Teoría de números en criptografía y su debilidad ante la posible era de las computadoras cuánticas

Marco Antonio Castillo Rubí*, Nancy Santana de la Cruz*, Alicia Mercedes Díaz Lobaton**, Germán Almanza Rodríguez*** y Felipe Castillo Rubí****

Recepción: 9 de diciembre de 2009
Aceptación: 17 de junio de 2011

*** Universidad Politécnica del Valle de Toluca, México, México.**
**** Facultad de Humanidades, Universidad Autónoma del Estado de México, México.**
***** Universidad Autónoma de Ciudad Juárez, México.**
****** Instituto de Estudios Superiores, Grupo ISIMA, México.**
Correo electrónico: mac@math.cinvestav.mx; nancysan_0620@hotmail.com; alixin_2000@hotmail.com; ralmanza@uacj.mx y felifermat@yahoo.com.mx

Resumen. La principal aplicación de la criptografía es proteger información para evitar que sea accesible a observadores no autorizados. Sin embargo, también tiene otras aplicaciones, por ejemplo verificar que un mensaje no haya sido modificado intencionadamente por un tercero, verificar que alguien es quien realmente dice ser, etc. El objetivo del trabajo es mostrar cómo la matemática juega un papel importante en la criptografía moderna y como ésta aprovecha los problemas difíciles (en el sentido computacional) que existen en la teoría de números para desarrollar protocolos criptográficos. Asimismo se menciona lo que pasaría con los protocolos criptográficos basados en la teoría de números si existiera una computadora cuántica.

Palabras clave: criptografía, teoría de números, computación cuántica.

Theory of Numbers in Cryptography and Its Weakness to the Possible Quantum Computer Age

Abstract. The main application of cryptography is to protect information and to prevent it from being accessible to non-authorized observers. However, it also has other applications, for example to verify that a message has not been intentionally modified by a third party, verify that someone is who they really say they are. The aim of this paper is to show how mathematics plays an important role in modern cryptography as it takes advantage of the difficult problems (in the computational sense) that exist in number theory to develop cryptographic protocols. We also mention what would happen to the cryptographic protocols based on the theory of numbers if there were to be a quantum computer.

Key words: cryptography, number theory, quantum computing.

Introducción

La *criptografía* es el estudio de las técnicas matemáticas relacionadas con los aspectos de seguridad de la información tal como: la confidencialidad, la integridad de datos, la autenticidad y el no rechazo. Desglosemos brevemente tales aspectos:

a) La *confidencialidad* es usada para guardar el contenido de información, donde sólo las personas autorizadas pueden saberlo.

b) La *integridad* de datos se refiere a la alteración no autorizada de datos.

c) La *autenticación* es relacionada con la identificación.

d) El *no rechazo* impide a una entidad negar los compromisos o acciones anteriores.

Hay dos tipos de criptografía: criptografía simétrica o de clave privada y criptografía asimétrica o de clave pública. La criptografía de clave pública se desarrolló en los años setenta y utiliza complicados algoritmos matemáticos relacionados con teoría de números, curvas elípticas, grupos infinitos no conmutativos, teoría de gráficas, teoría del caos, etc. De hecho en la siguiente sección definiremos los conceptos básicos de la criptografía simétrica y asimétrica, así como los protocolos de clave pública más utilizados como lo son el RSA, para cifrar y firma digital, y el Diffie-Hellman para intercambio de claves.

Cabe mencionar que aún no sabemos si es posible el desarrollo de las computadoras cuánticas. La diferencia crucial entre una computadora clásica (también llamada computadora digital) y una cuántica, es que las computadoras clásicas basan su funcionamiento en la existencia de bits, mientras que las computadoras cuánticas en *qubits*.

1. Esquemas de cifrado

En esta sección hablaremos sobre diversos esquemas de cifrado y los abordamos en dos clases. Intuitivamente, los esquemas de cifrado requieren de una clave para cifrar y otra para descifrar. Cuando de una clave, cualquiera de ellas, se obtiene fácilmente la otra, se les denomina esquemas de cifrado simétrico. Cuando de una de ellas no se puede obtener fácilmente la otra, se le denomina esquemas de cifrado asimétrico o de clave pública.

1.1. Criptografía simétrica

En este tipo de criptografía normalmente se utilizan dos claves: una para cifrar y otra para descifrar. Normalmente se dice que se emplea sólo una clave ya que conociendo la clave de cifrado, es fácil calcular la clave de descifrado, es decir; un esquema se dice *simétrico* si para cada par de claves: una de cifrado y otra de descifrado, al conocer una (cualquiera), es computacionalmente fácil determinar la otra. Existen dos clases de esquemas simétricos, éstos son:

a) *Cifradores de bloques*. Son aquellos que cifran de bloque en bloque.

b) *Cifradores de flujo*. Son aquellos que cifran bit a bit o byte a byte.

