(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 729 874**

(51) Int. CI.:

G06F 7/72 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **31.08.2015 PCT/EP2015/069867**

(87) Fecha y número de publicación internacional: **17.03.2016 WO16037885**

(96) Fecha de presentación y número de la solicitud europea: **31.08.2015 E 15756174 (7)**

(97) Fecha y número de publicación de la concesión europea: **13.03.2019 EP 3191936**

(54) Título: **Sistema y método de exponenciación del teorema chino del resto de uso único para algoritmos criptográficos**

(30) Prioridad:

10.09.2014 EP 14306393

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
06.11.2019

(73) Titular/es:

THALES DIS FRANCE SA (100.0%)
6, rue de la Verrerie
92190 Meudon, FR

(72) Inventor/es:

VIGILANT, DAVID

(74) Agente/Representante:

ELZABURU, S.L.P

DESCRIPCIÓN

Sistema y método de exponenciación del teorema chino del resto de uso único para algoritmos criptográficos

Antecedentes de la invención

La presente invención se refiere en general a la tecnología de criptografía electrónica, y en particular a la protección de un dispositivo de seguridad contra ataques de canal lateral utilizando enmascaramiento multiplicativo que utiliza técnicas de exponenciación simultánea.

La comunicación y el comercio electrónicos pueden ser herramientas poderosas pero peligrosas. Con la amplia disponibilidad de la tecnología de red, tal como la Internet, existe un uso cada vez mayor de herramientas en línea para la comunicación y el comercio. Cada año a más usuarios les resulta más fácil y más rápido realizar transacciones importantes, ya sea en forma de correspondencia o de comercio, utilizando ordenadores y redes informáticas. Sin embargo, siempre existe el riesgo de que la seguridad de las transacciones electrónicas se vea comprometida a través de la interceptación por terceras partes que no tienen derecho a participar en las transacciones. Cuando terceras partes maliciosas obtienen acceso a transacciones y datos privados de otra manera existe un riesgo de pérdida económica, pérdida de privacidad, e incluso pérdida de seguridad física. La criptografía es un mecanismo empleado para evitar la intrusión en la privacidad de las transacciones y datos electrónicos.

La criptografía es una tecnología para ocultar un mensaje en presencia de terceros que utiliza técnicas matemáticas en las que un mensaje es cifrado de tal manera que solo puede ser descifrado utilizando una clave secreta que debería ser conocida por el destinatario y/o el remitente de un mensaje.

Los algoritmos criptográficos tienen entradas y salidas. En el caso de cifrado, la entrada es un mensaje que ha de ser protegido en texto sin formato. El mensaje de texto sin formato es manipulado por el algoritmo criptográfico para producir un texto cifrado, la salida. Para producir el texto cifrado el algoritmo criptográfico realiza ciertas operaciones matemáticas que incluyen la utilización de una clave secreta. La clave puede ser un secreto compartido, por ejemplo, entre un remitente y un destinatario, o puede ser una clave privada mantenida por el destinario.

Una técnica criptográfica utilizada frecuentemente es el algoritmo RSA nombrado por sus inventores Rivest, Shamir, y Adelman. Para obtener un texto cifrado muy seguro, el algoritmo RSA se basa en la dificultad de factorizar números enteros grandes. Un usuario crea una clave pública seleccionando aleatoriamente dos números primos grandes de tamaño similar y multiplica estos dos números juntos. El resultado es la clave pública del usuario que el usuario puede publicar habilitando de este modo a otras entidades para cifrar mensajes para el usuario. Aunque la clave pública es pública y cualquiera puede cifrar un mensaje con su utilización, el mensaje cifrado solo puede ser descifrado utilizando la clave privada correspondiente que, en efecto, consiste en los dos números primos que se utilizaron para generar la clave pública. Por lo tanto, es crítico para la seguridad proporcionada por el algoritmo RSA que las claves privadas se mantengan secretas y no puedan ser discernidas por una tercera parte que intente subvertir el secreto de los mensajes cifrados por RSA.

Aunque los detalles del algoritmo RSA van más allá de este documento, para los fines de discusión en la presente memoria el algoritmo puede ser reducido a dos cálculos complementarios para el cifrado de un mensaje M en un texto cifrado C y el descifrado del texto cifrado C de nuevo al mensaje M. La clave pública es calculada a partir de dos números primos grandes p y q. A partir de p y q se ha calculado un número $n = pq$; n es el módulo para ambas claves privada y pública. Además e, el exponente de la clave pública es calculado a partir de p y q, como sigue:

Elegir e de tal manera que: $1 < e < \varphi(n)$ y el mayor divisor común de $(e, \varphi(n)) = 1$, es decir, e y $\varphi(n)$ son primos entre sí, en donde, $n = pq$ y $\varphi(n)$ es la función Totient de Euler.

Así, la clave pública consiste en el par de números enteros (n, e). La clave privada correspondiente consiste en el par de números enteros (n, d) donde $d \equiv e^{-1} \pmod{\varphi(n)}$ donde $\varphi(n)$ es la función Totient de Euler.

