(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 604 460**

(51) Int. CI.:

H04L 9/30 (2006.01)
G06F 7/72 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **26.07.2006 PCT/EP2006/064655**

(87) Fecha y número de publicación internacional: **15.03.2007 WO07028669**

(96) Fecha de presentación y número de la solicitud europea: **26.07.2006 E 06792567 (7)**

(97) Fecha y número de publicación de la concesión europea: **31.08.2016 EP 1922837**

(54) Título: **Procedimiento para codificar o decodificar con seguridad un mensaje**

(30) Prioridad:

06.09.2005 DE 102005042339

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
07.03.2017

(73) Titular/es:

SIEMENS AKTIENGESELLSCHAFT (100.0%)
Wittelsbacherplatz 2
80333 München, DE

(72) Inventor/es:

KARGL, ANTON;
MEYER, BERND y
BRAUN, MICHAEL

(74) Agente/Representante:

LOZANO GANDIA, José

PROCEDIMIENTO PARA CODIFICAR O DECODIFICAR CON SEGURIDAD UN MENSAJE

DESCRIPCIÓN

La invención se refiere a un procedimiento para codificar o decodificar con seguridad un mensaje o para generar o verificar una firma digital de un mensaje, en el que con el apoyo de un procesador se aplica al mensaje una operación matemática con una clave, la cual puede representarse como número binario con una secuencia de bits.

Los sistemas de criptografía asimétrica garantizan mediante la creación de pares de claves, compuestos por clave pública y privada, un nivel elevado de seguridad en la medida en que le es casi imposible a un atacante decodificar la clave privada o el mensaje codificado con clave pública en un tiempo finito. Sistemas de criptografía convencional, tales como los basados en curvas elípticas, se basan en una codificación que puede realizarse en tiempo polinómico, pero sólo puede invertirse en tiempo exponencial con respecto a la longitud de la clave en bits. En sistemas basados en curvas elípticas se utilizan actualmente longitudes de clave de n = 160 a 192 bits y en los sistemas basados en algoritmos RSA han de utilizarse para ello longitudes de n = 1024 a 1536 bits para niveles aproximadamente iguales de seguridad.

Una alternativa a los métodos de ataque basados en la inversión de la codificación, que hasta incluyen algoritmos para romper lo más eficientemente posible el algoritmo que sirve de base a la codificación, son los llamados ataques de canal lateral. Éstos pueden utilizarse en particular en elementos auxiliares móviles, como por ejemplo tarjetas inteligentes o dongles (llaves de seguridad), en los que está memorizado material secreto codificado, para permitir un intercambio de mensajes codificados o para generar firmas digitales o decodificarlas de nuevo.

El atacante utiliza la accesibilidad relativamente fácil de líneas de datos de los correspondientes circuitos para medir magnitudes físicas como intensidad de corriente, radiación electromagnética, resultados de errores inducidos o tiempos de duración de determinados cálculos. En la evaluación inmediata de los valores medidos usando un análisis de intensidad de corriente (SPA) simple o mediante el registro de los valores medidos como la corriente por medio de un osciloscopio con memoria y subsiguiente evaluación estadística, pueden obtenerse de manera eficiente informaciones sobre el algoritmo subyacente o también simplemente sólo una clave actualmente existente.

Esto último se ilustrará más en detalle mediante un ejemplo: Un procedimiento de codificación prevé, tanto para algoritmos basados en curvas elípticas como también basados en el procedimiento RSA, la aplicación de una operación matemática. En el caso del procedimiento RSA ha de realizarse una exponenciación modular

$$z = a^k \bmod N$$

siendo el módulo N así como a números naturales y k es una clave (privada o pública) o al menos una magnitud derivada de la misma. En el caso de curvas elípticas, es una multiplicación escalar

$$Q = k * P$$

Como operación matemática a realizar, donde P es un punto en una curva elíptica sobre un cuerpo finito K y k es de nuevo una clave o una magnitud derivada de la misma.

