E.M. nº 23 Mayo-Agosto 2006

LAS MATEMÁTICAS EN LA CRIPTOLOGÍA

Paz Morillo
Matemática Aplicada IV. Universidad Politécnica de Catalunya

1. INTRODUCCIÓN

Aunque se puede afirmar que la Criptología es casi tan antigua como la escritura, el nacimiento de lo que se conoce como Criptología *científica* se puede situar en la segunda guerra mundial, y es la matematización de la Criptología el hecho que marca el inicio de esta disciplina como ciencia. Así, uno de los momentos clave que se puede destacar en este proceso, es la aparición de la Teoría Matemática de la Información de **Shannon** y su trabajo sobre el *secreto perfecto* en los sistemas criptográficos [Sh49].

La máquina Enigma, utilizada para cifrar por los alemanes en la Segunda Guerra Mundial

Clásicamente, la Criptología se ha articulado en dos vertientes: la Criptografía, que se encarga del diseño de sistemas de escritura secreta, y el Criptoanálisis, que estudia las debilidades de estos sistemas con el fin de conseguir el secreto. La Criptología se ocupa no solamente de la escritura secreta sino también de otros aspectos directamente relacionados con las nuevas tecnologías y las comunicaciones. Así, la Criptografía se encarga de problemas relacionados con la seguridad de las comunicaciones y el Criptoanálisis de la rotura de dicha seguridad.

De hecho, el gran avance de la Criptología en las últimas décadas se debe principalmente a la existencia de ordenadores cada vez con mayor potencia de cálculo y a la generalización de las comunicaciones y la utilización de los recursos telemáticos para transacciones, gestiones o negocios, que han potenciado el uso de la Criptografía por parte de un número de usuarios cada vez mayor. En efecto, últimamente el volumen de información secreta que se debe almacenar o transmitir ha aumentado considerablemente: transacciones bancarias, comercio electrónico, información sanitaria,…

En este entorno, un rasgo en común entre la Criptología y otros campos de la informática y de las comunicaciones es que áreas de las matemáticas que eran consideradas puras o poco aplicadas, han resultado ser una pieza clave en avances tecnológicos muy importantes.

El objetivo de este artículo es presentar algunos de los problemas que estudia la Criptología, desde los más divulgados hasta otros prácticamente desconocidos pero de aplicaciones importantísimas. También se pretende mostrar cómo diferentes ramas de las matemáticas intervienen en el estudio y resolución de esos problemas.

Con este fin, el artículo comienza por una visión general de a qué se dedica, qué problemas ataca la Criptología y cómo los resuelve utilizando matemáticas, para pasar después a estudiar con detalle un problema menos conocido para aquél que no se dedique a la investigación en estos temas, y mostrar finalmente su solución.

Este trabajo se divide en tres secciones. En la Sección 2 se presentan las principales ideas de la *Criptografía de clave pública*, cuya introducción en el año 1976 supuso una revolución en el diseño de dos sistemas criptográficos: esquemas de cifrado de mensajes y esquemas de firma digital.

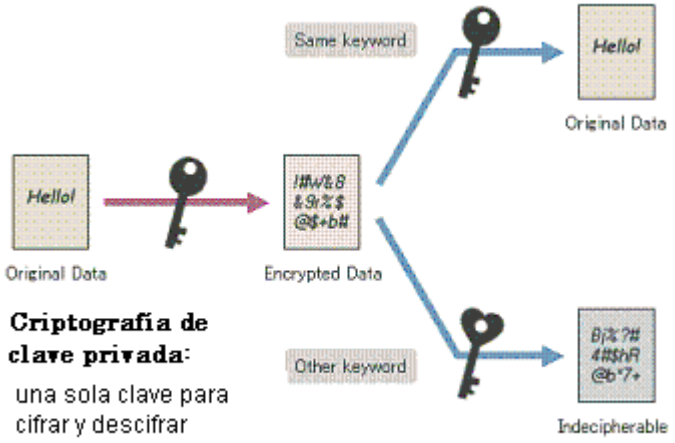
La Sección 3 trata sobre *esquemas para compartir secretos*, donde un secreto se reparte entre un grupo de usuarios de manera que sólo los conjuntos de usuarios autorizados pueden recuperar su valor. Una aplicación de estos esquemas la podemos encontrar, por ejemplo, en la necesidad de repartir el valor de una clave secreta, con cuyo conocimiento pueden tomarse decisiones o realizar acciones de especial trascendencia.

