FACULTAD DE MATEMÁTICAS

MAESTRÍA EN CIENCIAS MATEMÁTICAS

TESIS DE MAESTRÍA:

FUNDAMENTOS MATEMÁTICOS DEL ALGORITMO RSA

Tesis presentada por José Florentino Abarca Pita para obtener el grado de Maestría en Ciencias de la Universidad Autónoma de Guerrero

---

Asesor:
Dr. Jesús Romero Valencia

Índice general

Agradecimientos

La gratitud no sólo es la más grande de las virtudes,
sino que engendra todas las demás.
Cicerón.

Cada ciclo en nuestra vida es un círculo que debe cerrarse. Hoy se cierra un círculo particularmente complicado, de modo que cerrarlo me llena de satisfacción. El camino recorrido está poblado de personas que de un modo u otro, me ayudaron a recorrerlo, a veces con una palabra de aliento, otras con una desvelada conjunta, con un "jalón de orejas", con una invitacion al cine, en fin, las formas son muchas, pero para cada una de las personas que intervinieron e hicieron posible este logro, gracias de todo corazón, gracias...

Agradezco al Dr. Jesús Romero Valencia, por haber dicho que si todas las veces que el sentido común decía no.

Al Dr. Juan Carlos Hernández y al Dr. José María Sigarreta Almira, sin cuyo decidido apoyo simplemente esto no hubiera ocurrido.

A los profesores de la Maestría, que al compartirnos sus conocimientos, contribuyeron a nuestra formación académica y humana: Dr. Gerardo Reyna, Dra. Joana Luviano y Dr. Martín Patricio Árciga.

A mis padres, ya fallecidos, Florentino J. Abarca y María Elisa Pita Jerónimo, por traerme al escenario y por las enseñanzas.

Al comenzar el viaje éramos compañeros pero en el camino nos hicimos amigos, que digo amigos, hermanos, y fue gracias a ustedes que pude

soportar de pie la dura prueba que significaron estos dos años. Dondequiera que estén, gracias: Ángel (Cómico), Cristóbal, Daniel, Rosalío, Fleitas y desde luego, Lulú. Los llevo en el corazón.

A Enrique, Luz, José y Maricarmen, mis hermanos de sangre, porque siempre estuvieron ahí, sobre todo tú, Luz, que empezaste como médico de cuerpos y te has elevado a la estatura de médico de almas.

A mis hijas, Sarita y Lizy, porque soportaron durante dos años una doble ausencia. Con toda seguridad, la vida compensará.

Carissima, gracias, porque cada minuto valió la pena.

Si, también te agradezco a ti, Guadalupe, por todos los momentos compartidos y por todo lo que me enseñaste, que no fue poco.

José Florentino Abarca Pita.

Resumen

En todos los tiempos, los seres humanos se han comunicado protegiendo de algún modo la información sensible. A este fin, ha sido necesario desarrollar herramientas que permitan transmitir dicha información sensible. Tales herramientas han sido los algoritmos criptográficos, que cada uno en su momento, brindaron la protección necesaria.

El propósito de este trabajo es mostrar a detalle uno de los algoritmos criptográficos de más amplio uso en nuestros días, pese a su considerable antigüedad: el algoritmo RSA.

Como todo desarrollo humano, tiene antecedentes tanto históricos como matemáticos, de los cuales se ocupa el Capítulo 1, en el que se define la criptografía y se muestra un breve panorama de los algoritmos criptográficos utilizados en otros tiempos.

Las necesarias herramientas matemáticas que hacen posible el funcionamiento del algoritmo RSA, se ubican en el capítulo 2. La seguridad del algoritmo reside en la imposibilidad (aún en nuestros días) de desarrollar un algoritmo para factorizar el producto de dos números primos suficientemente grandes.

Finalmente, el Capítulo 3 ofrece una descripción detallada tanto del cifrado como el descifrado de *mensajes* con RSA. Aunque decimos mensajes, conviene desde ahora mencionar que dada la lentitud de los algoritmos criptográficos asimétricos, lo que usualmente se cifra con ellos son números que codifican claves de sesión y no mensajes de texto propiamente dichos. Naturalmente, el análisis de cada criptosistema lleva aparejado el algoritmo correspondiente.

Capítulo 1

Elementos de criptografía

En este primer capítulo definimos la criptografía y hacemos un recorrido somero por su desarrollo histórico, mencionando algunos de los algoritmos criptográficos que se utilizaron tanto en los tiempos antiguos como en tiempos más recientes.

1.1. Definición

La criptología se ocupa del estudio de los sistemas, claves y lenguajes secretos. Se divide a su vez en dos grandes ramas: la *criptografía* y el *criptoanálisis*. El presente trabajo se ubica en el contexto de la criptografía, palabra que según el Diccionario de la Real Academia Española, proviene del griego $\kappa\rho\upsilon\pi\tau o\zeta$ = oculto; $\gamma\rho\alpha\varphi\varepsilon\iota\nu$ = escribir y el sufijo -*ia* para crear sustantivos abstractos y su significado es *Arte de escribir con clave secreta o de un modo enigmático*. La criptografía trata sobre la protección de la información, englobando varias disciplinas, entre las que destacan la teoría de la información, las matemáticas discretas y la complejidad algorítmica. Al día de hoy, la criptografía, como técnica de enmascaramiento de la información, se encuentra muy unida a la informática y las redes de computadoras, completamente alejada de las máquinas de cifrar antiguas y de los métodos utilizados en el pasado para mantener el secreto de las comunicaciones.

El desarrollo de la criptografía moderna se sustenta en dos trabajos fundamentales: El primero de ellos es la publicación de los artículos *A mathemathical theory of communication* [22] y *Communication theory of secrecy systems*

[23] por Claude E. Shannon. El segundo es la publicación de *New directions in Cryptography* [4] de Diffie y Hellman, que introdujo el concepto de criptografía asimétrica, abriendo las posibilidades de aplicación de esta disciplina.

Shannon propuso dos técnicas de cifrado a las que llamó difusión y confusión. La difusión, mediante permutaciones o transposiciones, permitiría dispersar las propiedades estadísticas propias del lenguaje en el texto en claro sobre el criptograma. Por otro lado, la confusión permitiría generar mezcla, caos, confusión en el cifrado, de manera que la dependencia entre el texto en claro, clave y criptograma fuera lo más compleja posible, buscando con ello que no se pudiera romper el algoritmo de cifra. Para lograrlo, se sirve de sustituciones.

La publicación del artículo *New directions in cryptography* [4] por Diffie y Hellman representa un punto de quiebre en la historia de la criptografía, pues estableció las bases de la criptografía asimétrica o de clave pública, en la que cada participante en una comunicación secreta tendría dos claves, una pública y otra privada. Un emisor podría comunicarse con otra persona conociendo sólo su clave pública y sólo esa persona podría descifrar la comunicación, dado que sólo ella conocía su clave privada. Al mensaje cifrado se le llama *criptograma* (Figura 1.1).

Figura 1.1: Criptograma

La criptografía en la actualidad, además de proporcionar las técnicas para

cifrar/descifrar, debe garantizar:

Privacidad. El acceso a los datos lo realizan los usuarios autorizados para ello. Cuando se crea una comunicación entre dos puntos, es muy difícil determinar con toda seguridad que no está siendo captada por otras personas. Esto es así porque hay una ausencia de control en la comunicación por las partes que la han establecido. La única posibilidad factible para garantizar seguridad en la comunicación es cifrándola.

Autenticación. Dada la posibilidad que existe de suplantar identidades en transacciones en línea, es vital implementar medidas para verificar las identidades de los usuarios que establecen la comunicación. Por ejemplo, en un mensaje de correo electrónico, deben autenticarse tanto el emisor como el receptor.

Integridad. No deben permitirse alteraciones en los datos que se transmiten, ya que su integridad es fundamental. Imaginemos, por ejemplo, un mensaje alterado, donde se notifica una cantidad de dinero que es el doble de la original.

No repudio. Se acredita la autoría de un mensaje por parte de un usuario. Con esto no se puede negar la acción de haber comprado en línea o haber realizado la declaración de impuestos a través de Internet.

Para su análisis, la criptografía se divide en criptografía simétrica y criptografía asimétrica. La primera, también llamada criptografía de clave secreta o criptografía de clave privada se basa en el uso de *una* sola clave tanto para cifrar como para descifrar el mensaje. Tanto el emisor como el receptor del mensaje deben conocer la clave, que debe transmitirse por un canal seguro. Este es su talón de Aquiles, la distribución de las claves.

La criptografía asimétrica o de clave pública utiliza dos claves relacionadas inversamente entre sí, ya que funcionan como un par: una clave se utiliza para cifrar el mensaje y la otra para descifrarlo. Al momento de crear las dos claves, cualquiera de ellas puede usarse para cifrar o descifrar, pero una vez que se usa una clave para una de éstas operaciones, la otra necesariamente se usará para la operación inversa. La clave privada debiera conocerla sólo el propietario, mientras que la clave pública puede entregarse a cual-

quier persona. Mencionaremos aquí algunos de los algoritmos de cifrado más conocidos.

1.2. Cifrados antiguos

Los algoritmos de cifrado utilizados en la antigüedad han caído en desuso debido a que pueden descifrarse fácilmente empleando una computadora doméstica. Los métodos utilizados para ello son el análisis estadístico o en el peor caso, fuerza bruta.

