(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 337 925**

(51) Int. Cl.:
G06F 7/72 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Número de solicitud europea: **06743602 .2**
(96) Fecha de presentación : **27.03.2006**
(97) Número de publicación de la solicitud: **1864211**
(97) Fecha de publicación de la solicitud: **12.12.2007**

(54) Título: **Procedimiento de tratamiento de datos que implica una exponenciación modular y un dispositivo asociado.**

(30) Prioridad: **30.03.2005 FR 05 03083**

(45) Fecha de publicación de la mención BOPI:
30.04.2010

(45) Fecha de la publicación del folleto de la patente:
30.04.2010

(73) Titular/es: **OBERTHUR TECHNOLOGIES**
50, quai Michelet
92300 Levallois-Perret, FR

(72) Inventor/es: **Boscher, Arnaud;**
Giraud, Christophe y
Naciri, Robert

(74) Agente: **Elzaburu Márquez, Alberto**

Aviso: En el plazo de nueve meses a contar desde la fecha de publicación en el Boletín europeo de patentes, de la mención de concesión de la patente europea, cualquier persona podrá oponerse ante la Oficina Europea de Patentes a la patente concedida. La oposición deberá formularse por escrito y estar motivada; sólo se considerará como formulada una vez que se haya realizado el pago de la tasa de oposición (art. 99.1 del Convenio sobre concesión de Patentes Europeas).

Venta de fascículos: Oficina Española de Patentes y Marcas. Pº de la Castellana, 75 – 28071 Madrid

DESCRIPCIÓN

Procedimiento de tratamiento de datos que implica una exponenciación modular y un dispositivo asociado.

La invención se refiere a un procedimiento de tratamiento de datos que implica una exponenciación modular, y a un dispositivo asociado.

Los cálculos de exponenciación modular se utilizan frecuentemente en los algoritmos criptográficos y en este marco hacen intervenir en general un secreto, es decir un número almacenado por el dispositivo que pone en práctica el algoritmo criptográfico y que no es accesible desde el exterior.

Las etapas que ponen en práctica la exponenciación modular son particularmente objeto de ataques por parte de personas malintencionadas; puede tratarse de ataques por generación de fallo (especialmente del tipo DFA del inglés "Differential Fault Analysis") o por análisis del consumo de corriente del dispositivo que pone en práctica el algoritmo (del tipo SPA del inglés "Statistical Power Analysis" o DPA del inglés "Differential Power Analysis").

Debido a esto, se ha buscado ya proteger estas etapas, en particular en el caso en que el secreto que hay que proteger corresponde al exponente utilizado en la exponenciación modular.

Así, cuando se utiliza un algoritmo del tipo "square-and-multiply" (término inglés que significa "elevación al cuadrado y multiplicación") en el cual una variable es actualizada por multiplicación para cada bit del exponente que vale 1 (y solamente para estos bits), se ha buscado simetrizar el proceso, por ejemplo efectuando una multiplicación engañosa cuando el bit vale 0, con el fin de hacer frente a los ataques por medición de corriente (indicados a veces por SPA) o por medición de tiempo ("timing attacks").

Partiendo de esta idea general, se han desarrollado diversos algoritmos protegidos contra los ataques de tipo SPA, tales como el descrito en el artículo "The Montgomery Powering Ladder", de B.S. Kaliski Jr., C.Q. Koç y C. Paar, "Cryptographic Hardware and Embedded Systems" - CHES 2002, páginas 291-302.

Por otra parte, se ha buscado proteger los algoritmos criptográficos, y entre estos aquéllos que utilizan una exponenciación modular, de los ataques por fallos, por medio de los cuales un atacante intenta deducir informaciones sobre el funcionamiento interno que pone en práctica el procedimiento criptográfico generando un fallo de funcionamiento en el seno de este procedimiento.

Una solución utilizada habitualmente para luchar contra este último tipo de ataques consiste en doblar los cálculos efectuados con el fin de verificar que las dos iteraciones del mismo cálculo dan el mismo resultado, lo que en general tiende a probar que no se ha producido ningún fallo durante su desarrollo. Esta solución no obstante implica doblar el tiempo de cálculo para cada operación que se desee proteger (sin contar la subsiguiente etapa necesaria de verificación) lo que naturalmente no es deseable.

Con el fin de poner remedio a este inconveniente, la solicitud de patente WO 98/52319 propone, cuando se utiliza el teorema chino de los restos (o CRT del inglés "Chinese Remanider Theorem"), aprovechar la identidad supuesta de dos valores obtenidos cada uno en una de las derivaciones del algoritmo que utiliza este teorema para verificar anticipadamente el desarrollo sin fallos del algoritmo en sus dos derivaciones.

Esta solución, que se aprovecha de una particularidad de las puestas en práctica que utilizan el teorema chino de los restos, no es aplicable sin embargo a otros tipos de implementación. Por otra parte, puede recodarse a este respecto que la utilización del teorema chino de los restos implica el conocimiento de la descomposición en números primos p, q del módulo público n = p.q.

Finalmente, esta solución practica una verificación por medio de datos empleados en una etapa intermedia del proceso y, por tanto, no permite una verificación del funcionamiento sin fallo en cualquier punto del proceso, como podría desearlo el diseñador: por ejemplo, esta técnica no permite proteger el procedimiento en caso de ataques por fallo durante la recombinación del resultado obtenido por cada una de las derivaciones del algoritmo.

