Firmas digitales

Josep Domingo Ferrer

P05/75024/00990
Módulo 6

UOC
Universitat Oberta de Catalunya

Índice

Introducción

En este módulo presentamos el concepto de *firma digital*. Una firma digital es, para un documento electrónico, un elemento análogo a una firma manuscrita en un documento en papel. Veremos, sin embargo, que la seguridad que ofrecen las firmas digitales es muy superior.

Empezamos en el primer apartado con la definición del concepto de **firma digital**, y lo relacionamos con el de *criptosistema de clave pública*.

A continuación, en el segundo apartado tratamos las firmas digitales más utilizadas actualmente: la **firma RSA**, la **firma de ElGamal** y el ***Digital Signature Standard*** (**DSS**).

En el último apartado definimos el concepto de **función resumen** o ***hash***. Las funciones *hash* proporcionan un resumen de longitud fija de un mensaje arbitrariamente largo. Normalmente, para firmar un documento, se calcula la imagen para una función *hash* y se firma esta imagen en vez del documento original.

Objetivos

En los materiales didácticos asociados a este módulo se hallan los contenidos necesarios para alcanzar los objetivos siguientes:

1. Entender el concepto y la utilidad de las firmas digitales.

2. Conocer los principales algoritmos de firma digital utilizados hoy en día.

3. Entender el papel de las funciones *hash* en la firma digital.

1. Firma digital

La noción de **firma digital** es probablemente uno de los hallazgos fundamentales y más útiles de la criptografía moderna. Con un esquema de firma digital, cada usuario puede firmar mensajes de modo que cualquier persona pueda verificar las firmas más tarde.

1.1. Idea básica

Las firmas mantienen una estrecha relación con la criptografía de clave pública. Más concretamente, cada usuario crea un par de claves: una pública y una privada. El usuario utiliza su clave privada para firmar el mensaje, y cualquiera puede utilizar la clave pública del signatario para verificarla.

El concepto de *clave pública* se trata en el subapartado 2.1 del módulo "Criptosistemas de clave pública" de esta asignatura.

Con este procedimiento, el receptor queda convencido de que el mensaje no ha sido alterado porque está firmado y, además, posteriormente el signatario no puede negar que ha firmado el mensaje porque nadie, salvo el signatario, tiene la clave privada necesaria para generar la firma.

1.2. Formalización

Vamos a formalizar las ideas expuestas en los subapartados anteriores. Sea A un usuario que quiere firmar un mensaje m. Sea $f_A(\ )$ el cifrado bajo la clave pública de A y $f_A^{-1}(\ )$ el cifrado bajo la clave privada de A. Podemos formalizar un protocolo de firma y el protocolo de verificación asociado.

Protocolo de firma

A calcula $s = f_A^{-1}(m)$ para firmar m, es decir, cifra el mensaje con su clave privada; s es la firma del mensaje m.

Protocolo de verificación

Cualquier usuario B puede calcular $f_A(s)$ para verificar la firma del mensaje m y ver si se cumple $f_A(s) \stackrel{?}{=} m$. En caso afirmativo, se acepta la firma s como válida; en caso negativo, la firma no es válida.

Más adelante veremos que normalmente no se firma directamente el mensaje m, sino que se firma un resumen de longitud fija.

La firma de resúmenes se trata en el apartado 3 de este módulo.

2. Esquemas de firma digital

En este apartado vamos a estudiar tres de los esquemas de firma digital más utilizados:

1) El primero se basa en el criptosistema RSA y su infalsificabilidad está relacionada con la dificultad que supone el problema de la factorización.

2) El segundo esquema se basa en el criptosistema de ElGamal y su infalsificabilidad se relaciona con la dificultad que representa el problema del logaritmo discreto.

3) El tercer esquema es el algoritmo estándar de firma homologado por el gobierno de Estados Unidos, que es, de hecho, muy parecido a la firma de ElGamal (con algunas mejoras).