1.2.1. Cifrado César

Probablemente el cifrado por sustitución más famoso de la antigüedad. Se le llama así porque es el que usaba el emperador romano Julio César para enviar mensajes secretos a sus generales. Es uno de los algoritmos más simples. Sólo hay que sumar 3 al número de orden de cada letra. Por ejemplo, a la A le correspondería la D, a la B la E y así sucesivamente. Matemáticamente podemos ver el Cifrado César como una función lineal del tipo:

$$E(x) = (x + 3) \mod 27$$

La posición que cada letra *en claro* ocupa en el alfabeto, está indicada por x. $E(x)$ indica la posición de la letra cifrada correspondiente a x en el alfabeto. Así tenemos que E(0)=3, y E(26)=2, es decir, la A se cifra como D y la Z como C. La operación de descifrado podemos realizarla mediante la función:

$$D(x) = (x - 3) \mod 27$$

La Tabla 1 muestra la correspondencia de alfabetos.

0	1	2	3	4	5	6	7	8	9	10	11	12	13
A	B	C	D	E	F	G	H	I	J	K	L	M	N
D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P

14	15	16	17	18	19	20	21	22	23	24	25	26
Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Ejemplo: Para cifrar el texto CRIPTOGRAFIA SIMETRICA, tendríamos:

Mensaje en claro: CRIPTOGRAFIA SIMETRICA.
Mensaje cifrado: FULSWRJUDILD VLPHWUFD.

Los dos algoritmos mostrados a continuación, permiten el cifrado y el descifrado César, respectivamente.

Algoritmo 1. *Cifrado César.*
ENTRADA: Texto plano M.
SALIDA: Texto cifrado C.
1. $n = longitud(M)$.
2. Para i = 0 a (n-1) hacer
Calcular $C[i] = M[(i + 3)$ *mód* $27]$
3. Devolver (C).
4. Fin.

Algoritmo 2. *Descifrado César.*
ENTRADA: Texto cifrado C.
SALIDA: Texto cifrado M.
1. $n = longitud(C)$.
2. Para i = 0 a (n-1) hacer
Calcular $M[i] = C[(i - 3)$ *mód* $27]$
3. Devolver (M).
4. Fin.

1.2.2. Cifrado Afín

Lo expresado para el cifrado César puede generalizarse del siguiente modo:

$$E(x) = ax + b \text{ mód } n$$

en donde a se conoce como constante de decimación; b, constante de desplazamiento y n, el módulo de cifra. A este tipo de cifrado se le llama cifrado afín.

Es claro que los valores de a y de b no pueden ser arbitrarios. En primer lugar, a no puede ser 0, pues no existiría una equivalencia de alfabetos, por tanto, debe cumplirse $a \geq 1$. Por otro lado, para garantizar la existencia de

inversos, los valores de a y el módulo n deben ser primos relativos, es decir, $(a, n) = 1$. Como se trabaja en módulo $27 = 3^3$, los valores que puede tomar a son: $1, 2, 4, 5, 7, 8, 10, 11, 13, 14, 16, 17, 19, 20, 22, 23, 25$ *y* 26.
Los valores de b, en cambio, pueden moverse entre 0 y $n-1$, ya que se asegura siempre el inverso aditivo. Así, el número de claves que pueden generarse es $18 \cdot 27 = 486$. Sin embargo, $a = 1$, $b = 0$ debe descartarse, pues no cifra realmente, lo que deja 485 claves.

El descifrado viene dado por:

$$D(x) = a^{-1}(x - b) \; mód \; n$$

donde a^{-1} es el inverso multiplicativo de a módulo n.

Ejemplo:
Tomemos $a = 4$ y $b = 5$. El alfabeto original y el cifrado, se corresponderían como muestra la siguiente tabla:

0	1	2	3	4	5	6	7	8	9	10	11	12	13
A	B	C	D	E	F	G	H	I	J	K	L	M	N
F	J	N	Q	U	Y	C	G	K	Ñ	R	V	Z	D

14	15	16	17	18	19	20	21	22	23	24	25	26
Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
H	L	O	S	W	A	E	I	M	P	T	X	B

Así, para cifrar el texto CIFRADO AFIN, tendríamos:

Mensaje en claro: CIFRADO AFIN.
Mensaje cifrado: NKYWFQL FYKD.

Los algoritmos para el cifrado y el descifrado afín, respectivamente, son los siguientes:

Algoritmo 3. *Cifrado Afín.*
ENTRADA: Texto plano M, valores de a y de b.
SALIDA: Texto cifrado C.
1. $n = longitud(M)$.
2. Para i = 0 a (n-1) hacer

Calcular $C[i] = M[(a \cdot i) + b)$ *mód* $27]$
3. Devolver (C).
4. Fin.

Algoritmo 4. *Descifrado César.*
ENTRADA: Texto cifrado C.
SALIDA: Texto cifrado M.
1. $n = longitud(C)$.
2. Para i = 0 a (n-1) hacer
Calcular $M[i] = C[(i - 3)$ *mód* $27]$
3. Devolver (M).
4. Fin.

1.2.3. Cifrador de Vigenère

El cifrador polialfabético más conocido es el sistema de Vigenère, llamado así en honor al criptólogo francés Blaise de Vigenère (1523-1596). El sistema usa el mismo método que el cifrador del César, es decir, una sustitución monográmica por desplazamiento de k caracteres en el texto. El desplazamiento lo indica el valor numérico asociado a uno de los caracteres de una clave. Dicha clave debe escribirse cíclicamente bajo el mensaje. Veámoslo en detalle con el siguiente:

Ejemplo:

C	R	I	P	T	O	G	R	A	F	I	A	S	I	M	E	T	R	I	C	A
A	C	A	P	U	L	C	O	A	C	A	P	U	L	C	O	A	C	A	P	U

La clave ACAPULCO tiene una periodicidad de 8. A la primera letra del mensaje C, le aplicamos un desplazamiento equivalente a la primera letra de la clave A, lo que origina $C + A = (2 + 0)$ *mód* $27 = 2 = C$; la segunda letra R se cifra de forma análoga, sumándola con la segunda letra de la clave C, $R + C = (18 + 2)$ *mód* $27 = 20 = T$, etc. El resultado final será el criptograma: C = CTIFÑZIGAHIPNSÑSTTIRU, como se detalla a continuación:

$$
\begin{array}{rcccl}
C+A & = & (2{+}0) \text{ mód } 27 & = 2 & \Rightarrow C \\
R+C & = & (18{+}2) \text{ mód } 27 & = 20 & \Rightarrow T \\
I+A & = & (8{+}0) \text{ mód } 27 & = 8 & \Rightarrow I \\
P+P & = & (16{+}16) \text{ mód } 27 & = 5 & \Rightarrow F \\
T+U & = & (20{+}21) \text{ mód } 27 & = 14 & \Rightarrow \tilde{N} \\
O+L & = & (15{+}11) \text{ mód } 27 & = 26 & \Rightarrow Z \\
G+C & = & (6{+}2) \text{ mód } 27 & = 8 & \Rightarrow I \\
R+O & = & (18{+}15) \text{ mód } 27 & = 6 & \Rightarrow G \\
A+A & = & (0{+}0) \text{ mód } 27 & = 0 & \Rightarrow A \\
F+C & = & (5{+}2) \text{ mód } 27 & = 7 & \Rightarrow H \\
I+A & = & (8{+}0) \text{ mód } 27 & = 8 & \Rightarrow I \\
A+P & = & (0{+}16) \text{ mód } 27 & = 16 & \Rightarrow P \\
S+U & = & (19{+}21) \text{ mód } 27 & = 13 & \Rightarrow N \\
I+L & = & (8{+}11) \text{ mód } 27 & = 19 & \Rightarrow S \\
M+C & = & (12{+}2) \text{ mód } 27 & = 14 & \Rightarrow \tilde{N} \\
E+O & = & (4{+}15) \text{ mód } 27 & = 19 & \Rightarrow S \\
T+A & = & (20{+}0) \text{ mód } 27 & = 20 & \Rightarrow T \\
R+C & = & (18{+}2) \text{ mód } 27 & = 20 & \Rightarrow T \\
I+A & = & (8{+}0) \text{ mód } 27 & = 8 & \Rightarrow I \\
C+P & = & (2{+}16) \text{ mód } 27 & = 18 & \Rightarrow R \\
A+U & = & (0{+}21) \text{ mód } 27 & = 21 & \Rightarrow U
\end{array}
$$

Nótese que caracteres repetidos del mensaje se cifran de forma distinta, dependiendo de su posición relativa respecto a la clave. Es el caso de la letra C que se cifra una vez como C y la otra como R. Algo similar ocurre con la letra T que se cifra también dos veces, una como Ñ y la otra como T. Por otro lado, una letra repetida del criptograma puede originarse de caracteres distintos del texto en claro, como la letra S que proviene de los caracteres I y E del mensaje.

Como la clave está formada por un conjunto de d caracteres, $K = k_1...k_d$, en donde k_i $(i = 1, ..., d)$ representa la cantidad de desplazamiento del alfabeto $i - ésimo$, la función de transformación de Vigenère para cifrar viene dada por:

$$C_i = E_{k_i}(M_i) = (M_i + k_i) \; mód \; n$$

Acorde con la función de cifrado, la función de descifrado de Vigenère debe usar el inverso del desplazamiento aplicado, por lo que:

$$M_i = D_{k_i}(C_i) = (C_i - k_i) \ mód \ n$$

.