Con el fin de mejorar esta situación y, por tanto, de proponer un procedimiento de tratamiento de datos que implique una exponenciación modular protegida a la vez contra los ataques por análisis de corriente y los ataques por fallo, la invención propone un procedimiento de tratamiento (en general criptográfico) de datos, en el cual un mensaje es sometido a una primera operación, por ejemplo, con una clave secreta, que comprende una etapa de actualización por una segunda operación de una primera variable o de una segunda variable según que un bit correspondiente del operando valga 0 o 1, caracterizado por una etapa de prueba de una relación entre un primer valor resultante de la primera variable y un segundo valor resultante de la segunda variable, con el fin de detectar un fallo en el transcurso del cálculo.

En efecto, debido a la complementariedad de las actualizaciones de la primera variable y de la segunda variable, existe una relación que normalmente debe ser verificada entre las dos variables, o entre los valores que resultan de éstas. Por el contrario, la no-verificación de la prueba indica entonces un fallo en el transcurso del cálculo y de esta manera permite detectar un ataque, incluso cuando el ataque va dirigido contra una operación engañosa (caso de ataques "safe error").

El primer valor y el segundo valor son precisamente aquéllos de la primera variable y de la segunda variable; en este caso, la primera variable y la segunda variable pueden ser utilizadas ellas mismas para la prueba, lo que implica especialmente una ganancia de memoria.

La primera operación es, por ejemplo, una exponenciación modular, en cuyo caso la segunda operación es una multiplicación modular y el operando es un exponente.

La etapa de prueba puede comprender entonces la comparación del producto de la primera variable y de la segunda variable con una tercera variable actualizada cualquiera que sea el bit correspondiente del operando.

Una etapa de prueba alternativa comprende la comparación de la primera variable con el producto del elemento y de la segunda variable.

De acuerdo con otra realización posible, la primera operación es una multiplicación a lo largo de una curva elíptica y la segunda operación es una suma a lo largo de la curva elíptica.

La relación puede ser independiente del operando, lo que permite simplificar la etapa de prueba.

La etapa de verificación puede aplicarse especialmente al resultado de la primera operación, es decir después de las etapas de cálculo que implican la segunda operación.

El procedimiento puede comprender igualmente una etapa de verificación de la relación para cada bit del operando. Se detecta, así, un eventual ataque por fallo en cuanto éste haya perturbado un cálculo que pone en práctica la segunda operación, lo que garantiza un buen nivel de seguridad a todo lo largo de las etapas que ponen en práctica la primera operación.

De acuerdo con un modo de puesta en práctica posible, el primer valor resulta de una primera recombinación según el teorema chino de los restos que implica a la primera variable y el segundo valor resulta de una segunda recombinación según el teorema chino de los restos que implica a la segunda variable.

La invención propone igualmente un dispositivo de tratamiento de datos que permite una primera operación sobre un mensaje por medio de una clave secreta, que comprende medios de actualización por una segunda operación de una primera variable o de una segunda variable según que un bit correspondiente del operando valga 0 o 1, caracterizado por medios de prueba de una relación entre la primera y la segunda variable, con el fin de detectar un fallo en el transcurso del cálculo.

Un dispositivo de este tipo está incluido por ejemplo en una tarjeta de microcircuito.

Otras características y ventajas de la invención se pondrán de manifiesto a la luz de la descripción que sigue, hecha refiriéndose a los dibujos anejos, en los cuales:

- la figura 1 representa un procedimiento de tratamiento de datos de acuerdo con un primer modo de realización de la invención;

- la figura 2 representa un procedimiento de tratamiento de datos de acuerdo con un segundo modo de realización de la invención;

- la figura 3 representa un procedimiento de tratamiento de datos de acuerdo con un tercer modo de realización de la invención.

La figura 1 representa un ejemplo de procedimiento que permite un cálculo de exponenciación modular y realizado de acuerdo con las enseñanzas de la invención. El ejemplo dado aquí se describe en forma de un subprograma que recibe valores en la entrada y emite en la salida el resultado del cálculo de exponenciación modular. Sin embargo, se comprende que la invención no se limita a tales subprogramas.

La etapa E100 de la figura 1 corresponde la recepción en la entrada de los valores m, d y n en función de los cuales se desea realizar la exponenciación modular, es decir obtener el número m^d mod n.

En lo que sigue, se expresará el número d en forma de su descomposición binaria $(d_k, ..., d_1)$, donde k es el número de bits que componen el número d, donde cada d_i constituye un bit del número correspondiente siendo d_1 especialmente el bit de menor peso y d_k el bit de mayor peso.

Así pues, se tiene $d = \sum d_i 2^{i-1}$.

En los algoritmos criptográficos, un cálculo de exponenciación modular de este tipo se utiliza generalmente representando el número m un mensaje, el número d la clave secreta y el número n el módulo público. En este marco, se busca por tanto proteger en particular la determinación del número d (es decir de sus componentes binarios d_i) por

la observación de la puesta en práctica del procedimiento en una entidad electrónica o la generación de fallos en este procedimiento.