2.1. Firma RSA

Sea un usuario A, su clave pública RSA (n_A, e_A) y una clave privada d_A. Para firmar digitalmente el mensaje m con el RSA, el usuario A calcula la firma $s = m^{d_A} \bmod n_A$. A continuación, A publica el mensaje firmado y difunde la pareja (m, s).

El criptosistema RSA se trata en el subapartado 4.1 del módulo "Criptosistemas de clave pública" de esta asignatura.

Para verificar la firma s, cualquier usuario B puede comprobar si se cumple $s^{e_A} \bmod n_A \stackrel{?}{=} m$. En caso afirmativo, la firma es válida; en caso negativo, no lo es.

Utilización de la firma RSA

Supongamos que la clave pública del usuario A es $(n_A, e_A) = (34.121,15.775)$ y que su clave privada es $d_A = 26.623$. Si YES es el mensaje que A tiene que firmar, el mensaje codificado en base 26 es $m = 16.346$. Entonces A calcula la firma:

$$s = m^{d_A} \bmod n_A = 16.346^{26.623} \bmod 34.121 = 20.904.$$

Si descodificamos el valor de s en base 26, obtenemos la firma BEYA. De modo que se envía la pareja (YES, BEYA).

Cualquier usuario B que vea esta pareja, puede verificar la firma del mensaje. Para hacerlo, codifica en base 26 y obtiene $m = 16.346$, $s = 20.904$. Luego, B recupera la clave pública de A de un directorio de claves públicas. Finalmente, verifica que se cumpla la expresión siguiente:

$$s^{e_A} \bmod n_A = 20.904^{15.775} \bmod 34.121 = m.$$

2.2. Firma de ElGamal

Supongamos que el signatario A ha generado claves de ElGamal en un grupo G a partir de un elemento $\alpha \in G$. Sea a la clave privada de A, $\alpha^a \in G$ su clave pública, y n el orden del grupo G. Para firmar A un mensaje $m \in G$, efectúa el proceso siguiente:

1) A genera un número aleatorio h tal que $\text{mcd}(h, \phi(n)) = 1$.

2) A calcula $r = \alpha^h$ en G.

3) A resuelve la congruencia 2.1 para hallar el valor de s:

$$m \equiv ar + hs \bmod \phi(n). \qquad (2.1)$$

La firma digital para el mensaje m es la pareja (r, s).

El criptosistema de ElGamal se trata en el subapartado 4.2 del módulo "Criptosistema de clave pública" de esta asignatura.

La solución de la ecuación 2.1 es $s \equiv (m - ar)h^{-1} \bmod \phi(n)$ y existe porque la condición $\text{mcd}(h, \phi(n)) = 1$ asegura que h tiene elemento inverso multiplicativo módulo $\phi(n)$.

El cálculo de inversos se trata en el subapartado 1.2 del módulo "Criptosistemas de clave pública" de esta asignatura.

Por otra parte, cualquier usuario B puede efectuar el proceso siguiente para verificar la firma del mensaje m:

1) B calcula $r^s = (\alpha^h)^s$ en G. También calcula $(\alpha^a)^r$ en G.

2) B verifica si se cumple la relación: $(\alpha^a)^r(\alpha^h)^s \stackrel{?}{=} \alpha^m$, donde todas las operaciones están definidas en G.

Utilización de la firma de ElGamal

Supongamos que hemos escogido el grupo $G = \mathbb{Z}^*_{15.485.863}$ y $\alpha = 7$ (de hecho, las potencias de 7 generan todo G).

Al ser $p = 15.485.863$ primo, el orden de G es $\phi(p) = p - 1 = 15.485.862$.

El usuario A tiene como clave privada $a = 28.236$ y ha calculado su clave pública:

$$\alpha^a = 7^{28.236} \bmod 15.485.863 = 12.506.884.$$

Suponemos que A quiere firmar el mensaje $m = 128.688$. Para hacerlo seguirá los pasos siguientes:

1) A escoge el número aleatorio $h = 90.725$, que es coprimo con el orden del grupo, es decir, $\text{mcd}(90.725, 15.485.862) = 1$.