Una curva elíptica es un conjunto de lugares de ceros de una ecuación cuadrática

$$y^2 + a_1 xy + a_3 y = x^3 + a_2 x^2 + a_4 x + a_6,$$

con coeficientes a_i en un cuerpo finito. La curva elíptica no tiene puntos singulares y forma, con la adición de un punto infinitamente lejano como elemento neutro, un grupo aditivo cuya ley de grupo puede ser interpretada geométricamente. Cada recta intersecta una curva elíptica en tres puntos no necesariamente diferentes y para cada dos puntos se puede calcular un tercer punto, de modo que la suma de los tres puntos es el elemento neutro. La adición se realiza (ilustrada geométricamente) trazando una línea recta a través de dos puntos P y Q, determinando el tercer punto S que corta la curva elíptica S y realizando la simetría especular del punto en el eje x. Una multiplicación escalar de un punto por un escalar k resulta añadiendo k veces un punto a sí mismo.

Los criptosistemas basados en curvas elípticas son conocidos desde 1985 (introducidos por Koblitz, N. y Miller, V.). Éste y el procedimiento RSA (Rivest, R., Shamir, A., Adleman, L.) más antiguo se describen en detalle por ejemplo en Menezes, van Oorschot, Vanstone, “Manual de Criptografía Aplicada”, CRC Press, (1996). En contraste con las curvas elípticas, se calcula en los sistemas RSA en semigrupos multiplicativos. El análogo de la adición según las curvas elípticas es entonces la multiplicación en el sistema RSA. De la multiplicación escalar resulta aquí una exponenciación.

Volviendo al ejemplo ilustrativo, puede observarse que una implementación de cálculo de la operación matemática puede tener lugar mediante el siguiente algoritmo, viniendo predeterminado k por una representación binaria (b_i, i = n - 1 ... 0):

Algoritmo 1a: (RSA: $z = a^k \bmod N$)

(1) $z \leftarrow 1$
(2) $i \leftarrow n - 1$
(3) siempre que i > -1
(3.1) $z \leftarrow z^2 \bmod N$
(3.2) si bi = 1 entonces $z \leftarrow z * a \bmod N$
(3.3) $i \leftarrow i - 1$
(4) da como resultado z

Algoritmo 1b: (EC - curva elíptica: Q = k * P)

(1) $Q \leftarrow 0$
(2) $i \leftarrow n - 1$
(3) siempre que i > -1
(3.1) $Q \leftarrow 2 * Q$
(3.2) si b i = 1 entonces $Q \leftarrow Q + P$
(3.3) $i \leftarrow i - 1$
(4) da como resultado Q

Por lo tanto, se realizan así esencialmente dos operaciones aritméticas en cada ciclo de bucle, un denominado square-and-multiply (cuadrado-y-multiplicar) en el sistema RSA y un sistema denominado double-and-add (doble-y-sumar) en el sistema CE (CE se utilizará en lo que sigue como abreviatura de curvas elípticas) .

En el caso de un análisis de corriente simple (SPA), se analiza el perfil del consumo de corriente de una exponenciación o una multiplicación escalar. La multiplicación escalar por ejemplo consta principalmente en adiciones y duplicaciones. Las operaciones difieren significativamente en el número de operaciones elementales en K, de modo que también el consumo de corriente es diferente. Por lo tanto, se pueden sacar conclusiones incluso mediante un ataque de canal lateral correspondiente sobre los bits individuales y por tanto la propia representación binaria de k.

Un primer paso para defenderse contra tales ataques es igualar los flujos de corriente dependientes del valor de los respectivos bits y los tiempos de cálculo para ambos estados posibles de bits 0 y 1, como se muestra a continuación:

Algoritmo 2a: (RSA: $z = a^k \bmod N$)

(1) $z_0 \leftarrow 1$
(2) $i \leftarrow n - 1$
(3) siempre que i > -1
(3.1) $z_0 \leftarrow z_0^2 \bmod N$
(3.2) $z_1 \leftarrow z_0 * a \bmod N$
(3.3) $z_0 \leftarrow z_{bi}$ con $b_i = 0$ ó 1
(3.4) $i \leftarrow i - 1$
(4) da como resultado z_0

Algoritmo 2b: (EC - curva elíptica: Q = k * P)

(1) $Q_0 \leftarrow 0$
(2) $i \leftarrow n - 1$
(3) siempre que i > -1
(3.1) $Q_0 \leftarrow 2 * Q_0$
(3.2) $Q_1 \leftarrow Q_0 + P$
(3.3) $Q_0 \leftarrow Q_{bi}$ con $b_i = 0$ ó 1
(3.4) $i \leftarrow i - 1$
(4) da como resultado Q_0

Se ejecutan ahora, en cualquier caso independientemente del valor del correspondiente bit b_i, la primera operación aritmética (paso 3.1) al igual que también ahora la segunda operación aritmética (paso 3.2). Desde luego, el tiempo de ejecución aumenta aquí considerablemente, porque han de realizarse más operaciones.

