(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 800 038**

(51) Int. CI.:

H04L 9/00 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **12.10.2016 PCT/EP2016/074386**

(87) Fecha y número de publicación internacional: **20.04.2017 WO17064085**

(96) Fecha de presentación y número de la solicitud europea: **12.10.2016 E 16784814 (2)**

(97) Fecha y número de publicación de la concesión europea: **18.03.2020 EP 3363143**

(54) Título: **Método de consulta confidencial de una base de datos cifrada**

(30) Prioridad:

14.10.2015 FR 1559774

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
23.12.2020

(73) Titular/es:

COMMISSARIAT À L'ENERGIE ATOMIQUE ET AUX ENERGIES ALTERNATIVES (100.0%)
Bâtiment "Le Ponant D", 25, rue Leblanc
75015 Paris, FR

(72) Inventor/es:

CARPOV, SERGIU;
SIRDEY, RENAUD;
FAU, SIMON y
STAN, OANA

(74) Agente/Representante:

VALLEJO LÓPEZ, Juan Pedro

DESCRIPCIÓN

Método de consulta confidencial de una base de datos cifrada

Campo técnico

La presente invención se refiere al campo de la consulta confidencial de una base de datos cifrada. La invención utiliza criptografía totalmente homomórfica o FHE (Full *Homomorphic Encryption)* o incluso criptografía parcialmente homomórfica o SWHE *(SomeWhat Homomorphic Encryption).* Esta se aplica más particularmente a un contexto de informática en la nube *(cloud computing).*

Estado de la técnica anterior

El reciente desarrollo de la informática en la nube ha permitido a muchas empresas externalizar sus bases de datos a centros de datos *(data centers).* Sin embargo, esta nueva práctica, si bien permite reducir grandes inversiones en equipos informáticos, no está exenta de graves problemas de confidencialidad, tanto de los datos almacenados en la base como del contenido de las solicitudes de búsqueda. En la mayoría de los sectores de actividad, particularmente en el campo médico, financiero o económico, es esencial que se pueda garantizar la confidencialidad de los datos y del contenido de las solicitudes, no solo frente a un tercero malicioso, sino también frente al propio proveedor de servicios informáticos en la nube.

Para satisfacer esta exigencia de confidencialidad, es naturalmente necesario realizar un cifrado de todos o parte de los datos del usuario almacenados en la base. Sin embargo, un cifrado simple resulta insuficiente en la práctica en la medida en que la búsqueda de los registros que responden a una solicitud del usuario revela las ubicaciones donde estos se almacenan. Un tercero malintencionado, o incluso el propio proveedor, puede asociar entonces los registros cifrados con las solicitudes de los usuarios y deducir de ello información confidencial.

El artículo de S. Wang et al. titulado "Is homomorphic encryption the Holy Grail for data base queries on encrypted data ? ", Technical report, Universidad de California, Santa Barbara, 2012, propone un método de consulta confidencial de una base de datos cuyos registros están cifrados por medio de un criptosistema totalmente homomórfico.

En primer lugar, se recuerda que un cifrado totalmente homomórfico o FHE *(Full Homomorphic* Cryptography) es un cifrado asimétrico Enc_{pk} (de clave pública *pk)* que verifica las siguientes relaciones:

$$Enc_{pk}: X \to Y$$

$$\forall a, b \in X, Enc_{pk}\,(a+b) = Enc_{pk}\,(a) \oplus Enc_{pk}(b) \qquad (1\text{-}1)$$

$$\forall a, b \in X, Enc_{pk}\,(a.b) = Enc_{pk}\,(a) \otimes Enc_{pk}(b) \qquad (1\text{-}2)$$

donde X es el espacio de los mensajes sin cifrar (dicho de forma más sencilla espacio de claros) e *Y* es el espacio de los mensajes cifrados (dicho de forma más sencilla espacio de cifrados), + y son respectivamente una operación aditiva y una operación multiplicativa en el espacio de claros que confieren a X una estructura de anillo (X,+,.) $\oplus$ y $\otimes$ son operaciones correspondientes en el espacio de cifrados que confieren a *Y* una estructura de anillo (Y, $\oplus$ $\otimes$) De las expresiones (1-1) y (1-2) se entiende que la aplicación Enc_{pk} de (X,+,.) en (Y, $\oplus$, $\otimes$) es un homomorfismo de anillos.

En la práctica, un cifrado totalmente homomórfico es un cifrado probabilístico, es decir que depende de un parámetro aleatorio (o ruido) r. El cifrado de un mensaje sin cifrar *m* puede dar de este modo diferentes mensajes cifrados $Enc_{pk}(m,r)$ según el valor asumido por el parámetro r. No obstante, sea cual sea el valor asumido por este parámetro, el descifrado de $Enc_{pk}(m,r)$ siempre vuelve a dar el mensaje sin cifrar *m.* Si se denota Dec_{sk} la función de descifrado que corresponde a Enc_{pk} (donde *sk* es la clave secreta del usuario), por lo tanto tenemos:

$$Dec_{sk}\left(Enc_{pk}\,(m,r)\right) = m \qquad (2)$$

En lo siguiente, se adoptará una notación más simple para aligerar la exposición, es decir $\bar{m} = Enc_{pk}\,(m,r)$ y se convendrá en omitir en las expresiones de cifrado/descifrado la mención de las claves pública y privada. De este modo, tenemos $Dec(\bar{m}) = m$.

Se define un criptosistema por una para de una función de cifrado *Enc(.)* y de una función de descifrado *Dec(.).* De este modo, un criptosistema totalmente homomórfico, verificado con las notaciones anteriores:

$$Dec\left[\bar{a} \oplus \bar{b}\right] = a + b \qquad (3\text{-}1)$$

$$Dec\left[\bar{a} \otimes \bar{b}\right] = a.b \qquad (3\text{-}2)$$

Dicho de otra manera, un cifrado totalmente homomórfico permite calcular cualquier combinación de operaciones de suma y de multiplicación sin cifrar a partir de operaciones correspondientes en los cifrados. Como regla general, el espacio de claros es el cuerpo de los booleanos X=**Z**/2**Z**, siendo la operación aditiva un O exclusivo (XOR) y la operación multiplicativa un Y (AND). Entonces es posible realizar operaciones lógicas en valores booleanos a partir de operaciones correspondientes en sus valores cifrados.

Cabe señalar que cuando el método de cifrado solo permite que se calculen combinaciones de operaciones de suma y de multiplicación sin cifrar con una cierta profundidad de combinación, se prefiere utilizar el calificador "parcialmente homomórfico".

El artículo de S. Wang mencionado anteriormente propone calcular un booleano cifrado para cada registro de la base de datos, indicando este booleano si el registro en cuestión satisface o no la solicitud del usuario. Dado que el servidor no conoce el número de registros que satisfacen la solicitud, el usuario le proporciona un límite superior, M', del número de registros que le pueden ser devueltos, así como un parámetro de calidad de resultado γ. El servidor utiliza un búfer de tamaño fijo *B* proporcional a γ*M'* en el que almacena los registros, almacenándose cada registro del búfer de manera aleatoria entre las γ posiciones del búfer. Un registro que no responde a la solicitud se almacena en forma de un valor nulo cifrado ($\bar{0}$) y, por lo tanto, no tiene influencia sobre el descifrado. Por el contrario, un registro que corresponde a la solicitud se almacena y se podrá descifrar si no colisiona con otro registro. No obstante, este enfoque probabilístico no es del todo satisfactorio en la medida en que la respuesta del servidor puede no ser exhaustiva, dependiendo su grado de exhaustividad del parámetro de calidad γ.