2) Luego A calcula:

$$r = \alpha^h = 7^{90.725} \bmod 15.485.863 = 7.635.256.$$

3) A continuación, resuelve la congruencia $m \equiv ar + hs \bmod (p - 1)$, cuya solución es:

$$s = (m - ar)h^{-1} \bmod (p - 1) =$$
$$= (128.688 - 28.236 \cdot 7.635.256) \cdot 90.725^{-1} \bmod 15.485.862 = 11.047.464$$

donde el inverso de h se ha encontrado con el algoritmo de Euclides extendido.

La firma de A para el mensaje m es:

$$(r, s) = (7.635.256, 11.047.464).$$

Para comprobar la firma de A, cualquier usuario B puede calcular:

$$(\alpha^h)^s = r^s = 7.635.256^{11.047.464} \bmod 15.485.863 = 8.799.713.$$

Después B calcula:

$$(\alpha^a)^r = 12.506.884^{7.635.256} \bmod 15.485.863 = 1.260.686,$$

y también:

$$\alpha^m = 7^{128.688} \bmod 15.485.863 = 5.362.356.$$

Finalmente, B verifica que:

$$r^s \cdot (\alpha^a)^r = 8.799.713 \cdot 1.260.686 \bmod 15.485.863 = 5.362.356 = \alpha^m.$$

2.2.1. Velocidad de ElGamal

Firmar un mensaje con ElGamal requiere hacer una exponenciación, igual que con RSA*.

*** El tiempo de resolución de la congruencia es despreciable.**

La ventaja de la primera firma sobre la segunda es que el exponente h no depende del mensaje que se ha de firmar, de modo que la exponenciación se puede precalcular. De hecho, el signatario puede guardar en un lugar seguro unas cuantas exponenciaciones precalculadas, que utilizará a medida que tenga que firmar mensajes.

En cambio, verificar una firma de ElGamal es más lento que verificar una firma RSA porque implica llevar a cabo tres exponenciaciones, frente a una sola exponenciación para el RSA. Eso es un inconveniente grave, ya que se firma una sola vez, mientras que normalmente una firma se verifica muchas veces.

Finalmente, las firmas de ElGamal tienen el inconveniente de ser más largas que las firmas RSA, porque consisten en pares de enteros grandes (r, s), mientras que la firma RSA es un sólo entero grande s.

2.2.2. Seguridad de ElGamal

Para falsificar la firma de sobre el mensaje m, un criptoanalista enemigo tendría que resolver la ecuación $\alpha^m = (\alpha^\alpha)^r r^s$ con las incógnitas r y s: para hacerlo puede plantear su ataque de dos maneras diferentes:

a) Si fija r y trata de encontrar s, el criptoanalista ha de resolver el problema del logaritmo discreto (PLG).

b) Si fija *s* y trata de hallar *r*, el criptoanalista se enfrenta a una congruencia exponencial mixta, para la cual no hay algoritmo polinómico conocido, es decir, se encuentra con el problema de la firma de ElGamal (PSE).

El problema del logaritmo discreto se trata en el subapartado 1.4.2 del módulo "Criptosistemas de clave pública" de esta asignatura.

Por lo tanto, falsificar una firma de ElGamal no es equivalente a resolver el problema del logaritmo discreto. Ahora bien, si se supiera resolver el PLG, se podría falsificar la firma de ElGamal.

2.3. El estándar DSS

En 1991, el National Institute of Standards and Technology (NIST) de Estados Unidos propuso un **estándar de firma digital** y solicitó comentarios públicos para la adopción del estándar propuesto. El objetivo era que las oficinas gubernamentales norteamericanas tuvieran una manera estándar de firmar las comunicaciones en caso necesario.

El algoritmo propuesto es una variante de la firma de ElGamal y corre el rumor de que una de las razones de no haber adoptado exactamente la firma de ElGamal es una cuestión de patentes.

Protocolo de generación de claves DSS

Para generar una clave DSS hace falta que cada usuario escoja los elementos siguientes:

- *p*, número primo que cumpla que $2^{511} < p < 2^{512}$.
- *q*, número primo divisor de $p - 1$, que cumpla que $2^{159} < q < 2^{160}$.