En estas dos primeras secciones se presentan dos herramientas criptográficas de naturaleza muy distinta. En efecto, mientras que en los criptosistemas de clave pública la seguridad está basada en la dificultad computacional de problemas matemáticos como, por ejemplo, la descomposición en factores primos de un número entero de gran tamaño, la seguridad de los esquemas para compartir secretos es *incondicional*, es decir, no depende de los recursos computacionales del usuario. Otra diferencia importante son las herramientas matemáticas utilizadas: mientras que en los sistemas de clave pública la *Teoría de Números*, los *Cuerpos Finitos* y la *Teoría de la complejidad* juegan un papel fundamental, en la compartición de secretos intervienen principalmente la *Combinatoria* y el *Álgebra Lineal.*

Finalmente, en la Sección 4 se presenta brevemente la *computación multiparte* segura. El problema que se trata es el de n participantes (usuarios, ordenadores, procesadores,…) que quieren calcular el valor de una función de sus entradas de forma segura. Aquí seguridad significa que los datos de cada participante permanecerán secretos en todo momento, y el resultado de la función debe de ser correcto y obtenido por todos los participantes, incluso en presencia de algunos participantes con comportamiento incorrecto. La computación multiparte tiene aplicaciones tan importantes como la votación electrónica, estudios estadísticos de datos confidenciales, gestión de claves de chats en internet…

2. CRIPTOGRAFÍA DE CLAVE PÚBLICA

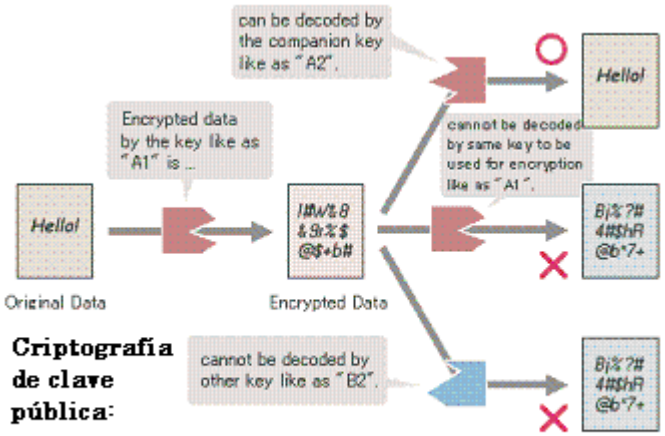
La Criptografía de clave pública, introducida por *Diffie y Hellman* [DH76], ha sido el punto de inflexión más importante en la Criptología.



Anteriormente sólo se utilizaba la Criptografía de clave privada, en la que se permite la comunicación secreta entre dos usuarios que comparten una clave de trabajo. La misma clave se usa

para cifrar y para descifrar los mensajes, y el emisor y el receptor han de acordar la clave secreta a través de un canal de comunicación seguro. Ésta es la principal dificultad de los sistemas de clave privada. Por ejemplo, en una red de comunicación con *n* usuarios, el número de claves secretas es *n(n-1)/2*, y así, en una red de 1000 usuarios se requerirían 499.500 conexiones totalmente seguras para poder distribuir las claves.

Sin embargo, en la Criptografía de clave pública, cada usuario posee una clave pública y una privada. Cuando un usuario quiere enviar un mensaje cifrado, lo hace utilizando la clave pública del receptor y éste es el único que puede descifrar dicho mensaje, usando para ello su clave secreta. Se puede imaginar que cada usuario tiene un buzón donde cualquiera puede depositar un mensaje, y él es el único que tiene la llave para abrirlo.



una clave pública para el cifrado y una secreta para el descifrado

Los criptosistemas de clave pública se basan en las llamadas *funciones unidireccionales con trampa*, funciones fáciles de calcular pero cuya inversa es difícil de obtener (se requiere de un tiempo excesivo, utilizando el mejor algoritmo conocido y con los mejores recursos computacionales), salvo que se conozca una cierta información adicional. Así, la seguridad se llama *condicional*: se basa en la capacidad computacional de los participantes. De hecho, la seguridad incondicional es imposible en este contexto, ya que un participante con recursos computacionales ilimitados siempre podría, a la vista del mensaje cifrado, y dado que el algoritmo de cifrado es público, probar todos los posibles valores del texto hasta hallar aquel cuyo cifrado coincidiera con el observado.

La seguridad de los criptosistemas de clave pública se basa pues, en la dificultad de resolver algún problema. Por ejemplo, la seguridad del criptosistema RSA [RSA78] se basa en la dificultad de factorizar un número entero producto de dos primos de gran tamaño. Otros criptosistemas se basan en la dificultad del cálculo del logaritmo discreto en ciertos grupos finitos [E85], como el grupo multiplicativo de un cuerpo finito o el grupo de puntos de una curva elíptica. Más adelante se verá el funcionamiento de dos criptosistemas de clave pública.

En la práctica, la velocidad de cifrado de los sistemas de clave pública es mucho menor que los de clave privada. Por tanto, para cifrar grandes volúmenes de información se usan sistemas de clave privada (por ejemplo al emitir TV en canales de pago). El criptosistema de clave pública se usa para transmitir, de forma secreta, la clave privada de trabajo a través de canales inseguros, resolviendo así el problema comentado de establecer claves secretas en ausencia de canales seguros de comunicación.