Para remediar esta falta de exhaustividad, el artículo de M. Mani et al. titulado "Enabling secure database as a service using fully homomorphic properties: challenges and opportunities", publicado en arXiv preprint, 13 de febrero de 2013, págs. 1-13 propone calcular un booleano cifrado para cada registro de la base de datos, como anteriormente, y a continuación proceder en dos tiempos. En un primer momento, el servidor del proveedor de servicios determina el número $\bar{M}$ de registros que responden a la solicitud (número de "hits") sumando los booleanos cifrados, y los transmite al usuario. Esto descifra este número y solicita al servidor que le transmita *M'>M* primeros registros. El número *M* se transmite sin cifrar. El servidor ordena entonces de manera confidencial los registros según los valores de los booleanos cifrados, dicho de otra manera los M' primeros registros de la tabla ordenada de este modo contienen los *M* registros que satisfacen la solicitud. Este método permite ocultar el número *M* de registros que corresponden al resultado.

Este método de consulta confidencial preserva bien la confidencialidad del resultado pero presenta el inconveniente de requerir una planificación de toda la base de datos en cada solicitud.

El objetivo de la presente invención es, por consiguiente, proponer un método de consulta confidencial de una base de datos, cifrada por cifrado totalmente homomórfico o incluso parcialmente homomórfico, que remedia todos o parte de los inconvenientes mencionados anteriormente, en particular que presenta un alto grado confidencialidad sin aumentar significativamente la complejidad de los cálculos.

Exposición de la invención

La presente invención se define mediante un método de consulta confidencial de una base de datos alojada por un servidor, según la reivindicación independiente 1, conteniendo la base de datos una tabla de registros, obteniéndose cada registro por medio de un cifrado totalmente homomórfico o incluso parcialmente homomórfico de valores sin cifrar, en el que:

(a) el usuario transmite una solicitud *(R),* que comprende un predicado al servidor, siendo dicho predicado una expresión lógica relacionada con uno o varios campos de la tabla de registros;
(b) el servidor calcula el valor booleano cifrado del predicado para cada registro de la tabla; caracterizado por que
(c) el servidor construye un contenedor *(B_m)* que comprende un número predeterminado *(K)* de ubicaciones y almacena ciegamente en dichas ubicaciones registros que verifican el predicado y no previamente transmitidos al usuario, a partir de los valores booleanos cifrados ($\bar{r}_i$, *i = 1,...,N)* obtenidos en la etapa (b);
(d) el servidor transmite al usuario el contenedor construido de este modo;
(e) el usuario recibe el contenedor, descifra el contenido de cada ubicación, y determina si el contenedor está lleno o no;
(f1) si el contenedor está lleno, el usuario transmite una solicitud de continuación *(RC_m)* al servidor para una nueva iteración de las etapas (c), (d) y (e);
(f2) si el contenedor no está lleno, el usuario obtiene la respuesta a dicha solicitud a partir de los registros almacenados en los contenedores recibidos y descifrados en la etapa o en las etapas (e).

La tabla de registros está representada por una matriz $\overline{\mathrm{T}}$ de tamaño *N x P* donde *N* es el número de registros de la tabla y *P* el número de campos de estos registros, obteniéndose un elemento cifrado *A* de la matriz a partir de su

valor sin cifrar $A = \sum_{q=0}^{Q-1} a_q 2^q$ por $\bar{A} = \overline{a_0}, \overline{a_1}, \ldots, \overline{a_{Q-1}}$ donde a_q, q=0, ..., Q-1 son los bits del valor sin cifrar y $\overline{a_q}$, *q=0,...,Q-1* son sus cifrados correspondientes, obtenidos mediante dicho cifrado totalmente homomórfico o incluso parcialmente homomórfico.

El predicado se evalúa en los diferentes registros, normalmente por medio de operaciones aditivas $\oplus$ y multiplicativas $\otimes$ relacionadas con los elementos cifrados.

El contenedor se representa ventajosamente mediante una matriz $\overline{\mathbf{B}}$ de tamaño *K×P* con *K < N,* construyendo el servidor la matriz $\overline{\mathbf{B}}$ inicializando los elementos de esta matriz a cero y actualizando las filas de esta matriz de manera iterativa barriendo todos los registros de la tabla de registros, efectuándose dicha actualización por medio de una operación de asignación:

$$\overline{\boldsymbol{B}} = Aff\_row\ (\overline{\boldsymbol{B}}, \bar{c} \otimes \overline{\boldsymbol{t}_i}, \bar{k})$$

que asigna ciegamente el vector $\bar{c} \otimes \overline{\mathbf{t}_i}$ a la *k*ésima fila de $\overline{\mathbf{B}}$, donde $\overline{t_i}$ es un vector de fila de $\overline{\mathrm{T}}$ que representa un *i*ésimo registro barrido y $\bar{c}$ es un booleano cifrado.

La operación de asignación de un vector $\overline{\mathbf{u}}$ de *P* elementos cifrados a la késima fila de una matriz $\overline{\mathrm{H}} = (\overline{\mathrm{H}_{ij}})$ de elementos cifrados de tamaño *K x P* da una matriz $\overline{\mathrm{G}} = g_{ij})$ de elementos cifrados, del mismo tamaño, tal que $Dec(\overline{g_{ij}}) = Dec(\overline{h_{ij}})\ \forall i \neq k$ y $Dec(\overline{g_{kj}}) = Dec(\overline{u_j})$, $1 \leq J \leq P$.

El booleano cifrado se calcula por medio de $\bar{c} = \overline{r_i} \otimes (\overline{n_{last}} < \overline{i_{dx}}) \otimes (\overline{i_{dx}} \leq \overline{n_{last} + K})$ donde $\overline{r_i}$ es el valor booleano cifrado del predicado para el *i*ésimo registro barrido, $\overline{i_{dx}}$ una variable cifrada que da el número de registros ya barridos que satisfacen el predicado, n_{last} = (m - 1) *K* donde *m* -1 es el número de contenedores ya transmitidos por el servidor al usuario y $\overline{n_{last}}$, $\overline{n_{last}} + K$ cifrados respectivos de n_{last} y n_{last} + *K*.

La variable cifrada $\overline{i_{dx}}$ se actualiza ventajosamente con cada registro barrido mediante $\overline{i_{dx}} = \overline{i_{dx}} + \overline{r_i}$.

El índice cifrado $\bar{k}$ de la fila de la matriz $\overline{\mathbf{B}}$ se actualiza, por su parte, con cada registro barrido mediante $\bar{k} = \bar{k} + \bar{c}$.
Según una variante, la base de datos se divide en porciones de tamaño *NL* con la posible excepción de una porción, efectuándose las etapas (c), (d), (e), (f1)-(f2), en serie o en paralelo, en cada una de dichas porciones de la base.

El método de cifrado totalmente homomórfico puede utilizar por ejemplo un criptosistema de Brakerski.

Breve descripción de los dibujos

Surgirán otras características y ventajas de la invención con la lectura de un modo de realización preferido de la invención con referencia a las figuras adjuntas, en las que:

La figura 1 ilustra de manera esquemática la implementación de un método de consulta confidencial según un modo de realización de la invención;
La figura 2 representa un diagrama de flujo de la construcción de los contenedores para el método de consulta confidencial según un modo de realización de la invención;
La figura 3 representa de manera esquemática un diagrama de flujo del método de consulta confidencial de una base de datos según un modo de realización de la invención.