El procedimiento propiamente dicho descrito en la figura 1 se inicia en la etapa E102 de inicialización de los registros utilizados en este subprogrma, a saber, la inicialización con el valor 1 de una variable A, la inicialización con el valor m de una variable B y de una variable S y la inicialización con el valor 1 de una variable i que sirve de índice.

Se pasa entonces a la etapa E104 (que constituye la primera etapa de un bucle como se describe más adelante) en la cual se prueba si el bit d_i (es decir el bit de rango i en el número d) vale 1.

En caso afirmativo, se pasa a la etapa E106 en la cual se procede al cálculo del valor A.S mod n, y después se memoriza el resultado de este cálculo en la variable A (con machacamiento en este caso del valor precedentemente almacenado en esta variable).

Dicho de otro modo, se actualiza la variable A por medio de una multiplicación modular por la variable S que, debido a la etapa E110 descrita en lo que sigue, vale m^{2i} durante esta actualización, cualquiera que sea la iteración i correspondiente en el bucle.

Si la prueba en la etapa E104 es negativa (es decir si el bit d_i es nulo), se procede en la etapa E108 al cálculo de B.S mod n, y después de actualiza la variable B con el resultado de este cálculo.

Así, según el valor del bit d_i, se actualiza la variable A, o la variable B por multiplicación modular por la variable S.

Cualquiera que sea el resultado de la prueba de la etapa E104, el procedimiento continúa (después de la etapa E106 o la etapa E108) en la etapa E110, en la cual se calcula S^2 mod n, y después se memoriza el resultado de este cálculo en la variable S (con machacamiento del valor precedentemente almacenado en esta variable).

Se incrementa entonces el valor del índice i en la etapa E112, y después se prueba el nuevo valor del índice i en la etapa E114: si i es estrictamente superior a k, se pasa a la etapa E116 descrita más adelante, mientas que en caso negativo (es decir en tanto que i sea inferior o igual a k) la etapa E114 va seguida de la etapa E104 anteriormente descrita (lo que realiza el bucle anteriormente citado).

Cuando i es estrictamente superior a k después del incremento, es decir, cuando han sido tratados el conjunto de los bits del número d, se pasa a la etapa E116 en la cual se prueba la validez de la relación siguiente: S = A.B mod n.

En funcionamiento normal, debido a las actualizaciones complementarias de las variables A y B y de la actualización sistemática de la variable S como se describió anteriormente, se verifica la relación S=A.B mod n.

Por consiguiente, si la prueba de la etapa E116 es positiva, se considera que ha tenido lugar un funcionamiento normal (es decir sin fallo) y se procede a la etapa E120 a la cual se reenvía el valor de salida A, que debido a su actualización por multiplicación modular para los únicos bits de d que valen 1, vale m^d mod n.

Por el contrario, si la prueba de la etapa E116 es negativa, se considera que ha tenido lugar un fallo durante el tratamiento descrito y por consiguiente se pasa a la etapa E118 a la cual se envía un valor de error.

En el marco de los algoritmos criptográficos, una detección de fallo de este tipo es considerada como la consecuencia de un ataque por fallo con miras a la determinación de la clave secreta d. Así pues, naturalmente, no se enviará ninguna información susceptible de ayudar al atacante en su búsqueda del conocimiento de la clave secreta. Por el contrario, podrán activarse entonces medidas de protección, tales como, por ejemplo, el bloqueo de la entidad electrónica que pone en práctica el procedimiento (entidad que, por ejemplo, es una tarjeta de microcircuito).

Se puede observar que la prueba utiliza aquí variables que son necesarias para la puesta en práctica del cálculo simetrizado de la exponenciación modular; este modo de realización permite, así, realizar la etapa de prueba sin necesitar la utilización de otras variables.

Se puede observar igualmente que la relación probada no implica a la clave secreta d y además no necesita el conocimiento de la clave pública (es decir el conocimiento del número e tal que d.e = 1 mod (p-1)(q-1).

Por otra parte, la relación hace intervenir a las dos variables A y B modificadas alternativamente en el transcurso del procedimiento de cálculo de exponenciación modular porque la modificación de una u otra de estas dos variables, por ejemplo por medio de un ataque por fallo, se traducirá en la no-verificación de la relación. De esta manera, se lucha eficazmente contra los ataques denominados “safe error” que buscan detectar la eventual ausencia de consecuencia de un fallo para deducir así que la operación correspondiente no es utilizada efectivamente por el cálculo.

En variante, podría procederse a la prueba de la etapa E116 en cada iteración del bucle (es decir por ejemplo insertar una etapa de prueba similar entre las etapas E110 y E112) puesto que en funcionamiento normal en cada iteración se verifica la relación.

En este caso, cualquier detección de un error (por no-verificación de la relación) podrá interpretarse como el resultado de un ataque por fallo; así pues, en este caso es preferible que la no-verificación de la relación ponga fin al cálculo de la exponenciación modular, mientas que su verificación implica la continuación normal de las iteraciones.

Cualquiera que sea la variante puesta en práctica (verificación al final del cálculo como en la figura 1 o en cada iteración como se acaba de describir), se observa, como se ve bien en la figura 1, que el número de etapas realizadas y el tipo de operaciones efectuadas en el transcurso de estas etapas no varía en función de la clave secreta d, lo que permite una protección del procedimiento contra los ataques por medición de corriente de tipo SPA.

La figura 2 representa un segundo modo de realización de la invención.