Un mensaje M es cifrado utilizando la clave pública (n, e) en el texto cifrado C por:

$$C = M^e \bmod n$$

El mensaje M es recuperado y descifrado a partir de C utilizando la clave privada correspondiente (n, d) por:

$$M = C^d \pmod{n}$$

El RSA también puede ser utilizado para formar criptográficamente un mensaje M en un mensaje firmado S, es decir,

$$S = M^d \pmod{n}$$

Habitualmente estos cálculos no son realizados directamente ya que las exponenciaciones en números enteros grandes son cálculos costosos. Un cálculo más eficiente, que implica exponenciación de números enteros mucho más pequeños, utiliza el Teorema Chino del Resto. Sin entrar en detalles, el enfoque del Teorema Chino del Resto incluye las exponenciaciones modulares:

$$Sp = M_p^{dp} \bmod p$$

$$Sq = M_q^{dp} \bmod q$$

En donde $dp = d \bmod (p - 1)$ y $dq = d \bmod (q - 1)$, y $M_p = M \bmod p$ y $M_q = M \bmod q$

El cálculo de la firma RSA-CRT está compuesto de 3 pasos principales:

- Calcular *Sp* (aproximadamente el 45% del cálculo)
- Calcular *Sq* (aproximadamente el 45% del cálculo)
- Recombinar *S* a partir de *Sp* y *Sq* (aproximadamente el 10% del cálculo)

Los ataques de canal lateral hacen uso de la temporización de programa, el consumo de energía y/o la emanación electrónica de un dispositivo que realiza un cálculo criptográfico. El comportamiento del dispositivo (temporización, consumo de energía y emanación electrónica) varía y depende directamente del programa y de los datos manipulados en el algoritmo criptográfico. Un atacante podría aprovecharse de estas variaciones para inferir datos confidenciales que conducen a la recuperación de una clave privada.

Los ataques por fallos derivan su nombre de la práctica de crear un fallo durante el cálculo y explotar el resultado producido por ese fallo para deducir la clave secreta. Generalmente, inyectar un fallo requiere un paso previo que consiste en determinar el momento más probable de éxito para la inyección del fallo. Este paso previo es realizado habitualmente mediante ingeniería inversa del programa a través del estudio de la potencia o de la traza de emanación electrónica. RSA-CRT es particularmente vulnerable a los ataques por fallos porque interrumpir o bien el cálculo de *Sp* solo o bien de *Sq* solo puede permitir al intruso deducir la clave privada, sea cual sea el efecto causado por fallo. Además, la configuración para inducir un fallo durante el cálculo bien de *Sp* o bien de *Sq* es relativamente fácil de hacer porque estos dos pasos confidenciales son de manera habitual fácilmente identificables en una traza de energía. Dado que *Sp* y *Sq* ocupan una gran parte del proceso, aproximadamente el 45% cada uno de la firma total, existe un tiempo suficiente para interrumpir cualquier cálculo. Así, un fallo que interrumpe el cálculo bien de *Sp* o bien de *Sq* podría permitir la recuperación no autorizada de los factores primos de la clave privada.

Un mecanismo utilizado para defenderse contra ataques por fallos es realizar la operación de firma dos veces para asegurar que no se ha introducido ningún fallo durante el cálculo. Realizar tales operaciones dos veces sería una contramedida costosa.

Otras técnicas anteriores incluyen Shamir (Shamir, Patente de los EE.UU 5.991.414, Method and apparatus for protecting public keys schemes from timing and fault attacks), Aumuller (Aumuller et al., Concrete results and practical countermeasures, Cryptographic Hardware and Embedded Systems – CHES 2002: 4th International Workshop, Volúmen 4), Giraud (Giraud, C., An RSA implementation resistant to fault attacks and to simple power analysis, IEEE Transactions on Computers (Volúmen: 55, Issue: 9), Sept. 2006), y Vigilant (Cryptographic Harware and Embedded Systems – CHES 2008, Lecture Notes in Computer Science Volúmen 5154, 2008, págs. 130-145).

Estas técnicas anteriores pueden estar divididas en dos tipos:

- La técnica de Shamir, a partir de la cual son divididas las técnicas de Aumuller y de Vigilant, consiste en multiplicar el módulo por un pequeño número aleatorio antes de la exponenciación. La exponenciación es realizada modulando este nuevo número y algunas verificaciones de consistencia pueden ser realizadas modulando el pequeño número aleatorio después de la exponenciación. Una verificación de consistencia global es realizada después de la recombinación. Si la verificación de consistencia global falla, un ataque por fallos puede haber sido detectado.

- La técnica de Giraud consiste en utilizar el algoritmo de exponenciación de escala de Montgomery que produce ($X^{(y-1)} \bmod Z$, $X^y \bmod Z$) cuando calcula $X^y \bmod Z$.

Común a estas técnicas anteriores es que todas detectan el fallo con alguna probabilidad, excepto la de Giraud. Pero la técnica de Giraud tiene el inconveniente de que requiere una gran cantidad de memoria RAM para su implementación. Además estas técnicas mantienen una estructura de tres pasos: cálculo de *Sp*, cálculo de *Sq*, y recombinación. Tener tres pasos proporciona al atacante múltiples oportunidades para configurar un ataque por fallos.

A partir de lo anterior será evidente que existe aún una necesidad de una tecnología mejorada para proporcionar un mecanismo seguro que es computacionalmente eficiente, que no requiere registros excesivamente grandes u otro almacenamiento, y en el que un dispositivo de seguridad portátil – por ejemplo, una tarjeta inteligente conectada a un ordenador anfitrión – puede proporcionar la capacidad de proporcionar servicios criptográficos que están protegidos contra ataques por fallos.

Breve descripción de los dibujos

La Figura 1 es una ilustración esquemática de un ordenador anfitrión con un dispositivo de seguridad portátil, por ejemplo, una tarjeta inteligente, conectada al mismo para realizar servicios criptográficos a través de la conexión sobre

una red a uno o más servidores.

La Figura 2 es una ilustración esquemática de un dispositivo de seguridad portátil.

La Figura 3 es una ilustración esquemática de programas almacenados en una memoria del dispositivo de seguridad portátil de la Figura 2.

La Figura 4 es una ilustración esquemática de un listado de programas de módulos de criptografía de la técnica anterior que puede estar almacenado en la memoria de un dispositivo de seguridad portátil como se ha ilustrado en la Figura 3 y que realiza un descifrado que incluye operaciones de exponenciación.