Exposición detallada de modos de realización particulares

El método de consulta confidencial de una base de datos, cifrada por cifrado totalmente homomórfico o incluso parcialmente homomórfico, según la presente invención se describirá a continuación en el contexto de una arquitectura cliente-servidor ilustrada en la figura 1.

Sin pérdida de generalidad, se supondrá en la siguiente descripción que el método de cifrado es totalmente homomórfico. Se entenderá, en efecto, que siempre será posible elegir un método de cifrado parcialmente homomórfico en la medida en que esto permita calcular combinaciones de operaciones de multiplicación y de suma sin cifrar con suficiente profundidad.

Se ha representado en 110 un servidor, por ejemplo un servidor de proveedor de servicios informáticos en la nube. Este servidor aloja una base de datos 130, por ejemplo una base de datos relacional. Se supondrá, en cualquier caso, que la base de datos comprende una tabla $\overline{\mathbf{T}}$ (por ejemplo una relación en una base de datos relacional) constituida por *N* registros cifrados. Más concretamente, cada registro consta de *P* campos y cada campo está cifrado por medio de un criptosistema totalmente homomórfico como se explica a continuación. Cada registro puede,

por lo tanto, considerarse como una *P*-upla de valores cifrados o de manera equivalente como un vector de tamaño *P* cuyos elementos están cifrados. De manera similar, la tabla $\overline{\mathbf{T}}$ puede ser considerada como una matriz de tamaño *N x P* de elementos cifrados, correspondiendo las filas a los diferentes registros y las columnas a los diferentes campos.

Sin pérdida de generalidad, se supondrá en lo sucesivo que el valor sin cifrar, *A,* de un campo puede estar representado por una palabra de Q bits, Dicho de otra manera: $A=\sum_{q=0}^{Q-1} a_q 2^q$, estando el espacio de claros constituido por valores binarios {0,1}, dicho de otra manera X = **Z**/2**Z**. Cada bit a_q se cifra por medio del criptosistema totalmente homomórfico mencionado anteriormente, por ejemplo el criptosistema de Brakerski cuya descripción se encontrará en el artículo de Z. Brakerski et al. titulado "(Leveled) fully homomorphic encryption without bootstrapping" publicado en Proc. of ITCS 2012, págs. 309-325. El valor cifrado $\overline{a_q}$ = *Enc(a*$_q$*)* es una palabra binaria y se denotará $\bar{A}$ la secuencia de las palabras $\overline{a_q}$, *q=0,..,Q-1,* dicho de otra manera:

$$\bar{A} \equiv \overline{a_0}, \overline{a_1}, ..., \overline{a_{Q-1}} \qquad (4)$$

$\bar{A}$ se denominará por abuso del lenguaje "cifrado" de A.

De la misma forma, se denotará recíprocamente *Dec(*$\bar{A}$*),* la palabra de *Q* bits definida por:

$$Dec\left(\bar{A}\right) = \sum_{q=0}^{Q-1} Dec\left(\overline{a_q}\right) 2^q \qquad (5)$$

Dec($\bar{A}$*)* se denominará por abuso del lenguaje "descifrado" de $\bar{A}$.

Volviendo a la figura 1, un usuario transmite al servidor una solicitud, *R,* por ejemplo una solicitud SQL *(Structured Query Language)* que consta, de manera general, de un predicado, es decir una expresión lógica relacionada con uno o varios campos de la tabla $\bar{T}$ y que puede asumir, según los registros, un valor verdadero (TRUE) o falso (FALSE). El predicado puede comprender operadores booleanos AND, OR, NOT y/u operadores de comparación (<, ≤, >, ≥) o incluso un operador de igualdad (==).

El servidor evalúa el predicado (es decir calcula su valor lógico) en los diferentes registros de la tabla $\bar{T}$ y forma un vector $\bar{r}$ de tamaño *N* cuyos elementos son booleanos que representan los valores lógicos así obtenido. Es esencial señalar que la evaluación del predicado se efectúa en cifrados y sin cifrar, y que los booleanos que dan los valores lógicos de los diferentes registros también están cifrados. En otras palabras, el servidor no puede distinguir en la tabla los registros que corresponden a un valor verdadero y los que corresponden a un valor falso del predicado. Cabe destacar que el servidor no tiene más acceso al número de resultados, es decir al número de registros *M* que dan lugar a un valor verdadero del predicado.

El método de consulta confidencial según la invención implementa un contenedor, también llamado cubo *(bucket),* de tamaño fijo, que puede contener hasta *K* registros, por medio del cual los resultados de la solicitud son devueltos al usuario. Más concretamente, el servidor revisa todos los registros de la tabla y transfiere ciegamente al contenedor lo que satisfacen la solicitud, es decir aquellos para los cuales el valor del predicado es verdadero. La transferencia a ciegas se efectúa por medio de una operación de asignación detallada más adelante. Esta garantiza que el servidor (o un tercero malintencionado que efectúa un ataque sobre el servidor) no pueda determinar los registros efectivamente transferidos al contenedor. Las ubicaciones del contenedor a las cuales no ha sido transferido un registro contienen valores cifrados $\bar{0}$ (que indican convencionalmente que la ubicación está vacía).

La respuesta del servidor al usuario se distribuye en un número $\tilde{m}$ de contenedores, $B_1,...,B_{\tilde{m}}$ donde $\tilde{m}$ = ⌈M/K⌉ es la parte entera por exceso de la fracción M/K. El servidor no puede determinar de antemano el número *m* de contenedores ya que el número *M* de resultados no le es revelado. Para cada contenedor B_m recibido, el usuario descifra el contenido de las diferentes ubicaciones del contenedor. Si el usuario llega a una ubicación que solamente contiene 0 entonces sabe que ha obtenido todos los resultados. En sentido contrario, si dicha ubicación no es detectada, el usuario reenvía al servidor una solicitud de continuación, RC_m. La transferencia de los resultados se realiza de este modo por paso de contenedores (o cubos) a petición del usuario. Al final, el servidor conoce indirectamente el número de contenedores transferidos para una solicitud es decir el valor $\tilde{m}$ pero solo conoce el número de resultados dentro de un número *K* - 1 de registros.

Se ha representado en la figura, la solicitud inicial R del usuario y sus solicitudes de continuación RC_m, *m=1,..,m-* 1 sucesivas. En respuesta a las solicitudes *R,* $RC_1,...,RC_{\tilde{m}-1}$ el servidor transmite respectivamente los contenedores

B_m, $m=1,..,\tilde{m}$, como se explica a continuación.

En respuesta a la solicitud *R* del usuario, el servidor evalúa el predicado en cada uno de los registros de la tabla T. Esta evaluación se realiza, por supuesto, en los valores cifrados. Por ejemplo, si la solicitud consiste en obtener todos los registros tales que el contenido *F* de un campo es igual a un valor cifrado dado $\overline{D}$, es decir si el predicado es $\overline{F} = \overline{D}$, donde $\overline{f_{Q-1}}...\overline{f_1 f_0}$ et $\overline{d_{Q-1}}... \overline{d_1 d_0}$ son las representaciones binarias respectivas de $\overline{F}$ y $\overline{D}$, tendremos:

$$\left(\overline{F} == \overline{D}\right) = \bigotimes_{q=0}^{Q-1}\left(\overline{1} \oplus \overline{f_q} \oplus \overline{d_q}\right) \qquad (6)$$

Dicho de otra manera, la evaluación del predicado ($\overline{F} = \overline{D}$) da un valor booleano cifrado igual a $\overline{1}$ (valor lógico TRUE) si el campo $\overline{F}$ del registro es igual a $\overline{D}$ e igual a $\overline{0}$ (valor lógico FALSE) en el caso contrario.