Como en el primer modo de realización, se ha designado por etapa de entrada E200 la recepción por el subprograma descrito aquí de los valores m, d y n. Como anteriormente, se trata, naturalmente, solamente de un ejemplo posible de puesta en práctica de la invención.

En consideración a estos valores, se procede a una etapa de inicialización E202, en el transcurso de la cual se inicializa una variable a_0 con 1, una variable a_1 con m y una variable i con el valor k que, como anteriormente, representa el número de bits de la clave secreta d (k es un dato fijo del sistema criptográfico utilizado). Se diminuye igualmente d en una unidad debido al modo de realización utilizado en este caso para el cálculo de exponenciación modular.

Se entra entonces en un bucle que hablando en propiedad permite el cálculo de la exponenciación modular, para empezar por una etapa E204 de prueba del valor del bit d_i. (Las notaciones relativas a la descomposición de la clave d en bits son idénticas a aquéllas presentadas a propósito del primer modo de realización y, por tanto, no serán tomadas aquí nuevamente).

Si el bit de rango i en esta clave d vale 1, se pasa a la etapa E206 en la cual se procede en primer lugar a la multiplicación de la variable a_0 por la variable a_1 (es decir al cálculo de $a_0 * a_1 \bmod n$), cuyo resultado se memoriza en la variable a_0 (con machacamiento). En el seno de la etapa E206 se procede igualmente al cálculo del valor $a_1^2 \bmod n$ y a la actualización de la variable a_1 por el resultado de este cálculo.

Si en la etapa E204 se determina que el bit de rango i de la clave secreta d vale cero (es decir si $d_i = 0$), se procede a la etapa E208 en el transcurso de la cual se calcula el producto de la variable a_1 por la variable a_0 (es decir que se calcula $a_1 * a_0 \bmod n$), se memoriza el valor obtenido en a_1 con machacamiento, se calcula el cuadrado modular de la variable a_0 (es decir, el valor $a_0^2 \bmod n$) y se almacena el resultado de esta última operación en la variable a_0 con machacamiento.

Puede observarse que las etapas E206 y E208, puestas en práctica respectivamente cuando el bit de la clave secreta d correspondiente durante la iteración i vale 1 o 0 son simétricas con respecto a las variables a_0 y a_1, siendo actualizadas cada una de estas variables en cada caso por multiplicación por la otra variable.

Cualquiera que sea el resultado de la prueba de la etapa E204, se procede, después de la etapa E206 o la etapa E208, a la etapa E210 en la cual se prueba la relación siguiente: $m * a_0 = a_1$.

En funcionamiento normal, es decir cuando las operaciones de la etapa E206 o E208 precedentes han sido realizadas sin fallo, debe verificarse esta relación.

Por consiguiente, si la prueba de la etapa E210 es positiva, se considera que el desarrollo del cálculo ha tenido lugar sin fallo y, por tanto, se continúa el tratamiento en la etapa E214, como se describe a continuación.

Por el contrario, si la etapa E210 no permite la verificación de la relación $m * a_0 = a_1$, se deduce de esto que uno de los cálculos efectuados en la etapa precedente E206 o E208 ha experimentado una perturbación, tal como por ejemplo un ataque por fallo.

Por esta razón, si la etapa de prueba E210 conduce a un resultado negativo, se procede a una etapa E212 en la cual se considera que se ha detectado un fallo y que se necesita un tratamiento adecuado. Como se ha visto anteriormente, este tratamiento puede variar según las aplicaciones.

Como ya se indicó, el funcionamiento normal conduce a la etapa E214, en la cual se disminuye la variable i. Se prueba entonces en la etapa E216 si la variable i ha llegado a 0. En caso negativo, no han sido tratados todos los bits de la clave secreta d y se procede a la iteración siguiente del bucle pasando a la etapa E204 ya descrita. En caso afirmativo, ha sido tratado el conjunto de los bits de la clave secreta y entonces el cálculo de exponenciación modular ha terminado: la variable a_1 corresponde al resultado deseado, o sea $m^d \bmod n$. Se envía, por tanto, a la salida del subprograma descrito aquí el valor a_1 a la etapa E218.

En variante, entre las etapas E216 y E218 puede realizarse la etapa E210, en sustitución o en suplemento de la etapa E210 descrita anteriormente. En el caso en que la etapa E210 no se realice en el bucle, sino solamente después del bucle (por ejemplo entre las etapas E216 y E218), se aligeran las fases de prueba del algoritmo.

Refiriéndose a la figura 3, se va a describir ahora un tercer modo de realización de la invención, puesto en práctica en el marco de una exponenciación modular que utiliza el teorema chino de los restos (o CRT).

El algoritmo presentado en la figura 3 recibe en la entrada el número m (o mensaje) del que se desea realizar la exponenciación modular, los dos números primos p, q que componen el módulo público n = p.q, los componentes d_p, d_q de la clave secreta d en relación con p y q (donde $d_p = d \bmod (p-1)$ y $d_q = d \bmod (q-1)$ y el número a tal que $a = p^{-1} \bmod q$.

Se procede entonces en la etapa E302 a la exponenciación modular parcial del número m con el exponente d_p según un algoritmo del tipo de cálculo descrito anteriormente refiriéndose a la figura 2. Al final de este algoritmo, se obtiene, así, un número S_p (correspondiente al número a_1 en la figura 2) resultado de la exponenciación modular parcial (es decir $S_p = m^{dp} \bmod p$) y un número S'_p (correspondiente al número a_0 en la figura 2) tal que $S_p = mS'_p$.