La Figura 5 ilustra un método de la técnica anterior para realizar una operación de descifrado que utiliza exponenciación modular de acuerdo con el algoritmo de cuadrado y multiplicación siempre con elementos de la mitad de tamaño para realizar dos operaciones de exponenciación, en particular, $S_p = M^{dp}) \bmod p$ y $S_q = (M^{dq}) \bmod q$.

La Figura 6 ilustra un módulo criptográfico que implementa un algoritmo de descifrado modificado que utiliza elementos de la mitad de tamaño, de acuerdo con una realización preferida para realizar el descifrado utilizando el mismo material clave que en los algoritmos de las figs. 4 y 5 mientras que realiza solo una exponenciación.

Descripción detallada de la invención

En la siguiente descripción detallada, se hace referencia a los dibujos adjuntos que muestran, a modo de ilustración, realizaciones específicas en las que se puede poner en práctica la invención. Estas realizaciones se han descrito con suficiente detalle para habilitar a los expertos en la técnica a poner en práctica la invención. Ha de entenderse que las distintas realizaciones de la invención, aunque diferentes, no son de forma necesaria mutuamente excluyentes. Por ejemplo, un rasgo, una estructura o una característica particular descrito en la presente memoria en conexión con una realización puede ser implementado dentro de otras realizaciones sin salirse del alcance de la invención. Además, ha de entenderse que la ubicación o disposición de los elementos individuales dentro de cada realización descrita puede ser modificada sin salirse del espíritu y alcance de la invención. La siguiente descripción detallada, por lo tanto, no ha de ser tomada en un sentido limitativo, y el alcance de la presente invención está definido solo por las reivindicaciones adjuntas, interpretadas apropiadamente, junto con el intervalo completo de equivalentes al que tienen derecho las reivindicaciones. En los dibujos, números similares se refieren a la misma funcionalidad o a una similar a o largos de las diversas vistas.

En una realización de la invención, se ha proporcionado una tecnología que habilita la utilización de tarjetas inteligentes, u otros dispositivos de seguridad portátiles, que han de ser utilizados para firmar digitalmente documentos o para descifrar documentos o mensajes cifrados utilizando claves privadas almacenadas en las tarjetas inteligentes de una manera que reduce eficientemente el riesgo de ataques de análisis de potencia diferencial.

Las tarjetas inteligentes son tarjetas de plástico con microprocesador integrado y un almacenamiento seguro. Son portátiles, seguras, y resistentes a la manipulación. Las tarjetas inteligentes proporcionan servicios de seguridad en muchos campos incluyendo telecomunicación, banca, comercio, e identidad ciudadana. Las tarjetas inteligentes pueden adoptar diferentes formas, tales como tarjetas con forma de tarjeta de crédito con conectores eléctricos para conectar la tarjeta inteligente a un lector de tarjeta inteligente, autentificadores USB con tarjetas inteligentes insertadas, y tarjetas SIM para utilizar en teléfonos móviles y tabletas. Las tarjetas inteligentes son utilizadas en la presente memoria como ejemplos de dispositivos de seguridad portátiles que pueden ser utilizados en implementaciones de la tecnología descrita en la presente memoria. Otros ejemplos de dispositivos de seguridad portátiles incluyen tarjetas de memoria inteligentes, memoria flash, etc. En una realización preferida, el dispositivo de seguridad portátil tiene un procesador, una memoria para almacenar programas y datos, y algunas características de seguridad para hacer el dispositivo relativamente a prueba de manipulaciones. Las tarjetas inteligentes son utilizadas en la presente memoria como ejemplos de tales dispositivos.

Aunque el mecanismo para enmascarar un cálculo criptográfico descrito en la presente memoria puede ser utilizado ventajosamente en tarjetas inteligentes y otros autentificadores de seguridad portátiles utilizados para realizar cálculos criptográficos, los mimos mecanismos también pueden ser utilizados con otros procesadores criptográficos. Así, las tarjetas inteligentes son utilizadas en la presente memoria solamente con fines ilustrativos.

La firma digital y otra criptografía son ejemplos de funciones que proporcionan las tarjetas inteligentes. La tarjeta inteligente almacena claves privadas o secretas compartidas en su almacenamiento seguro y realiza operaciones criptográficas para generar una firma digital para una entrada dada o para descifrar una entrada dada. Una tarjeta inteligente funciona con un dispositivo principal, tal como un ordenador personal (PC), un teléfono celular, una tableta o un terminal bancario. Una aplicación de PC, tal como un cliente de email o un navegador de internet, funciona típicamente con una tarjeta inteligente para firmar, cifrar, o descifrar un documento. La operación criptográfica puede ser parte de un mecanismo de respuesta a un desafío para la autentificación de usuario. La aplicación de PC y la tarjeta inteligente interactúan a través de algún API criptográfico llamado middleware, que está diseñado para comunicar con la tarjeta inteligente. En este escenario, la tarjeta inteligente proporciona servicios localmente al PC.

La Figura 1 es una ilustración esquemática de una red 111 que conecta un ordenador anfitrión 103 con un dispositivo 109 de seguridad portátil, p. ej., una tarjeta inteligente, conectada al mismo, a uno o más servidores remotos 113. El

ordenador anfitrión 103 es operado por un usuario 101 quien interactúa con uno de los servidores 113 a través de una ventana 105 de navegador de red de un navegador de red. En el escenario ejemplar ilustrado en la Figura 1, la tarjeta inteligente 109 proporciona las operaciones criptográficas de parte del usuario 101, p. ej., para firmar criptográficamente documentos, para descifrar mensajes recibidos desde la parte dependiente 113, o para realizar una operación criptográfica como parte de un mecanismo de autentificación de respuesta a un desafío.