Un ejemplo de un esquema simétrico cifrado por bloques, llamado c*ifrado por sustitución simple*, es el siguiente: sean Σ un alfabeto de q símbolos, $\mathcal{M}$ el conjunto de cadenas de longitud t sobre Σ, $\mathcal{K}$ el grupo de permutaciones de Σ (recordemos que el grupo de permutaciones de Σ es el conjunto de todas las biyecciones que existe de Σ en Σ). Defínase para cada $e \in \mathcal{K}$ una transformación de cifrado como:

$$E_e(m) = (e(m_1) \bullet\bullet\bullet e(mt)) = (c_1 \bullet\bullet\bullet ct) = c$$

donde $m = (m_1 \bullet\bullet\bullet mt) \in \mathcal{M}$. Para descifrar $(c_1 \bullet\bullet\bullet ct)$ se calcula la inversa de e, la cual denotamos por d:

$$Dd(c) = (d(c_1) \bullet\bullet\bullet d(ct)) = (m_1 \bullet\bullet\bullet mt) = m.$$

1.2. Ejemplo

Sea $\Sigma = \{A, B, C, \bullet\bullet\bullet, Z\}$ y sean $\mathcal{M}$ y C conjuntos de cadenas de longitud cinco sobre $\mathcal{A}$. Tomemos el espacio de claves $\mathcal{K}$ como el grupo de permutaciones de Σ, luego si $e \in \mathcal{K}$ es una clave para cifrar, entonces para descifrar se usa la inversa $d = e^{-1}$. En particular si $e \in \mathcal{K}$ es la permutación que recorre tres posiciones a la derecha

$$e = \begin{pmatrix} A\,B\,C \bullet\bullet\bullet X\,Y\,Z \\ D\,E\,F \bullet\bullet\bullet A\,B\,C \end{pmatrix}$$

el mensaje

$$m = CRIPT\ OGRAF\ IAENE\ LGRUP\ ODETR\ ENZAS$$

está cifrado como

$$c = Ee(m) = FULSW\ RJUDI\ LDHQH\ OJUXS\ RGHWU\ HQCDV.\blacksquare$$

Existen otros tipos de cifrado por bloques como: cifrado por sustitución homofónica, cifrado por sustitución polialfabético, cifrado de transposición, cifrado de composiciones, cifrado producto. Por otro lado, cabe mencionar que el AES (Advanced Encription Standard) es el nuevo estándar de cifrado simétrico dispuesto por el NIST (National Institute of Standards and Technology), después de un periodo de competencia entre 15 algoritmos sometidos. El 2 de octubre de 2000 fue designado el algoritmo Rijndael como AES, el estándar reemplazó al Triple Des, para ser usado en los próximos 20 años (ver Murphy y Robshaw, 2002).

Las ventajas de la criptografía simétrica son que los algoritmos son *fáciles* de implementar y requieren *poco* tiempo de cómputo. La desventaja principal es que las claves han de transmitirse por un canal seguro, algo difícil de realizar, es decir, la seguridad depende de un secreto compartido exclusivamente por el emisor y receptor. De hecho unos de los problemas mayores de la criptografía simétrica es encontrar un método eficaz para estar de acuerdo en el intercambio de claves seguras. Este problema se le llama *el problema de distribución de claves*.

1.3. Criptografía asimétrica

La criptografía asimétrica surge para solucionar el problema que tiene la criptografía simétrica de distribución de claves. Una solución a esto la dieron en el año de 1976 W. Diffie y M. Hellman. De manera creativa es también inventado el concepto de firma digital, que resuelve el problema de autenticidad de una entidad.

Un *esquema de cifrado de clave pública* es una quíntupla (M, C, K, E, D), donde:

a) M es el conjunto de textos llanos.

b) C es el conjunto de textos cifrados.

c) K es el conjunto de claves.

d) $\mathcal{E} = \{E_e : M \to C \mid e \in \mathcal{K}\}$ y $\mathcal{D} = \{D_d : C \to M \mid d \in \mathcal{K}\}$.

e)Para cada $e \in \mathcal{K}$, existe una única clave $d \in \mathcal{K}$ tal que $Dd(Ee(m)) = m$, para todo $m \in M$.

1.3.1. Observación

a) Normalmente *a e* se le llama clave pública y *a d* clave privada.

b) El proceso de aplicar la transformación E_e al mensaje m usualmente es llamado cifrado de *m,* y al proceso de aplicar la transformación D_d al texto cifrado $c = E_e(m)$ usualmente es llamado descifrado de *c*.

c) Por lo regular (pero no necesariamente), el conjunto de textos llanos, el conjunto de textos cifrados y el conjunto de claves son iguales, de hecho estos conjuntos pueden ser finitos (por ejemplo $\mathbb{Z}_n$) o infinitos (por ejemplo el grupo de trenzas).

d) Un *esquema de cifrado de clave pública*, en la práctica, está compuesto de tres algoritmos eficientes: un algoritmo de generación de claves (de hecho genera el par (e, d), un algoritmo de cifrado y un algoritmo de descifrado (ver por ejemplo Schneier, 1996).

e) Para que estos esquemas sean seguros ha de cumplirse que a partir de la clave pública resulte computacionalmente imposible calcular la clave privada.

1.4. Ejemplos

1.4.1. Algoritmo RSA[1]

Este algoritmo es de clave pública y debe su nombre a sus tres inventores: Rivest Ron, Shamir Adi y Adleman Leonard. La descripción del esquema es la siguiente:

a) Elegimos dos números primos p y q suficientemente grandes, y tomamos $n = pq$.

b) Buscamos e tal que sea primo con $\phi(n) = (p-1)(q-1)$.

c) Como e y $\phi(n)$ son primos entre sí, entonces existe un d tal que:

$$e \cdot d \equiv 1 (\text{mod } \phi(n) = n + 1 - p - q),$$

donde d se puede calcular mediante el algoritmo de Euclides.

d) Definimos

$Ee(x) = x^e \bmod n$ función de cifrado

$Dd(y) = y^d \bmod n$ función de descifrado x, y $\in \mathbb{Z}_n$.

