0 1	0 1
1	111	$TK_1 = R_1$ $R_1 = F(TK_1, 111, 1)$			TK_2	entrada salida	1 1	1 1	1 2
2	112	$TK_2 = R_1$ $R_1 = F(TK_2, 112, 1)$			TK_3	entrada salida	1 1	1 1	2 3
3	113	$TK_3 = R_1$ borrar R_1			X	entrada salida	1 1	1 1	3 4
4	114	$TK_4 = R_2$ $R_2 = F(TK_4, 114, 2)$, $R_1 = F(TK_4, 114, 1)$		TK_7	TK_5	entrada salida	1 1	1 2	4 2
5	122	$TK_5 = R_1$ $R_1 = F(TK_5, 122, 1)$			TK_6	entrada salida	1 1	2 2	2 3
6	123	$TK_6 = R_1$ borrar R_1			X	entrada salida	1 1	2 2	3 4
7	124	$TK_7 = R_2$ $R_2 = F(TK_7, 124, 2)$, $R_1 = F(TK_7, 124, 1)$		TK_9	TK_8	entrada salida	1 1	2 3	4 3
8	133	$TK_8 = R_1$ borrar R_1			X	entrada salida	1 1	3 3	3 4
9	134	$TK_9 = R_2$ borrar R_2		X		entrada salida	1 1	3 4	4 4
10	144	$TK_{10} = R_3$, $R_3 = F(TK_{10}, 144, 3)$ $R_2 = F(TK_{10}, 144, 2)$, $R_1 = F(TK_{10}, 144, 1)$	TK_{16}	TK_{13}	TK_{11}	entrada salida	1 2	4 2	4 2

(continuación)

iteración	*st* enviado	clave de transacción usada y actualización de registros clave	evolución registros clave			tabla de distancia			
			R_3	R_2	R_1		D_3	D_2	D_1
11	222	$TK_{11} = R_1$ $R_1 = F(TK_{11}, 222, 1)$			TK_{12}	entrada salida	2 2	2 2	2 3
12	223	$TK_{12} = R_1$ borrar R_1			X	entrada salida	2 2	2 2	3 4
13	224	$TK_{13} = R_2$ $R_2 = F(TK_{13}, 224, 2)$, $R_1 = F(TK_{13}, 224, 1)$		TK_{15}	TK_{14}	entrada salida	2 2	2 3	4 3
14	233	$TK_{14} = R_1$ borrar R_1			X	entrada salida	2 2	3 3	3 4
15	234	$TK_{15} = R_2$ borrar R_2		X		entrada salida	2 2	3 4	4 4
16	244	$TK_{16} = R_3$, $R_3 = F(TK_{16}, 244, 3)$ $R_2 = F(TK_{16}, 244, 2)$, $R_1 = F(TK_{16}, 244, 1)$	TK_{19}	TK_{18}	TK_{17}	entrada salida	2 3	4 3	4 3
17	333	$TK_{17} = R_1$ borrar R_1			X	entrada salida	3 3	3 3	3 4
18	334	$TK_{18} = R_2$ borrar R_2		X		entrada salida	3 3	3 4	4 4
19	344	$TK_{19} = R_3$ borrar R_3	X			entrada salida	3 4	4 4	4 4

Tabla 2: Ejemplos de entradas principales y una distancia de evolución de las tablas sobre el lado cliente, con parámetros del sistema N =3 y R = 3. Se denota por TK_i la clave utilizada para la iteración i-ésima. Una X\ en las columnas de claves registros de evolución significa que el cliente borra el contenido a partir de este registro.

En este ejemplo, supongamos que el servidor recibe st = 224. Posiciona d = 1, p = 3 y calcula $K = F(IK; 000; 3)$. Posteriormente, no entra en el bucle de iteración, ni en la primera condicional si calcula p' = 1 y posteriormente Dp = 4 = N + 1, la clave $K = F(K; 144; 2)$ es la clave compartida con el cliente. Se da en la tabla 3 un ejemplo más complicado.