Durante una etapa E304 se procede asimismo a la exponenciación modular parcial por medio del exponente d_q según un algoritmo como el descrito en la figura 2, que permite obtener $S_q = m^{dq} \bmod q$ (S_q corresponde al número a_1 en la figura 2), así como S'_q (correspondiente a a_0 en la figura 2) que verifica en funcionamiento normal $S_q = mS'_q$.

Se pasa entonces a una etapa E306 de recombinación de los resultados modulares parciales según la fórmula china de los restos. Se procede, así, por una parte, a la combinación de los valores S'_p y S'_q que da como resultado S' y, por otra, a la recombinación de los valores S_p y S_q, lo que permite obtener el valor S.

Siendo S_p y S_q los resultados de la exponenciación modular parcial como se vio respectivamente en las etapas E302 y E304, la variable S contiene el resultado de la exponenciación modular (o sea $S = m^d \bmod (p * q)$). Por otra parte, debido a las relaciones mencionadas anteriormente entre S'_p y S_p, por una parte, y S'_q y S_q, por otra, se tiene igualmente después de la recombinación en funcionamiento normal la relación $S = m * S' \bmod (p * q)$.

Por esta razón, se verifica la exactitud de esta relación en la etapa de prueba E308.

En caso de verificación negativa durante la etapa de prueba, se procede a una etapa E310 en la que se considera que se ha producido un error durante el cálculo, causado probablemente por un ataque por generación de fallo. Entonces puede aplicarse un tratamiento adaptado como se describió a propósito de los otros modos de realización.

Si, por el contrario, en la etapa E308 se verifica la relación $S = m * S' \bmod (p * q)$, se considera que el algoritmo se ha desarrollado sin fallo y se procede a la etapa E312, a la cual se reenvía el valor S que, como se indicó anteriormente, corresponde al resultado de la exponenciación modular.

Los modos de realización que acaban de describirse, son únicamente ejemplos posibles de puesta en práctica de la invención. Ésta naturalmente se aplica en otros casos, por ejemplo a otras operaciones distintas a la exponenciación modular descompuesta en multiplicaciones modulares: ésta se aplica, así, igualmente, a los algoritmos criptográficos basados en curvas elípticas en las cuales se efectúan multiplicaciones a lo largo de estas curvas descompuestas como un conjunto de sumas.

Además, la invención puede aplicarse a otros sistemas de cálculo distintos a los descritos, tales como por ejemplo la aritmética de Montgomery.

Por otra parte, existen diferentes fórmulas de recombinación según el método CRT y la invención puede aplicarse a estas diferentes fórmulas.

REIVINDICACIONES

1. Procedimiento de tratamiento criptográfico de datos, en el cual un mensaje (m) es sometido a una primera operación con una clave secreta (d), que comprende una etapa de actualización por una segunda operación de una primera variable ($B; a_0; S'_p, S'_q$), o de una segunda variable ($A; a_1; S_p, S_q$) según que un bit correspondiente de la clave secreta valga 0 o 1, **caracterizado** por una etapa de prueba (E116, E210; E308) de una relación entre un primer valor (B; a_0; S') resultante de la primera variable y un segundo valor (A; a_1; S) resultante del asegunda variable, con el fin de detectar un fallo en el transcurso del cálculo.

2. Procedimiento de acuerdo con la reivindicación 1, **caracterizado** porque el primer valor es el valor de la primera variable y porque el segundo valor es el valor de la segunda variable.

3. Procedimiento de acuerdo con las reivindicaciones 1 o 2, **caracterizado** porque la etapa de prueba (E116) es aplicada al resultado de la primera operación.

4. Procedimiento de acuerdo con la reivindicación 1, **caracterizado** porque el primer valor (S') resulta de una primera recombinación según el teorema chino de los restos que implica a la primera variable (S'_p, S'_q) y porque el segundo valor S resulta de una segunda recombinación según el teorema chino de los restos que implica a la segunda variable (S_p, S_q).

5. Procedimiento de acuerdo con una de las reivindicaciones 1 a 4, **caracterizado** porque la primera operación es una exponenciación modular, porque la segunda operación es una multiplicación modular y porque la clave secreta es un exponente.

6. Procedimiento de acuerdo con la reivindicación 5, **caracterizado** porque la etapa de prueba comprende la comparación del producto de la primera variable (B) y de la segunda variable (A) con una tercera variable (S) actualizada por el cálculo de un cuadrado, cualquiera que sea el bit correspondiente de la clave secreta.

7. Procedimiento de acuerdo con la reivindicación 5, **caracterizado** porque la etapa de prueba comprende la comparación de la segunda variable (a_1) con el producto del elemento (m) y de la primera variable (a_0).

8. Procedimiento de acuerdo con una de las reivindicaciones 1 a 4, **caracterizado** porque la primera operación es una multiplicación a lo largo de una curva elíptica y porque la segunda operación es una suma a lo largo de la curva elíptica.

9. Procedimiento de acuerdo con una de las reivindicaciones 1 a 8, **caracterizado** porque la relación es independiente de la clave secreta.