- Clave pública: (e, n)
- Clave privada: (d, p, q).

Cualquiera que conozca p, q y d podrá descifrar los mensajes del propietario de la clave (de hecho en este caso basta conocer p y q para romper el sistema). Cabe mencionar que la justificación de este esquema depende sólo del hecho de que

$$x^{ed+1} \equiv x \bmod n$$

para cualquier entero libre de cuadrado n. Además de estos datos hemos de fijar la longitud de bloque: a saber, la longitud del bloque que vamos a cifrar y longitud del bloque cifrado.

Por ejemplo, si elegimos dos primos $p = 281$ y $q = 167$, entonces $n = 281 \cdot 167 = 46927$ y $\phi(n) = (281 - 1)(167 - 1) = 46480$. Buscamos e y d tales que $e \cdot d \equiv 1 \bmod \phi(46927)$, digamos $e = 39423$ y $d = 26767$. Así la clave pública será: (39423, 46927) y la clave privada será: (26767, 281, 167). Supongamos que vamos a cifrar bloques de dos letras en bloques de tres letras y que queremos cifrar HOLA utilizando un alfabeto de 36 símbolos, a saber

$$\Sigma = \{1, 2, 3, ..., 9, A, B, C, ..., Z\}$$

El procedimiento refiere los siguientes pasos:

a) Asignamos a cada letra un número según el alfabeto, en este caso:

$$\text{HOLA} \rightsquigarrow (17, 24, 21, 10).$$

b) Bloques a cifrar: (17, 24) y (21, 10).

c) Expresamos ambos bloques como un número en base 36:

$$(17, 24) = 17 \cdot 36^0 + 24 \cdot 36 = 881$$
$$(21, 10) = 21 \cdot 36^0 + 10 \cdot 36 = 381$$

d) Elevamos estos números a la *e-ésima* potencia y tomamos el residuo módulo 46927:

$$881^{39423} \equiv 45840 \bmod 46927$$
$$381^{39423} \equiv 26074 \bmod 46927.$$

e) Expresamos estos números en base 36, teniendo en cuenta que vamos a tener tres componentes:

$$45840 = 12 \cdot 36^0 + 13 \cdot 36 + 35 \cdot 36^2 = (12, 13, 35)$$
$$26074 = 10 \cdot 36^0 + 4 \cdot 36 + 20 \cdot 36^2 = (10, 4, 20)$$

f) Según el alfabeto considerado a cada número le asignamos una letra:

$$(12, 13, 35) \rightsquigarrow CDY \qquad (10, 4, 20) \rightsquigarrow A4K$$

1. Ver por ejemplo Menezes *et al.* 1997:285-291.

g) Por lo tanto el mensaje cifrado es *CDYA4K*.

h) Para descifrar habría que hacer el mismo proceso, pero partiendo de bloques de tres letras y terminando en bloques de dos letras, después aplicar la función de descifrado $D_d(y)$.

Para romper este esquema de cifrado lo podemos intentar de varias formas: *A* fuerza bruta, intentando resolver cualquiera de los dos logaritmos discretos:

$45840^d \equiv 881 \bmod 46927$ $26074^d \equiv 381 \bmod 46927\}$

o resolviendo:
$e \cdot d \equiv 1 \bmod \phi(46927)$,

Lo cual equivale a conocer $\phi(46927)$, que a su vez equivale a conocer la factorización en números primos de 46927. Aún no se conoce un algoritmo en tiempo polinomial que factorice, en primos, números lo suficientemente grandes.■

1.4.2. Para el siguiente ejemplo necesitamos saber cómo encontrar raíces cuadradas en Z_n.

Supongamos que tenemos $c = m^2 \bmod n$ y queremos encontrar las raíces cuadradas de *c* módulo *n*. A continuación daremos un algoritmo para el caso más usual que es cuando *n* es un número compuesto de la forma $n = pq$, con $p \equiv q \equiv 3 \bmod 4$. Para los demás casos (por ejemplo cuando *n* es primo, etc.) ver [1] *p*

Algoritmo que encuentra raíces cuadradas módulo *n* dado sus factores primos *p* y *q*, con $p \equiv q \equiv 3 \bmod 4$.

a) Usar el algoritmo extendido de Euclides (ver [1] p. 67) para encontrar enteros *a* y *b* satisfaciendo $ap + bq = 1$.

b) Calcular $r = c^{(p+1)/4} \bmod p$.

c) Calcular $s = c^{(q+1)/4} \bmod q$.

d) Calcular $x = (aps + bqr) \bmod n$.

e) Calcular $y = (aps - bqr) \bmod n$.

f) Las cuatro raíces de *c* módulo *n* son *x*, $-x \bmod n$, *y* y $-y \bmod n$.

Cifrado de clave pública de Rabin (ver por ejemplo [1] pp. 292-293). En este cifrado cada entidad crea una clave pública y una clave privada correspondiente.

*A*lgoritmo de generación de claves. La entidad *A* debe hacer lo siguiente:

a) Generar dos primos aleatoriamente grandes (y distintos) *p* y *q*, cada uno aproximadamente del mismo tamaño.

b) Calcular $n = pq$.

c) La clave pública será *n*; y la clave privada (*p*, *q*).

Algoritmo de cifrado. La entidad *B* cifra un mensaje *m* para *A*, que *A* descifra.

a) *Cifrado*. La entidad *B* debe hacer lo siguiente

• Obtener la clave pública *n*.