Tabla 3: Ejemplos de deducción de clave sobre el lado servidor, con los parámetros del sistema N = 8, R = 8 y st = 12466689. La clave se determina con ocho iteraciones en el servidor.

iteración	Actualización clave	Valores locales			
			d	p	p'
inicial	$K = F(IK, 00000000, 7)$	entrada salida	 1	 7	7
1	$K = F(K, 11999999, 6)$	entrada salida	1 2	7 6	6
2	$K = F(K, 12299999,6)$ $K = F(K, 12399999, 5)$	entrada salida	2 4	6 5	5
3	$K = F(K, 12449999, 5)$ $K = F(K, 12459999, 2)$	entrada salida	4 6	5 2	2
4	$K = F(K, 12466669, 2)$ $K = F(K, 12468679, 2)$	entrada salida	6	2	1

5.2.1.3. Otra característica

En al menos un modo de realización complementario de la invención, el cliente no transmite una estructura de datos *st* en tanto que tal al servidor, sino un dato específico que permite al servidor encontrar esta estructura de datos *st* para, a continuación, obtener la clave de cifrado.

Un modo de realización de ese tipo permite reducir el volumen de datos transmitidos desde el cliente hacia un servidor.

5.3 Otras características opcionales y ventajas

Se presenta, en relación con la figura 4, un modo de realización de un cliente que comprende unos medios de cálculo de las claves y unos medios de transmisión de la estructura de datos st que permite al servidor calcular a su vez la clave utilizada por el cliente.

Un cliente de ese tipo comprende una memoria 41 (que comprende una memoria protegida que comprende unos registros *R* constituidos por una memoria tampón), una unidad 42 de procesamiento, equipada por ejemplo con un microprocesador P, y controlada por un programa de ordenador 43, que implementa el procedimiento de la invención.

En la inicialización, las instrucciones del código del programa de ordenador 43 se cargan por ejemplo en una memoria RAM antes de ser ejecutadas por el procesador de la unidad 42 de procesamiento. La unidad 42 de procesamiento recibe en la entrada al menos una información I, tal como el número *N* de iteraciones del servidor. El microprocesador de la unidad 42 de procesamiento implementa las etapas del procedimiento descrito anteriormente, según las instrucciones del programa de ordenador 43, para entregar una información procesada T, tal como la estructura de datos *st* y sus datos cifrados con la ayuda de la clave *K* calculada por el terminal. Para ello, el dispositivo comprende, además de la memoria tampón 41, unos medios de determinación de un número R de registros disponibles dentro de dicho cliente para realizar una pluralidad de cálculos de claves de cifrado, unos medios de determinación de un número *N* máximo de iteraciones necesarias para una obtención de al menos una clave de cifrado en dicho servidor, unos medios de obtención de una estructura de datos representativos del estado de cálculo de la clave realizado dentro de dichos *R* registros disponibles y unos medios de cálculo de dicha al menos una clave de cifrado.

Estos medios están controlados por el microprocesador de la unidad 42 de procesamiento.

Se presenta, en relación con la figura 5, un modo de realización de un servidor que comprende unos medios de recepción de la estructura de datos *st* y unos medios de cálculo de la clave utilizada por el cliente.

Un servidor de ese tipo comprende una memoria 51 (que comprende una memoria protegida que comprende unos registros *R* constituidos por una memoria tampón), una unidad 52 de procesamiento, equipada por ejemplo con un microprocesador P, y controlada por un programa de ordenador 53, que implementa el procedimiento de la invención.

En la inicialización, las instrucciones del código del programa de ordenador 53 se cargan por ejemplo en una memoria RAM antes de ser ejecutadas por el procesador de la unidad 52 de procesamiento. La unidad 52 de procesamiento recibe en la entrada al menos una información I, tal como la estructura de datos *st* procedente del cliente, y los datos cifrados por él con la ayuda de la clave *K*. El microprocesador de la unidad 42 de procesamiento implementa las etapas del procedimiento descrito anteriormente, según las instrucciones del programa de ordenador 43. Para entregar una información procesada T, tal como la clave de cifrado *K*, obtenida a partir de la estructura de datos *st*.

Para ello, el dispositivo comprende, además de la memoria tampón 51, unos medios de determinación de un número *R* de registros disponibles dentro de dicho cliente para realizar una pluralidad de cálculos de claves de cifrado. Unos medios de determinación de un número *N* máximo de iteraciones necesarias para una obtención de al menos una clave de cifrado, unos medios de obtención de una estructura de datos representativos de un estado de cálculo de la clave realizado dentro de dichos *R* registros disponibles y unos medios de cálculo de dicha al menos una clave de cifrado *K*.