El valor del propio bit incide solamente en la asignación en el paso (3.3). Si $b_i = 0$, el punto Q_0 (caso CE) se actualiza en este ciclo del bucle sólo por sí mismo. En otras palabras, el valor calculado en el paso

(3.2) para Q_1 sigue ignorándose, se calculó “inútilmente” en términos de la valoración de su resultado. Si por el contrario es $b_i = 1$, se asigna a Q_0 el valor Q_1. En base a las dos variables auxiliares W_0, Q_1 que hay ahora, se logra así en cualquier caso al menos una aproximación en cuanto al consumo de corriente y al tiempo de cálculo.

Desde luego esto no es completo, tal como se ha comprobado. El paso (3.3) implica precisamente, similarmente a una consulta if/else (si/entonces), un salto de direcciones. En función de la dirección, puede notarse también aquí un consumo de corriente diferente, por lo que resulta posible un ataque de canal lateral.

Un método alternativo de defensa contra ataques de canal lateral es la llamada escala de Montgomery. Este algoritmo es particularmente eficaz en el caso de curvas elípticas, pero por la publicación DE 10151129 A1 también se conoce una aplicación a sistemas RSA. En el cálculo de la multiplicación escalar se puede prescindir precisamente del cálculo a la vez de la coordenada y si se calcula en cada caso el producto del punto sobre la curva elíptica por un factor k y al mismo tiempo por un factor k + 1. La coordenada y puede ser reconstruida en el caso de las curvas elípticas a partir de ambos resultados parciales.

El cálculo simultáneo de ambos resultados de las multiplicaciones se integra, como consecuencia del algoritmo, ventajosamente en un esquema común, en el que las variables auxiliares definidas para ello (aquí llamadas R y S) se actualizan en cada ciclo de bucle eficientemente con ayuda mutua:

Algoritmo 3: (EC - curva elíptica: Q = k * P)

Escala de Montgomery:

(1) $R \leftarrow P, S \leftarrow 0$
(2) $i \leftarrow n - 1$
(3) siempre que $i > -1$
(3.1) si $b_i = 1$ {$S \leftarrow S + R, R \leftarrow 2 * R$}
(3.2) caso contrario {$R \leftarrow R + S, S \leftarrow 2 * S$ }
(3.3) $i \leftarrow i - 1$
(4) da como resultado R, S
(5) reconstruye k * P a partir de los puntos R, S y P

En el ejemplo mostrado precede en cada ciclo de bucle la variable auxiliar R y el punto R en 1 * P a la magnitud auxiliar o el punto S. P se había predeterminado. Para k = (1010) = 10 es por ejemplo S = 10 * P y R = 11 * P. La adición y la duplicación discurren con independencia de los bits por completo de la misma forma. Por parte de la secuencia de las operaciones no puede sacarse ninguna conclusión relativa a la secuencia de bits. Sin embargo constituye la instrucción de salto ("si" o bien "caso contrario"), tal como se mencionó al principio, un punto débil para un ataque.

Por CHEVALIER-MAMES B “Self-randomized exponentiation algorithms” (Algoritmos de exponenciación auto-aleatorizados) se conoce un algoritmo en el que se protegen procedimientos criptográficos asimétricos frente a ataques de canal lateral mediante la llamada “Self-Randomized Exponentiation” (exponenciación auto-aleatorizada).

El exponente D se considera entonces como fuente adicional de aleatoriedad durante la exponenciación. El procedimiento puede utilizarse sobre RSA y algoritmos de curvas elípticas y se implementa en una ejecución sin ramificaciones.

El documento XP055155620: A Practical Countermeasure against Address-Bit Differential Power Analysis (Una contramedida práctica contra el análisis de potencia diferencial de bit de dirección) de Kouichi Itohs, Tetsuya Izu y Masahiko Takenakas describe una contramedida práctica frente al bit de dirección DPA, la contramedida del direccionamiento aleatorio, que se aplica con o sin tablas previamente calculadas a exponentes ECC o RSA. Esta contramedida casi no tiene sobrecoste para la protección, no siendo la velocidad de cálculo inferior a la que se tiene sin contramedida.