2.1 El criptosistema RSA

En el año 1978 *Rivest, Shamir y Adleman* pusieron en práctica por primera vez, la idea de Criptografía de clave pública introducida por Diffie y Hellman en 1976.

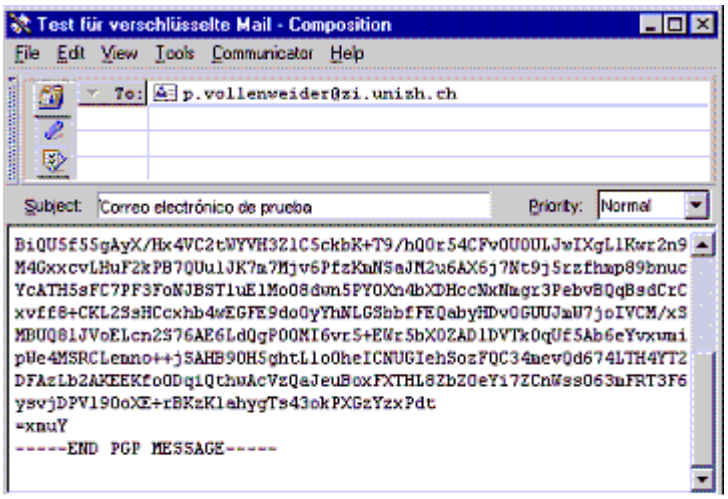
Sea Z_n el conjunto de restos que se obtienen al dividir números enteros entre *n*. Por ejemplo, $Z_{17}=\{0,1,2,...,15,16\}$. Si dos enteros *a* y *b* dan el mismo resto al dividir entre *n*, escribimos $a=b \pmod n$. Así, $19=2 \pmod{17}$, y también $-21=13 \pmod{17}$ porque $-21=(-2)*17+13$.

En el sistema de cifrado RSA cada participante elige un número entero producto de dos primos grandes, $n=pq$, y elige dos enteros e,d tal que $ed=1 \pmod{\phi(n)}$, siendo ϕ la función de Euler, y por tanto $\phi(n)=(p-1)(q-1)$. La clave pública del participante será (n,e) y la clave privada (p,q,d).

El método de cifrado y descifrado se basa en el Teorema de Euler, que afirma que si *x* es un entero primo con *n*, entonces $x^{\phi(n)}=1 \pmod n$. Cuando A quiere enviar un mensaje cifrado a B, mira su clave pública (n,e) y codifica el mensaje de forma que sea un elemento $x \in Z_n$. El mensaje cifrado que A envía a B es $c=x^e \pmod n$. Para descifrar el mensaje, B calcula $c^d \pmod n$. Por el Teorema de Euler vemos que $x=x^{ed} \pmod n$.

Observemos que para descifrar es necesaria la clave privada *d*, pero si un adversario fuera capaz de factorizar *n* tendría $\phi(n)$ y podría hallar *d* y descifrar todos los mensajes dirigidos a B.

Se ha demostrado que encontrar la clave privada *d* de un usuario es tan difícil como factorizar *n*. Más concretamente, cualquier algoritmo que permita calcular *d* a partir de *n* y *e*, se puede utilizar como subrutina en un algoritmo probabilístico eficiente para factorizar *n*.



PGP es una de las aplicaciones más utilizadas para cifrar mensajes de correo electrónico y utiliza el esquema RSA.

Sin embargo, podría ser que un mensaje se pudiera descifrar sin tener que calcular el valor *d*. Se trataría de buscar alguna estrategia que permitiera deducir el valor de *x*, a partir del conocimiento de x^e módulo *n*. No se ha demostrado que romper el RSA sea equivalente a factorizar, y algunos trabajos indican justamente lo contrario [BV98] y [Co98]. Para evidenciar la dificultad de la cuestión, vale la pena comentar que existen criptosistemas del tipo RSA en los que sí se demuestra que hay equivalencia entre su rotura y la factorización del entero *n* [Sc98].

En la actualidad, el tamaño de los primos *p* y *q* que se considera seguro es de 160 cifras decimales, aunque se empieza a considerar que tal vez esta medida ya no es suficientemente segura.

2.2 Criptosistemas basados en el logaritmo discreto

Otro problema difícil utilizado en el diseño de criptosistemas de clave pública es el logaritmo discreto. Esto es, dados α y β en un grupo finito, hallar un entero *a* tal que $\alpha^a = \beta$. Se dice que *a* es el logaritmo de β en base α, es decir $a=\log_\alpha(\beta)$.