Estos medios se controlan por el microprocesador de la unidad 52 de procesamiento.

REIVINDICACIONES

1. Procedimiento de obtención de al menos una clave de cifrado de al menos unos datos transmitidos desde un cliente hacia un servidor **caracterizado porque** comprende:

- una etapa de determinación de un número *R* de registros disponibles dentro de dicho cliente para realizar una pluralidad de cálculos de claves de cifrado;
- una etapa de determinación de un número *N* máximo de iteraciones necesarias en una obtención de al menos una clave de cifrado en dicho servidor;
- una etapa de obtención de una estructura de datos representativos del estado de cálculo de la clave realizado dentro de dichos *R* registros disponibles;
- una etapa de cálculo de dicha al menos una clave de cifrado en función:

  - de dicho número de registros disponibles *R* de dicho cliente,
  - efectuando como máximo *N* llamadas a una función *F* pseudoaleatoria y
  - *de dicha estructura de datos*;

de manera que dicha al menos una clave de cifrado pueda obtenerse entre una combinación de como máximo $T = C_{R+N}^{N} - 1$ claves de cifrado a partir de un secreto previamente compartido entre dicho servidor y dicho cliente.

2. Procedimiento según la reivindicación 1, **caracterizado porque** el número R de registros disponibles y el número *N* máximo de iteraciones se definen en función de las limitaciones del servidor y de las limitaciones del cliente y de las limitaciones de seguridad.

3. Procedimiento según una cualquiera de las reivindicaciones anteriores, **caracterizado porque** comprende además, cuando se implementa dentro de dicho cliente, una etapa de emisión de un dato que representa dicha estructura de datos representativos del estado de cálculo de la clave realizado dentro de dichos *R* registros disponibles, registros denotados por R_R, …R_1.

4. Procedimiento según la reivindicación 3, **caracterizado porque** dicha estructura de datos representativos de un estado de cálculo de la clave realizado dentro de dichos *R* registros disponibles es una concatenación de *R* números enteros, $st = D = D_R$, … D_1 representando cada entero D_i una distancia entre dicho secreto previamente compartido entre dicho servidor y dicho cliente, y una clave almacenada en un registro R_i de dicho cliente, correspondiendo dicha distancia a un número de iteraciones de dicha función *F* pseudoaleatoria.

5. Procedimiento según la reivindicación 4 **caracterizado porque** dicho procedimiento comprende:

- una etapa de determinación, partiendo del registro R_1 hasta como máximo el registro R_R, de una posición *p* del registro más pequeño R_p, para el cual una distancia D_p asociada es estrictamente más pequeña que el número *N*+1, comprendiendo dicho registro R_p una clave *K* de cifrado, utilizada para una transacción dada;
- una etapa de actualización de dicha estructura de datos que comprende:

  cuando $D_p < N$:

  - una etapa de actualización de los *p*–1 registros R_p, …R_1 calculando a partir de dicha clave *K* de registro, de dicha función *F* pseudoaleatoria, de dicha estructura de datos *D* y de un número entero *i*, tal que $R_i = F(K, D, i)$ para $1 <= i <= p$;
  - una etapa de actualización de dicha estructura de datos de manera que $D_i = D_p + 1$ siendo $1 <= i <= p$;

  cuando $D_p = N$, una etapa de borrado del contenido de dicho registro R_p y
  una etapa de actualización del número $D_p = D_p + 1 = N + 1$.

6. Procedimiento según una cualquiera de las reivindicaciones 1 o 2, **caracterizado porque**, cuando se implementa dentro de dicho servidor, dicha etapa de obtención comprende:

- una etapa de recepción de un dato que representa dicha estructura de datos representativos de un estado de cálculo de la clave realizado dentro de dichos *R* registros disponibles;
- una etapa de lectura de dicho dato recibido que suministra dicha estructura de datos representativos;
- una etapa de cálculo de dicha estructura de datos representativos de un estado de cálculo de la clave realizado dentro de dichos *R* registros disponibles cuando dicha estructura de datos representativos no se lee directamente en dicho dato recibido.