• Representar el mensaje como un entero *m* en el rango $\{0, 1,..., n - 1\}$.

• Calcular $c = m^2 \bmod n$.

• Mandar el texto cifrado *c* a *A*.

b) *Descifrado*. Para recuperar el texto llano *m* de *c, A* debe hacer lo siguiente:

• Usar un algoritmo que encuentra la raíces cuadradas módulo *n*, dados sus factores primos *p* y *q*. sean m_1, m_2, m_3 y m_4 las cuatro raíces de c módulo *n*.

• El mensaje enviado puede ser m_1, m_2, m_3 o m_4.

1.4.2.1. Observación

Una forma de decidir cuál es el mensaje original *m* de los cuatro posibles m_1, m_2, m_3, m_4, podemos hacer uso de la redundancia, por ejemplo:

Generación de claves. La entidad *A* elige los primos $p = 277$ y $q = 331$, y calcula $n = pq = 91687$. La clave pública es $n = 91687$, mientras la clave privada de *A* es ($p = 277$, $q = 331$).

a) *Cifrado*. Supóngase que se requieren los últimos seis bits de mensajes originales para ser reproducidos anterior al cifrado. En el orden para cifrar el mensaje de 10-*bit* $\overline{m} = 1001111001$, *B* reproduce los últimos seis bits de $\overline{m}$ para obtener el mensaje de 16-*bit* $m = 1001111001111001$, que en notación decimal es $m = 40569$. La entidad *B* entonces calcula:

$c = m^2 \bmod n = 40569^2 \bmod 91687 = 62111$
y envía esto a *A*.

b) *Descifrado*. Para descifrar *c, A* usa un algoritmo (podría ser el descrito arriba) para calcular las cuatro raíces cuadradas de *c* módulo *n*, las cuales son:

$m_1 = 69654$, $m_2 = 22033$, $m_3 = 40569$, $m_4 = 51118$,

en forma binaria:

$m_1 = 10001000000010110$, $m_2 = 101011000010001$,

$m_3 = 1001111001111001$, $m_4 = 1100011110101110$.

Ya que solamente m_3 tiene la redundancia requerida, *A* descifra *c* al m_3 y recupera el mensaje original $\overline{m}$.

La seguridad de este cifrado se basa en que en la actualidad, aún no existe un algoritmo eficiente que calcule raíces cuadradas módulo un número compuesto (para números suficientemente grandes).■

1.4.3. Protocolo de Intercambio de Diffie-Hellman (ver por ejemplo Menezes, *et al.* 1997).
Este intercambio de claves propuesto por Diffie y Hellman utiliza la función de exponenciación modular en canales abiertos. Sean A y B dos personas que quieren compartir un secreto, la descripción del esquema es la siguiente:

a) A y B eligen un primo p suficientemente grande.

b) Luego se elige un $g \in \mathbb{Z}_p^{\times}$ tal que $< g > = \mathbb{Z}_p^{\times}$.

c) Los valores p y g son públicos.

d) Tanto A como B eligen valores aleatorios, privados, x_A y x_B en $\mathbb{Z}_p^{\times}$.

e) A manda a B, y $y_A \equiv g^{xA} \bmod p$

f) B manda a A, y $y_B \equiv g^{xB} \bmod p$

g) Ambos lados de la comunicación construyen la misma clave simétrica.

• A calcula: $z_{BA} \equiv y_B^{xA} \equiv g^{xBxA} \bmod p$

• B calcula: $z_{AB} \equiv y_A^{xB} \equiv g^{xAxB} \bmod p$

luego $z_{AB} = z_{BA}$.

En este esquema los datos públicos son (p, g, y_A, y_B). En el caso de que alguien quisiera conocer la clave secreta a partir de los datos públicos, tendría que conocer x_A o x_B para generar la clave secreta z_{AB}, lo que equivale a resolver una de estas dos ecuaciones:

$$x_A \equiv \log_g y_A \bmod p$$
$$x_B \equiv \log_g y_B \bmod p$$

pero en la actualidad no se conocen algoritmos eficientes que resuelvan tales ecuaciones (con p suficientemente grande).■

2. Firmas digitales

Las firmas digitales son una solución que ofrece la criptografía para verificar la integridad de documentos y la procedencia de documentos. Las firmas digitales se pueden realizar tanto con criptografía simétrica como asimétrica.

Un *esquema de firma digital* es una quíntupla (M, A, K, S, V), donde:

1. M es el conjunto de mensajes que pueden ser firmados.

2. A es el conjunto de elementos llamados firmas.

3. K es el conjunto de claves.

4. $S = \{sig_K: M \to A \mid K \in K\}$ y $V = \{ver_K: M \times A \to \{\text{verdadero, falso}\} \mid K \in K\}$.

5. Para cada $K \in K$, existe un algoritmo de firmado $sig_K \in S$ y un correspondiente algoritmo de verificación $ver_K \in V$. Cada

$sig_K: M \to A$ y $ver_K: M \times A \to \{\text{verdadero, falso}\}$

Son funciones tal que la siguiente ecuación se satisface para cualquier mensaje $x \in M$ y para cualquier firma $y \in A$:

$$ver_K(x, y) = \begin{cases} verdadero & \text{si } y = sig_k(x) \\ falso & en\ caso\ contrario \end{cases}$$

2.1. Observación

Por lo regular (pero no necesariamente), el conjunto de textos a firmar, el conjunto de textos firmados y el conjunto de claves son iguales, de hecho estos conjuntos pueden ser finitos (por ejemplo $\mathbb{Z}_n$) o infinitos (por ejemplo el grupo de trenzas).