El criptosistema propuesto por *ElGamal* en 1985 [E85], se basa en la dificultad de resolver el problema del logaritmo discreto en el grupo multiplicativo Z_p^* (los elementos de Z_p distintos del 0), siendo p un primo adecuado. Concretamente, teniendo en cuenta los métodos y la capacidad de cálculo actual, p ha de ser un primo de más de 150 dígitos decimales y tal que $p-1$ tenga un factor primo grande.

En el criptosistema de ElGamal, cada usuario elige un primo p con las características mencionadas y un elemento α cuyas potencias generen el grupo multiplicativo Z_p^*. Tal α existe y es fácil de encontrar. Además, elige un entero a tal que $a \in Z_{p-1}$ y calcula $\beta=\alpha^a$. La clave pública del usuario es (p,α, β) y su clave secreta es a. Para cifrar un mensaje m se elige al azar un elemento $r \in Z_{p-1}$ (que se mantiene secreto) y se calcula $c=(\alpha^r, m\beta^r)$. El receptor legítimo puede descifrar el mensaje usando su clave privada a. En efecto, dado el mensaje cifrado $c=(c_1,c_2)$, se tiene que $m=c_2/(c_1)^a$. Todos los cálculos se realizan en el grupo Z_p^*.

A diferencia del criptosistema RSA, el criptosistema ElGamal no es determinista. De hecho, el mensaje cifrado depende del valor del número aleatorio r, y así hay muchos cifrados diferentes correspondientes a un mismo mensaje en claro.

Observemos que se puede construir un criptosistema como ElGamal en cualquier grupo finito en el que el problema del logaritmo discreto sea intratable. Por ejemplo, *Koblitz* [K87] y *Miller* [M85] propusieron el uso del grupo de puntos de una curva elíptica.

2.3 Firmas electrónicas

Una de las ventajas de la Criptografía de clave pública es que permite implementar otras aplicaciones aparte del cifrado de mensajes, y las firmas electrónicas o los sistemas de identificación son dos de ellos. Veamos, por ejemplo, cómo utilizar el criptosistema RSA para la firma de mensajes. Supongamos que el usuario A con clave pública (n,e) y privada (p,q,d) quiere enviar a B un mensaje m firmado. Para ello, el usuario A calcula $\sigma = m^d \pmod n$ y envía (m, σ) (recordemos que sólo A puede calcular σ, dado que para ello se necesita su clave secreta). Luego B puede verificar que A ha firmado el mensaje comprobando que $\sigma^e = m \pmod n$.

Una firma debe estar ligada al mensaje que se firma. En un documento escrito la firma está en la misma hoja que el mensaje. En la firma electrónica, la firma debe depender del mensaje y, además, cualquiera ha de poder verificarla. Observemos que en RSA la firma se verifica utilizando la clave pública de A.

En el criptosistema de ElGamal, dado que no es determinista, la firma no es posible de la manera antes descrita. Sin embargo, en el mismo trabajo de 1985, ElGamal presenta un esquema de firma electrónica basado en la dificultad del cálculo del logaritmo discreto. Una modificación de este esquema, el Digital Signatura Standard (DSS) se adoptó como estándar el año 1994 por el National Institute of Standards and Technology de Estados Unidos [DSS94].

3. ESQUEMAS PARA COMPARTIR SECRETOS

Supongamos que en un colectivo se necesita el acuerdo de un grupo determinado de personas para emprender alguna acción o tomar alguna decisión. Esta situación se resuelve con un esquema para compartir secretos.

En un *esquema para compartir secretos* se reparte el valor de un secreto en fragmentos entre los participantes de un conjunto P, de forma que sólo los subconjuntos autorizados pueden reconstruir el secreto a partir de sus fragmentos.

Los esquemas para compartir secretos se introdujeron de forma independiente por *Blackley* [B79] y *Shamir* [S79] en 1979.

La familia de subconjuntos autorizados Γ se denomina *estructura de acceso* y ha de ser *monótona*, es decir, si un subconjunto contiene un subconjunto autorizado, también debe ser autorizado.

En un esquema para compartir secretos, con estructura de acceso Γ y conjunto de posibles secretos *K*, a partir de un valor secreto $k \in K$ y de una cierta elección aleatoria, cada participante P_i recibe un fragmento $s_i \in S$ donde S denota el conjunto de posibles fragmentos, de manera que:

1. Los subconjuntos de P autorizados pueden reconstruir el valor del secreto *k* a partir de sus fragmentos. Es decir, si $A \in \Gamma$ existe un único valor posible *k* para los fragmentos s_i dados.

2. Los participantes de un subconjunto no autorizado no pueden obtener ninguna información sobre el secreto a partir del valor de sus fragmentos. Es decir, si $A \notin \Gamma$ todos los valores posibles del secreto son igualmente probables, incluso conociendo los fragmentos s_i de todos los $P_i \in A$.