7. Procedimiento según la reivindicación 6, **caracterizado porque**, dicha estructura de datos se define como una concatenación de *R* números enteros $st = D = D_R$, … D_1 en la que cada número entero D_i representa una distancia entre dicho secreto previamente compartido entre dicho servidor y dicho cliente, y una clave almacenada en un registro R_i de dicho cliente, y **porque** la etapa de cálculo de dicha al menos una clave de cifrado comprende:

- una etapa de inicialización de una instancia local *d* a 1 y una posición de registro *p* tal que $p = p'$, siendo *p*' un

número que corresponde a la posición de mayor peso, partiendo del registro R_R hasta el registro R_1, tal que $D_{p'} > 1$;
- una primera etapa de cálculo para obtener una clave $K = F(secreto\ previamente\ compartido, 0^R, p')$ en la que 0^R corresponde a R veces 0,
- y al menos una iteración:

• de un cálculo para obtener una clave $K = F(K, g_{p,d}(D), p)$ en la que $g_{p,v}(D) = D'_R, \ldots D'_1$ para la que $D'_i = N+1$ si i está comprendido entre 1 y p–1, $D'_i = d$ si $i = p$ y $D'_i = D_i$ si i está comprendido entre p+1 y R y de una etapa para actualizar la variable d tal que $d = d$+1 en tanto que se verifique la condición siguiente: $d < D_p - 1$;
• si el número $p = 1$, una etapa de cálculo que consiste en obtener una clave $K = F(K, g_{p,i}(D), p)$ que corresponde entonces a la clave compartida temporalmente con el cliente y detener la iteración de las etapas;
• determinar una posición p' tal que $D_{p'} > D_p$;
• Si $D_p! = N+1$, una etapa que consiste en calcular una clave $K = F(K, g_{p,d}(D), p')$ y una etapa de actualización de las variables $p = p'$ y $d = d+1$.
• Si no, efectuar una etapa que consiste en calcular una clave $K = F(K, g_{p,d}(D), p'+1)$ que corresponde entonces a la clave compartida temporalmente con el cliente y detener la iteración de las etapas.

8. Dispositivo de obtención de al menos una clave de cifrado de al menos unos datos transmitidos desde un cliente hacia un servidor **caracterizado porque** comprende:

- unos medios de determinación de un número R de registros disponibles dentro de dicho cliente para realizar una pluralidad de cálculos de claves de cifrado;
- unos medios de determinación de un número N máximo de iteraciones necesarias en una obtención de al menos una clave de cifrado en dicho servidor;
- unos medios de obtención de una estructura de datos representativos del estado de cálculo de la claves realizado dentro de dichos R registros disponibles;
- unos medios de cálculo de dicha al menos una clave de cifrado en función:

  - de dicho número de registros disponibles R de dicho cliente,
  - efectuando como máximo N llamadas a la función F pseudoaleatoria y
  - de dicha estructura de datos;

de manera que dicha al menos una clave de cifrado pueda obtenerse entre una combinación de como máximo $T = C_{R+N}^{N} - 1$ claves de cifrado a partir de un secreto previamente compartido entre dicho servidor y dicho cliente.

9. Producto de programa de ordenador que puede telecargarse desde una red de comunicación y/o almacenado en un soporte legible por ordenador y/o ejecutable por un microprocesador, **caracterizado porque** comprende unas instrucciones de código de programa para la ejecución del procedimiento de obtención según una al menos de las reivindicaciones 1 a 7 cuando se ejecuta en un ordenador.