Un esquema de firma digital, en la práctica, está compuesto de tres algoritmos eficientes: un algoritmo de generación de claves (genera el par (e, d) donde e es llamada clave privada y d clave pública), un algoritmo de firmado y un algoritmo de verificación.

Definamos un tipo especial de función que se utiliza para firmar digitalmente.

2.2. Definición

Una función hash (o función resumen) es una función que mapea cadenas de bits de longitud arbitraria a cadenas de caracteres de longitud fija, o sea h: $\{0, 1\}^* \to \{0, 1\}^d$, y además es una función computacionalmente eficiente, es decir satisface las siguientes características:

a) Dado m, es computacionalmente fácil (tiempo polinomial) calcular $h(m)$.

b) Dado $h(m)$, es computacionalmente intratable (tiempo exponencial) recuperar m.

c) Dado m, es computacionalmente intratable obtener un m′ tal que $h(m) = h(m')$.

d) Debe ser difícil encontrar dos mensajes aleatorios m y m′ tales que $h(m) = h(m')$.

El proceso de firmar digitalmente con una función hash es como sigue: primero se produce un resumen del mensaje, luego se cifra este resumen con nuestra clave privada, de esta forma la única persona que conozca la clave privada será capaz de firmar digitalmente en nuestro nombre. Para verificar la firma procederemos de la siguiente forma: desciframos la firma digital usando la clave pública, obtenemos el resumen del mensaje original, hacemos un hash sobre el mensaje original. Comprobamos nuestro resumen con el obtenido al descifrar y si coinciden, la firma digital es válida. Cabe mencionar que se puede firmar digitalmente sin utilizar la función hash.

Note que el mensaje a ser firmado puede ser el texto llano o cifrado, porque el espacio del mensaje de la firma digital puede ser cualquiera subconjunto de $\{0, 1\}^*$. Ahora mencionamos dos tipos básicos de ataques.

a) *Ataque de clave única*: en este ataque el adversario conoce sólo la llave pública del firmante y por consiguiente sólo tiene la capacidad de verificar la validez de firmas de mensajes.

b) *El ataque de la firma conocida*: el adversario conoce la clave pública del firmante y ha visto el mensaje y la firma escogidos y producidos por el firmante legal.

Uno puede decir que el adversario ha roto un esquema de firma de un usuario A si su ataque le permite hacer cualquiera de los siguientes:

a) *Falsificación existencial*: el adversario tiene éxito falsificando la firma de un mensaje, no necesariamente de su elección.

b) *Falsificación selectiva*: el adversario tiene éxito falsificando la firma de algún mensaje de su elección.

c) *Falsificación universal:* el adversario encuentra un algoritmo de firma eficiente que funcionalmente equivale al algoritmo de firma de A.

d) *Rotura total:* el adversario puede computar la clave privada del firmante A.

2.3. Ejemplo

2.3.1. Esquema de firma RSA.

La descripción del esquema es la siguiente:

a) Un usuario elige dos números primos p y q suficientemente grandes.

b) Sea $\mathcal{M} = C = \mathbb{Z}_n$ el espacios de textos llanos y textos cifrados respectivamente.

c) Definamos el espacio de claves como

$$\mathcal{K} = \{(n, p, q, e, d): n = pq, ed \equiv 1 \bmod \phi(n)\}$$

d) Clave pública: (n, d)

e) Clave privada: (p, q, e).

f) Para $\mathcal{K} = (n, p, q, e, d)$, definamos:

$$\sigma_K(x) = x^e \bmod n$$
$$V_K(x, y) = \text{verdadero} \Leftrightarrow x \equiv y^d \bmod n$$
$$x, y \in \mathbb{Z}_n.\blacksquare$$

Otros algoritmos empleados en firmas digitales son: El Gammal, MD5 desarrollado por Ron Rivest (fue una modificación del MD4), y SHA desarrollado por la NSA (ver Menezes *et al.* 1997 o Koblitz, 2000).

3. Computación cuántica

Para ver la conexión que existe entre la Teoría de Números en Criptografía y su debilidad ante la posible era de las computadoras cuánticas, como bien dice el título del trabajo, primeramente veremos la estructura matemática básica de una computadora cuántica, posteriormente veremos su debilidad.

Se está investigando sobre computación cuántica y muchos de los problemas que corren en tiempo exponencial en computadoras clásicas correrían en tiempo polinomial sobre computadoras cuánticas. La computación cuántica es un nuevo desarrollo tecnológico para el procesamiento de información que depende del aprovechamiento de fenómenos característicos de la mecánica cuántica como: la cuantización, la superposición, la interferencia y el entrelazamiento.