- g, generador del único subgrupo cíclico de $\mathbb{Z}_p^*$ de orden q.
- x, clave privada del usuario, con $0 < x < q$.
- $y = g^x \bmod p$, clave pública del usuario.

Para escoger p, q y g se procede de la manera siguiente:

a) Se genera un número aleatorio e impar q tal que $2^{159} < q < 2^{160}$ y se comprueba su primalidad. El proceso se repite hasta obtener un q primo.

b) A continuación se genera el primo p repitiendo la elección aleatoria de un entero n tal que verifique:

$$\frac{2^{511}-1}{2q} < n < \frac{2^{512}-1}{2q},$$

hasta que $p = 2nq + 1$ sea primo.

c) Finalmente, se genera un elemento g de orden q del grupo $\mathbb{Z}_p^*$ repitiendo la elección aleatoria de un entero h con $1 < h < p - 1$ y calculando $g = h^{(p-1)/q}$ hasta que $g \neq 1$.

Protocolo de firma DSS

Si m es el mensaje que se tiene que firmar y H: $\mathbb{N} \to \mathbb{Z}$ es una función *hash* unidireccional, el signatario tendrá que efectuar los pasos siguientes:

Las funciones *hash* se tratan en el apartado 3 de este módulo.

1) Escoger un entero aleatorio k específico para m, con $0 < k < q$.

2) Calcular $r = (g^k \bmod p) \bmod q$.

3) Encontrar s resolviendo la congruencia $H(m) \equiv -xr + ks \bmod q$.

4) Indicar que la firma de m es el par (r, s).

Protocolo de verificación DSS

Si (r, s) es la firma DSS que se quiere verificar, el receptor tendrá que dar los pasos siguientes:

1) Calcular $w = s^{-1} \bmod q$.

2) Calcular $u_1 = H(m)w \bmod q$ y $u_2 = rw \bmod q$.

3) Verificar si se cumple la relación:

$$r = (g^{u_1} y^{u_2} \bmod p) \bmod q.$$

2.3.1. Velocidad del DSS

Dado que el DSS trabaja con un subgrupo de $\mathbb{Z}_p^*$ de orden q, las firmas digitales son más cortas que la firma de ElGamal (son dos enteros de magnitud parecida a q, en vez de dos enteros de magnitud semejante a p). Este rasgo, aparte de representar una reducción de la capacidad de almacenaje, supone una verificación más rápida porque las exponenciaciones que se calculan para verificar una firma tienen exponentes del orden de q (160 bits).

A pesar de estas ventajas, la firma DSS requiere más cálculo que la firma RSA, particularmente con respecto a la verificación. También se señala como inconveniente la necesidad de generar un entero aleatorio k de 160 bits para firmar cada mensaje.

Además, a diferencia de RSA y de ElGamal, la firma DSS no tiene la doble utilidad de ser al mismo tiempo un criptosistema de clave pública para cifrar mensajes.

2.3.2. Seguridad del DSS

La seguridad del DSS es similar a la de la firma de ElGamal, con la única diferencia de que los logaritmos discretos se hallan sobre el subgrupo cíclico de orden q de $\mathbb{Z}_p^*$ generado por g.

Hasta ahora, el mejor algoritmo para encontrar logaritmos discretos en el subgrupo cíclico generado por g requiere calcular logaritmos en $\mathbb{Z}_p^*$, con lo que la seguridad del DSS es, de momento, la misma que la de ElGamal. Sin embargo, no se puede excluir la posibilidad de que en el futuro se encuentren algoritmos para calcular logaritmos en un subgrupo cíclico de un grupo finito dado. Una de las razones de no disponer de tales algoritmos es probablemente que este problema no presentaba demasiado interés antes de la introducción del DSS.

También se ha criticado el hecho de que la longitud de p y de q quede tan fijada en el estándar. Han surgido voces que señalan que las longitudes son insuficientes para proporcionar una seguridad adecuada. Futuras versiones del estándar permitirán posiblemente más flexibilidad en la elección de p y q.