Algoritmo 5. *Cifrado Vigenère.*
ENTRADA: Texto plano $M, Pass$.
SALIDA: Texto cifrado C.
1. $n = longitud(M)$.
2. Para $i = 0$ *a (n-1) hacer*
Calcular $C[i] = (M[i] + Pass[i])$ *mód* 27
3. Devolver (C).
4. Fin.

Algoritmo 6. *Descifrado Vigenère.*
ENTRADA: Texto cifrado $C, Pass$.
SALIDA: Texto en claro M.
1. $n = longitud(C)$.
2. Para $i = 0$ *a (n-1) hacer*
Calcular $M[i] = (C[i] - Pass[i])$ *mód* 27
3. Devolver (M).
4. Fin.

1.2.4. Cifrador de Vernam

En 1917, Gilbert S. Vernam, ingeniero del MIT que trabajaba en AT&T, diseña un dispositivo criptográfico basado en los 32 códigos Baudot de los teletipos creados por su compañía. Vernam sugiere una clave continua con la que se cifraría el mensaje carácter por carácter para lograr el texto cifrado. Después, junto con Joseph Mauborgne vieron la dificultad de criptoanalizar comunicaciones si la clave era aleatoria y se cumplían algunas propiedades. Esto hizo posible el nacimiento de la libreta de un solo uso. En la década de los 40's, Claude Shannon demostró que este sistema tenía el así llamado, secreto perfecto, es decir, que el texto cifrado no ofrece absolutamente ninguna información sobre el texto sin cifrar. El cifrado de Vernam (libreta de un solo uso) es el único algoritmo criptográfico irrompible si se tienen las siguientes condiciones: libretas de un solo uso aleatorias, la creación y el intercambio de

las libretas de un solo uso debe ser segura y la libreta tiene que ser al menos tan larga como el mensaje. El cifrador de Vernam representa cada carácter M_i con 5 bits en código Baudot que se suma OR-EXCLUSIVO (XOR) con la correspondiente clave k_i de una secuencia binaria aleatoria. Así, el cifrador de Vernam, crea un flujo de bits de texto cifrado de la forma:

$$C = E_k(M) = C_1C_2C_3...C_N \tag{1.1}$$

De donde:

$$C_i = (M_i + k_i)\, mod\, 2 \qquad con\, i\, =\, 1, 2, ...N \tag{1.2}$$

$$C_i = (M_i \oplus k_i) \tag{1.3}$$

Ejemplo: Usando el código de Baudot cifre el mensaje M = CIFRAR con la clave K = MONEDA.
Solución: Haciendo la suma OR-exclusivo:

$C \oplus M = 01110 \oplus 11100 = 10010 = L$
$I \oplus O = 00110 \oplus 11000 = 11110 = V$
$F \oplus N = 01101 \oplus 01100 = 00001 = E$
$R \oplus E = 01010 \oplus 00001 = 01011 = J$
$A \oplus D = 00011 \oplus 01001 = 01010 = R$
$R \oplus A = 01010 \oplus 00011 = 01001 = D$

Obtenemos el texto cifrado C = LVEJRD

Capítulo 2

Preliminares matemáticos

Gran parte de la criptografía moderna está basada sobre los fundamentos del álgebra y la teoría de números. En especial, el algoritmo RSA debe su funcionamiento a resultados importantes de esta última. En este capítulo, haremos un recorrido por los elementos fundamentales de la teoría de números que sustentan el algoritmo RSA, describiéndolos y probándolos.

2.1. Divisibilidad

La *Teoría de los Números* es el estudio de las propiedades de los números naturales

$$1, 2, 3, 4, ...$$

o más generalmente, el estudio de las propiedades de los enteros

$$..., -3, -2, -1, 0, 1, 2, 3...$$

Denotamos el conjunto de los números enteros con el símbolo $\mathbb{Z}$. Los enteros pueden sumarse, restarse y multiplicarse del modo usual y satisfacen las reglas usuales de la aritmética: conmutatividad, asociatividad, distributividad, entre otras.
Consideremos a y b enteros. Podemos sumarlos $a + b$, restarlos $a - b$ y multiplicarlos $a \cdot b$, obteniendo en cada caso otro número entero. Sin embargo, no siempre es posible dividir un entero por otro. Por ejemplo, no podemos

dividir 5 por 2, pues no existe un entero que sea igual a $\frac{5}{2}$. Esto conduce a la importante noción de *divisibilidad*.

Definición. *Sean a y b enteros con $b \neq 0$. Decimos que b divide* a *a o que a es divisible por b, si existe un entero c tal que:*

$$a = bc$$

Escribimos $b|a$ para señalar que b divide a a, en el caso contrario, escribimos $b \nmid a$.

Ejemplo: $3|111$, dado que $111 = 3 \cdot 37$.

Ejemplo: $27 \nmid 765$, dado que si tratamos de dividir 765 por 27, la división no es exacta.

2.2. El algoritmo de la división.

Dado un entero n, el conjunto $\mathbb{Z}$ de todos los enteros puede dividirse en dos subconjuntos: los que son múltiplos de n y los que no lo son. Gran parte de la teoría de números se basa en el refinamiento de esta partición obtenida clasificando a los enteros que no son múltiplos de n de acuerdo con el residuo que se obtiene al dividirlos por n.

Teorema 1. *(Algoritmo de la División)*
Si a y b son números enteros con $b > 0$, entonces existen dos enteros, q y r, únicos, tales que $a = bq + r$, con $0 \leq r < b$. A los números a, b, q y r se les denomina, respectivamente, dividendo, divisor, cociente y residuo.

Demostración. **Existencia de q y r.**
Basta tomar q como un número entero tal que bq sea el mayor de los múltiplos de b menor o igual que a, es decir, $bq \leq a$. Una vez obtenido el cociente q, podemos calcular el residuo r haciendo

$$r = a - bq.$$

Por otra parte, si $bq \leq a$, entonces el siguiente múltiplo de q, $b(q + 1)$, será estrictamente mayor que a, es decir,

$$bq \leq a < b(q + 1).$$

Entonces,

$$bq \leq a < b(q+1)$$

O sea que

$$bq - bq \ \leq \ a - bq \ < \ b(q+1) - bq$$

Por tanto,

$$0 \leq a - bq < b$$

Luego,

$$0 \leq r < b$$

Así pues, existen q y r, enteros tales que

$$a = bq + r, \ con \ 0 \leq r < b.$$

Unicidad de q y r.
Supongamos que existen q_1, q_2, r_1 y r_2, enteros tales que verifican el teorema, o sea,

$$a = bq_1 + r_1 : 0 \leq r_1 < b$$

$$a = bq_2 + r_2 : 0 \leq r_2 < b.$$

Entonces,

$$bq_1 + r_1 = bq_2 + r_2 \Longrightarrow b(q_1 - q_2) = r_2 - r_1 \Longrightarrow b|q_1 - q_2| = |r_2 - r_1|$$

y al ser

$$0 \leq r_1, r_2 < b$$

tenemos

$$0 \leq |r_2 - r_1| < b$$

luego,

$b|q_1 - q_2| = |r_2 - r_1| < b$, lo que implica $b|q_1 - q_2| < b$
es decir,

$$b(1 - |q_1 - q_2|) > 0$$

y al ser $b > 0$, tendremos que

$$1 - |q_1 - q_2| > 0$$

de donde se sigue que

$$0 \leq |q_1 - q_2| < 1$$

y como q_1 y q_2 son enteros, tendrá que ser

$$|q_1 - q_2| = 0$$

por tanto,

$$q_1 = q_2$$

de donde se sigue también que

$$r_1 = r_2$$

□

Ejemplos:

1. Sean $a = 83$ y $b = 5$.
   El mayor múltiplo de 5 menor o igual que 83 es $5 \cdot 16$, así que tomando $q = 16$ y $r = 83 - 5 \cdot 16 = 3$, tendremos que

   $$83 = 5 \cdot 16 + 3, \quad con \;\; 0 \leqslant 3 < 5$$

2. Sean $a = 12$ y $b = 20$.
   El mayor múltiplo de 20 menor o igual que 12 es $20 \cdot 0$, luego, tomando $q = 0$ y $r = 12 - 20 \cdot 0 = 12$, tendremos que

   $$12 = 20 \cdot 0 + 12, \quad con \;\; 0 \leqslant 12 < 20$$

3. Sean $a = -37$ y $b = 7$.
   El mayor múltiplo de 7 menor o igual que -37 es $7 \cdot (-6)$, por lo que tomando $q = -6$ y $r = -37 - 7(-6) = 5$, se tiene

   $$-37 = 7 \cdot (-6) + 5, \quad con \;\; 0 \leqslant 5 < 7$$

Corolario 1. *Si a y b son enteros, con $b \neq 0$, entonces existen dos enteros q y r, únicos, tales que $a = bq + r$, donde $0 \leq r < |b|$.*

Demostración. Si $b > 0$, entonces se cumplen las hipótesis del teorema anterior, lo que verifica el corolario.
Si $b < 0$, entonces $-b > 0$ y aplicando el teorema anterior, existirán dos enteros q_1 y r, únicos, tales que $a = (-b)q_1 + r$, con $0 \leq r < -b$
de aquí que

$$a = b(-q_1) + r, \ con \ 0 \leq r < -b = |b|$$

tomando $q = -q_1$, tendremos que

$$a = bq + r, \ con \ 0 \leq r < |b|$$

siendo q y r únicos, ya que q_1 y r lo eran. □

Ejemplo: Sean $a = 59$ y $b = -6$.
El mayor múltiplo de -6 menor o igual que 59 es $(-6)(-9)$. Tomando $q = -9$ y $r = 59 - (-6)(-9) = 59 - 54 = 5$, tenemos que:

$$59 = (-6)(-9) + 5, \quad con \ \ 0 \leqslant 5 < |-6| = 6$$

2.3. Máximo común divisor

Vamos ahora a prestar atención a los divisores comunes de dos números enteros.