Aunque la Figura 1 proporciona una ilustración de un escenario en el cual la criptografía puede jugar un papel importante, existen otros muchos usos importantes para la criptografía. Así, la tecnología utilizada en la presente memoria no está limitada en esta aplicación al ejemplo de uso que se ha ilustrado en la Figura 1.

La Figura 2 es una ilustración esquemática de un dispositivo 109 de seguridad portátil, por ejemplo, una tarjeta inteligente. El dispositivo 109 de seguridad portátil puede incluir un procesador 201 conectado a través de un bus 202 a una memoria de acceso aleatorio (RAM) 203, una memoria de solo lectura (ROM) 204, y una memoria no volátil (NVM) 205. El dispositivo 109 de seguridad portátil incluye además una interfaz 207 de entrada/salida para conectar el procesador 201, de nuevo típicamente a través del bus 202, a un conector 211 mediante el cual el dispositivo 109 de seguridad portátil puede estar conectado al ordenador anfitrión 103.

En realizaciones alternativas, la conexión entre el ordenador anfitrión 103 y el dispositivo 109 de seguridad portátil es inalámbrica, por ejemplo, utilizando comunicación de campo cercano (NFC) u otras tecnologías de comunicación por radio o microondas.

La NVM 205 y/o ROM 204 puede incluir programas informáticos 301 como se ha ilustrado en la Figura 3. Aunque se ha representado aquí que los programas informáticos 301 están todos ubicados en la ROM 204 o la NVM 205, en la práctica real no existe tal restricción ya que los programas pueden ser dispersados sobre múltiples memorias e incluso instalados temporalmente en la RAM 203. Además, el dispositivo 109 de seguridad portátil puede incluir múltiples ROM o NVM. Los programas 301 incluyen programas de sistema operativo así como programas de aplicación cargados sobre el dispositivo 109 de seguridad portátil. La NVM 205 o la ROM 204 también pueden contener datos privados, tales como una clave privada 209 o una clave secreta compartida 210, almacenada bien en su forma básica o bien en cantidades derivadas.

Los programas 301 del dispositivo 109 de seguridad portátil pueden incluir un módulo criptográfico 213, un módulo 215 de autentificación de usuario, un módulo 217 de comunicación, y el sistema operativo OS 219.

Así, el dispositivo 109 de seguridad portátil puede recibir un documento o un mensaje a través del conector 211. El procesador 201, ejecutando instrucciones del módulo 213 de criptografía, puede firmar digitalmente el documento/mensaje o puede descifrar el documento/mensaje utilizando la clave privada 209 o la clave secreta compartida 210. Utilizar la funcionalidad proporcionada a través del módulo 217 de comunicaciones, el procesador 201 puede recibir y transmitir comunicaciones con el ordenador anfitrión 103.

La Figura 4 es un esquema de una implementación posible de la técnica anterior del módulo 213 de criptografía. El módulo 213 de criptografía contendría una o más funciones, métodos, o rutinas. Una posible función sería, como se ha ilustrado en la Figura 4, una función llamada Función Criptográfica ( ) que toma el argumento M, el mensaje a firmar o descifrar. En el módulo 213 de criptografía la firma S es calculada utilizando la ecuación RSA estándar 401, en particular, $S = m^D \bmod n$.

Un enfoque alternativo de la técnica anterior implementa la Función Criptográfica ( ) utilizando el Teorema Chino del Resto para realizar una operación criptográfica; incluye cálculos 401 de exponenciación modular en elementos de la mitad de tamaño.

Como apreciaría un experto en la técnica, esta operación sería reducida a declaraciones aritméticas de nivel inferior en aras de la eficiencia. Un enfoque común para calcular de manera eficiente $M^{dp} \bmod p$ es el algoritmo de cuadrado y multiplicación siempre. La Figura 5 es un listado de programas para un módulo criptográfico 213' que ilustra un procedimiento de cuadrado y multiplicación siempre para calcular $Sp = M^{dp} \bmod p$ (algoritmo 401a) y $Sq = M^{dq} \bmod q$ (algoritmo 401b) que utiliza las cantidades dp, dq e iq (paso 501) que son definidas como:

$$dp = d \bmod (p - 1)$$

$$dq = d \bmod (q - 1)$$

$$iq = q^{-1} \bmod p$$

en donde dp y dq son escritos en las representaciones binarias

$$dp = [dp_{n-1}, dp_{n-2}, \ldots, dp_2, dp_1, dp_0]$$

y

$$dq = [dq_{n-1}, dq_{n-2}, \ldots, dq_2, dq_1, dq_0]$$

S puede ser entonces calculada utilizando la fórmula de Garner, paso 503:

$$S = Sq + q^*(iq^*(Sp - Sq) \bmod p$$

El algoritmo de la Figura 5 es mucho más eficiente que el algoritmo de la Figura 4 porque utiliza elementos de mitad de tamaño.

De acuerdo con una realización de la invención descrita en la presente memoria a continuación, el módulo criptográfico 213' (Figura 6) utiliza una modificación a los algoritmos de exponenciación en las figs. 4 y 5 que, como el algoritmo en la Figura 5, utiliza exponentes de mitad de tamaño mientras que realiza sola una exponenciación. Este enfoque modificado se ha ilustrado en la Figura 5.

La Figura 6 es un listado de programas que ilustra un cálculo 401c de exponenciación modular modificado para calcular el resultado $S = m^D \bmod n$ utilizado en una realización preferida de un módulo criptográfico 213" incorporado, por ejemplo, a una memoria, p.ej., la ROM 204 o NVM 205 de un dispositivo 109 de seguridad portátil, utilizando exponentes de mitad de tamaño mientras que realiza solo una exponenciación. El cálculo 401c de exponenciación calcula *S* mediante una utilización alternativa del Teorema Chino del Resto.