"No hay información sin representación", esta es la idea que nos llevará intuitivamente a plantearnos la información cuántica. Formalmente la unidad básica de información de la computación cuántica es el qubit o bit cuántico. El *espacio de* 1 *qubit* es un espacio de Hilbert $\mathcal{E}_1$ isomorfo a $\mathbb{C}^2$ (recordemos que un espacio de hilbert se define como un espacio dotado de un producto interior que es completo con respecto a la norma vectorial definida por el producto interior), donde $\{|0\rangle, |1\rangle\}$ es una base ortogonal de $\mathcal{E}_1$ (de hecho es la base canónica de $\mathcal{E}_1$), la cual le llamaremos base computacional de $\mathcal{E}_1$. Las representaciones de la matriz de los vectores $|0\rangle$ y $|1\rangle$ están dadas por

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \ |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Sean $\alpha, \beta \in \mathbb{C}$, un *qubit genérico* es

$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \text{ con}$$
$$|\alpha|^2 + |\beta|2 = 1 \qquad \text{(Superposición lineal)} \quad (1)$$

Más generalmente, llamaremos estados a los vectores en $\mathcal{E}$ un espacio de Hilber de dimensión finita N (donde N es potencia de dos), y tales vectores los escribimos como $|v\rangle$ (notación de Dirac), les decimos *kets*. Cada ket tiene asociado un *bra*, que es una funcional lineal $\langle v| : \mathcal{E} \to \mathbb{C}$ definida por:

$$\langle v|(|w\rangle) = \langle v|w\rangle,$$

que es un producto interno, llamado bracket. Con bras y kets también podemos armar operadores lineales, de la forma: $|v\rangle\langle w| : \mathcal{E} \to \mathcal{E}$ que actúan según:

$$|v\rangle\langle w|(|x\rangle) = |v\rangle\langle w|x\rangle.$$

Un matriz hermitiana A es aquella que satisface la igualdad $A = A^\dagger$, es decir es aquella matriz que coincide con la conjugación de su propia transpuesta. Las matrices hermitianas

son importantes en mecánica cuántica, en particular en computación cuántica, porque sus autovalores son reales y se asocian a magnitudes físicas observables, de hecho llamaremos observable a una matriz hermitiana A.

Sea A un operador hermitiano que actúa en $\mathcal{E}$. Dado que A es en particular un operador normal, el teorema espectral afirma que, mediante un operador unitario, A puede ser diagonalizado de tal forma que sólo tiene entradas reales en la diagonal principal (los autovalores correspondientes). Consecuentemente, del conjunto de autovectores de A podemos extraer una base $\{u_1, u_2,..., u_N\}$ para $\mathcal{E}$. Supongamos que $a_1,..., a_N$ son los autovalores de A, es decir

$A|u_i\rangle = a_i|u_i\rangle$ para $i = 1,..., N$.

Sea $|\Psi\rangle \in \mathcal{E}$ un estado normalizado, de modo que $\langle\Psi|\Psi\rangle = 1$, entonces

$$|\Psi\rangle = \sum_{n=1}^{N} c_n|u_n\rangle$$

con $\langle u_n|u_m\rangle = \delta_{nm}$ (delta de Krönecker) y $c_n = \langle u_n|\Psi\rangle$. La probabilidad p_n de obtener el autovalor a_n es

$p_n = |c_n|^2 = |\langle u_n|\Psi\rangle|^2$.

Obsérvese que $\Sigma_{n=1}^{N}$, pues $\langle\Psi|\Psi\rangle = \Sigma_{n=1}^{N}|c_n|^2 = 1$. En particular medir un estado $|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$ en $\mathcal{E}_1$ implica obtener $|0\rangle$ con probabilidad $|\alpha|^2$ y $|1\rangle$ con probabilidad $|\beta|^2$.

Para seguir el estudio de los qubits, definamos el producto tensorial de dos matices. *El producto tensorial* de dos matrices A y B se define y denota como:

$$A\otimes B = \begin{pmatrix} A_{11}B & \dots & A_{1m}B \\ \vdots & \ddots & \vdots \\ A_{n1}B & \dots & A_{nm}B \end{pmatrix}.$$

Por ejemplo si $A = \begin{pmatrix} 2 & 5 \\ 5 & 6 \end{pmatrix}$ y $B = \begin{pmatrix} 7 \\ 4 \\ 8 \end{pmatrix}$, entonces

$$A\otimes B = \begin{pmatrix} 2\begin{pmatrix} 7 \\ 4 \\ 8 \end{pmatrix} & 3\begin{pmatrix} 7 \\ 4 \\ 8 \end{pmatrix} \\ 5\begin{pmatrix} 7 \\ 4 \\ 8 \end{pmatrix} & 6\begin{pmatrix} 7 \\ 4 \\ 8 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 14 & 21 \\ 8 & 12 \\ 16 & 24 \\ 35 & 42 \\ 20 & 24 \\ 40 & 48 \end{pmatrix}. \blacksquare$$

Un *espacio de* 2 *qubit* $\mathcal{E}_2$ es el producto tensoril de 1 qubit

$\mathcal{E}_2 = \mathcal{E}_1\otimes\mathcal{E}_1 := \mathcal{E}_1^{\otimes 2}$

Donde su base computacional es: $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$, la cual está dictada por las reglas del producto tensorial

$$|00\rangle = |0\rangle\otimes|0\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad |01\rangle = |0\rangle\otimes|1\rangle = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

$$|10\rangle = |1\rangle\otimes|0\rangle = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \quad |11\rangle = |1\rangle\otimes|1\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}.$$

En notación compacta o decimal, escribimos:

$|0\rangle = |00\rangle\ |1\rangle = |01\rangle\ |2\rangle = |10\rangle\ |3\rangle = |11\rangle$,

la dimensión de $\mathcal{E}_2$ es $2^2 = 4$, luego *un qubit genérico* será:

$$|\Psi\rangle = \alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle := \sum_{n=0}^{3} c_n|n\rangle \qquad (2)$$

con

$$|\alpha|^2 + |\beta|^2 + |\gamma|^2 + |\delta|^2 := \sum_{n=0}^{3} |c_n|^2 = 1$$

En general un *Espacio de N qubit* $\mathcal{E}_N$ es el producto tensorial de 1 qubit

$\mathcal{E}_N = \mathcal{E}_1 \otimes ... \otimes \mathcal{E}_1 := \mathcal{E}_1^{\otimes N}$,

Su base computacional (notación compacta) es:

$\{|0\rangle, |1\rangle,..., |2^N - 1\rangle\}$,

y su dimensión es 2^N, es decir su dimensión crece exponencialmente con el número de qubits, lo cual hace rápidamente inmanejable la simulación de un problema cuántico en una máquina clásica. Luego un qubit genérico de $\mathcal{E}^N$ se representa por

$\sum_{n=0}^{2^N-1} c_n|n\rangle$ con $\sum_{n=0}^{2^N-1} c_n|^2 = 1$.