Definición. *Dados dos números enteros a y b, diremos que el entero $d \neq 0$, es un* ***divisor común****, si los divide a ambos.*

Ejemplo: $2|6$ y $2|14$, por tanto, 2 es un divisor común de 6 y 14.

Ejemplo: $-5|10$ y $-5|25$, luego, -5 es un divisor común de 10 y 25.

Definición. *Sean a y b números enteros. Diremos que d es el* ***máximo común divisor*** *de a y b, si d es el entero positivo más grande tal que $d|a$ y $d|b$. Se denota (a, b). Si $a = b = 0$, se define $(a, b) = 0$.*

Ejemplo: El máximo común divisor de 15 y 35 es 5, ya que 5|15 y 5|35.

Ejemplo: De manera similar,

$$(356, 1090) = 2$$

Una manera de verificar esto, es desde luego, enlistar todos los divisores positivos de 356 y 1090.

$$Divisores\ de\ 356 = \{1, 2, 4, 89, 356\}$$
$$Divisores\ de\ 1090 = \{1, 2, 5, 10, 109, 218, 545, 1090\}$$

y corroborar que, en efecto, 2 es el divisor común más grande.

2.3.1. Propiedades

Sean a y b dos números enteros distintos de cero, entonces:

I) $(a, 0) = |a|$

II) $(a, b) = (|a|, |b|)$

Demostración. I) $(a, 0) = |a|$, *para todo* $a \in \mathbb{Z} \setminus \{0\}$.
Efectivamente, el máximo común divisor de a y 0 es, por definición, el máximo del conjunto de los divisores comunes de a y de 0, ordenado de acuerdo a la relación de divisibilidad. Dado que todos los números enteros son divisores de cero, dicho conjunto estará formado sólo por los divisores de a y $|a|$ es el mayor divisor de a por lo que

$$(a, 0) = |a|$$

II) $(a, b) = (|a|, |b|)$.
Sea d un divisor de a y de b. Como a y b son distintos de cero, pueden ocurrir 4 casos:
$a < 0\ y\ b > 0$. Entonces:

$$d|a\ y\ d|b$$

lo que implica

$$d| - a\ y\ d|b$$

Así,

$$d||a| \ y \ d||b|$$

Análogamente los demás casos.
En cualquier caso, el conjunto de los divisores comunes de a y de b se corresponde con el de los divisores comunes de $|a|$ y de $|b|$, por lo que el máximo común divisor el mismo,

$$(a, b) = (|a|, |b|)$$

□

Nótese que si a y b son enteros positivos, esto es lo mismo que decir:

$$(-a, b) = (a, -b) = (-a, -b) = (a, b)$$

Se demostrará ahora que todo par de enteros a y b poseen un divisor común que puede expresarse como combinación lineal de a y b. Para arribar a este importante resultado, enunciamos en primera instancia el Principio de Buen Orden.

El Principio de Buen Orden

Sea $A \subset \mathbb{N}$ con $A \neq \emptyset$, entonces A tiene elemento mínimo.

Teorema 2. *Sea $d = (a, b)$, entonces existen $x, y \in \mathbb{Z}$ tales que $d = ax + by$.*

Demostración. Considere el conjunto:

$$S = \{ax + by | a, b \in \mathbb{Z}\}$$

y $m = ax_0 + by_0$, donde m es el menor elemento natural de S. Supongamos que $m \nmid a$, entonces:

$$a = mq + r, \quad 0 < r < m$$

Es decir,

$$r = a - mq = a - (ax_0 + by_0) \cdot q$$
$$r = a - ax_0q - by_0q$$
$$r = a(1 - x_0q) + b(-y_0q)$$

Esto significa que $r \in S$, lo cual es absurdo pues $r > 0$ y $r < m$ y m es el menor elemento de S. Por tanto, $m|a$. Análogamente se prueba que $m|b$. Así, m es divisor común de a y b. Ahora sólo resta mostrar que $m = d$.

Si $d = (a, b)$ entonces

$$d|a \Rightarrow a = d \cdot q_1$$
$$d|b \Rightarrow b = d \cdot q_2$$
$$m = ax_0 + by_0$$
$$m = (dq_1)x_0 + (dq_2)y_0$$
$$m = d(q_1x_0 + q_2y_0)$$
$$d|m \Longrightarrow d \leqslant m \Longrightarrow d = m$$

□

Ejemplos:

1. $(21, 15) = 3$. Veamos que 3 puede escribirse como combinación lineal de 21 y 15. Buscamos enteros u y v tales que $21u + 15v = 3$. Sean $a = 21$ y $b = 15$),

$$\begin{array}{rcll} a & = & b \cdot 1 + 6 & 6 = a - b \\ b & = & (a - b) \cdot 2 + 3 & 3 = -2a + 3b \end{array}$$

   Es decir, $u = -2$ y $v = 3$, que en efecto cumplen:

$$3 = 21(-2) + 15(3)$$

2. El máximo común divisor de 2024 y 748 es 44. Del mismo modo, buscaremos u y v enteros tales que $2024u + 748v = 44$. Tomemos $a = 2024$ y $b = 748$. Entonces,

$$\begin{array}{rcll} a & = & b \cdot 2 + 528 & 528 = a - 2b \\ b & = & (a - 2b) + 220 & 220 = -a + 3b \\ a - 2b & = & (-a + 3b) \cdot 2 + 88 & 88 = 3a - 8b \\ -a + 3b & = & (3a - 8b) \cdot 2 + 44 & 44 = -7a + 19b \end{array}$$

   Así, $u = -7$ y $v = 19$, que verifican:

$$44 = 2024(-7) + 748(19)$$

El método utilizado para encontrar las combinaciones lineales de los ejemplos anteriores, es el mismo que desde la antigüedad permite calcular el máximo común divisor de dos números. Se trata del Algoritmo de Euclides, que analizamos ahora detalladamente.

2.4. El algoritmo de Euclides

Teorema 3. *(Algoritmo de Euclides).*
Sean a y b enteros positivos con $a \geqslant b$. El siguiente algoritmo calcula (a, b) en un número finito de pasos.

(1) Sean $r_0 = a$ y $r_1 = b$.

(2) Hagamos $i = 1$.

(3) Divide r_{i-1} por r_i para obtener un cociente q_i y un residuo r_{i+1},

$$r_{i-1} = r_i \cdot q_i + r_{i+1} \quad con \quad 0 \leqslant r_{i+1} < r_i.$$

(4) Si el residuo $r_{i+1} = 0$, entonces $r_i = (a, b)$ y el algoritmo finaliza.

(5) En otro caso, si $r_{i+1} > 0$, hágase $i = i + 1$ y repítase el Paso 3.

Demostración. El Algoritmo de Euclides consiste de una secuencia de divisiones con residuo como se muestra a continuación:

$$a = b \cdot q_1 + r_2 \quad con \quad 0 \leqslant r_2 < b.$$

$$b = r_2 \cdot q_2 + r_3 \quad con \quad 0 \leqslant r_3 < r_2.$$

$$r_2 = r_3 \cdot q_3 + r_4 \quad con \quad 0 \leqslant r_4 < r_3.$$

$$r_3 = r_4 \cdot q_4 + r_5 \quad con \quad 0 \leqslant r_5 < r_4.$$

$$\vdots \quad \vdots \qquad \vdots$$

$$r_{t-2} = r_{t-1} \cdot q_{t-1} + r_t \quad con \quad 0 \leqslant r_t < r_{t-1}.$$

$$r_{t-1} = r_t \cdot q_t$$

$$Entonces, \quad r_t = (a, b)$$

Los valores r_i son estrictamente decrecientes y tan pronto como llegan a valer 0, el algoritmo termina, esto prueba que efectivamente, el algoritmo tiene un número finito de pasos.
Además, en cada iteración del Paso 3, tenemos una ecuación de la forma:

$$r_{i-1} = r_i \cdot q_i + r_{i+1}$$

Esta ecuación implica que cualquier divisor de r_{i-1} y r_i es también un divisor de r_{i+1} y similarmente implica que cualquier divisor común de r_i y de r_{i+1} es también un divisor de r_{i-1}. De aquí que:

$$(r_{i-1}, r_i) = (r_i, r_{i+1}) \ \ para \ \ todo \ \ i = 1, 2, 3, ... \tag{2.1}$$

Sin embargo, como podemos notar, finalmente obtenemos un r_i que es 0, digamos $r_{r+1} = 0$. Entonces, $r_{t-1} = r_t \cdot q_t$, así:

$$(r_{t-1}, r_t) = (r_t \cdot q_t, r_t) = r_t$$

Pero la ecuación (2.1) dice que esto es igual a (r_0, r_1), i.e., (a, b), esto completa la prueba de que el último residuo distinto de 0 en el Algoritmo de Euclides es igual al máximo común divisor de a y de b. □

Ejemplos

1. Se había mostrado que $(356, 1090) = 2$; enlistando sus divisores comunes. Veámoslo ahora formalmente con el Algoritmo de Euclides:

$$\begin{array}{rcrcll} 1090 & = & 356 \cdot 3 & + & 22 & \\ 356 & = & 22 \cdot 16 & + & 4 & \\ 22 & = & 4 \cdot 5 & + & 2 & \longleftarrow \quad mcd = 2 \\ 4 & = & 2 \cdot 2 & + & 0 & \end{array}$$

2. Aplicando el Algoritmo de Euclides a 400 y 2048, vemos que el máximo común divisor es 16.

$$\begin{array}{rcrcll} 2048 & = & 400 \cdot 5 & + & 48 & \\ 400 & = & 48 \cdot 8 & + & 16 & \longleftarrow \quad mcd = 16 \\ 48 & = & 16 \cdot 3 & + & 0 & \end{array}$$

2.5. Números primos y compuestos

Definición. *Sea $p \in \mathbb{Z}^+$ distinto de 1, se dice que p es primo si sus únicos divisores son 1 y él mismo. Si p no es primo se denomina compuesto.*

Teorema 4. *Existen infinitos números primos.*

Demostración. Supongamos que el conjunto de los números primos es finito y sea $P = \{p_1, p_2, ..., p_n\}$ dicho conjunto.