Podrán utilizarse otros predicados que implementan operadores de comparación y/o y operadores aritméticos ($\oplus$ y $\otimes$). De manera general, el valor lógico de un predicado se puede obtener a partir de operaciones $\oplus$ y $\otimes$ en los elementos cifrados de un registro. Se encontrará por ejemplo el cálculo de un predicado de comparación en el artículo de M. Mani *et al.* mencionado anteriormente.

En cualquier caso, el servidor evalúa el predicado en cada registro *i=1,...,N* de la tabla y deduce de ello un vector $\bar{r}$ de tamaño *N* cuyos elementos $\bar{r}_i$, son los resultados de esta evaluación.

A continuación se define un operador de asignación condicional *Sel* que permite asignar una primera palabra cifrada $\bar{A}$ o una segunda palabra cifrada $\bar{B}$ a una variable $\bar{V}$ según el valor de un booleano cifrado $\bar{c}$, de la siguiente manera:

$$\overline{V} = Sel\left(\overline{A}, \overline{B}; \overline{c}\right)$$

$$\overline{v_q} = \left(\overline{c} \otimes \overline{a_q}\right) \oplus \left(\left(\overline{1} \oplus \overline{c}\right) \otimes \overline{b_q}\right),\ q = 0,...,Q-1 \qquad (7)$$

Se entiende de la expresión (7) que la palabra cifrada *A* se asigna a *V* si *c* es un cifrado de 1 y que la palabra cifrada $\bar{B}$ se asigna a $\bar{V}$ si $\bar{c}$ es un cifrado de 0. Es esencial señalar que esta asignación se realiza ciegamente, es decir que el servidor que calcula la expresión (7) no tiene ningún medio de determinar cuál de las palabras $\bar{A}$ y $\bar{B}$ es asignada efectivamente a $\bar{V}$. En efecto, en ausencia de la clave de descifrado, el servidor no sabe si $\bar{c}$ es un cifrado de 1 o 0. Además, al estar la palabra cifrada asignada modificada por el cálculo de la expresión (7), no es posible deducir esta asignación de una simple comparación de $\bar{V}$ con $\bar{A}$ y $\bar{B}$.

El operador de asignación *Sel* permite asignar ciegamente un valor cifrado a un elemento de un vector. Más concretamente, si se denota $\bar{\mathbf{v}} = (\overline{v_1},..., \overline{v_P})$ un vector de cifrado y *k* el cifrado de un índice k, $1 \leq k \leq P$, se puede asignar ciegamente un valor cifrado *u* al elemento de índice *k* del vector v para generar un nuevo vector $\bar{\mathbf{w}} = (\overline{w_1},..., \overline{v_P})$, tal que:

$$Dec(\overline{w_i}) = Dec\left(\overline{v_i}\right),\ \forall i \neq k \quad \text{y} \quad Dec(\overline{w_k}) = Dec\left(\overline{u}\right) \qquad (8)$$

Esta asignación puede obtenerse ventajosamente por medio de:

$$\overline{w_i} = Sel\left(\overline{v_i}, \overline{u}; \left(\overline{i} == \overline{k}\right)\right),\ i = 1,..., P \qquad (9)$$

Al estar el índice $\bar{k}$ cifrado, el servidor que efectúa el cálculo de la expresión (9) no sabrá determinar a qué elemento del vector v se le habrá asignado la palabra cifrada $\bar{u}$. Además, el descifrado de los elementos del vector w proporcionará los mismos elementos descifrados que para el vector v, excepto el descifrado del elemento de índice *k* que dará la palabra u. Cabe señalar que el cálculo de la expresión ($\bar{9}$) requiere simplemente conocer la clave (pública) de cifrado para obtener los índices cifrados $\bar{i}$, $\bar{i}$ = 1,...,P.

Según el mismo principio, es posible asignar ciegamente una palabra cifrada *u* a un elemento cualquiera de índices *k,ℓ*en una matriz $\bar{\mathbf{H}}$ (de tamaño K×P) de cifrados$\bar{\mathbf{H}}$ = $(\overline{h_{ij}})$, $1 \leq i \leq K$, $1 \leq j \leq P$ para obtener una matriz del mismo tamaño $\bar{\mathbf{G}}$ = $(\bar{\mathbf{G}}_{ij})$, $1 \leq i \leq K$, $1 \leq j \leq$ P:

$$\overline{g_{ij}} = Sel\left(\overline{h_{ij}}, \overline{u}; \left(\overline{i} == \overline{k}\right) \otimes \left(\overline{j} == \overline{\ell}\right)\right) \quad (10)$$

La matriz $\overline{\mathbf{G}}$ verifica entonces:

$$Dec\left(\overline{g_{ij}}\right) = Dec\left(\overline{h_{ij}}\right) \ \forall (i,j) \neq (k,\ell) \quad \text{y} \quad Dec\left(\overline{g_{k\ell}}\right) = Dec\left(\overline{u}\right) \quad (11)$$

Dicho de otra manera, el descifrado de los elementos de la matriz $\overline{\mathbf{G}}$ proporcionará los mismos elementos descifrados que para la matriz $\overline{\mathbf{H}}$, excepto el descifrado del elemento de índices *k,ℓ* que dará la palabra u. La expresión de asignación (10) se denotará en lo sucesivo de manera más compacta:

$$\overline{\mathbf{G}} = Aff\left(\overline{\mathbf{H}}, \overline{u}; \left(\overline{k}, \overline{\ell}\right)\right) \quad (12)$$

La asignación puede referirse a una fila completa o una columna completa de la matriz $\overline{\mathbf{H}}$. Por ejemplo, si se denota $\overline{\mathbf{u}}$ =(u_1,...,u_P) un vector de palabras cifradas a asignar respectivamente a los *P* elementos de la k[ésima] fila de la matriz $\overline{\mathbf{H}}$, se puede definir una nueva matriz $\overline{\mathbf{G}}$:

$$\overline{g_{ij}} = Sel\left(\overline{h_{ij}}, \overline{u_j}; \left(\overline{i} == \overline{k}\right)\right) \ 1 \leq i \leq K \ , \ 1 \leq j \leq P \quad (13)$$

Aún aquí, al estar el índice $\overline{k}$ cifrado, el procesador que efectúa el cálculo de la expresión (13) no sabe qué fila de la matriz ha sido objeto de una asignación y el descifrado da:

$$Dec\left(\overline{g_{ij}}\right) = Dec\left(\overline{h_{ij}}\right) \ \forall i \neq k \quad \text{y} \quad Dec\left(\overline{g_{kj}}\right) = Dec\left(\overline{u_j}\right), \ 1 \leq j \leq P \quad (14)$$

Dicho de otra manera, Dicho de otro modo, el descifrado de los elementos de la matriz G proporcionará los mismos elementos descifrados que para la matriz **H**, excepto el descifrado de los elementos de la fila *k* que volverán a dar los elementos del vector $\overline{\mathbf{u}}$. La expresión (14) se denotará en lo sucesivo de manera más compacta:

$$\overline{\mathbf{G}} = Aff\_row\left(\overline{\mathbf{H}}, \overline{\mathbf{u}}; \overline{k}\right) \quad (15)$$

La operación de asignación *Aff_row* permite al servidor extraer a ciegas los registros que satisfacen la solicitud del usuario y cambiarlos en contenedores sucesivos como se ha explicado en relación con la figura 2.

Se supone en esta figura que el servidor ha recibido una solicitud de continuación RC_m por parte del usuario y que debe, por lo tanto, construir el contenedor B_m.