Es por lo tanto el objetivo de la invención ofrecer un procedimiento mejorado en cuanto a la seguridad contra ataques de canal lateral para codificar o decodificar o bien para generar o verificar una firma digital de mensajes.

El objetivo se consigue mediante un procedimiento para codificar o decodificar de forma segura un mensaje o para generar o verificar una firma digital de un mensaje con las características de las reivindicaciones independientes 1 y 9. De las reivindicaciones dependientes pueden tomarse variantes ventajosas.

Los procedimientos mencionados en las reivindicaciones independientes se refieren a las orientaciones generales mostradas por ejemplo en los algoritmos 2 y 3, consistiendo una idea que conecta los aspectos en la evitación de instrucciones de salto medibles en el perfil de corriente.

Se propone determinar las direcciones de memoria de las variables auxiliares con las que se realizan las operaciones aritméticas en la exponenciación o la multiplicación escalar y realizar incluso operaciones con las mismas. Se prevé al respecto calcular la diferencia de las direcciones de memoria, cada una expresada como números o reproducida en la zona de los números. Si en función del bit de clave actual se agrega la diferencia calculada a una dirección de memoria cuando ésta tuviera el valor menor, o se sustrae si tenía el valor mayor, se obtiene precisamente la dirección de memoria de la otra variable auxiliar (R o S o bien z_0 ó z_1) en cada caso. Por lo tanto, puede lograrse mediante dicho cálculo un intercambio de valores de las dos variables auxiliares o una reproducción de una de las dos variables auxiliares sobre en cada caso la otra.

El intercambio de los valores de las magnitudes auxiliares es ventajoso en la escala de Montgomery: para ambas vías de ramificación dependientes de los bits se realizan operaciones aritméticas idénticas, pero en disposición intercambiada simétricamente de las variables auxiliares.

El cálculo puede realizarse por ejemplo en el contexto de una multiplicación del valor del bit por la diferencia previamente calculada de los valores de dirección de memoria. Si el valor del bit es 0, entonces no se produce a continuación tampoco ningún cambio en la adición o sustracción de las direcciones de memoria. Por el contrario si el valor del bit es 1, se resta o se suma idénticamente la diferencia.

Un aspecto prevé llevar el valor del bit al final de una palabra de cálculo, es decir, hacer que sea el bit menos significativo (LSB). Una multiplicación subsiguiente de la palabra de cálculo por la diferencia aporta un producto que se puede añadir inicialmente a una dirección de memoria. Si ahora de borra el bit (LSB) al valor 0, se vuelve a multiplicar y se sustrae el producto de nuevo, obtenemos la dirección original, si el bit 0 LSB era 0 al principio y por lo tanto no se ha modificado nada debido al proceso de borrado. Sin embargo, si tenía el valor 1, llegamos exactamente en la diferencia en la zona de direcciones más allá a la dirección de memoria de la otra magnitud auxiliar utilizada en la operación matemática.

Los pasos esenciales de la invención son la formación de la diferencia de las direcciones y la adición o sustracción, dependiente del valor actual de los bits, de la diferencia de una de las direcciones de memoria en función de los bits para llegar en cada caso a la otra. Por lo tanto, se omite la instrucción de salto, lo que resulta ventajoso especialmente en la escala de Montgomery. Las operaciones aritméticas a ejecutar en función del valor del bit se encontraban hasta ahora en diversas posiciones en la secuencia del programa. Por el contrario ahora se puede acceder, debido al carácter simétrico de las magnitudes auxiliares en las instrucciones del programa, a las mismas operaciones de cálculo, intercambiándose sólo los contenidos simétricamente. Al omitirse la instrucción de salto, es casi imposible una detección del consumo de energía diferente en función del valor del bit o un tiempo de cálculo correspondientemente diferente, con lo que aumenta la seguridad frente a un ataque de canal lateral.

La invención se describirá ahora más en detalle en base a ejemplos de ejecución.

Se desarrolla partiendo de la secuencia mostrada en el algoritmo 3 de una escala de Montgomery, según la técnica anterior, un algoritmo SPA seguro: Las etapas

(3.1) si $b_i = 1$: {S ← S + R, R ← 2 * R}
(3.2) caso contrario {R ← R + S, S ← 2 * S} pueden expresarse simplificadamente con

F1 = {S ← S + R, R ← 2 * R}

y

F2 = {R ← R + S, S ← 2 * S }

si $b_i = 1$: F1 y en caso contrario F2.