3.1 **El esquema de Shamir**

El esquema de Shamir es, como ya se ha mencionado, uno de los primeros propuestos. La estructura de acceso está formada por los subconjuntos con al menos t participantes de un conjunto con n participantes. Los esquemas con estructura de acceso de este tipo se denominan *esquemas de umbral* (t,n).

Adi Shamir es una de las figuras más destacadas de la criptografía

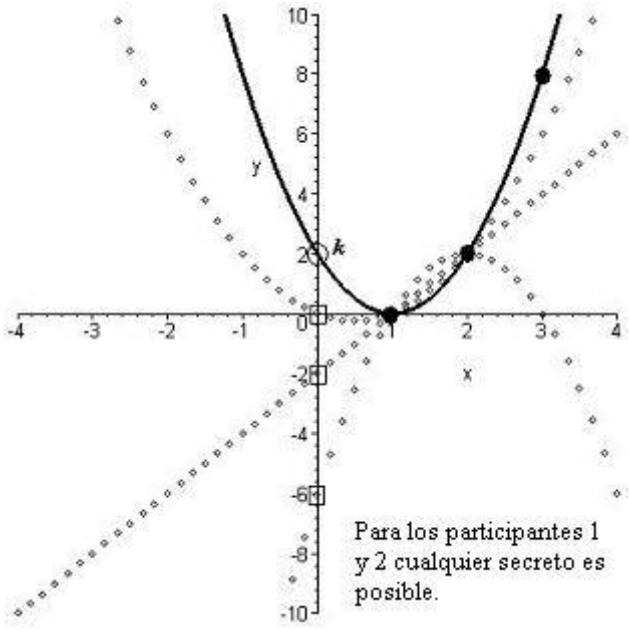
Consideremos el conjunto de participantes $P=\{P_1,P_2,\ldots,P_n\}$ y un entero *t* tal que $1 \leq t \leq n$. El valor secreto que se reparte es un elemento del cuerpo finito Z_p^*, siendo *p* primo.

En la fase inicial, a cada participante se le asigna un elemento x_i del cuerpo Z_p^*, donde los valores x_i son no nulos y diferentes dos a dos. Los valores x_i son públicos. Para distribuir un secreto *k* se toma al azar un polinomio de grado menor o igual que *t-1*, *p(x)*, tal que *p(0)=k*. Y cada participante recibe como fragmento $s_i=p(x_i)$. Debido a que el valor del secreto y los coeficientes del polinomio han de mantenerse secretos, esta fase la lleva a cabo un participante especial D llamado *distribuidor*.

Veamos cómo efectivamente el esquema de Shamir es un esquema para compartir secretos, de umbral *(t,n)*. Cualquier conjunto de *t* participantes puede obtener el polinomio *p* y, por tanto, el secreto *p(0)*. Por ejemplo, esto puede hacerse utilizando la interpolación de Lagrange que da la fórmula para calcular el único polinomio de grado menor o igual que *t-1*, que pasa por *t* puntos conocidos (*polinomio interpolador*). Por otra parte, si sólo se conocen *t-1* fragmentos, para cualquier posible valor del secreto *k*, existe un único polinomio *q(x)* con grado menor o igual que *t-1* tal que *q(0)=k* y

$q(x_i)=s_i$ para todo $i=1,\ldots,t-1$. Por tanto, los fragmentos de los participantes de un subconjunto no autorizado no dan ninguna información sobre el valor del secreto.

Veamos un ejemplo. En Z_{17} diseñamos un esquema de umbral (3,4), los participantes son P_1, P_2, P_3, P_4 y los elementos asignados son $x_1=1$, $x_2=2$, $x_3=3$, $x_4=6$, respectivamente. El valor secreto a repartir es $k=2$ y el polinomio elegido por el distribuidor es $p(x)=2x^2-4x+2$. Entonces, los fragmentos de cada participante serán $s_1=0$, $s_2=2$, $s_3=8$, $s_4=16$, respectivamente. Si ahora suponemos que se unen los participantes P_1, P_2 y P_3 y buscan el único polinomio que pasa por los puntos (1,0), (2,2) y (3,8) obtendrán $p(x)$ y en el término independiente tendrán el secreto. En cambio, si sólo se unen dos participantes, por ejemplo, P_1 y P_2, para cualquier valor del secreto s existe un polinomio que pasa por los puntos (1,0) y (2,2) y tiene por término independiente s. Efectivamente, basta tomar $p(x)=9(2+s)x^2+(7s-1)x+s$.



Observemos que la seguridad del esquema de Shamir es incondicional, ya que no está basada en ninguna hipótesis computacional. Es decir, un subconjunto no autorizado no obtendría ninguna información ni con recursos computacionales infinitos. De hecho, y por definición, esta propiedad debe tenerla cualquier esquema para compartir secretos.