Consideremos ahora un número q, tal que $q = p_1 p_2 ... p_n + 1$.

Si q es primo, $q > p_i$, con $i = 1, 2, ..., n$ y entonces sería un nuevo primo, contradicción. Si q no es primo, entonces es compuesto, es decir, tiene un divisor primo p_j. Como $p_j | q$ y $p_j | (p_1 p_2 ... p_n) \implies p_j | 1$, lo cual es imposible pues $p_j > 1$. □

Ejemplo: $2, 3, 31, 97, 211, ...$ son todos números primos.

En la criptografía moderna, varios algoritmos de cifrado hacen uso de los números primos. En particular, el algoritmo RSA, que se describirá más adelante, basa su seguridad y fortaleza en la dificultad de factorizar un número compuesto muy grande, producto a su vez, de dos números primos.

Definición. *Sean $a, b \in \mathbb{Z}$, se dice que a y b son primos relativos ó coprimos si $(a, b) = 1$.*

Ejemplos:

1. 7 y 16 son primos relativos, pues $(7, 16) = 1$.
2. 12 y 40 no son primos relativos, ya que $(12, 40) = 4 \neq 1$.

2.5.1. Lema de Euclides

Lema 1. *(Lema de Euclides).*
Sean m, n, p números enteros con p primo, tales que $p|mn$. Entonces, $p|m$ o $p|n$.

Demostración. Si $p|mn$ entonces existe $q \in \mathbb{Z}$ tal que $mn = pq$.
La demostración se hará por casos:

1. Si $p|m$, no hay nada que probar.
2. Si $p \nmid m$ entonces, $(p, m) = 1$, es decir, existen s, t enteros, tales que: $1 = ps + mt$. Luego,

$$n = psn + mtn$$
$$n = psn + tpq$$
$$n = p(sn + tq)$$

Es decir,

$p|n$.

□

Ejemplo:
Sean $m = 28$, $n = 6$ y $p = 7$ primo. Obtenemos $m \cdot n = 28 \cdot 6 = 168$. Vemos que $7|168$ y consecuentemente $7|28$ aunque $7 \nmid 6$.

2.5.2. Teorema Fundamental de la Aritmética

Como una aplicación del *Lema de Euclides*, probaremos que cada entero positivo tiene, esencialmente, una única factorización como producto de primos.

Teorema 5. *(Teorema Fundamental de la Aritmética).*
Sea $a \geqslant 2$. Entonces a puede ser factorizado como un único producto de números primos

$$a = p_1 p_2 ... p_n.$$

Demostración. Demostraremos existencia y unicidad.
Existencia: Sea

$$A = \{n \in \mathbb{N} : n \geqslant 2 \text{ y } n \text{ no es producto de primos}\}$$

Bastará con probar que $A = \emptyset$.
Supongamos que $A \neq \emptyset$. Por el Principio de Buen Orden, A tiene elemento

mínimo m, es decir, m no es producto de primos, en particular, m no es primo. Lo que implica que existen $a, b > 1$ tales que $m = a \cdot b$, así, $a, b < m$. Es decir, $a, b \notin A$, en consecuencia, a, b son producto de primos. Esto nos lleva a que m es producto de primos, hecho que contradice nuestra hipótesis. Por tanto, $A = \emptyset$.

Unicidad:
Supongamos que a tiene dos factorizaciones como producto de números primos,

$$a = p_1 p_2 ... p_s = q_1 q_2 ... q_t.$$

donde los p_i y los q_j son todos primos, no necesariamente distintos y s no necesariamente igual a t.
Dado que $p_1|a$, vemos que p_1 divide al producto $q_1 q_2 ... q_t$. Así, por el Lema de Euclides, podemos encontrar p_1 que divide a algún q_j. Reajustando el orden de los q_j si es necesario, podemos suponer que $p_1|q_1$. Pero p_1 y q_1 son ambos primos, por lo que tenemos $p_1 = q_1$. Esto nos permite cancelar ambos factores,

$$p_2 p_3 ... p_s = q_2 q_3 ... q_t.$$

Repitiendo este proceso s veces, obtenemos al final:

$$1 = q_{t-s} q_{t-s+1} ... q_t.$$

Se sigue inmediatamente de aquí que $t = s$ y que las factorizaciones originales de a son idénticas, reajustando tan sólo el orden de los factores. □

Ejemplos:

1. Consideremos 45. Su factorización es: $45 = 3 \cdot 3 \cdot 5$
2. Si tomamos ahora 196, lo podemos factorizar como: $196 = 2 \cdot 2 \cdot 7 \cdot 7$

2.6. Congruencias

Definición. *Sea m un entero positivo y a, b dos números enteros. Diremos que a y b son congruentes módulo m si $m|(a - b)$. Denotaremos esto*

$$a \equiv b \ (mód\ m)$$

Teorema 6. *Sea m cualquier número entero positivo. Se cumple:*

a) Cualquier número entero es congruente módulo m exactamente con uno de los enteros $0, 1, ..., m-1$.

b) Dos números enteros son congruentes entre si módulo m si y sólo si ambos dan el mismo residuo al dividirlos por m.

Demostración. a) Se probará que si a es un entero arbitrario, entonces es congruente módulo m exactamente con uno de los enteros $0, 1, ..., m-1$. En efecto,

$$\begin{aligned}
\text{a}\in\mathbb{Z}\text{ y m}\in\mathbb{Z}^+ &\implies \text{Existen q y r enteros y únicos : a = mq + r, con } 0\leq\text{r}<\text{ m}\\
&\iff \text{a}-\text{r}=\text{mq, con } 0\leq \text{r} < \text{m}\\
&\iff \text{m}|\text{a}-\text{r, con } 0\leq \text{r} < \text{m}\\
&\iff \text{a}\equiv \text{r}\ \ (\text{mód}\ \ \text{m}), \text{ con } 0\leq \text{r} < \text{m}\\
&\iff \begin{cases}
\text{a}\equiv 0\ (\text{mód m})\\
\text{ó}\\
\text{a}\equiv 1\ (\text{mód m})\\
\text{ó}\\
\text{a}\equiv 2\ (\text{mód m})\\
\vdots\\
\text{ó}\\
\text{a}\equiv \text{m}-1\ (\text{mód m})
\end{cases}
\end{aligned}$$

b) Sean a y b dos enteros arbitrarios.

Si $a \equiv b\,(mód\, m)$, entonces

$$a - b = mq,\ con\ q \in \mathbb{Z}$$

Por otro lado, por el Algoritmo de la División, podemos encontrar q_1, q_2, r_1 y r_2 enteros, tales que:

$$a = mq_1 + r_1$$

$$y$$

$$a = mq_2 + r_2$$

de dónde

$$a - b = m(q_1 - q_2) + r_1 - r_2$$

Tenemos que:

$$a - b = mq$$

$$y$$

$$a - b = m(q_1 - q_2) + r_1 - r_2$$

Ahora bien, el residuo de dividir $a - b$ entre m es único, así que:

$$r_1 - r_2 = 0$$

es decir,

$$r_1 = r_2$$

O sea que al dividir a y b entre m, obtenemos el mismo residuo.

Supongamos ahora que a y b, ambos, dan el mismo residuo al dividirlos por m, es decir, existen q_1, q_2 y r enteros, tales que:

$$a = mq_1 + r \ y \ b = mq_2 + r$$

Si restamos miembro a miembro, obtenemos:

$$a - b = m(q_1 - q_2) \ \ si \ y \ sólo \ si,$$
$$m|a - b \ \ si \ y \ sólo \ si,$$
$$a \equiv b \, (mód \, m).$$

□

Ejemplos:

1. $27 \equiv 7 \ (mód \ 5)$, ya que 5 divide a $20 = 27 - 7$.
2. $121 \equiv 31 \ (mód \ 15)$, dado que 15 divide a $90 = 121 - 31$.
3. $200 \not\equiv 72 \ (mód \ 3)$, pues claramente 3 no divide a $128 = 200 - 72$.

2.7. Función ϕ de Euler

Definición. *Para $n > 1$ definimos la función $\phi(n)$ como el número de enteros positivos menores o iguales que n que son primos relativos con n. Ésta función es la llamada función ϕ de Euler.*

Notemos que $\phi(1) = 1$ pues $(1, 1) = 1$. Si $n > 1$ entonces $(n, n) = n \neq 1$ y por lo tanto $\phi(n) \leqslant n - 1$.