10. Dispositivo de tratamiento criptográfico de datos que permite una primera operación sobre un mensaje por medio de una clave secreta, que comprende:

- medios de actualización por medio de una segunda operación de una primera variable o de una segunda variable según que un bit correspondiente de la clave secreta valga 0 o 1,

caracterizado por:

- medios de prueba de una relación entre un primer valor resultante de la primera variable y un segundo valor resultante de la segunda variable, con el fin de detectar un fallo en el transcurso del cálculo.

11. Dispositivo de acuerdo con la reivindicación 10, **caracterizado** porque el primer valor es el valor de la primera variable y porque el segundo valor es el valor de la segunda variable.

12. Dispositivo de acuerdo con las reivindicaciones 10 u 11, **caracterizado** porque los medios de prueba son aplicados al resultado de la primera operación.

13. Dispositivo de acuerdo con la reivindicación 10, **caracterizado** por medios para obtener el primer valor (S') como resultado de una primera recombinación según el teorema chino de los restos que implica a la primera variable (S'_p, S'_q) y por medios para obtener el segundo valor (S) como resultado de una segunda recombinación según el teorema chino de los restos que implica a la segunda variable (S_p, S_q).

14. Dispositivo de acuerdo con una de las reivindicaciones 10 a 13, **caracterizado** porque la primera operación es una exponenciación modular, porque la segunda operación es una multiplicación modular y porque la clave secreta es un exponente.

15. Dispositivo de acuerdo con la reivindicación 14, **caracterizado** porque los medios de prueba comprenden medios de comparación del producto de la primera variable (B) y de la segunda variable (A) con una tercera variable (S) actualizada por el cálculo de un cuadrado, cualquiera que sea el bit correspondiente de la clave secreta.

16. Dispositivo de acuerdo con la reivindicación 14, **caracterizado** porque los medios de prueba comprenden medios de comparación de la primera variable (a_1) con el producto del elemento (m) y de la segunda variable (a_0).

17. Dispositivo de acuerdo con una de las reivindicaciones 10 a 13, **caracterizado** porque la primera operación es una multiplicación a lo largo de una curva elíptica y porque la segunda operación es una suma a lo largo de la curva elíptica.

18. Dispositivo de acuerdo con una de las reivindicaciones 10 a 17, **caracterizado** porque la relación es independiente de la clave secreta.