En muchas ocasiones, se necesita el acuerdo de los n participantes para la recuperación del secreto compartido, y en esta situación se necesita un esquema de umbral (n,n).

Un aspecto a tener en cuenta en la compartición de secretos es su compatibilidad con las operaciones básicas, esto es, a partir de los fragmentos de dos secretos a y b, es conveniente poder calcular los fragmentos de su suma $a+b$ y de un múltiplo λa (en este caso se habla de *esquema lineal*), o de su producto ab. Es fácil ver que el esquema de Shamir es un esquema lineal.

3.2 **Generación compartida de secretos**

Otro tema importante en los esquemas para compartir secretos es la *generación compartida de secretos*. En este caso no existe ese participante especial denominado distribuidor y el secreto se genera entre todos los participantes. Este tema es de especial relevancia en entornos en los que no existe una autoridad de confianza. Veamos un ejemplo de repartición de un secreto generado de forma compartida. Cada participante P_i, elige un valor secreto, r_i, y lo reparte mediante un esquema lineal de umbral (t,n) entre el colectivo de participantes (por ejemplo, según el esquema de Shamir), es decir, cada participante hace de distribuidor de su secreto. A continuación, cada participante puede sumar los

fragmentos obtenidos del resto de participantes y el fragmento que le corresponde de su propio secreto. Es fácil ver que la suma de los fragmentos obtenidos es el fragmento de la suma de los secretos. Así el secreto que se tiene compartido, mediante un esquema de umbral *(t,n)*, es la suma de los secretos r_i generados por cada participante.

Veamos un ejemplo.

Supongamos tres participantes P_1, P_2, P_3 y a cada uno le asignamos el valor igual a su índice $x_i=i$. Ahora cada participante elige un valor secreto, por ejemplo $r_1=1$, $r_2=3$, $r_3=-1$. Vamos a suponer que las operaciones se realizan en Z_{17}, como en el ejemplo anterior, y que el esquema es de umbral (3,3). A continuación cada participante elige un polinomio y distribuye su secreto entre los demás participantes, tal como se ha descrito anteriormente. Así por ejemplo, $p_1(x)=x^2+x+1$, $p_2(x)=2x^2-x+3$, $p_3(x)=3x^2-1$ y, en consecuencia, el participante P_1 recibirá los fragmentos 4 y 2 de los otros dos participantes, y obtendrá el fragmento 3 de su propio secreto; P_2 recibirá 7 y 11, y 9 será el fragmento de su propio secreto; y, finalmente, P_3 recibirá 13 y 1 y calculará 9 (fragmento de su secreto). El último paso a realizar por parte de cada participante será la suma de los fragmentos que posee, por lo que los fragmentos resultantes serán $s_1=9$, $s_2=10$, $s_3=6$.

Podemos comprobar que el polinomio de grado menor o igual que 2, que pasa por los puntos (1,9), (2,10), (3,6), tiene por término independiente el valor 3, que es el secreto generado de forma compartida como suma de los valores secretos elegidos por cada participante.

En la Sección siguiente utilizaremos la notación [*a*] para una repartición del valor a, es decir, [*a*] denota la colección de toda la información relacionada con *a* mantenida por todos los participantes. Así, si *M* es una matriz, [*M*] denotará una repartición de cada una de las coordenadas de *M*.

4. COMPUTACIÓN MULTIPARTE

La computación multiparte segura puede definirse como el problema de *n* participantes que quieren calcular una función de sus entradas de manera segura. Aquí la seguridad significa obtención del resultado correcto, así como privacidad de las entradas de los participantes, todo ello incluso en presencia de participantes corruptos. Concretando, tenemos entradas $x_1, x_2, \ldots, x_n$, y cada jugador P_i conoce x_i y se quiere calcular $(y_1,y_2,\ldots,y_n) = f(x_1, x_2, \ldots, x_n)$, de manera que el jugador P_i obtendrá y_i pero nada más que eso. Una completa introducción al tema de la computación multiparte puede hallarse en [CD05].

Veamos un ejemplo sencillo, un conjunto de 50 personas quiere conocer cuál es su sueldo medio sin revelar qué cobra cada una de ellas. En este caso la función a calcular es la suma de las 50 entradas (sueldos de cada uno) dividida por 50. Para realizarlo de forma segura, cada uno de los trabajadores debe repartir a todos los demás un fragmento del secreto que es su sueldo. Esto se puede realizar mediante un esquema de Shamir con 50 participantes y umbral 50, es decir, esquema de umbral (50,50). Una vez hechas todas las comunicaciones, cada persona tiene un fragmento de los 50 sueldos y, en privado, puede calcular la media aritmética de dichos fragmentos. Ahora, uniendo la información obtenida por cada usuario y recuperando el secreto según se ha mostrado anteriormente en el esquema de Shamir, se obtiene el sueldo medio buscado.