Las entradas al algoritmo son:

m – el mensaje que ha de ser descifrado

q y *p* – los dos números primos grandes que son multiplicados para calcular *n*

El cálculo 401c de exponenciación comienza realizando tres cálculos preliminares, 601:

$$iq = q^{-1} \bmod p$$

$$mq = 1 + q^*iq^*(m - 1) \bmod n$$

$$mp = 1 + (1 - q^*iq^*) * (m - 1) \bmod n$$

Se ha mostrado a través de la aritmética modular que a partir de los cálculos anteriores, las siguientes relaciones sostienen:

$$mq \bmod p = 1$$

$$mq \bmod q = m \bmod q$$

$$mp \bmod q = 1$$

$$mp \bmod p = m \bmod p$$

El cálculo también utiliza las cantidades *dp* y *dq*, que son definidas a partir de las cantidades *p* y *q*, respectivamente, como se ha descrito anteriormente, como:

$$dp = d \bmod (p - 1)$$

$$dq = d \bmod (q - 1)$$

Un valor *A* de acumulador es inicializado a 1, Paso 603.

Después, con la representación binaria de *dp* como $dp = [dp_0, dp_1, \ldots, dp_{k-1}, dp_k]$ y $dq = [dq_0, dq_1, \ldots, dq_{k-1}, dq_k]$, *S* es calculada iterativamente (bucle 605) modificando el acumulador *A* sobre los bits de *dp* y *dq* y dependiendo del valor de cada bit dp_i y dq_i que realiza actualizaciones del valor *A*, como sigue:

En el inicio de cada iteración, *A* es establecida como $A = A^*A \bmod n$, paso 607.

El par de valor dp_i y dq_i presenta cuatro alternativas exclusivas mutuamente posibles: $dp_i = 0$ y $dq_i = 0$, $dp_i = 1$ y $dq_i = 0$, $dp_i = 0$ y $dq_i = 1$, y $dp_i = 1$ y $dq_i = 1$.

Para la primera de estas alternativas ($dp_i = 0$ y $dq_i = 0$), *A* es establecida como $A = A^*1 \bmod n$, pasos 609. Como esto es una operación de identidad, en una implementación real, el paso es puenteado no haciendo nada cuando la operación no cambia el valor de *A*.

Para la segunda alternativa ($dp_i = 1$ y $dq_i = 0$), *A* es establecida como $A = A^*mp \bmod n$, pasos 611.

Para la tercera alternativa ($dp_i = 0$ y $dq_i = 1$), *A* es establecida como a $A = A^*mq \bmod n$, pasos 613.

Para la cuarta alternativa ($dp_i = 1$ y $dq_i = 1$), *A* es establecida como a $A = A^*m \bmod n$, pasos 615.

En la conclusión, después de que todos los bits de dp_i y dq_i han sido procesados por el bucle 605, el resultado mantenido en A mantiene el valor $S = m^D \bmod n$ y puede ser devuelto a la rutina de llanada como el mensaje firmado S, Paso 617.

En cada iteración i de la exponenciación, el acumulador A es igual a S_i de tal manera que:

$$S_i \bmod p = m^{(dp0\ dp1\ dp2\ldots\ dpi)} \bmod p$$

$$S_i \bmod q = m^{(dq0\ dq1\ dq2\ldots\ dqi)} \bmod q$$

Estas relaciones son verdaderas porque:

- en el paso 615, cuando se multiplica por m, la multiplicación del acumulador $A*m$ toma el módulo $p*q$ porque n es definido como $n = pq$

- en el paso 611, cuando se multiplica por mp, la multiplicación del acumulador $A*mp$ es equivalente a $A*m$ módulo p porque la multiplicación es 1 módulo q; consecuentemente no existe así ningún cambio en A debido a q

- en el paso 613, cuando se multiplica por mq, la multiplicación del acumulador $A*mq$ es equivalente a $A*m$ módulo q porque la multiplicación es 1 módulo p; consecuentemente no existe ningún cambio en A debido a p

- en el paso 609, cuando se multiplica por 1, la multiplicación de $A*1$ es $A*1$ módulo p y q, consecuentemente no existe ningún cambio debido ni al módulo p ni al modulo q

Así, después de la iteración final – es decir, dónde $i = k$:

$$Sp = S_k \bmod p = m^{dp} \bmod p$$

$$Sq = S_k \bmod q = m^{dq} \bmod q$$

En otras palabras, porque

$$Sp = S \bmod p$$

$$Sq = S \bmod q$$

Se deduce que

$$S = S_k = A$$

A partir de lo anterior es evidente que se ha presentado un mecanismo en la presente memoria que calcula el mensaje firmado S de una manera muy eficiente utilizando valores de exponente de mitad de tamaño sin exponer múltiples exponenciaciones a ataques por fallo protegiendo de este modo contra la detección del material clave utilizado en el cifrado.

El mecanismo descrito anteriormente ha sido descrito en el contexto de la técnica cuadrado y multiplicación siempre. El mecanismo es adaptado fácilmente a otras técnicas de exponenciación.

Aunque se han descrito e ilustrado realizaciones específicas de la invención, la invención no está limitada a formas o disposiciones de partes así descritas e ilustradas. La invención está limitada solo por las reivindicaciones.