Si n es un número primo, entonces cada entero menor que n es primo relativo con n, así que $\phi(n) = n - 1$. Por otro lado, si $n > 1$ y $\phi(n) = n - 1$ entonces n tiene que ser primo pues de lo contrario n tendría un divisor d tal que $1 < d < n$ y entonces $\phi(n) \leqslant n - 2$. Esto demuestra la siguiente:

Proposición 1. *$n > 1$ es primo sí y sólo sí, $\phi(n) = n - 1$.*

Ejemplos:

1. $\phi(2) = 1$
2. $\phi(4) = 2$
3. $\phi(30) = 8$

2.8. Teorema de Euler

Teorema 7. *(Teorema de Euler).*
Sean n un entero positivo y a un número primo relativo con n. Entonces: $a^{\phi(n)} \equiv 1$ (mód n).

Demostración. Sean $P = \{x \in \mathbb{Z}, x < n \text{ tal que } (x, n) = 1\}$ y $Q = \{a \cdot x, \text{ con } x \in P\}$
Los elementos de Q son congruentes con los elementos de P (en diferente orden). Hagamos ahora:

$$u = \Pi x, \text{ con } x \in P$$

$$v = \Pi y, \ con \ y \in Q$$

Los números u y v son congruentes pues sus factores lo son: $u \equiv v \ (mód \ n)$. El entero v es igual a u multiplicado por $a^{\phi(n)}$, es decir, $v \equiv u \cdot a^{\phi(n)}$. Así, se tiene que:

$$u \equiv u \cdot a^{\phi(n)} \ (mód \ n)$$

De dónde se concluye,

$$1 \equiv a^{\phi(n)} \ (mód \ n)$$

□

Ejemplo:

1. Encontrar el residuo al dividir 3^{50} entre 14. Aquí, $a = 3$ y $n = 14$. Claramente, $50 = 6 \cdot 8 + 2$, por lo que:

   $$3^{50} = 3^{6 \cdot 8 + 2} = (3^6)^8 \cdot 3^2 \equiv 1^8 \cdot 3^2 \ (mód \ 14)$$

   De donde:

   $$3^{50} \equiv 9 \ (mód \ 14)$$

   Por tanto, el residuo solicitado es 9.

Teorema 8. *(Pequeño Teorema de Fermat).*
Sea $a \in \mathbb{N}$ y p un número primo, entonces $a^p \equiv a \, (mód \, p)$.

Demostración. Se puede demostrar el Pequeño Teorema de Fermat como un caso particular del Teorema de Euler. Si a es múltiplo de p, entonces ambos lados de la congruencia son 0. Consideremos entonces el caso en que a no es múltiplo de p. Entonces a y p no tienen factores en común, así que a tiene inverso módulo p, y en la sucesión $a, 2a, ..., (p-1)a$ no hay dos números congruentes entre si y son congruentes, en un cierto orden, a los residuos $1, 2, ..., p-1$. De esta manera su producto es el mismo módulo p:

$$a \cdot 2a \cdot ... \cdot (p-1)! a \equiv (p-1)! \ (mód \ p)$$

Cancelando $(p-1)!$ de ambos lados de la congruencia, obtenemos:

$$a^{p-1} \equiv 1 \, (mód \, p)$$

Multiplicando a en ambos lados de la relación de equivalencia se tiene:

$$a^p \equiv a \,(mód\, p)$$

□

2.9. Teorema chino del residuo

Un sistema de dos o más congruencias lineales no tiene necesariamente solución, aunque cada una de las congruencias individuales la tenga. Por ejemplo, no existe ningún x que satisfaga simultáneamente $x \equiv\ 1\ (mód\ 2)$ y $x \equiv\ 0\ (mód\ 4)$, a pesar de que cada una de ellas, separadamente, tiene solución. En este ejemplo, los módulos 2 y 4 no son primos entre sí. Se demostrará enseguida que todo sistema de dos o más congruencias lineales que, separadamente admiten solución única, puede resolverse simultáneamente también si los módulos son primos relativos entre sí, dos a dos.

El Teorema Chino del Residuo describe las soluciones para un sistema de congruencias lineales simultáneas. Sean:

$$x \equiv a\ (mód\ m) \quad y \quad x \equiv b\ (mód\ n)$$

con (m,n)=1, en este caso el teorema establece que existe una única solución módulo mn.

Teorema 9. *(Teorema Chino del Residuo)*
Sea $m_1, m_2, ...m_k$ un conjunto de números primos relativos entre sí dos a dos, es decir,

$$(m_i, m_j) = 1 \ \ para\ todo\ i \neq j$$

Sean $a_1, a_2, ..., a_k$ enteros arbitrarios. El sistema de congruencias simultáneas:

$$x \equiv a_1\ (mód\ m_1),\ \ x \equiv a_2\ \ (mód\ m_2), ..., x \equiv a_k\ (mód\ m_k),$$

tiene una única solución. Además, si $x = w$ y $x = w'$ son ambas soluciones, entonces,

$$w \equiv w'\ \ (mód\ \ m_1 m_2 ... m_k)$$

Demostración. Sea $M = m_1 m_2 \cdots m_r$ y sea $b_k = M/m_k$. Entonces $(b_k, m_k) = 1$, por tanto, cada M_k admite un inverso único c_k módulo m_k. Sea ahora:

$$x = a_1 b_1 c_1 + a_2 b_2 c_2 + ... + a_r b_r c_r$$

Consideremos cada uno de los términos de esta suma módulo m_k. Dado que $M_i \equiv 0 \ (mód \ m_k)$ si $i \neq k$ tenemos:

$$x \equiv a_k b_k c_k \equiv a_k \ (mód \ m_k)$$

Por tanto, x satisface cada una de las congruencias del sistema. Se prueba además que el sistema posee una única solución módulo M. En efecto, si x e y, son dos soluciones del sistema tenemos $x \equiv y \ (mód \ m_k)$ para cada k y, dado que los m_k son primos relativos entre si dos a dos, tendremos también que $x \equiv y \ (mód \ M)$, lo que termina la demostración. □

Ejemplo:
Considérese el sistema de congruencias:

$$x \equiv 2 \ (mód \ 4)$$
$$x \equiv 3 \ (mód \ 7)$$
$$x \equiv 5 \ (mód \ 9)$$

que resolveremos utilizando el Teorema Chino del Residuo.

Para este caso, $M = 4 \cdot 7 \cdot 9 = 252$. Calculemos ahora los b_k:

$$b_1 = M/m_1 = 252/4 = 63$$
$$b_2 = M/m_2 = 252/7 = 36$$
$$b_3 = M/m_3 = 252/9 = 28$$

Se tiene por tanto que:

$$x_0 = 2 \cdot 63 \cdot c_1 + 3 \cdot 36 \cdot c_2 + 5 \cdot 28 \cdot c_3$$

De donde:

$$63 \cdot c_1 \equiv 1 \ (mód \ 4) \quad \Longrightarrow \ c_1 = 3$$
$$36 \cdot c_2 \equiv 1 \ (mód \ 7) \quad \Longrightarrow \ c_2 = 1$$

$$28 \cdot c_3 \equiv 1 \ (mód\ 9) \qquad \Longrightarrow \ c_3 = 1$$

Así,

$$x_0 = 2 \cdot 63 \cdot 3 + 3 \cdot 36 \cdot 1 + 5 \cdot 28 \cdot 1 = 656$$

Por lo que el conjunto solución del sistema de congruencias tiene la forma:

$$x = 152 + 252 \cdot y$$

.

Capítulo 3

El algoritmo RSA

Este capítulo muestra un estudio detallado del algoritmo de cifrado asimétrico más utilizado en la actualidad: el algoritmo RSA. ¿Cómo se cifra? ¿Cómo se descifra? ¿Qué consideraciones importantes deben hacerse al momento de cifrar? Estas y otras cuestiones encuentran aquí su respuesta.

Varios meses después de que Diffie y Hellman mostraran el intercambio de claves que lleva su nombre, los investigadores Ron Rivest, Adi Shamir y Leonard Adleman, propusieron un sistema de cifra asimétrico. El algoritmo llegó a patentarse con las iniciales de los apellidos de los autores: RSA (Rivest, Shamir, Adleman). La fortaleza de RSA se basa (en teoría), en la dificultad de factorizar un número compuesto muy grande, producto a su vez de dos números primos grandes.

Aunque los inicios de este algoritmo no fueron fáciles, su seguridad y versatilidad acabaron imponiéndose, lo que derivó en un estándar: PKCS No. 1, RSA Cryptography Standard, ampliamente utilizado en la actualidad en Internet.

Según el gobierno británico, el mismo algoritmo de cifra asimétrico basado en la dificultad de factorizar números grandes fue descubierto mucho antes. En 1969, el Government Communications Headquarters (GCHQ) en Inglaterra, trabajó en la distribución de claves a través de una cifra no simétrica, llegando a la misma conclusión que los inventores de RSA. El proyecto lo lideró el matemático Clifford Cocks. Sin embargo, el trabajo fue considerado como secreto por el gobierno inglés, por lo que no se publicó ni se patentó como

invento. El algoritmo RSA es elegantemente sencillo y puede expresarse con las siguientes ecuaciones:

$$C = M^e \mod n \qquad (Para\ cifrar)$$
$$M = C^d \mod n \qquad (Para\ descifrar)$$

M es el mensaje, C el texto cifrado (criptograma), e la clave pública del destino, d la clave privada del destino y n es el cuerpo de cifra o módulo público del destino. *Mód* es una operación clásica en aritmética modular y devuelve el residuo que resulta al dividir los dos valores que se le proporcionan.