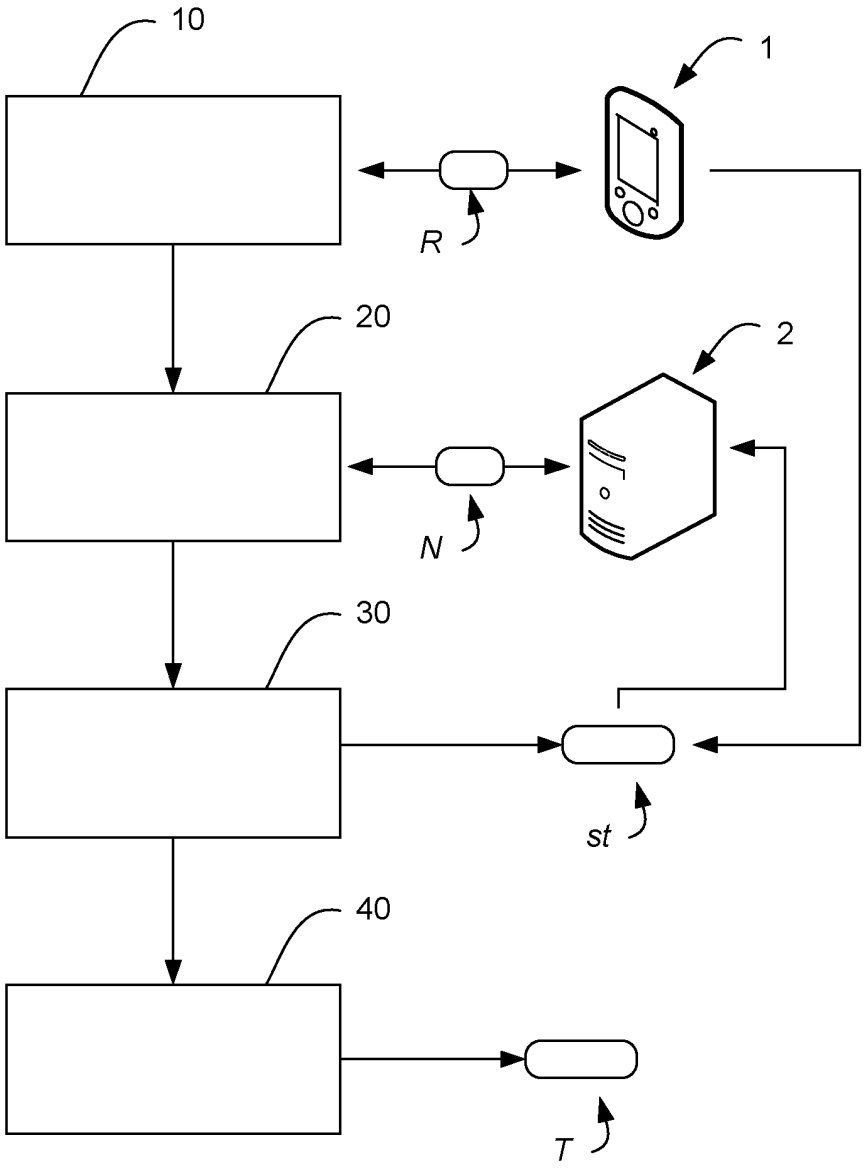

Figura 1

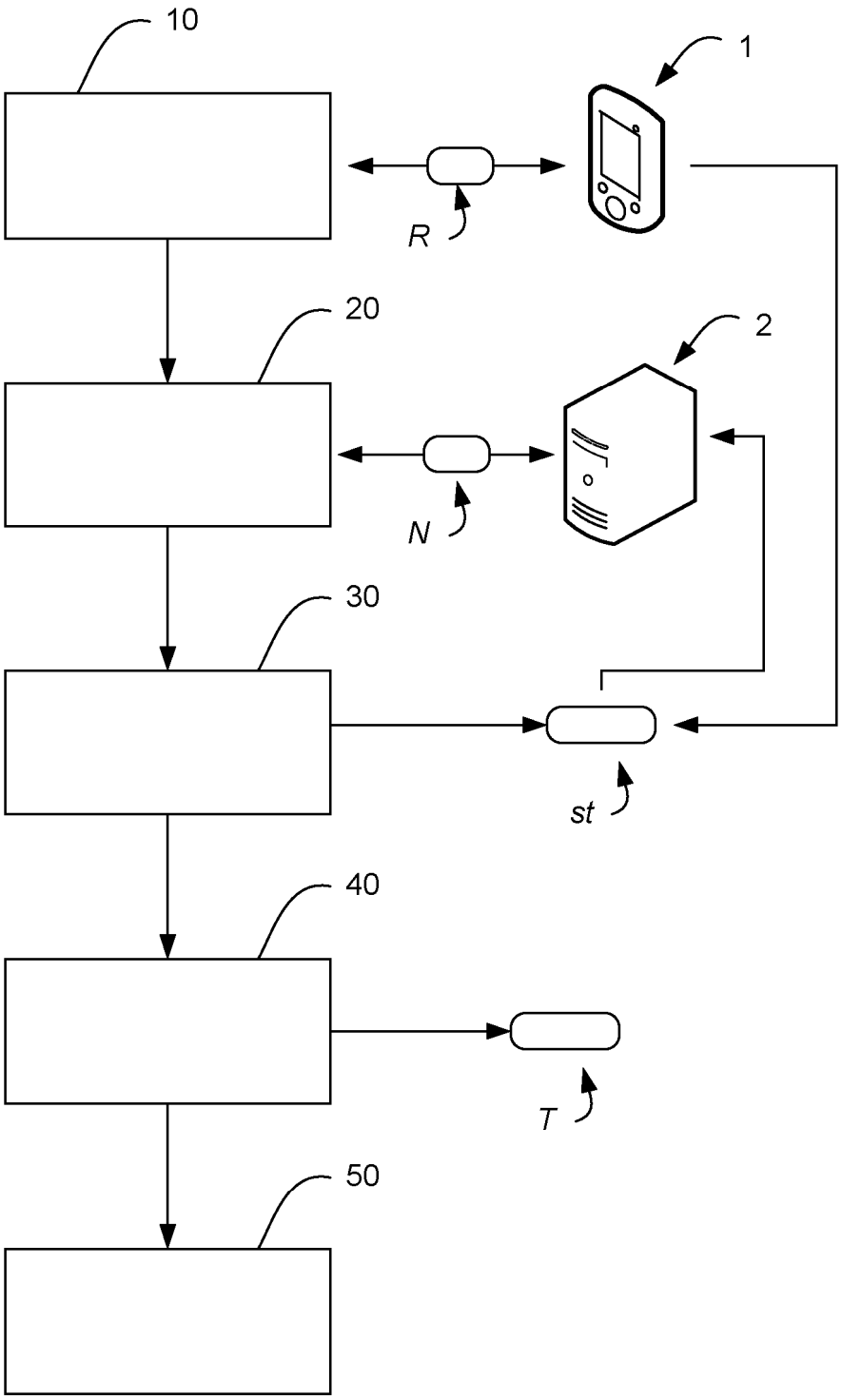