REIVINDICACIONES

1. Un método para operar un aparato de criptografía para realizar una operación de descifrado que tiene una operación de exponenciación X, protegiendo el método al aparato para que no revele información en relación con la operación de exponenciación X cuando la operación está expuesta a un ataque por fallos, comprendiendo el método producir un resultado equivalente a la exponenciación mediante:

recepción de un mensaje *m* sobre el cual realizar una operación criptográfica equivalente a la operación de exponenciación $S = m^d \bmod n$;

determinación de dos exponentes de la mitad de tamaño a partir del exponente *d*;

división de la base *m* en dos sub-bases *mp* y *mq* determinadas a partir de la base *m*;

cálculo iterativo de *S* multiplicando repetidamente un acumulador *A* por *m*, *mp*, *mq* o 1 dependiendo de los valores del bit *i*-th de *dp* y *dq* para cada iteración *i*;

devolución cuando el valor *S* es el valor final del acumulador *A*; y

finalización de la operación criptográfica utilizando el valor *S* obtenido a partir de la operación.

2. El método de la reivindicación 1 en donde los dos exponentes de la mitad de tamaño son *dp* y *dq* de tal manera que $dp = d \bmod (p - 1)$ y $dq = d \bmod (q - 1)$ donde *p* y *q* son números primos de tal manera que $n = pq$.

3. El método de la reivindicación 1 o 2 en donde:

$$mp = 1 + q^* iq^* (m - 1) \bmod n;$$

y

$$mp = 1 + (1 - q^* iq) * (m - 1) \bmod n$$

en donde

$$iq = q^{-1} \bmod p.$$

4. El método de cualquier reivindicación precedente en donde *dp* y *dq* tienen bits indexados de 0 a *k* y la iteración es una iteración de 0 a *k* que realiza los cálculos:

$$A = A^* A \bmod n$$

IF ($dp_i = 0$ && $dq_i = 0$)

$$A = A^*1 \bmod n$$

IF ($dp_i = 1$ && $dq_i = 0$)

$$A = A^* mp \bmod n$$

IF ($dp_i = 0$ && $dq_i = 1$)

$$A = A^* mq \bmod n$$

IF ($dp_i = 1$ && $dq_i = 1$)

$$A = A^*m \bmod n.$$

5. Un dispositivo electrónico protegido del ataque por fallos que comprende una unidad central de procesamiento, una memoria, y un almacenamiento de instrucciones en donde el almacenamiento de instrucciones contiene instrucciones para hacer que la unidad central de procesamiento realice el método de cualquiera de las reivindicaciones precedentes.