3. Funciones *hash*

Las firmas digitales en uso actualmente son lentas en relación con criptosistemas de clave compartida como el DES. Es deseable, pues, firmar sólo un resumen del mensaje en vez del mensaje entero. Las funciones *hash* sirven para crear resúmenes.

Una **función *hash*** hace corresponder a un mensaje m de longitud variable una representación $H(m)$ de longitud fija. Típicamente, $H(m)$ tiene de 64 a 160 bits y se denomina el *valor hash del mensaje*.

La proyección...

... de los enteros $\mathbb{Z}$ en $\mathbb{Z}_p$ por p fijado es una función *hash*. En efecto, a un entero de longitud arbitraria se le hace corresponder un entero de longitud como máximo $|p|$ bits, donde $|p|$ es la longitud de p en bits.

Si una función *hash* tiene que ser utilizada para aplicaciones criptográficas, no basta con que resuma su entrada de manera aparentemente aleatoria. Hace falta que sea también unidireccional.

Una **función *hash* unidireccional** es una función *hash* H para la que, dado cualquier mensaje m' del recorrido de H, es difícil de encontrar m tal que $m' = H(m)$.

Una función *hash* unidireccional es, pues, una función *hash* que es también una función unidireccional.

La combinación de funciones *hash* unidireccionales con firmas digitales da pie al protocolo de firma con *hash* y al correspondiente protocolo de verificación.

Protocolo de firma con *hash*

Para firmar un mensaje m con una función *hash*, A ejecuta dos acciones:

1) Calcula $H(m)$, donde H es una función *hash* unidireccional públicamente conocida.

2) Firma $H(m)$ con su clave privada y obtiene la firma s. Se considera s la firma de m.

Protocolo de verificación con *hash*

Cualquier usuario B puede verificar la firma s del mensaje m del modo siguiente:

1) Primero calcula $H(m)$.

2) A continuación, verifica si s es una firma válida para $H(m)$.

Observad que es fundamental que H sea unidireccional, ya que se considera equivalente firmar $H(m)$ y firmar m. Mal andaríamos si, dado el valor *hash* $H(m_1)$ de un mensaje m_1 firmado por A, fuera fácil encontrar otro mensaje $m_2 \neq m_1$ tal que $H(m_1) = H(m_2)$. En este caso, cualquiera podría pretender que A ha firmado m_2.

Lectura complementaria

El estudiante interesado puede encontrar descripciones de las funciones *hash* de la familia MD y también de la función SHA-1 en la obra siguiente:

B. Schneier (1996). *Applied cryptography: protocols, algorithms and source code in C* (2.ª ed.). Nueva York: John Wiley & Sons.

Las funciones *hash* más utilizadas son los *message digest* de Rivest, conocidos como MD2, MD4 y MD5. Estas funciones generan resúmenes de 128 bits y el único ataque que se conoce contra éstas es el de la busca exhaustiva. Recientemente, el NIST ha propuesto una función *hash* estándar, conocida como *Secure Hash Algorithm* (SHA-1).

Los algoritmos que implementan estas funciones tienen una apariencia similar a los de los criptosistemas de tipo DES*, con la diferencia de que no dependen de ninguna clave. Su velocidad en *software* es superior a la de los criptosistemas de tipo DES.

*** Iteraciones, operaciones en el nivel de bit fáciles de implementar en *hardware*, etc.**

Resumen

Con un **esquema de firma digital** cada usuario puede firmar mensajes en formato electrónico de modo que las firmas puedan ser verificadas por cualquier persona en cualquier momento posterior.

Los dos algoritmos de firma digital más utilizados hoy en día se obtienen de los dos criptosistemas de clave pública más empleados: RSA y ElGamal. La infalsificabilidad de la firma RSA está relacionada con el problema de la factorización, mientras que la infalsificabilidad de la firma de ElGamal se vincula al problema del logaritmo discreto.

Hay un **estándar de firma digital**, el DSS, propuesto por el gobierno de Estados Unidos, que en realidad es una variante de la firma de ElGamal.