Este contenedor puede ser considerado como una matriz $\overline{\mathbf{B}}$ de tamaño *K* × *P.* Se recuerda aquí que *K* es el número de registros que pueden ser almacenados en el contenedor y *P* el número de campos. Se recuerda también que la tabla de los registros puede considerarse como una matriz $\overline{\mathbf{T}}$ de tamaño *N* × *P*: $\overline{\mathbf{T}}$ = (t_{ij}), *i=1....,N,j=1,...,P.* Se supone que *K* < *N,* incluso *K* << *N.*

En la etapa 210, la matriz B es inicializada por la matriz $\mathbf{0}_{KxP}$ de la cual todos los elementos son nulos.

En la etapa 220, se determina n_{last} = (m - 1).K el índice del último registro transmitido al usuario. El servidor sabe, en efecto, que ya ha reenviado al usuario n_{last} registros que satisfacen la solicitud. Cabe destacar, en cambio, que el servidor no sabe dónde se encuentra este último registro en la tabla $\overline{\mathbf{T}}$.

En la etapa 230, se inicializa un contador de registros en la tabla $\overline{\mathbf{T}}$, es decir *i* = 1. Cabe destacar que este contador está sin cifrar. Sirve para barrer los registros sucesivos de la tabla.

En la etapa 240, el servidor inicializa una variable $\overline{i_{dx}}$ que da el número de registros barridos que satisfacen el predicado, es decir $\overline{i_{dx}} = \overline{0}$. Cabe destacar que esta variable está cifrada pero que su valor inicial nulo es conocido por el servidor.

En la etapa 250, el servidor inicializa un contador $\overline{k}$ que apunta a la fila actual del contenedor, $\overline{k} = \overline{1}$. Aún aquí, el contador está cifrado pero su valor inicial es conocido por el servidor.

A continuación se barren la totalidad de los registros de la tabla $\overline{\mathbf{T}}$, es decir se entra en un bucle en el que se repiten las siguientes etapas para i que va de 1 a *N*:
En la etapa 260, se calcula el booleano cifrado:

$$\overline{c} = \overline{r_i} \otimes \left(\overline{n_{last}} < \overline{i_{dx}}\right) \otimes \left(\overline{i_{dx}} \leq \overline{n_{last} + K}\right) \quad (16)$$

Dicho de otra manera, se determina ciegamente si el i^{ésimo} registro de la tabla $\overline{\mathbf{T}}$ satisface la solicitud (término r_i, y si de hecho forma parte de los n_{last} +1 a n_{last} + *K* registros que pueden ser transferidos al contenedor actual (término $(\overline{n_{last}} < \overline{i_{dx}}) \otimes (\overline{i_{dx}} \leq \overline{n_{last} + K})$). Esta última prueba es necesaria en la medida en que se barre la totalidad de la tabla ciegamente.

En la etapa 270, se transfiere el i^{ésimo} registro de la tabla $\overline{\mathbf{T}}$ a la *k^{ésima}* ubicación del contenedor, actualizando la matriz $\overline{\mathbf{B}}$ por medio de la operación de asignación:

$$\overline{\mathbf{B}} = Aff\_row\left(\overline{\mathbf{B}}, \overline{c} \otimes \overline{\mathbf{t}_i}; \overline{k}\right) \quad (17)$$

donde se ha denotado $\overline{t_i}$ el vector de tamaño *P* que representa el i^{ésimo} registro de la tabla $\overline{\mathbf{T}}$ y donde $\overline{c} \otimes \overline{\boldsymbol{t}_i}$es el vector definido por:

$$\overline{c} \otimes \overline{\mathbf{t}_\mathbf{i}} = \left(\overline{c} \otimes t_{i1}, \ldots, \overline{c} \otimes t_{iP}\right) \quad (18)$$

En la etapa 280, se actualiza el número de registros barridos que satisfacen la solicitud, es decir:

$$\overline{i_{dx}} = \overline{i_{dx}} + \overline{r_i} \quad (19)$$

En la etapa 290, se actualiza el puntero de fila en el contenedor por:

$$\overline{k} = \overline{k} + \overline{c} \quad (20)$$

Dicho de otra manera, se incrementa (de manera oculta) el índice *k* solo en la medida en que el i^{ésimo} registro es a la vez un registro que satisface la solicitud y está destinado a ser almacenado en el contenedor actual.

Un ejemplo numérico se da a continuación para ilustrar el procedimiento de construcción de los contenedores. La solicitud *R* del usuario consiste en este contexto a extraer todos los registros de la base comenzando por el octeto (cifrado) $\overline{192}$.

Se supone que la tabla $\overline{\mathbf{T}}$ está constituida por cinco registros compuestos por cuatro campos, es decir:

i/p	1	2	3	4	$\overline{r_i}$	$\overline{i_{dx}}$
1	$\overline{192}$	$\overline{168}$	$\overline{132}$	$\overline{20}$	$\overline{1}$	$\overline{1}$
2	$\overline{201}$	$\overline{141}$	$\overline{132}$	$\overline{1}$	$\overline{0}$	$\overline{1}$
3	$\overline{192}$	$\overline{168}$	$\overline{201}$	$\overline{20}$	$\overline{1}$	$\overline{2}$
4	$\overline{121}$	$\overline{42}$	$\overline{2}$	$\overline{255}$	$\overline{0}$	$\overline{2}$
5	$\overline{192}$	$\overline{178}$	$\overline{101}$	$\overline{2}$	$\overline{1}$	$\overline{3}$

La primera columna de la tabla proporciona los índices de los registros, la penúltima contiene los booleanos cifrados $\overline{r_i}$ y la última los booleanos cifrados $\overline{i_{dx}}$.

Se supone que el contenedor es de tamaño *K* = 2. En ese caso, el primer contenedor reenviado por el servidor es el siguiente:

$\overline{192}$	$\overline{168}$	$\overline{132}$	$\overline{20}$
$\overline{192}$	$\overline{168}$	$\overline{201}$	$\overline{20}$

En la recepción de este contenedor, el usuario descifra los (campos de los) diferentes registros que están almacenados en él. Al no detectar el usuario fila nula transmite una primera solicitud de continuación RC_1 al servidor. Este barre de nuevo los cinco registros de la tabla y solamente transfiere al segundo contenedor aquellos que no han sido ya almacenados en el primero. El servidor construye entonces el segundo contenedor:

$\overline{192}$	$\overline{178}$	$\overline{101}$	$\overline{2}$
$\overline{0}$	$\overline{0}$	$\overline{0}$	$\overline{0}$

El usuario descifra los (campos de los) diferentes registros y detecta la presencia de una fila nula, es decir que solamente comprenden valores nulos cifrados. De ello deduce que todos los registros que satisfacen la solicitud ya le han sido transmitidos.

Se entiende que el método de construcción de contenedores descrito anteriormente garantiza que el servidor solamente tiene acceso a un límite superior del número de resultados ($\tilde{m}$K). El tamaño *K* del contenedor es un compromiso entre el grado de ambigüedad aceptable en el número de resultados, por un lado, y la necesidad de recursos de comunicación (entre el usuario y el servidor), por otro lado. En efecto, una solución trivial, pero sin interés, sería elegir *K* = *N,* es decir cargar la totalidad de la tabla $\overline{\mathbf{T}}$ en el contenedor. La ambigüedad en el número de resultados sería entonces máxima pero el número de registros transmitidos inútilmente también sería máximo (N - *M).* Por el contrario, para un tamaño pequeño *K* del contenedor, la ambigüedad en el número de resultados sería menor pero el número de registros transmitidos inútilmente sería también menor *($\tilde{m}$K* - M).