De forma compartida se pueden realizar operaciones más complicadas en criptografía, subastas electrónicas, votaciones electrónicas, controles de audiencia (visitas a páginas web),… Si estas operaciones no se realizan de forma compartida, deben dejarse en manos de una autoridad de confianza, lo que puede suponer un riesgo elevado para la seguridad, además de una carga de trabajo excesiva para dicha autoridad.

A continuación vamos a ver un ejemplo en el que el cálculo a realizar de forma compartida pertenece al ámbito del Álgebra Lineal. Supongamos un número *n* de personas que buscan trabajo y un número *m* de trabajos ofertados. Para llevar a cabo la asignación de trabajos de la manera más satisfactoria posible se solicita a los trabajadores que manifiesten sus preferencias y esa información se quiere mantener en secreto. Así, de forma compartida, se va a generar una matriz de preferencias en la que el número de filas corresponderá al número de trabajadores y el número de columnas al de trabajos. En la fila *i* columna *j* se pondrá un 1 si el trabajador *i* ha manifestado interés por el trabajo *j* y un 0 en caso contrario. Cada trabajador reparte su fila, por ejemplo con un esquema de umbral (*n*,*n*). Se puede demostrar que es posible un emparejamiento (trabajadores-trabajos) si y sólo si esta matriz de adyacencia tiene rango máximo.

Así pues el problema que estamos considerando ahora consiste en: dada una matriz generada de forma compartida, saber si tiene rango máximo, de forma segura, es decir, sin revelar la matriz.

4.1 **Rango de una matriz compartida**

Este es un ejemplo de cálculo en el que la exigencia de la seguridad de los datos hace que los algoritmos conocidos, y considerados los mejores, dejen de serlo. Efectivamente, aunque los cálculos del *Método de eliminación de Gauss* para calcular el rango de una matriz pueden realizarse de forma compartida y segura, este algoritmo es poco o nada eficiente en este contexto. En cada paso de la eliminación de Gauss, cada uno de los participantes debe comunicarse con todos los demás para intercambiar la información obtenida. Es decir, el número de rondas del algoritmo depende del número *n* de trabajadores. Se entiende por ronda una etapa en la que cada participante se comunica con los restantes para transmitir los resultados de los cálculos hechos privadamente. El número de rondas es el principal parámetro para medir la complejidad de un algoritmo de computación multiparte segura.

Por tanto, va a ser necesario considerar otro algoritmo para hallar el rango de una matriz compartida.

Presentamos a continuación un método para el cálculo distribuido seguro del rango de una matriz que funciona con un número constante de rondas, esto es, independiente del tamaño de la matriz.

El algoritmo para saber si una matriz compartida, no cuadrada, tiene rango máximo se basa en los algoritmos que se explicarán a continuación.

4.1.1 *Cálculo compartido del determinante de una matriz invertible*

Sea *A* una matriz compartida, de la que se sabe que el determinante es no nulo. Se quiere calcular el determinante de *A* de forma segura, es decir, sin revelar nada más sobre la matriz.

Para ello se genera de forma compartida entre todos los participantes otra matriz aleatoria *B*, también invertible, y su determinante. Un protocolo que permite generar una matriz y su determinante ([*B*],[det(*B*)]), puede hallarse en [CD01].

A continuación por un método de inversión segura, por ejemplo el método de *Bar-Ilan y Beaver* [BB89], se calcula el inverso del determinante de *B*, $[\det(B)^{-1}]$.

En el siguiente paso se calculan los fragmentos del producto de matrices *BA*, a partir de los fragmentos de ambas, [*BA*]=[*B*][*A*]. Se calcula y se hace público el producto *BA*.

Todos los participantes pueden calcular el determinante det(*BA*) y con el conocimiento de los fragmentos del inverso del determinante de *B*, se obtienen ya los fragmentos del determinante de *A* (haciendo uso de la linealidad del esquema considerado).

Observemos que el conocimiento de *BA* y de su determinante, no da ninguna información sobre la matriz *A*, si se supone que *A* es invertible. Si estamos en el caso que *A* pudiera ser no invertible, al hacerse pública la matriz *BA*, se filtraría información sobre *A* (por ejemplo, su rango) que debería ser secreta.

4.1.2 *Cálculo compartido del determinante de una matriz A*

Ahora se supone desconocido si *A* es una matriz invertible o no. El protocolo para calcular de forma compartida y segura su determinante consta de los siguientes pasos.