Se llega al conocimiento que sirve de base a la invención de que las instrucciones de salto representan un punto débil frente a ataques de canal lateral, por lo que un paso siguiente consiste en intercambiar las variables auxiliares al principio y al final de un ciclo de bucle, si el bit de clave tiene el valor 0. Entonces sólo necesita ser remitido a una de las dos direcciones de salto, por ejemplo F1:

(3.1) si $b_i = 1$: F1
(3.2) caso contrario {intercambio (R, S), F1, intercambio (R, S)}. En este punto, la detectabilidad mediante ataques de canal lateral se incrementa significativamente de nuevo, porque en función del valor del bit se requiere al copiar dos accesos a memoria por cada palabra de ordenador. Para elementos de cuerpo más largos, se requieren muchos accesos, lo cual se refleja significativamente en el consumo de corriente cuando el bit de clave no está activado en ese momento, por lo que es 0.

Se puede hacer una aproximación como se muestra en el siguiente algoritmo:

Algoritmo 4: (EC - curva elíptica: Q = k * P)

Escala de Montgomery:

(1) R ← P, S ← 0
(2) i ← n - 1
(3) siempre que i > -1
(3.1) si b_i = 1{r ← dirección (R), s ← dirección (S)}
(3.2) caso contrario {r ← dirección (S), s ← dirección (R)}
(3.3) cargar contenido s a S ', cargar contenido r a R'
(3.4) S'← S' + R '
(3.5) R'← 2' * R '
(3.6) guardar S' en s, guardar R' en r
(3.7) i ← i - 1
(4) da como resultado R, S
(5) reconstruye k * P a partir de los puntos R, S y P.

r y s son aquí variables de dirección y provocan el intercambio de las magnitudes auxiliares en función del valor del bit. r muestra la dirección de la magnitud auxiliar o el punto R, cuando b_i = 1 y el punto S, cuando b_i = 0.

Se evita la instrucción de salto en (3.1) según la invención formando la diferencia entre las variables de dirección r y s y multiplicando la misma por una palabra de ordenador h, en la que está archivada la dependencia de los bits. Supongamos ahora que la dirección (S) es mayor que la dirección (R). Además, el escalar k está en un campo de palabras de ordenador.

Algoritmo 5: (EC - curva elíptica: Q = k * P)

Escala de Montgomery:

(1) R ← P, S ← 0
(2) d ← dirección (S) - dirección (R)
(3) i ← n – 1
(4) siempre que i > -1
(4.1) girar b_i cíclicamente sobre el LSB de la palabra de ordenador
(4.2) copiar h a h', borrar el LSB de h'
(4.3) r ← dirección (S), s ← dirección (R)
(4.4) m ← h * d
(4.5) r ← r - m, s ← s + m
(4.6) m ← h' * d
(4.7) r ← r + m, s ← s - m
(4.8) cargar contenido s a S', cargar contenido r a R'
(4.9) S'← S' + R'
(4.10) R'← 2' * R'
(4.11) da como resultado S' en s, guardar R' en r
(4.12) i ← i - 1
(5) da como resultado R, S
(6) reconstruye k * P a partir de los puntos R, S y P.

Las palabras de ordenador h y h' difieren numéricamente como máximo en el valor 1, porque sólo pueden ser distintas en el LSB. Si b_i = 1, entonces r desciende en los pasos (4.4) a (4.7) de la dirección (S) a la dirección (R). Esto corresponde al resultado del paso (3.1) del algoritmo 4, pero aquí no se requirió ninguna instrucción de salto. Se realizó sólo una multiplicación real de escalares (pasos 4.4) y (4.6).

Una aplicación de la invención a un procedimiento RSA tal como existe en el algoritmo 2a según la técnica anterior, muestra la siguiente secuencia:

Algoritmo 6: (RSA: $z = a^k \bmod N$)

(1) $z_0 \leftarrow 1$
(2) d ← dirección (z_0) - dirección (z_1)
(3) i ← n - 1
(4) siempre que i > -1
(4,1) $z_0 \leftarrow z_0^2 \bmod N$
(4.2) $z_1 \leftarrow z_0 * a \bmod N$
(4.3) girar b_i cíclicamente sobre el LSB de la palabra de ordenador

(4.4) copiar h a h', borrar el LSB de h'
(4.5) r ← dirección (z_0)
(4.6) m ← h * d
(4.5) r ← r - m
(4.6) m ← h' * d
(4.7) r ← r + m
(4.8) guardar el contenido r en z_0
(4.9) i ← i - 1
(5) da como resultado z_0

Contrariamente a la escala de Montgomery para curvas elípticas, sólo se tiene que calcularse una dirección en este algoritmo. Sólo ha de guardarse un valor intermedio.