Ya se ha señalado anteriormente que la construcción de cada contenedor necesitaría barrer la totalidad de la tabla $\overline{\mathbf{T}}$. Para simplificar los cálculos, se puede dividir la tabla en *L* porciones de tamaño igual a *N/L* (excepto la última porción de tamaño N-⌊N/L⌋. *L*), aplicándose el procedimiento de extracción a cada porción de la tabla. La complejidad de construcción de un contenedor se reduce de este modo en un factor L. Como contrapartida, la transferencia de resultados moviliza a menos *L* veces más recursos de comunicación en la medida en que las *L* porciones de la tabla serán barridas cada una al menos una vez. En definitiva, la elección del tamaño *K* del contenedor y del factor de división *L* resulta de un compromiso entre el grado de ambigüedad requerido, los recursos de cálculo y los recursos de comunicación disponibles.

Se ha representado esquemáticamente en la figura 3 un diagrama de flujo de un método de consulta confidencial de una base de datos según un modo de realización de la invención.

Los intercambios entre el usuario (cliente) y el servidor (del proveedor de servicios) son los ya ilustrados en la figura 1.

En la etapa 310, el usuario transmite una solicitud (R), que comprende un predicado, al servidor.

En la etapa 320, el servidor calcula el valor booleano cifrado del predicado para cada registro de la tabla $\overline{\mathbf{T}}$, dicho de otra manera, calcula los valores $\bar{r}_i$, *i* = 1,...N.

En la etapa 325, el índice del contenedor se inicializa, es decir *m* = 1.

En la etapa 330, el servidor construye un contenedor B_m que comprende un número predeterminado *K* de ubicaciones y almacena ciegamente en estas ubicaciones registros que verifican el predicado y no previamente transmitidos al usuario. La construcción del contenedor se efectúa como se ha explicado anteriormente en relación con la figura 2.

En la etapa 340, el servidor transmite al usuario el contenedor B_m.

En la etapa 350, el usuario descifra el contenido de cada ubicación del contenedor B_m, dicho de otro modo cada elemento de la matriz B.

En la etapa 360, el usuario determina si una fila nula está o no presente en el contenedor B_m. De ello deduce así que el contenedor está lleno o no.

Si se detecta dicha fila, el usuario deduce de ello que ha recibido todos los registros que satisfacen la solicitud. En efecto, se comprenderá que el último contenedor $B_{\tilde{m}}$ consta de *K* - (*M* - $\tilde{m}$*K)* filas nulas mientras que los contenedores anteriores B_m, m=1,...., $\tilde{m}$-1 no constan de ninguna. El usuario obtiene en 370 la respuesta a la solicitud *R* a partir de los registros almacenados en los contenedores $B_{\tilde{m}}$, *m=1,....,* $\tilde{m}$, descifrados anteriormente en la etapa 350. El procedimiento de la solicitud se completa en 375.

A la inversa, si el usuario no detecta dicha fila nula después del descifrado, transmite al servidor una solicitud de continuación RC_m en 380 e incrementa el índice del contenedor, *m,* en 385 antes de volver a la etapa 330.

REIVINDICACIONES

1. Método de consulta confidencial de una base de datos alojada por un servidor, conteniendo la base de datos una tabla de registros, obteniéndose cada registro por medio de un cifrado totalmente homomórfico o incluso parcialmente homomórfico de valores sin cifrar, en el que:

(a) el usuario transmite (310) una solicitud (R), que comprende un predicado al servidor, siendo dicho predicado una expresión lógica relacionada con uno o varios campos de la tabla de registros;
(b) el servidor (320) calcula el valor booleano cifrado del predicado para cada registro de la tabla;

caracterizado por que:

(c) el servidor (330) construye un contenedor (B_m) que comprende un número predeterminado (K) de ubicaciones y almacena ciegamente en dichas ubicaciones registros que verifican el predicado y no previamente transmitidos al usuario, a partir de los valores booleanos cifrados ($\overline{r_i}$, i =1,...,N) obtenidos en la etapa (b);
(d) el servidor transmite (340) al usuario el contenedor construido de este modo;
(e) el usuario recibe el contenedor, descifra el contenido de cada ubicación (350), y determina si el contenedor está lleno o no (360);
(f1) si el contenedor está lleno, el usuario transmite (380) una solicitud de continuación *(RC_m)* al servidor para una nueva iteración de las etapas (c),(d) y (e);
(f2) si el contenedor no está lleno, el usuario obtiene la respuesta a dicha solicitud (370) a partir de los registros almacenados en los contenedores recibidos y descifrados en la etapa o en las etapas (e).

2. Método de consulta confidencial de una base de datos según la reivindicación 1, **caracterizado por que** la tabla de registros está representada por una matriz $\overline{\mathbf{T}}$ de tamaño *N x P* donde *N* es el número de registros de la tabla y *P* el número de campos de estos registros, obteniéndose un elemento cifrado $\bar{A}$ de la matriz a

partir de su valor sin cifrar $A = \sum_{q=0}^{Q-1} a_q 2^q$ por $\bar{A} = \overline{a_0}, \overline{a_1}, \ldots, \overline{a_{Q-1}}$ donde a_q, *q=0,...,Q-1* son los bits del valor sin cifrar y $\overline{a_Q}$, *q=0,...,Q-1* son sus cifrados correspondientes, obtenidos mediante dicho cifrado totalmente homomórfico o incluso parcialmente homomórfico.

3. Método de consulta confidencial de una base de datos según la reivindicación 2 **caracterizado por que** el predicado se evalúa en los diferentes registros por medio de operaciones aditivas $\oplus$ y multiplicativas $\otimes$ relacionadas con los elementos cifrados.

4. Método de consulta confidencial de una base de datos según la reivindicación 3, **caracterizado por que** el contenedor está representado por una matriz $\overline{\mathrm{B}}$ de tamaño *K x P* con *K < N,* construyendo el servidor la matriz $\overline{\mathrm{B}}$ inicializando los elementos de esta matriz a cero y actualizando las filas de esta matriz de manera iterativa barriendo todos los registros de la tabla de registros, efectuándose dicha actualización por medio de una operación de asignación:

$$\overline{\mathbf{B}} = Aff\_row\left(\overline{\mathbf{B}}, \overline{c} \otimes \overline{\mathbf{t}_i}; \overline{k}\right)$$

que asigna ciegamente el vector $\overline{\mathrm{c}} \otimes \overline{\boldsymbol{t}_i}$ a la *k*^{ésima} fila de $\overline{\mathrm{B}}$, donde $\overline{\boldsymbol{t}_i}$ es un vector de fila de T que representa un *i*^{ésimo} registro barrido y *c* es un booleano cifrado.

5. Método de consulta confidencial de una base de datos según la reivindicación 4, caracterizado por que la operación de asignación de un vector u de *P* elementos cifrados a la k^{ésima} fila de una matriz H = (h_{ij} de elementos cifrados de tamaño *K* x *P* da una matriz G = (g_{ij}) de elementos cifrados, del mismo tamaño, tal que $Dec(\overline{g_{ij}}) = Dec(\overline{h_{ij}})$ $\forall i \neq k$
y $Dec(\overline{g_{kj}}) = Dec((\overline{u}_j)$, $1 \leq j \leq P$.