Como los criptosistemas de clave pública, los algoritmos de firma digital son lentos comparados con los criptosistemas de clave compartida. Por esta razón, se tiende a firmar resúmenes de los mensajes en vez de los mensajes enteros. Las **funciones *hash*** permiten obtener un resumen de longitud fija a partir de un mensaje de longitud arbitraria, de modo que para un criptoanalista enemigo sea difícil encontrar un mensaje diferente con el mismo resumen.

Actividades

1. Buscad en Internet implementaciones de las funciones *hash* de la familia MD y también de la función estándar SHS.

2. Escribid programas que implementen las firmas RSA y ElGamal con números pequeños. Podéis aprovechar implementaciones de los criptosistemas RSA y ElGamal.

Ejercicios de autoevaluación

1. Demostrad que la firma RSA es existencialmente falsificable, es decir, que dadas firmas legítimas para dos mensajes m_1 y m_2, cualquier criptoanalista puede calcular la firma para el mensaje producto $m_1 \cdot m_2$.

2. En la firma de ElGamal, tomad $p = 23$, $\alpha = 5$ (α es primitivo en $\mathbb{Z}_{23}^*$). Detallad cómo un usuario A generaría su par de claves pública y privada. Detallad paso a paso cómo firmaría el usuario A vuestra edad en años módulo 23.

3. Seguid los pasos indicados para generar los parámetros del DSS suponiendo que $p = 23$:
a) Hallad un factor primo q de $p - 1$ tal que $q > 10$.
b) Comprobad que $x^{q-1} = 1 \bmod q$ para todo $1 \leq x \leq q - 1$.
c) Encontrad un número $h < p - 1$ tal que $g = h^{(p-1)/q} \bmod p > 1$.
d) Detallad los elementos del subgrupo multiplicativo generado por g.

4. Suponed que A usa un esquema de firma de ElGamal y que firma dos mensajes, m_1 y m_2 con firmas (r, s_1) y (r, s_2), utilizando el mismo r para las dos firmas. Suponed también que $\text{mcd}(s_1 - s_2, p - 1) = 1$. Demostrad cómo se puede calcular eficientemente la clave secreta a del signatario y como consecuencia se puede romper el esquema de firma.

Solucionario

Ejercicios de autoevaluación

Si habéis hecho las actividades del módulo "Criptosistemas de clave pública" podéis aprovechar las implementaciones que hayáis realizado para llevar a cabo la actividad 2.

1. Si el signatario tiene clave pública (e, n) y clave privada d, la firma RSA de m_1 es $s(m_1) = m_1^d \bmod n$ y la firma RSA de m_2 es $s(m_2) = m_2^d \bmod n$. La firma del mensaje producto puede ser calculada a partir de las firmas anteriores de la manera siguiente:

$$s(m_1m_2) = (m_1m_2)^d \bmod n = ((m_1)^d \bmod n)((m_2)^d \bmod n) \bmod n = \\ = (s(m_1)s(m_2)) \bmod n$$

2. Tenemos $p = 23$ y $\alpha = 5$. Supongamos que el usuario A elije como clave privada $a = 3$. Entonces calcula su clave pública como $\alpha^a = 23 \bmod 23 = 10$. Supongamos que nuestra edad en años módulo 23 es $m = 11$. Entonces, para firmar m, A elige aleatoriamente un h tal que $\text{mcd}(h, \phi(23)) = \text{mcd}(h, 22) = 1$. Por ejemplo, $h = 5$. Entonces, A calcula $r = \alpha^h = 5^5 \bmod 23 = 20$. Finalmente, A encuentra el valor de s como:

$$s = (m - ar)h^{-1} \bmod 22 = (11 - 3 \cdot 20)\ 9 \bmod 22 = 21.$$

La firma del mensaje $m = 11$ es $(r, s) = (20, 21)$.