REIVINDICACIONES

1. Procedimiento para codificar o decodificar de forma segura un mensaje o para generar o verificar una firma digital de un mensaje, en el que con el apoyo de un procesador se aplica al mensaje una operación matemática con una clave k, la cual puede representarse como número binario con una secuencia de bits b_i, que incluye las etapas:

- inicializar una primera magnitud auxiliar z_0 y una segunda magnitud auxiliar z_1;
- determinar en cada caso una dirección de memoria para la primera z_0 y la segunda z_1 magnitud auxiliar y asignar la primera dirección de memoria a una variable de dirección r;
- calcular un valor diferencial d a partir de la diferencia entre las direcciones de memoria;
- secuencialmente para cada bit b_i, comenzando por el bit más significativo MSB, realizar las etapas siguientes:

(a) actualizar la primera magnitud auxiliar z_0 aplicando una primera operación aritmética a la primera magnitud auxiliar z_0;
(b) actualizar la segunda magnitud auxiliar z_1 aplicando una segunda operación aritmética a la primera magnitud auxiliar z_0;
(c) añadir el valor diferencial calculado d a la variable de dirección r en función del valor del bit actual b_i de modo que la variable de dirección r lleve asociado como valor bien la dirección de memoria de la primera magnitud auxiliar z_0 o bien la dirección de memoria de la segunda cantidad auxiliar z_1;
(d) actualizar la primera magnitud auxiliar z_0 mediante la transferencia del valor de aquella magnitud auxiliar z_0, z_1 cuya dirección de memoria está asociada a la variable de dirección r.

Después de completar los pasos para cada uno de los bits b_i, emisión de la primera variable auxiliar z_0 como resultado de la operación matemática,
añadiéndose en la etapa (c.) el valor diferencial d a la variable de dirección r en función del valor del bit actual b_i, tal que

- se forma una primera palabra de ordenador h1, en la que el bit actual b_i en el procesamiento secuencial es el bit menos significativo LSB;
- la primera palabra de ordenador h1 se multiplica por la diferencia con un primer producto m;
- el primer producto m1 se sustrae de la variable de dirección r;
- a partir de la primera palabra de ordenador h1 se forma una segunda palabra ordenador h2, colocando el bit menos significativo LSB a cero;
- la segunda palabra de ordenador h2 se multiplica por la diferencia d a un segundo producto m2;
- el segundo producto m2 se añade a la variable de dirección r.

2. Procedimiento según la reivindicación 1,
en el que a la primera magnitud auxiliar z_0 se le asigna un valor 1 durante la inicialización y la primera operación aritmética incluye una cuadratura o multiplicación de la primera magnitud auxiliar z_0 por sí misma.

3. Procedimiento según la reivindicación 2,
en el que se predetermina una base fija a y la segunda operación aritmética incluye una multiplicación de la primera magnitud auxiliar z_0 por la base a.

4. Procedimiento según la reivindicación 3,
en el que se predetermina un número N, que se utiliza para una operación de módulo en las operaciones aritméticas.

5. Procedimiento según la reivindicación 4,
que es un procedimiento RSA de codificación/decodificación o de firma, en el que la operación matemática es una exponenciación modular a^k mod N y en el que el número predeterminado N para la operación de módulo es el producto de dos grandes números primos p*q.

6. Procedimiento según la reivindicación 1,
en el que

- la primera magnitud auxiliar z_0 representa un punto Q_0 en una curva elíptica E sobre un cuerpo finito K y se le asigna el valor 0 en la etapa de la inicialización,
- en la etapa de actualización de la primera magnitud auxiliar z_0, la primera operación aritmética incluye una multiplicación escalar del punto Q_0 por el factor 2,
- la operación matemática incluye una multiplicación escalar k*P.

7. Procedimiento según la reivindicación 6,

en el que en la etapa de actualización de la segunda magnitud auxiliar z_1, la segunda operación aritmética incluye una adición del punto Q_0 con un punto fijamente predeterminado P en la curva elíptica E.