6. Método de consulta confidencial de una base de datos según las reivindicaciones 4 o 5, caracterizado por que el booleano cifrado se calcula por medio de $\overline{\mathrm{c}} = \overline{r_i} \otimes (\overline{n_{last}} < \overline{i_{dx}}) \otimes (\overline{i_{dx}} \leq \overline{n_{last} + K})$ donde r_i es el valor booleano cifrado del predicado para el *i*^{ésimo} registro barrido, i_{dx} una variable cifrada que da el número de registros ya barridos que satisfacen el predicado, n_{last} = (m-1)K donde *m* -1 es el número de contenedores ya transmitidos por el servidor al usuario y $\overline{n_{last}}$, $\overline{n_{last}} + K$ cifrados respectivos de n_{last} y n_{last} + *K*.

7. Método de consulta confidencial de una base de datos según la reivindicación 6, caracterizado por que la variable cifrada $\overline{i_{dx}}$ se actualiza con cada registro barrido mediante $\overline{i_{dx}} = \overline{i_{dx}} + \overline{r_i}$.

8. Método de consulta confidencial de una base de datos según las reivindicaciones 6 o 7, caracterizado por que el

índice cifrado $\bar{k}$ de la fila de la matriz $\bar{\mathbf{B}}$ se actualiza con cada registro barrido mediante $\bar{k} = \bar{k} + \bar{c}$.

9. Método de consulta confidencial de una base de datos según una cualquiera de las reivindicaciones anteriores, **caracterizado por que** la base de datos se divide en porciones de tamaño *N/L* con la posible excepción de una porción, efectuándose las etapas (c),(d),(e), (f1)-(f2), en serie o en paralelo, en cada una de dichas porciones de la base.

10. Método de consulta confidencial de una base de datos según una cualquiera de las reivindicaciones anteriores, **caracterizado por que** el método de cifrado totalmente homomórfico utiliza un criptosistema de Brakerski.