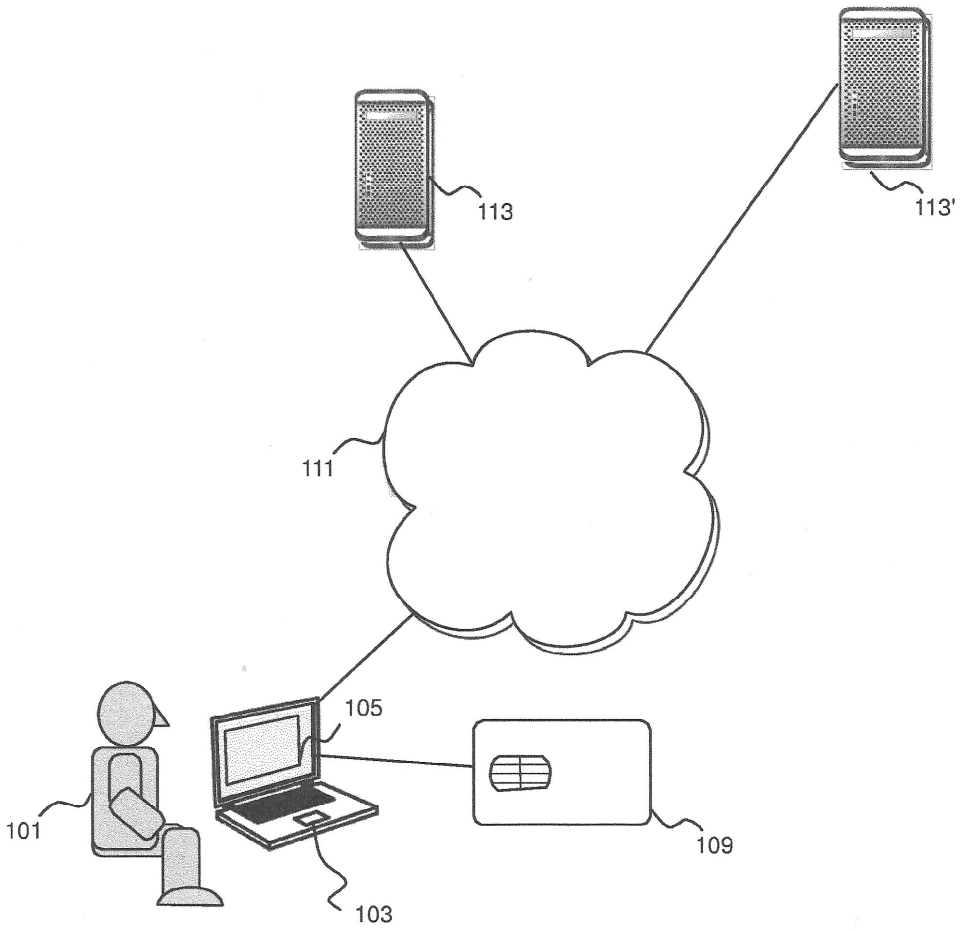

Fig. 1

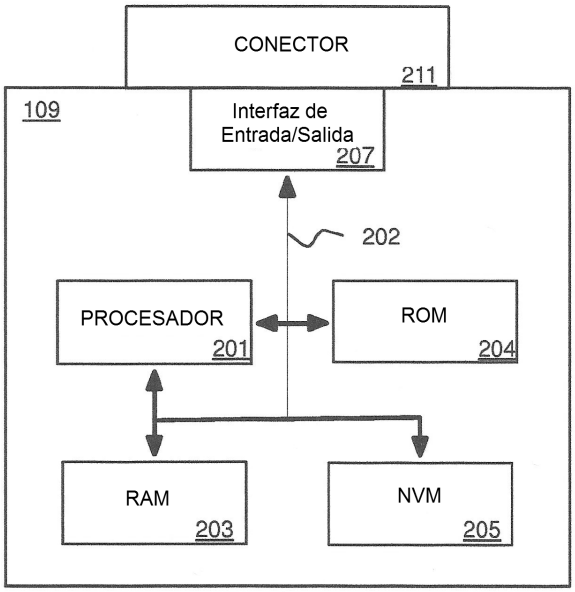
CONECTOR
211
109
Interfaz de Entrada/Salida
207
202
PROCESADOR
201
ROM
204
RAM
203
NVM
205

Fig. 2

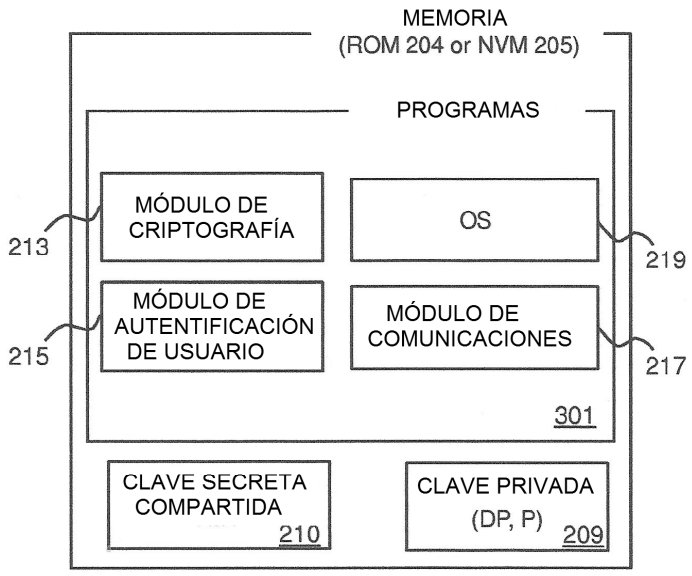
MEMORIA
(ROM 204 or NVM 205)
PROGRAMAS
MÓDULO DE CRIPTOGRAFÍA
213
OS
219
MÓDULO DE AUTENTIFICACIÓN DE USUARIO
215
MÓDULO DE COMUNICACIONES
217
301
CLAVE SECRETA COMPARTIDA
210
CLAVE PRIVADA
(DP, P)
209

Fig. 3

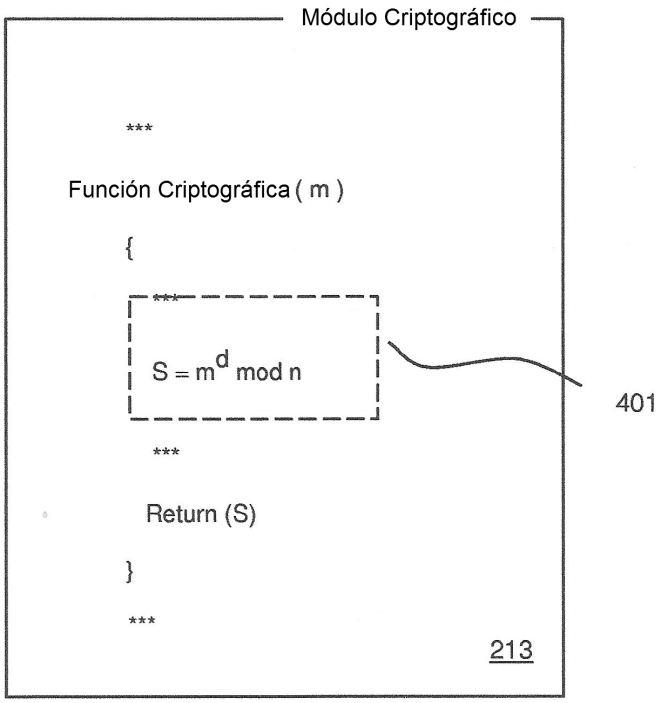

Fig. 4 (Técnica Anterior)