Figura 2

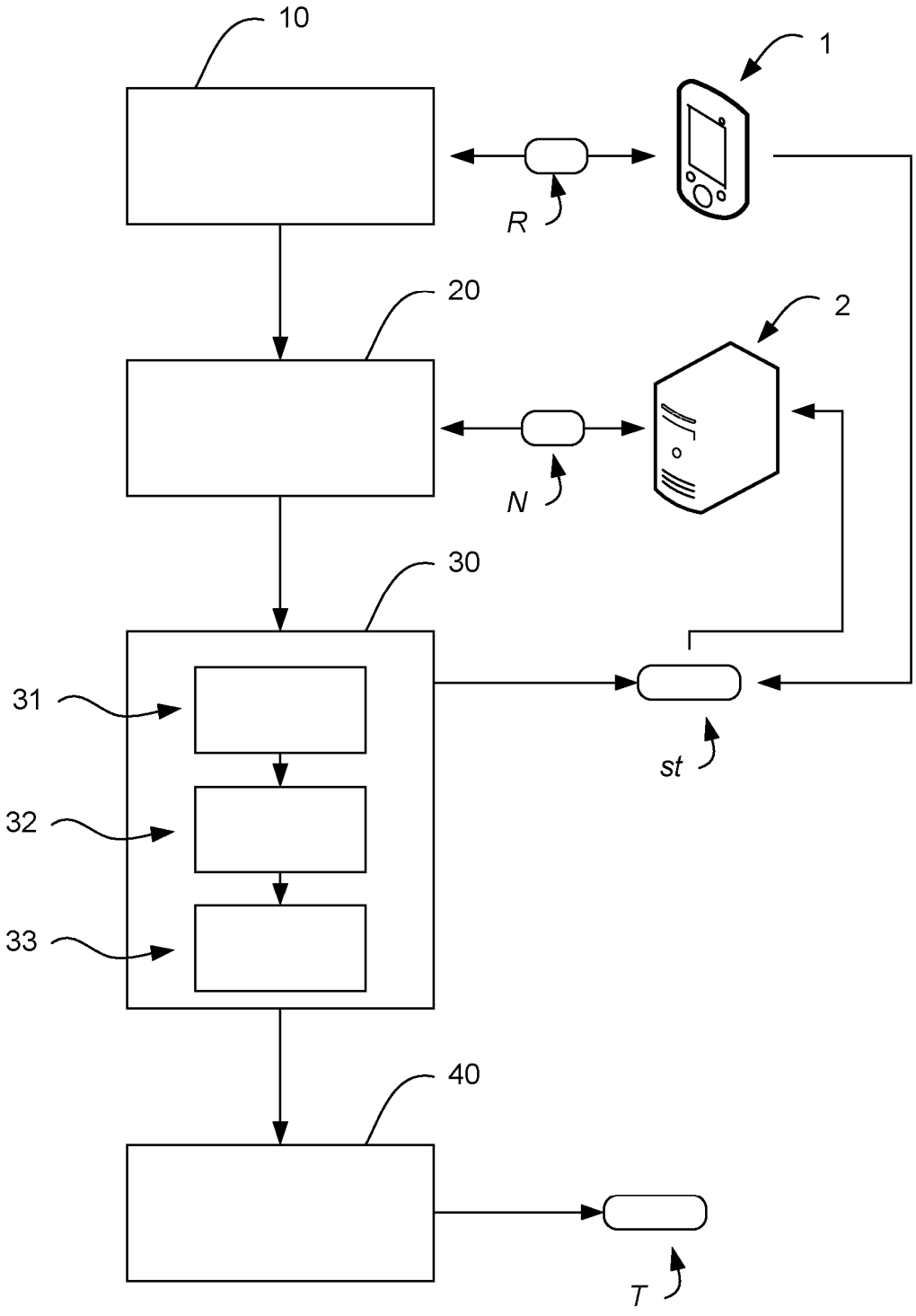

Figura 3