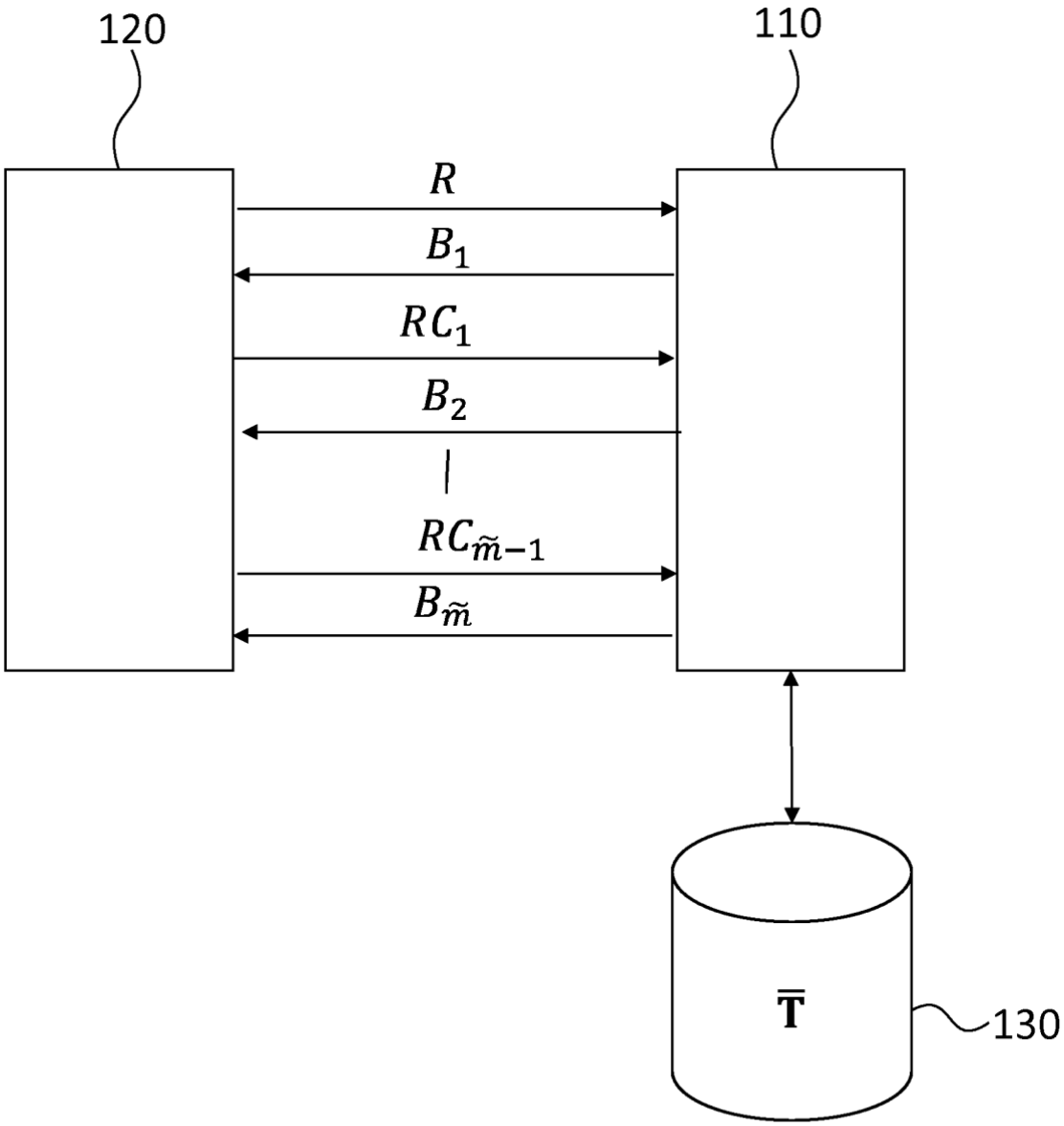

Fig. 1

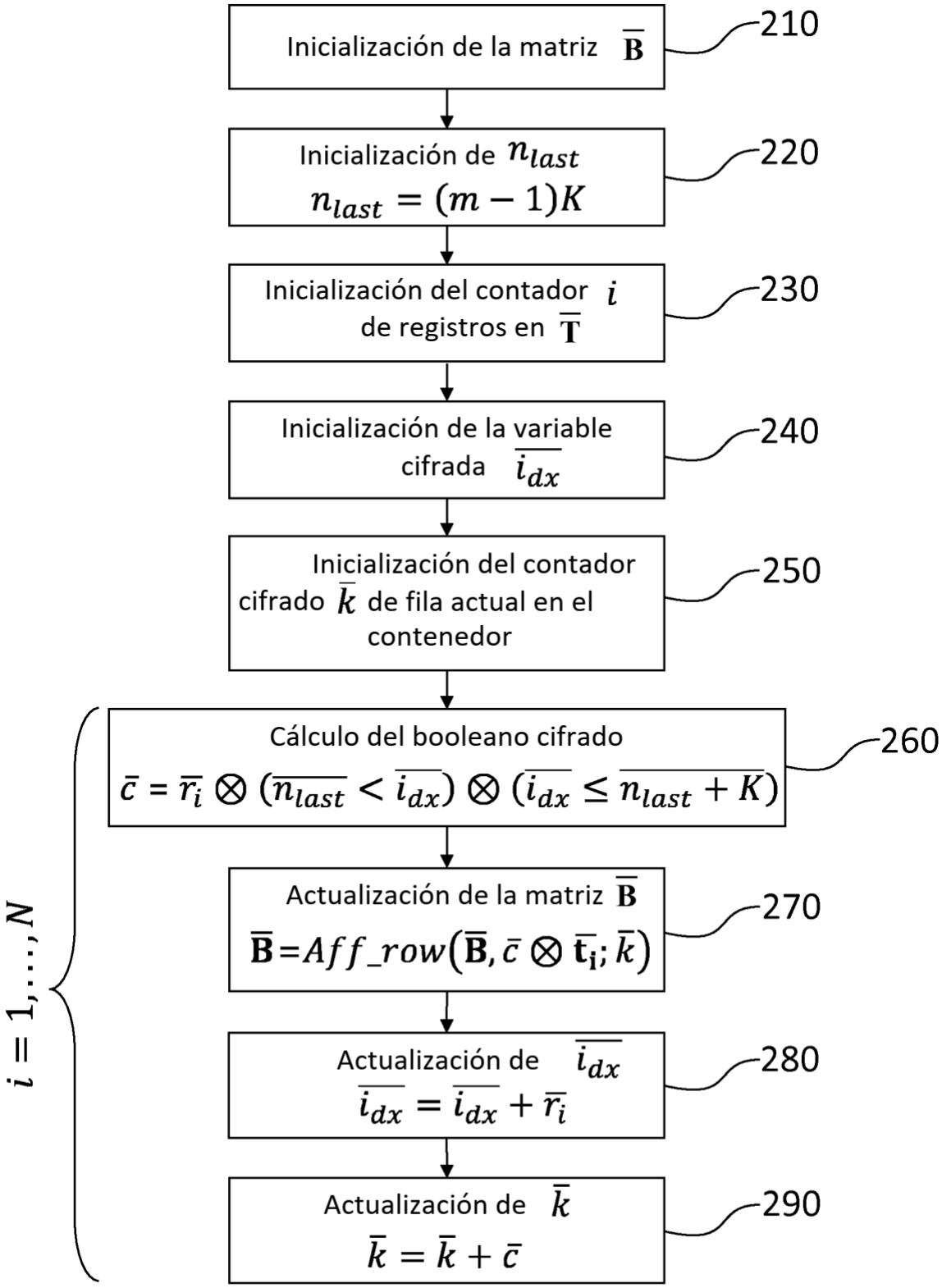
Inicialización de la matriz $\overline{\mathbf{B}}$
210
Inicialización de n_{last}
$n_{last} = (m-1)K$
220
Inicialización del contador i de registros en $\overline{\mathbf{T}}$
230
Inicialización de la variable cifrada $\overline{i_{dx}}$
240
Inicialización del contador cifrado $\overline{k}$ de fila actual en el contenedor
250
Cálculo del booleano cifrado
$\bar{c} = \overline{r_i} \otimes (\overline{n_{last}} < \overline{i_{dx}}) \otimes (\overline{i_{dx}} \leq \overline{n_{last} + K})$
260
Actualización de la matriz $\overline{\mathbf{B}}$
$\overline{\mathbf{B}} = Aff\_row(\overline{\mathbf{B}}, \bar{c} \otimes \overline{\mathbf{t}_i}; \bar{k})$
270
Actualización de $\overline{i_{dx}}$
$\overline{i_{dx}} = \overline{i_{dx}} + \overline{r_i}$
280
Actualización de $\overline{k}$
$\overline{k} = \overline{k} + \bar{c}$
290
$i = 1, \ldots, N$

Fig. 2

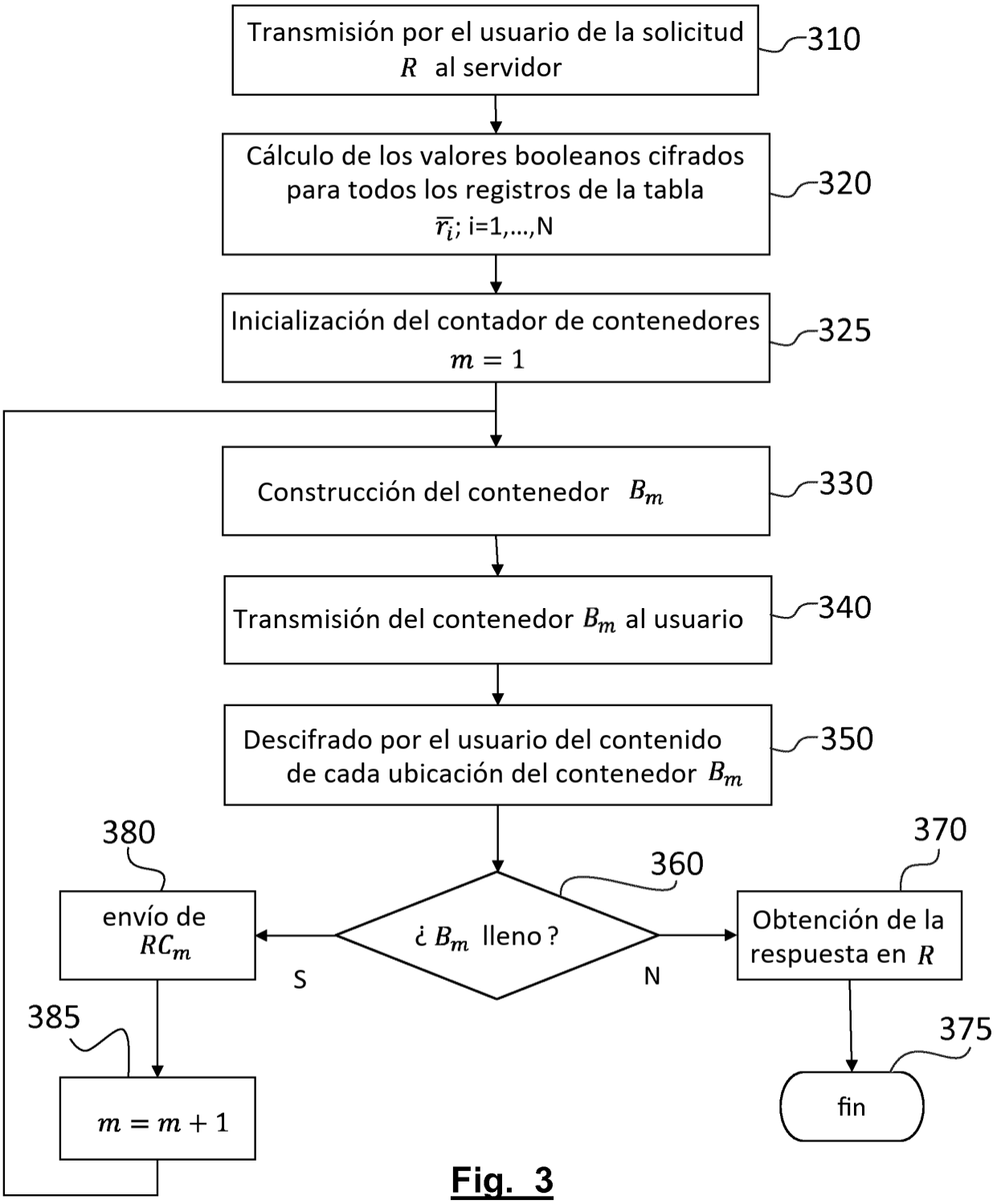
Transmisión por el usuario de la solicitud R al servidor
310
Cálculo de los valores booleanos cifrados para todos los registros de la tabla $\overline{r_i}$; i=1,...,N
320
Inicialización del contador de contenedores $m = 1$
325
Construcción del contenedor B_m
330
Transmisión del contenedor B_m al usuario
340
Descifrado por el usuario del contenido de cada ubicación del contenedor B_m
350
380
envío de RC_m
360
¿ B_m lleno ?
S
N
370
Obtención de la respuesta en R
385
$m = m + 1$
375
fin

Fig. 3