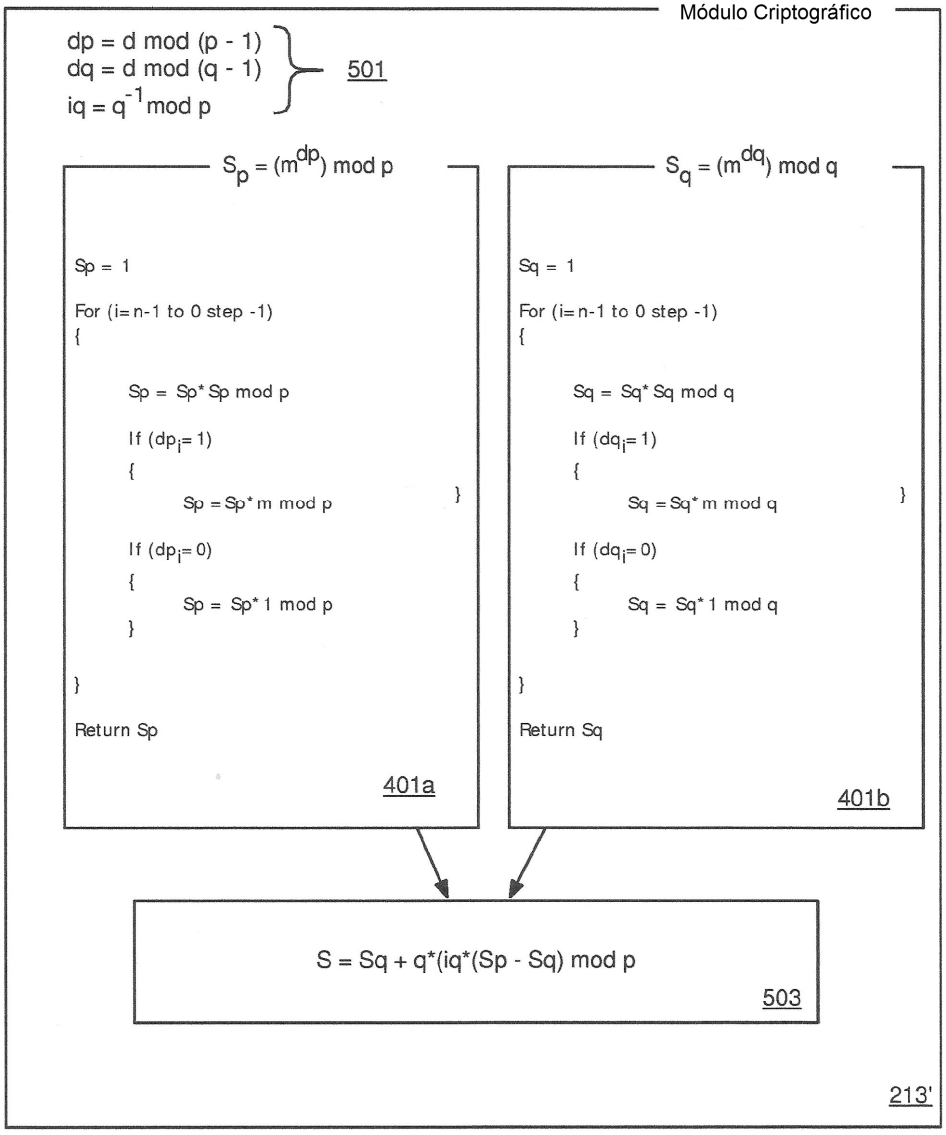

Fig. 5 (Técnica Anterior)

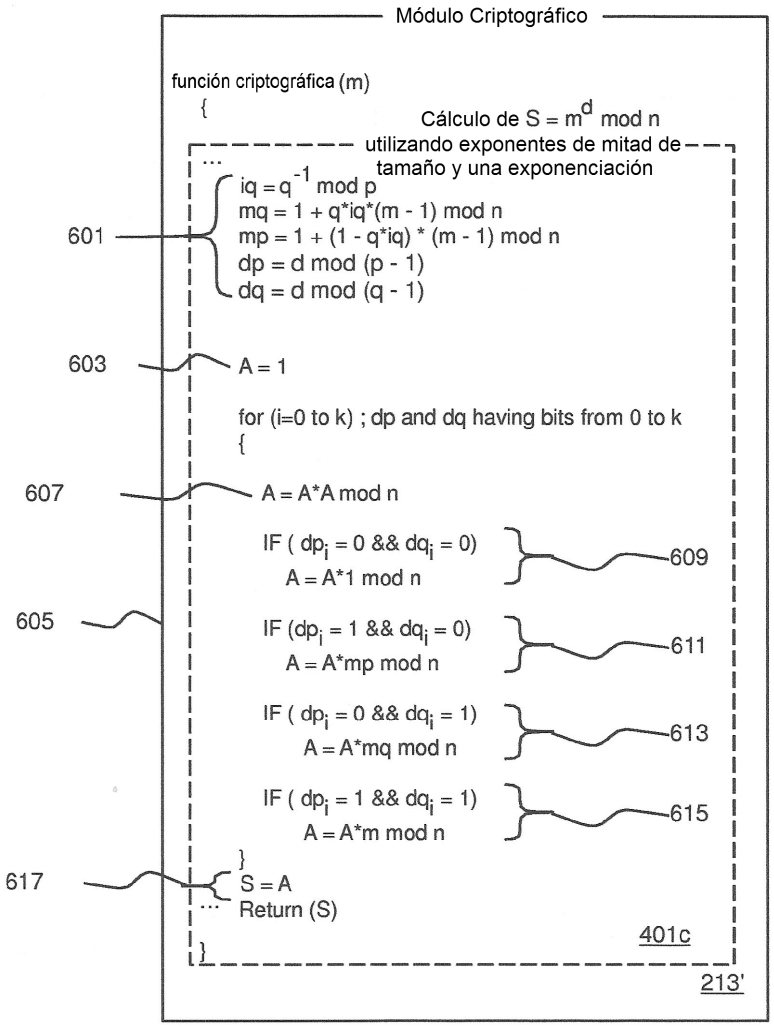
Módulo Criptográfico
función criptográfica (m)
{
Cálculo de S = m^d mod n
utilizando exponentes de mitad de tamaño y una exponenciación
...
iq = q^-1 mod p
mq = 1 + q*iq*(m - 1) mod n
mp = 1 + (1 - q*iq) * (m - 1) mod n
dp = d mod (p - 1)
dq = d mod (q - 1)
601
603
A = 1
for (i=0 to k) ; dp and dq having bits from 0 to k
{
607
A = A*A mod n
IF (dp_i = 0 && dq_i = 0)
A = A*1 mod n
609
605
IF (dp_i = 1 && dq_i = 0)
A = A*mp mod n
611
IF (dp_i = 0 && dq_i = 1)
A = A*mq mod n
613
IF (dp_i = 1 && dq_i = 1)
A = A*m mod n
615
}
617
S = A
...
Return (S)
401c
}
213'

Fig. 6