4. Debilidad

Mucha de criptografía basada en la Teoría de Números, tales como RSA, llegarían a ser obsoletas si el algoritmo de Shor es implementado alguna vez en una computadora cuántica práctica. Un mensaje cifrado con RSA puede ser descifrado descomponiendo en factores la llave pública N, que es el producto de dos números primos. Los

algoritmos clásicos conocidos no pueden hacer esto en tiempo $O((logN)k)$ para ningún k, así que llegan a ser rápidamente imprácticos a medida que se aumenta N. Por el contrario, el algoritmo de Shor puede romper RSA en tiempo polinómico.

Como todos los algoritmos de computación cuántica, el algoritmo de Shor es probabilístico: da la respuesta correcta con alta probabilidad, y la probabilidad de fallo puede ser disminuida repitiendo el algoritmo. El algoritmo de Shor fue demostrado en 2001 por un grupo en IBM, que descompuso 15 en sus factores 3 y 5, usando una computadora cuántica con 7 qubits.

4.1. Algoritmo de Shor

Este algoritmo halla el orden de un elemento dentro de un grupo cíclico en tiempo polinomial, con ello nos permite encontrar los factores primos de un número compuesto N en tiempo cuántico polinomial. De esta manera, de existir computadoras cuánticas que manejen los qubits necesarios para implementar el algoritmo de Shor con números de 4096 qubits, por ejemplo, los esquemas de firma y cifrados basados en RSA quedarían completamente inseguros, debido a que al momento de escribir esta tesis, las longitudes de las claves son de entre 1024 y 2048 bits.

El problema es factorizar el número entero N. Dado un número aleatorio $x < N$ coprimo con N, se define el orden de x módulo N como el menor entero positivo no nulo r tal que

$$x^r \equiv 1 \bmod N,$$

resulta entonces que

$$x^r - 1 \equiv 0 \bmod N.$$

Si r es par, entonces

$$(x^{r/2} + 1)\,(x^{r/2} - 1) \equiv 0 \pmod N,$$

es decir, $(x^{r/2} + 1)$ y/o $(x^{r/2} - 1)$ deben contener factores comunes a N, y hallando éstos se puede factorizar N. Por supuesto, la probabilidad de que r sea par es1/2. Por ejemplo consideremos $N = 21$. La sucesión de equivalencias

$$2^4 \equiv 16 \bmod 21$$
$$2^5 \equiv 11 \bmod 21$$
$$2^6 \equiv 11 \times 2 \equiv 1 \bmod 21$$

muestra que el orden de $x = 2$ módulo 21 es $r = 6$. Por lo tanto

$$x^{r/2} \equiv 2^3 \equiv 8 \bmod 21,$$

luego $x^{r/2} - 1$ produce el factor 7 y $x^{r/2} + 1$ produce el factor 3.

Describamos brevemente el algoritmo de Shor.

Paso 1. Escoja un número aleatorio $a<N$

Paso 2. Calcule $mcd(a, N)$. Esto se puede hacer eficientemente usando el algoritmo de Euclides.

Paso 3. Si $mcd(a, N) \neq 1$, entonces es un factor no trivial de N, así que terminamos.

Paso 4. Si $mcd(a, N) = 1$, entonces hallar r, el período de la función siguiente:

$$f(x) = a^x \bmod N,$$

es decir el número entero más pequeño r para el cual $f(x + r) = f(x)$.

Paso 5. Si r es impar, vaya de nuevo al paso 1.

Paso 6. Si $a^{r/2} \equiv -1 \bmod N$, vaya de nuevo al paso 1.

Paso 7. Los factores de N son el mcd $(a^{r/2} \pm 1, N)$.

4.1.1 Observación

En el paso 4 entra la parte cuántica, pues en la actualidad no existe un algoritmo eficiente para encontrar el periodo, cuando N es suficientemente grande. Sin embargo, existe un algoritmo cuántico que lo encuentra en tiempo polinomial, tal algoritmo involucra la transformada de Fourier cuántica, para más detalles ver [15] o [19], y para una implementación experimental ver [21].