Bien configurados, los números e, d y n, permiten recuperar un mensaje M de un mensaje cifrado C. Estos números, e, d y n son las claves utilizadas en el algoritmo.

3.1. Cifrado con RSA

RSA es un algoritmo asimétrico, por lo que utiliza dos claves: una clave pública, formada por los números e y n; y una clave privada formada por los números d y n. Conviene desde ahora mencionar que los algoritmos asimétricos son entre cien y mil veces más lentos que su contraparte simétrica; razón por la que se recomienda su uso en un contexto exclusivo: el intercambio de claves (típicamente claves de sesión) o el firmado digital (cifrado de claves hash). Se ilustrará el proceso RSA mediante un ejemplo, cifrando paso a paso el número 123456.

1. *Se eligen dos números primos p y q.* En la actualidad, dichos números deben ser del orden de 1024 bits, por lo menos. Para nuestro ejemplo, sean $p = 859$ y $q = 1013$.

2. *Se calcula el módulo n, es decir, el producto de p y q.* En este caso:

$$n = p \cdot q = 859 \cdot 1013 = 870167$$

3. *Calculamos el valor de* $\phi(n)$. Dado que p y q son primos, se tiene:

$$\phi(n) = (p-1)\cdot(q-1) = \phi(870167) = 858 \cdot 1012 = 868296$$

4. *Se elige un valor de e en el intervalo* $1 < e < \phi(n)$. Para asegurarse que exista el inverso multiplicativo y por ende, la clave privada d, se debe cumplir que $(e, \phi(n)) = 1$. Para nuestro ejemplo tomaremos $e = 97$, verificando que efectivamente $(97, 868296) = 1$, con el Algoritmo de Euclides.

$$\begin{array}{rcrcll} 868296 & = & 97 \cdot 8951 & + & 49 & \\ 97 & = & 49 \cdot 1 & + & 48 & \\ 49 & = & 48 \cdot 1 & + & 1 & \longleftarrow \quad mcd = 1 \\ 48 & = & 1 \cdot 48 & & & \end{array}$$

5. *Se calcula d, el inverso multiplicativo de e módulo n.*
Como $(e, \phi(n)) = 1$, expresaremos 1 como combinación lineal, es decir, se buscarán u y v enteros tales que:

$$e \cdot u + \phi(n) \cdot v = 1 \tag{3.1}$$

Hagamos $a = 868296$ y $b = 97$. Entonces,

$$\begin{array}{rcl@{\qquad\qquad}rcl} a & = & b \cdot 8951 + 49 & 49 & = & a - 8951b \\ b & = & (a - 8951b) \cdot 1 + 48 & 48 & = & -a + 8952b \\ a - 8951b & = & (-a + 8952b) \cdot 1 + 1 & 1 & = & 2a - 17903b \end{array}$$

O sea que $u = -17903$ y $v = 2$. Despejando $e \cdot u$ de (3.1), tenemos:

$$e \cdot u = -\phi(n) \cdot v + 1 \tag{3.2}$$

y como $\phi(n) > 1$, el residuo es 1, luego,

$$e \cdot u \ \mathit{mód}\ \phi(n) = 1 \tag{3.3}$$

Nótese que u es casi el valor buscado, pero cuando no satisface $0 < u < \phi(n)$, es posible convertir u en el valor adecuado haciendo:

$$d = u \ \mathit{mód}\ \phi(n) \tag{3.4}$$

Así, $d = -17903\ (\mathit{mód}\ 868296) = 850393$.

6. Se cifra el "mensaje". $C = M^e \ (mód\ n)$. Para nuestro ejemplo:

$$C = 123456^{97} \ (mód\ 870167)$$

Recordemos que los números involucrados en las claves RSA actuales son del orden de los 1024 bits por lo menos. El método tradicional para elevar a una potencia, digamos x^8, es el siguiente:

$$x \xrightarrow{CUAD} x^2 \xrightarrow{MUL} x^3 \xrightarrow{MUL} x^4 \xrightarrow{MUL} x^5 \xrightarrow{MUL} x^6 \xrightarrow{MUL} x^7 \xrightarrow{MUL} x^8$$

Es decir, se eleva al cuadrado la primera vez, y para calcular las potencias restantes, se hacen 6 multiplicaciones sucesivas. Considerando una potencia de 1024 bits, como es usual en RSA, realizar 2^{1024} multiplicaciones no es para nada conveniente. Existe otra forma de realizar la exponenciación de una manera más rápida. Si consideramos el mismo ejemplo mostrado anteriormente, podemos calcular x^8 del siguiente modo:

$$x \xrightarrow{CUAD} x^2 \xrightarrow{CUAD} x^4 \xrightarrow{CUAD} x^8$$

Sólo se requirieron 3 operaciones de elevar al cuadrado, que tiene más o menos la misma complejidad que una multiplicación. Este método recibe el nombre de *algoritmo de exponenciación rápida*. Sin embargo, está restringido a exponentes que son potencias de 2. ¿Cómo extender el método a exponentes arbitrarios? Convirtiendo el exponente a su equivalente en binario, donde cada bit es una potencia de 2. Veamos un ejemplo con un exponente arbitrario: x^{25}.

$$x \xrightarrow{CUAD} x^2 \xrightarrow{MUL} x^3 \xrightarrow{MUL} x^4 \ \ldots \ \xrightarrow{MUL} x^{25}$$

el enfoque tradicional requeriría una elevación al cuadrado y 23 multiplicaciones sucesivas.

Convertimos 25_{10} a su equivalente en base 2, 11001_2. Se tiene, por tanto:

$$x^{25} = x^{11001_2} = x^{b_4 b_3 b_2 b_1 b_0}$$

expresión en la que ya podemos utilizar el algoritmo de exponenciación rápida, que recorre los bits del exponente, iniciando en el de la izquierda, b_4, y terminando con el de la derecha, b_0. El algoritmo es el siguiente:

3.1.1. Algoritmo de Exponenciación Rápida.

Sea $A^B (mód\ n)$ la operación a realizar. Entonces:

Algoritmo 7. *Algoritmo de Exponenciación Rápida.*
ENTRADA: A, B, n.
SALIDA: $x = A^B$ *mód n.*
1. Representar B (de k bits) en binario: $b_{k-1}b_{k-2}...b_i...b_2b_1b_0$.
2. Hacer x=1
3. Para i = (k-1) .. 0 hacer
$x = x^2\ (mód\ n)$
Si $b_i = 1$ *entonces*
$x = x \cdot A\ (mód\ n)$
Fin-Si
Fin-Para
4. Devolver (x).
5. Fin.

Aplicaremos ahora el algoritmo para cifrar $C = 123456^{97}\ (mód\ 870167)$.

$i = 6$	$b_6 = 1$	$x = 1^2 \cdot 123456\ (mód\ 870167)$	$x = 123456$
$i = 5$	$b_5 = 1$	$x = 123456^2 \cdot 123456\ (mód\ 870167)$	$x = 506697$
$i = 4$	$b_4 = 0$	$x = 506697^2\ (mód\ 870167)$	$x = 816793$
$i = 3$	$b_3 = 0$	$x = 816793^2\ (mód\ 870167)$	$x = 727285$
$i = 2$	$b_2 = 0$	$x = 727285^2\ (mód\ 870167)$	$x = 277937$
$i = 1$	$b_1 = 0$	$x = 277937^2\ (mód\ 870167)$	$x = 770711$
$i = 0$	$b_0 = 1$	$x = 770711^2 \cdot 123456\ (mód\ 870167)$	$x = 688983$

Con esto concluye el proceso de cifrado, del cual resulta:

$$C = 123456^{97}\ (mód\ 870167) = 688983.$$

3.2. Descifrado con RSA

El descifrado es la operación más costosa en términos de tiempo de cómputo, ya que si se consideran los tamaños de clave actuales, el servidor deberá resolver algo como:

$$C^{dSv}\ (mód\ n_{Sv}) = 2048\ bits^{2048\ bits}\ (mód\ 2048\ bits)$$

En general, si $N \equiv\ a\ (mód\ n)$ y $p|n$, entonces $N \equiv a\ mód\ p$. Análogamente tenemos que $N \equiv a\ mód\ q$.

Así, en lugar de calcular $C^d\ mód\ n$, resolvemos:

$$X \equiv C^d\ mód\ p$$
$$X \equiv C^d\ mód\ q$$

Como p y q son primos, podemos utilizar el Pequeño Teorema de Fermat:

$$C^{p-1} \equiv 1\ mód\ p$$
$$C^{q-1} \equiv 1\ mód\ q$$

De modo que podemos calcular C^{p-1}, C^{q-1} en lugar de una potencia mucho más grande.