19. Tarjeta de microcircuito que comprende un dispositivo de acuerdo con una de las reivindicaciones 10 a 18.

FIG. 1

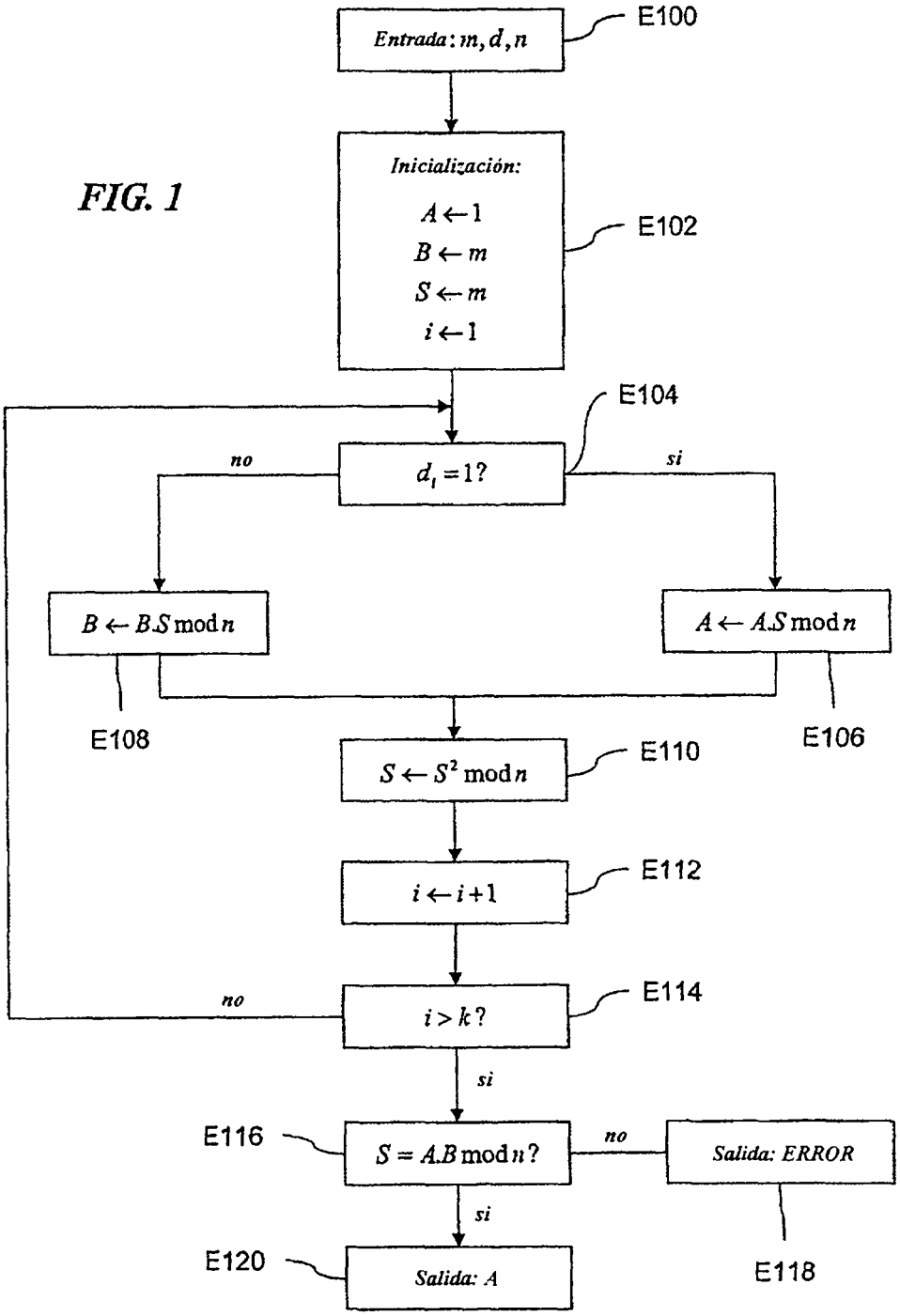

Entrada: m, d, n
E100
Inicialización:
A ← 1
B ← m
S ← m
i ← 1
E102
E104
no
d_i = 1?
si
B ← B.S mod n
A ← A.S mod n
E108
E106
S ← S² mod n
E110
i ← i + 1
E112
no
i > k?
E114
si
E116
S = A.B mod n?
no
Salida: ERROR
si
E120
Salida: A
E118

FIG. 2

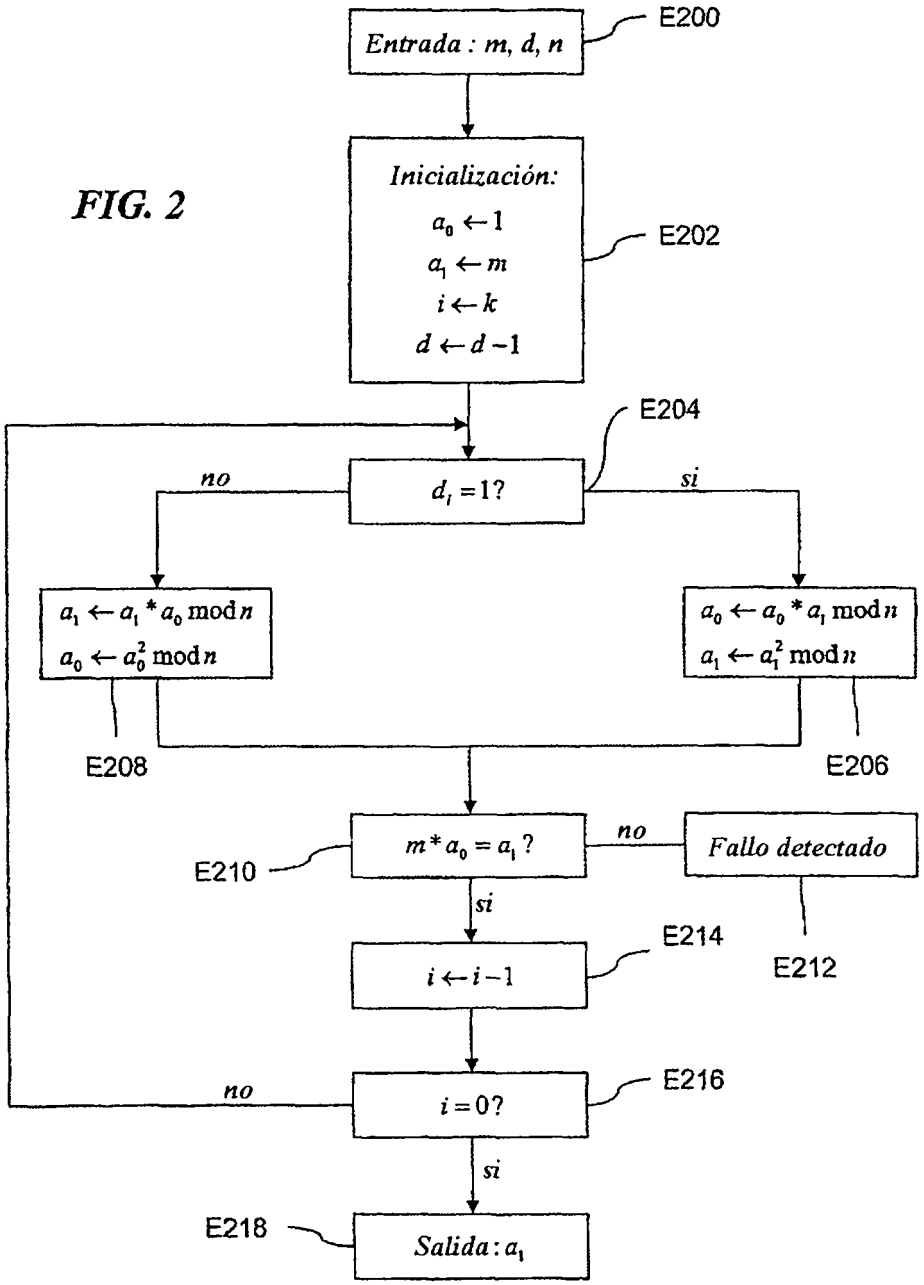
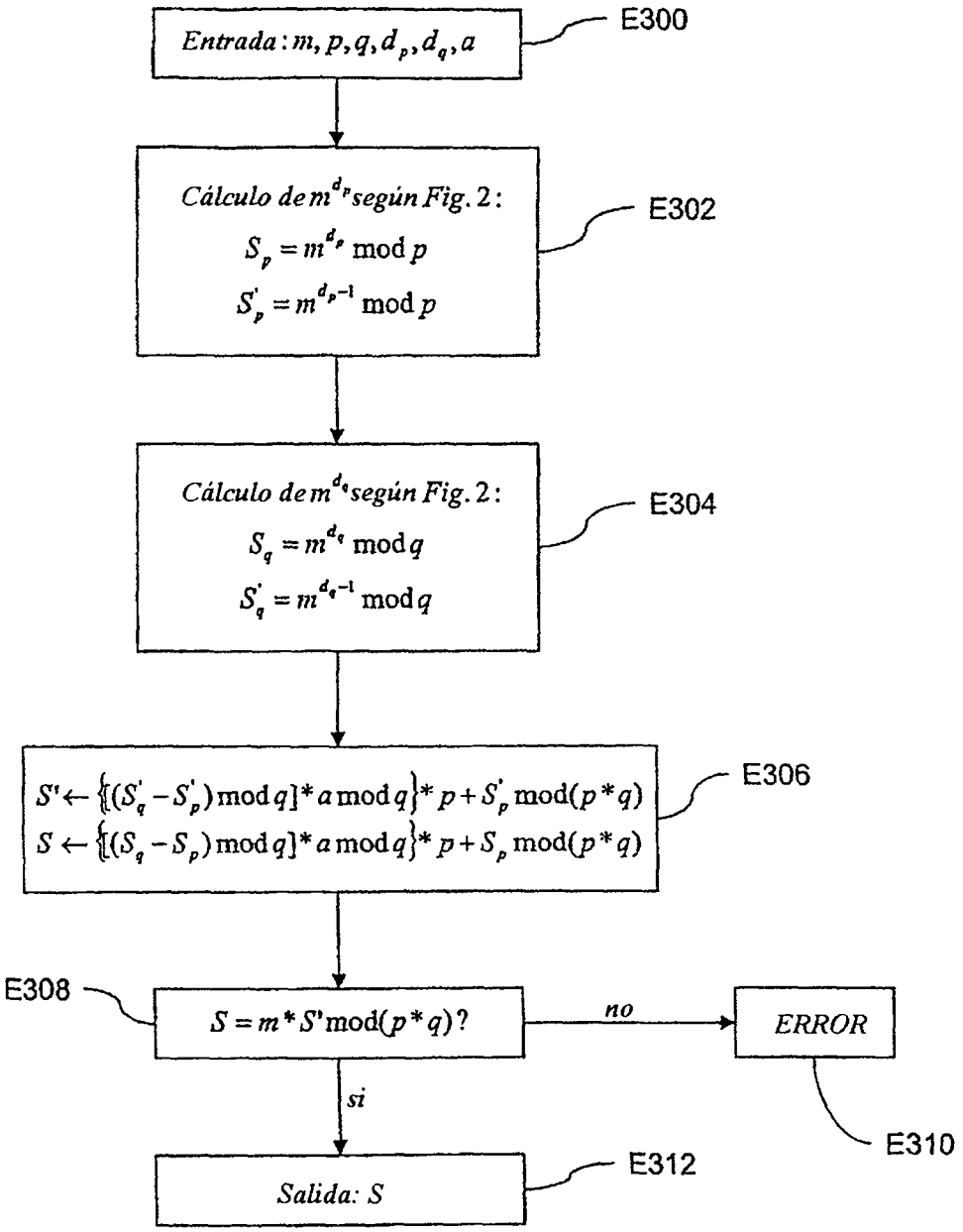