Al posible uso de computadoras cuánticas para realizar ataques a los sistemas criptográficos actuales se le ha dado en llamar criptoanálisis cuántico. La computación cuántica como disciplina es muy reciente y aún no se ha desarrollado un método de programación intuitivo y fácil, por lo que el diseño de algoritmos cuánticos es complicado, y tiene que basarse en las operaciones elementales con puertas cuánticas. Finalmente mencionemos algunas diferencias entre el computador clásico y cuántico:

a) Una computadora actual se estima que tardaría varios años para factorizar un número grande (supongamos, por ejemplo 1 000 dígitos), mientras que un computador cuántico lo haría en minutos (algoritmo de Shor). Así podríamos romper RSA, y con pocas adaptaciones otros criptosistemas similares de clave pública, como los basados en curvas elípticas.

b) Las búsquedas en bases de datos no ordenadas se realizan actualmente al azar y para localizar un dato en especial se requiere en promedio de $N/2$ intentos, donde N es el número total de datos. Una computadora cuántica podría realizar lo anterior en un número de intentos

igual a la raíz cuadrada de *N* (algoritmo de Grover). Esto podría usarse para atacar de manera más eficiente que en cómputo clásico a los criptosistemas de clave privada DES, triple DES, AES, etcétera.

c) El qubit no puede ser construido a partir del transistor; más bien se deben utilizar partículas o sistemas de partículas que manifiesten el fenómeno de la interferencia cuántica.

Conclusiones

En este trabajo vimos una introducción de cómo la teoría de números juega un papel importante en la criptografía moderna, la cual tiene muchas aplicaciones en seguridad de información. Por otro lado, cuando la física cuántica se fusionó con la informática, dieron nacimiento a una nueva línea de investigación que es la *computación cuántica*. Se dio una idea intuitiva de la estructura matemática para construir los qubits que es información básica de una computadora cuántica. La posible construcción eficiente de tal maquinaria tan poderosa, romperá todos los protocolos criptográficos basados en la teoría de números, tales como son: firmas digitales, comercio electrónico, certificados digitales, sellos digitales, votos digitales, transferencias bancarias, intersección de información de gobiernos y ejércitos, etc. El que logre construir tal máquina tendría el control del mundo.

Bibliografía

Advanced Encryption Standard (1997). FIPS-197, <http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>.

Anshel, I.; M. Anshel; B. Fisher y D. Goldfeld (2001). "New Key Agreement Protocols in Braid Group Cryptography", in *Topics in Cryptology CT-RSA*. Lecture notes in computer science. Vol. 2020. Berlin Heidelberg New York: Springer.

Artin, E. (1947). *Theory of Braids*. Ann Math 48(2), 101-126.

Birman, J. S. (1974). "Braids, Links and Mapping Class Groups", *Annals of Math. Study,* 82. Princeton University Press.

Cha, J. C.; K. H. Ko; S. J. Lee; J. W. Han and J. H. Cheon (2001). "An Effcient Implementation of Braid Groups", in *Advances in Cryptology -ASIACRYPT 2001* (Gold Coast). Lecture notes in computer science, vol. 2248, pp. 144-156. Berlin Heidelberg New York: Springer.

Dehornoy P.(2004). "Braid-based Cryptography", in Myasnikov, A., Shpilrain, V., (eds.) *Group theory, statistics and cryptography*. *Contemporary mathematics*, vol 360, pp 5-33. American Mathematical Society. Online available at <http://www.math.unicaen.fr/ dehornoy/Surveys/Dgw.ps>.

Ekert, A. and R. Jozsa (1996). "Quantum Computation and Shor's Factoring Algorithm". *Reviews of Modern Physics*.

Epstein, D.; J. Cannon; D. Holt; S. Levy; M. Paterson and W. Thurston (1992). *Word Processing in Groups*. Boston, MA: Jones and Bartlett Publishers.

Franco, N. and J. Gonzales-Meneses (2001). "Conjugacy Problem for Braid Groups and Garside Groups", J. Algebra, to appear; <http://xxx.lanl.gov/abs/math.GT/0112310>.

Ko, K. H.; D. H. Choi; M. S. Cho & J. W. Lee, "New Signature scheme Using conjugacy Problem", Preprint <http://eprint.iacr.org/2002/168>.

Ko, K. H.; S. J. Lee; J. H. Cheon; J. W. Han; J.-S. Kang and C. Park (2000). "New public-key cryptosystem using braid groups", in *Advances in cryptology -CRYPTO 2000* (Santa Barbara, C. A). Lecture notes in computer science, vol 1880, Berlin Heidelberg New York: Springer.

Menezes, A.; P. van Oorschot and S. Vanstone (1997). *Handbook of Applied Cryptography*. CRC Press.

Michael, A. N. and L. C. Isaac (2000). Q*uantum Computation and Quantum Information*, Cambridge University Press.

Murphy, S. and M. Robshaw (2002). *Essential Algebraic Structure within the AES,* Crypto, and LNCS 2442.

Neal Koblitz (2003). *A Course in Number Theory and Cryptography*, Second Edition, Springer Verlag.

Peter, W. S. (1996). "*P*olynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer, lanl.arXiv.org ePrint Archive", Online available at <http://

www.arxiv.org/PScache/quantph/pdf/9508/9508027v2.pdf> (25 of June 1996).

Schneier, B. (1996). *Protocols, Algorithms, and Source Code.* in C. Ed. Wiley, segunda edición.

Sunder L. and A. Chaturvedi (2005). *Authentication Schemes Using Braid Groups,* <http://arxiv.org/pdf/cs.CR/0507066> (julio 2005).

Vandersypen, M. K.; M. Steffen and G. Breyta (2001). *Experimental Realization of Shor quantum Factoring Algorithm Using Nuclear Magnetic Resonance.* <http://arxiv.org/PScache/quantph/pdf/0112/0112176v1.pdf> (30 de diciembre de 2001).

Volker Gebhardt (2006). *Conjugacy Search in Braid Groups from a Braid-based Cryptography Point of View*, Springer-Verlag 2006.