Primero se generan de forma segura valores $x_0, x_1, x_2, \ldots x_n$ aleatorios (siendo *n* el tamaño de *A*). Notemos que el determinante de $A-zId$ es 0 sólo si *z* es un autovalor de la matriz *A*, y como máximo *A* tiene *n* autovalores, siendo *n* el tamaño de la matriz. Así, con alta probabilidad las matrices $A_i=A-x_iId$ son invertibles y su determinante puede calcularse con el protocolo explicado en el apartado anterior.

Una vez realizado este cálculo se tienen *n+1* valores del polinomio característico de *A*, $q(x)=\det(A-xId)$, que tiene grado *n*, y realizando de forma privada una interpolación se obtiene el valor del polinomio característico en el 0, que coincide con el determinante de *A*, $q(0)=\det(A)$.

4.1.3 *Cálculo compartido del rango máximo de A*

Supongamos ahora que la matriz *A* es no cuadrada, entonces *A* tiene rango máximo si y sólo si el determinante de la matriz de Gramm A^TA o AA^T es no nulo, según sea mayor el número de columnas que de filas o viceversa.

Así pues, para saber si una matriz compartida tiene rango máximo basta aplicar el algoritmo del punto 4.1.2 a la matriz A^TA o AA^T.

Debemos tener en cuenta que la condición necesaria y suficiente de rango máximo de una matriz no cuadrada es válido si estamos trabajando con elementos del cuerpo de los reales y, sin embargo, no tiene porqué ser cierto sobre cuerpos finitos. En este último caso, debe aplicarse el argumento mencionado, no sobre la matriz *A* sino sobre una matriz *A*' obtenida como el producto de *A* por una matriz diagonal de la forma $D=\text{diag}(1,\alpha,\alpha^2,\ldots,\alpha^{n-1})$ para un cierto valor α. Se demuestra que con probabilidad elevada *A'* tiene rango máximo si y sólo si su matriz de Gramm tiene determinante no nulo.

En resumen, aplicando los tres algoritmos explicados en estos apartados, se consigue un algoritmo para saber si una matriz compartida, no necesariamente cuadrada, tiene rango máximo. El número de rondas de este método es constante y, por tanto, es más eficiente que el conocido método de Gauss.

BIBLIOGRAFÍA:

[B79] G.R.Blakley. Safeguarding cryptographic keys. AFIPS Conference Proc.48, pp.313-317, 1979.

[BB89] J.Bar-Ilan, D.Beaver. Non-cryptographic fault-tolerant computing in constant number of rounds of interaction. Proc. ACM PODC'89, pp.201-209, 1989.

[BV98] D. Boneh, R.Venkatessan. Breaking RSA may not be equivalent to factoring. Advances in Cryptology EUROCRYPT'98, pp.59-71, 1998.

[CD01] R.Cramer, I.Damgård. Secure Distributed Linear Algebra in a Constant Number of rounds. Advances in Cryptology CRYPTO '01, pp.119-136, 2001.
[CD05] R.Cramer, I.Damgård. Multiparty Computation, an Introduction. *Contemporary Cryptology*, CRM, Birhäuser, 2005.
[Co98] D.Coppersmith. Finding a small root of a univariate modular equation. Advances in Cryptology EUROCRYPT'96, pp.155-165, 1996.
[DH76] W.Diffie, M.E.Hellman. New directions in cryptography. IEEE Trans. on Information Theory, n.22, pp.644-654, 1976.
[DSS94] Digital Signature Standard. National Bureau of Standards. FIPS Publication 186, 1994.
[E85] T.ElGamal. A public key Cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. on Information Theory, n.31, pp.469-472, 1985.
[K87] N.Koblitz. Elliptic curve cryptosystems. Mathematics of Computation, n.48, pp.203-209, 1987.
[M85] V.Miller. Uses of elliptic curves in cryptography. Advances in Cryptology, CRYPTO'85, pp.417-426, 1985.
[RSA78] R.L.Rivest, A.Shamir, L.Adleman. A method for obtaining digital signatures and public key cryptosystems. Communications of the ACM, n.21, pp.120-126, 1978.
[S79] A.Shamir. How to share a secret. Communications of the ACM, n.22, pp.612-613, 1979.
[Sc98] R.Scheidler. A Public-Key Cryptosystem Using Purely Cubic Fields. Journal of Cryptology, n.11, pp.109-124, 1998.
[Sh49] C.Shanon. Communication Theory of secrecy systems. Bell Systems Technical Journal, n.28, pp.656-715, 1949.

Paz Morillo es Licenciada en Matemáticas por la Universidad de Barcelona y Doctora en Informática por la Universidad Politécnica de Cataluña. Sus primeros trabajos de investigación fueron sobre Teoría de Grafos y su aplicación al diseño de redes de interconexión. En 1992 fundó el grupo de investigación de Matemática Aplicada a la Criptografía (MAK). Ha trabajado en temas como la aplicación de curvas elípticas a criptografía, diseño de criptosistemas demostrablemente seguros, criptografía distribuida,…