Para verificar la firma, se calcula:
- $r^s = 20^{21} \bmod 23 = 15$,
- $(\alpha^a)^r = 10^{20} \bmod 23 = 3$,
- $\alpha^m = 5^{11} \bmod 23 = 22$

y se comprueba que:

$$r^s(\alpha^a)^r = (15 \cdot 3) \bmod 23 = 22 = \alpha^m.$$

3.
a) Podemos tomar $q = 11$.
b) No es necesario comprobar que $x^{q-1} = 1 \bmod q$ para todo $1 \leq x \leq q - 1$ porque eso se sigue del teorema pequeño de Fermat.
c) Tomamos $h = 3 < 22$ y comprobamos que:

$$g = 3^{22/11} \bmod 23 = 9 \neq 1$$

d) Los elementos del subgrupo multiplicativo generado por g son:

$$\{9, 12, 16, 6, 8, 3, 4, 13, 2, 18, 1\}.$$

El subgrupo en cuestión tiene orden 11, que es divisor del orden del grupo multiplicativo entero, que es 22.

4. Si r es igual a las dos firmas, entonces $h = \log_\alpha r$ también lo es. Luego, se cumplen las dos relaciones siguientes:
- $m_1 \equiv ar + hs_1 \bmod (p - 1)$.
- $m_2 \equiv ar + hs_2 \bmod (p - 1)$.

Restamos las dos ecuaciones anteriores y obtenemos:

$$m_1 - m_2 \equiv h(s_1 - s_2) \bmod (p - 1).$$

Como se supone que $\text{mcd}(s_1 - s_2, p - 1)$, podemos encontrar su inverso módulo $p - 1$ y así obtener:

$$h = (m_1 - m_2)(s_1 - s_2)^{-1} \bmod (p - 1).$$

Una vez hallado h, podemos aislar a de cualquiera de las dos ecuaciones.

Hemos encontrado la clave privada del signatario y, por lo tanto, hemos roto el esquema. Por eso es muy importante que el valor h (y por lo tanto r) sea aleatorio y diferente en cada firma de ElGamal.

Glosario

falsificación *f* Ataque criptoanalítico que pretende obtener la firma de un cierto mensaje sin la intervención del signatario.

firma de ElGamal *f* Firma digital basada en el criptosistema de clave pública de ElGamal.

firma digital *f* Procedimiento para firmar documentos en formato electrónico que consiste en un algoritmo de firma privado del signatario y un algoritmo público para la verificación de la firma.

firma DSS *f* Firma digital estándar del gobierno de Estados Unidos, muy parecido a la firma de ElGamal.

firma RSA *f* Firma digital basada en el criptosistema de clave pública RSA.

función *hash* *f* Función que da como salida un resumen de longitud fija a partir de una entrada consistente en un mensaje arbitrariamente largo.

función *hash* unidireccional *f* Función *hash* que además es unidireccional; es decir, la salida es fácil de calcular a partir de la entrada, pero la entrada es difícil de calcular a partir de la salida.

RSA Criptosistema de clave pública, basado en el problema de la factorización, publicado por Rivest, Shamir y Adleman en 1978.

verificación *f* Comprobación de que una firma es válida, es decir, de que ha sido efectuada por el pretendido signatario. Ha de ser posible para todo el mundo, es decir, no tiene que requerir conocimiento de parámetros secretos.

Bibliografía

ElGamal, T. (1985). "A public-key cryptosystem and a signature scheme based on discrete logarithms". *IEEE transactions on information theory* (vol. 31, pág. 469-472).

Fuster, A.; de la Guía, D.; Hernández, L.; Montoya, F.; Muñoz, J. (1997). *Técnicas criptográficas de protección de datos*. Madrid: Ra-ma.

Goldwasser, S.; Bellare, M. (1996). *Lecture on cryptography* (Manuscrito).

National Institute of Standards and Thecnology (1994). "Digital signature standard (DSS)". *Federal Information Processing Standards* (núm. 186). Washington: NIST.

Rivest, R. L.; Shamir, A.; Adleman, L. (1978). "A method for obtaining digital signatures and public-key cryptosystems". *Communications of the ACM* (núm. 2, vol. 21, pág. 120-126).

Schneier, B. (1996). *Applied cryptography: protocols, algorithms and source code in C* (2.ª ed.). Nueva York: John Wiley & Sons.