8. Procedimiento para codificar o decodificar de forma segura un mensaje o para generar o verificar una firma digital de un mensaje, en el que el procesador apoya una operación matemática con una clave que se aplica al mensaje, que se puede representar como número binario con una secuencia de bits b_i que comprende las etapas:

- inicializar una primera R y una segunda S magnitud auxiliar;
- determinar en cada caso una dirección de memoria para la primera R y la segunda S magnitud auxiliar;
- asociar la primera dirección de memoria a una primera variable de dirección r y la segunda dirección de memoria a una segunda variable de dirección S;
- calcular un valor diferencial d a partir de la diferencia entre las direcciones de memoria;
- secuencialmente para cada bit b_i, realizar los pasos siguientes:

  (a) añadir el valor diferencial calculado d a la primera variable de dirección r y restar el valor diferencial (d) de la segunda variable de dirección s en función del valor del bit b_i, tal que

  - bien esté asignada como valor a la primera variable de dirección r la dirección de memoria de la primera magnitud auxiliar R y a la segunda variable de dirección s la dirección de memoria de la segunda variable auxiliar S,
  - o bien esté asignada como valor a la primera variable de dirección r la dirección de memoria de la segunda magnitud auxiliar S y a la segunda variable de dirección S la dirección de memoria de la primera magnitud auxiliar R;

  (b) actualización de la variable auxiliar R, S que está asociada a la primera variable de dirección r, aplicando una primera operación aritmética;

  (c) actualización de la variable auxiliar R, S que está asociada a la segunda variable de dirección s, aplicando una segunda operación aritmética.

- después de finalizar las etapas para cada uno de los bits b_i, emisión de la primera R y segunda S variables auxiliares y cálculo de un resultado de la operación matemática de la primera (R) y la segunda S variable auxiliar,

añadiéndose en la etapa a el valor diferencial d a la primera variable de dirección r en función del valor del bit actual b_i, tal que

- se forma una primera palabra de ordenador h1, en la que el bit actual b_i es en el procesamiento secuencial el bit menos significativo LSB;
- la primera palabra de ordenador (h1) se multiplica por la diferencia a un primer producto m1;
- el primer producto m1 se resta de la variable de dirección r;
- a partir de la primera palabra de ordenador h1 se forma una segunda palabra de ordenador h2, colocando el bit menos significativo LSB en cero;
- la segunda palabra de ordenador h2 se multiplica por la diferencia d a un segundo producto m2;
- el segundo producto m2 se añade a la variable de dirección r.

9. Procedimiento según la reivindicación 8,
en el que la primera variable auxiliar R representa un punto de una curva elíptica E sobre un cuerpo finito K y en la etapa de la inicialización se le asigna un punto fijo P.

10. Procedimiento según la reivindicación 8 ó 9,
en el que la segunda magnitud auxiliar S representa un punto de una curva elíptica E sobre un cuerpo finito K y se le asigna un valor 0 en la etapa de la inicialización.

11. Procedimiento según una de las reivindicaciones 8 a 10,
en el que la operación matemática incluye una multiplicación escalar k*P.

12. Procedimiento según cualquiera de las reivindicaciones 8 a 11,
en el que la primera operación aritmética en la etapa de actualizar la magnitud auxiliar asociada a las primeras variables de dirección, incluye una adición de dos puntos de una curva elíptica.

13. Procedimiento según una cualquiera de las reivindicaciones 8 a 12,
en el que la segunda operación aritmética en la etapa de actualización de la magnitud auxiliar asociada a las segundas variables de dirección, incluye una multiplicación escalar de un punto en una curva elíptica por un factor 2 o bien la adición a sí mismo.

14. Procedimiento según la reivindicación 8,
que es un procedimiento RSA de codificación/decodificación o procedimiento de firma, siendo la operación matemática una exponenciación modular a^k mod N con un número predeterminado N para la operación de módulo, que es el producto de dos grandes números primos p*q.

15. Procedimiento según cualquiera de las reivindicaciones 8 a 14,
en el que en la etapa (a) se resta el valor diferencial d de la segunda variable de dirección s en función del valor del bit actual b_i

- añadiendo el primer producto m1 a la variable de dirección r;
- restando el segundo producto m2 de la variable de dirección s.