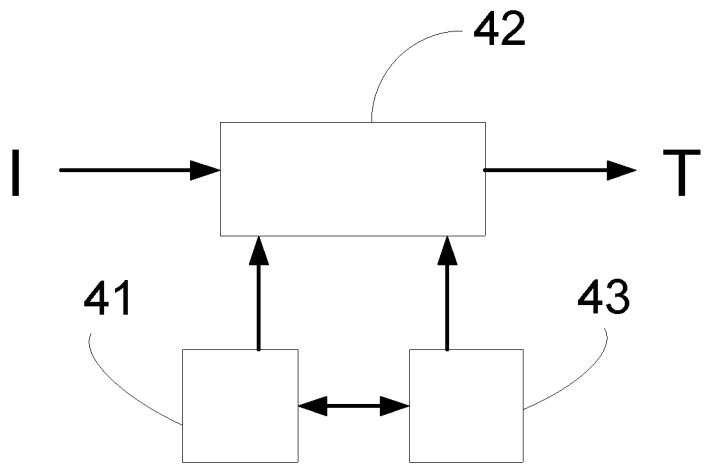

Figura 4

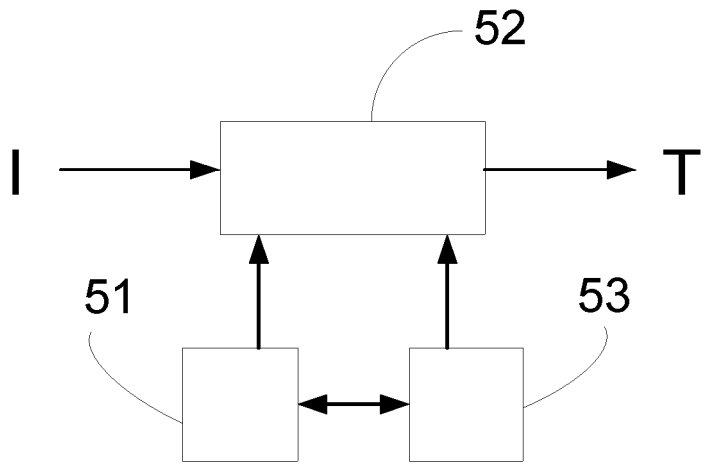

Figura 5