Entrada : m, d, n
E200
Inicialización:
$a_0 \leftarrow 1$
$a_1 \leftarrow m$
$i \leftarrow k$
$d \leftarrow d-1$
E202
E204
$d_i = 1$?
no
si
$a_1 \leftarrow a_1 * a_0 \bmod n$
$a_0 \leftarrow a_0^2 \bmod n$
$a_0 \leftarrow a_0 * a_1 \bmod n$
$a_1 \leftarrow a_1^2 \bmod n$
E208
E206
$m * a_0 = a_1$?
E210
no
Fallo detectado
E212
si
$i \leftarrow i-1$
E214
$i = 0$?
E216
no
si
Salida : a_1
E218

FIG. 3


Entrada: m, p, q, d_p, d_q, a
E300
Cálculo de m^{d_p} según Fig. 2:
$S_p = m^{d_p} \bmod p$
$S'_p = m^{d_p - 1} \bmod p$
E302
Cálculo de m^{d_q} según Fig. 2:
$S_q = m^{d_q} \bmod q$
$S'_q = m^{d_q - 1} \bmod q$
E304
$S' \leftarrow \{[(S'_q - S'_p) \bmod q] * a \bmod q\} * p + S'_p \bmod (p*q)$
$S \leftarrow \{[(S_q - S_p) \bmod q] * a \bmod q\} * p + S_p \bmod (p*q)$
E306
E308
$S = m * S' \bmod (p*q)$?
no
ERROR
E310
si
Salida: S
E312