Retomemos ahora nuestro ejemplo. Debemos *descifrar* 688983, utilizando para ello la ecuación:

$$M = 688983^{850393}\ (mód\ 870167)$$

Dado que tanto $p = 859$, como $q = 1013$, dividen a $n = 870167$, podemos escribir la ecuación anterior como:

$$M = 688983^{850393}\ (mód\ 859)$$
$$M = 688983^{850393}\ (mód\ 1013)$$

Reducimos ahora las bases por sus respectivos módulos:

$$M = 65^{850393}\ (mód\ 859)$$
$$M = 143^{850393}\ (mód\ 1013)$$

Se reducen también los exponentes, aplicando el Pequeño Teorema de Fermat:

$$850393 = 858 \cdot 991 + 115$$
$$850393 = 1012 \cdot 840 + 313$$

Resolvamos para el primer exponente:
$M \equiv (65^{991})^{858} \cdot 65^{115} = (65^{991})^{858}$ *mód* $859 \cdot 65^{115}$ *mód* 859.
$M \equiv 1$ *mód* $859 \cdot 65^{115}$ *mód* $859 \equiv 65^{115}$ *mód* 859.

El cálculo para el segundo exponente es:
$M \equiv (143^{840})^{1012} \cdot 143^{313} = (143^{840})^{1012}$ *mód* $1013 \cdot 143^{313}$ *mód* 1013.
$M \equiv 1$ *mód* $1013 \cdot 143^{313}$ *mód* $1013 \equiv 143^{313}$ *mód* 1013.

Tenemos pues el sistema de congruencias:
$M \equiv 65^{115}$ *mód* 859.
$M \equiv 143^{313}$ *mód* 1013.

Es decir,
$M \equiv 619$ *mód* 859.
$M \equiv 883$ *mód* 1013.

Resolviendo el sistema anterior mediante el Teorema Chino del Residuo, encontramos:

$$619 \cdot 1013 \cdot 608 \equiv 619 \text{ } mód \text{ } 859$$

$$883 \cdot 859 \cdot 296 \equiv 883 \text{ } mód \text{ } 1013$$

Cuya solución es:

$$(619 \cdot 1013 \cdot 608 + 883 \cdot 859 \cdot 296) \text{ } mód \text{ } (859 \cdot 1013) = 123456$$

Se obtiene así el *mensaje descifrado* $M = 123456$, como se esperaba.

El ejemplo que hemos desarrollado ilustra muy a detalle los procesos para generar las claves RSA, así como el cifrado y el descifrado. Damos ahora los algoritmos respectivos.

3.2.1. Algoritmos RSA

.

Algoritmo 8. *Algoritmo para generar claves RSA.*
SALIDA: Clave pública RSA (e,n) y clave privada (d,n).
1. Elegir dos números primos p y q de por lo menos 512 bits.
2. Calcular $n = p \cdot q$ *y* $\phi(n) = (p-1)(q-1)$.
3. Elegir un entero e impar arbitrario con $1 < e < \phi(n)$ *y tal que cumpla* $(e, \phi(n)) = 1$.
4. Calcular el entero d que satisfaga $1 < d < \phi(n)$ *y* $e \cdot d \equiv 1$ *mód* $\phi(n)$.
5. Devolver (n, e, d).
6. Fin.

Algoritmo 9. *Algoritmo RSA para cifrar.*
ENTRADA: Clave pública RSA (n, e)*; mensaje a cifrar* $M \in [0, n-1]$.
SALIDA: Mensaje cifrado C.
1. Calcular $C = M^e$ *mód n.*
2. Devolver C.
3. Fin.

Algoritmo 10. *Algoritmo RSA para descifrar.*
ENTRADA: Clave privada RSA (n, d)*; mensaje cifrado C.*
SALIDA: Mensaje en claro M.
1. Calcular $M = C^d$ *mód n.*
2. Devolver M.
3. Fin.

3.3. Diseño y elección de claves RSA: valores de p, q y e.

Aunque al momento de elegir los valores p, q y e (indirectamente d), pudiera pensarse que no hay restricciones, la realidad es muy diferente. Damos a continuación algunos criterios a considerar.

3.3.1. Recomendaciones en la elección del valor e.

Se recomienda usar un valor de la clave pública e muy pequeño comparado con el módulo n. Hay varias razones para hacerlo así:

- La trampa $\phi(n)$ será igual a $(p-1)(q-1)$ siendo p y q primos de al menos 1024 bits, el tamaño de $\phi(n)$ será aproximadamente igual al de n, un poco menor pero con igual número de bits.

- Como la clave pública e y la clave privada d son inversas en el cuerpo $\phi(n)$, es decir, $d = inv[e, \phi(n)]$, entonces se cumple la siguiente relación $e \cdot d \; mod \; \phi(n) = 1$.

- Para que se cumpla esa igualdad, el producto $e \cdot d$ debe salir del cuerpo $\phi(n)$ al menos una vez para que la operación en ese módulo nos devuelva el valor 1.

- Dicho de otro modo, se cumplirá que $e \cdot d = k \cdot \phi(n) + 1$, con k = 1, 2, 3, 4, ...

- Para que ese producto salga al menos una vez del cuerpo $\phi(n)$, es decir, con $k = 1$, dado que la clave pública e tiene por ejemplo 17 bits, el valor de la clave privada d debería ser como mínimo mayor a 1007 bits, para el caso hipotético (y con probabilidad casi nula) que se cumpla la ecuación para $k = 1$.

- En la práctica, ese valor de $k = 1$ o un valor bajo de k, será muy poco probable y, por tanto, se puede esperar una clave privada d muy cercana o igual en bits al valor de n como sucede en la práctica.

- En otras palabras, será computacionalmente difícil adivinar el valor de la clave privada d, puesto que encontrar un número dentro de un cuerpo de 1024 bits significa un tiempo de cómputo inabordable, con una media de 2^{1023} intentos.

- Forzar entonces que la clave pública e sea un valor pequeño, menor de 20 bits dentro de un cuerpo de 1024 bits o mayor, garantiza que la clave privada d sea un valor muy grande y, por ende, muy segura, pues es computacionalmente intratable encontrarla por fuerza bruta.

Realmente, todas las claves RSA comerciales utilizan un valor único para e de 17 bits, el número 65537. Este valor relativamente pequeño de e obliga a que la clave privada d tenga un tamaño similar al módulo n y haga imprácticco un ataque por fuerza bruta. El valor 65537 es un número primo conocido como número 4 de Fermat o F_4. Es el valor que se observa en los certificados digitales (Ver *Figuras 3.1 y 3.2*), lo que será diferente en cada clave serán los primos p y q, y por lo tanto, el módulo n.

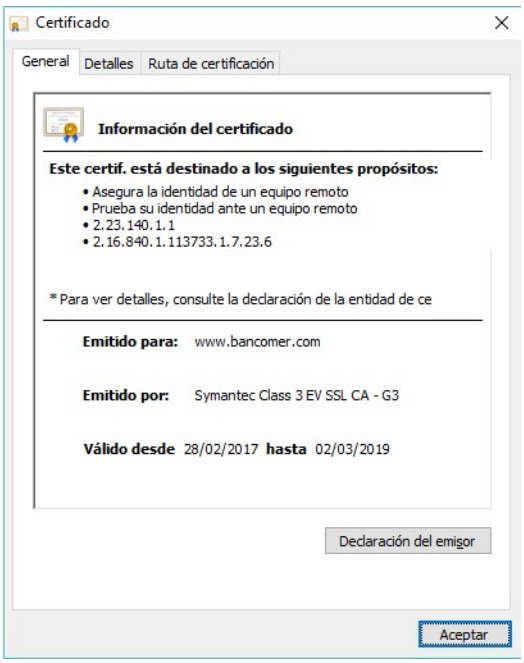

Figura 3.1: Carátula

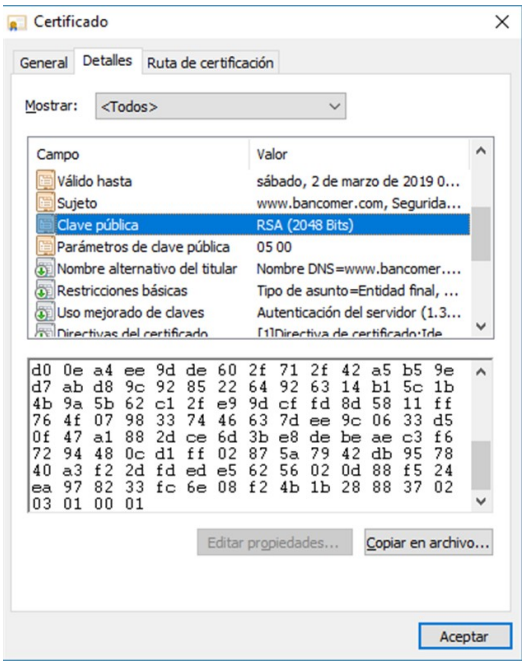

Figura 3.2: Clave pública

Además de las ventajas mencionadas, el número 4 de Fermat tiene una característica significativa que vale la pena destacar. La representación en base 2 de este valor sólo tiene dos bits iguales a 1: $65,537 = 10000000000000001_2 = 010001_{16}$. Este hecho tiene una gran utilidad en la eficiencia computacional.

Se recomienda el uso de los sistemas de cifra asimétricos o de clave pública exclusivamente para el intercambio de claves (cifrar información pequeña, por ejemplo, una clave simétrica de sesión de a lo más, algunas centenas de bits) o para la firma digital (cifrar un hash criptográfico también de algunas centenas

de bits a lo sumo). Ello debido a que estos sistemas son bastante lentos en relación a los sistemas de cifra simétrica.

3.4. Claves privadas y públicas parejas

Cuando se genera una clave RSA se utiliza como trampa la función $\phi(n)$ de Euler para calcular la clave privada d conociendo la clave pública e. Como $(e, \phi(n)) = 1$, se garantiza que el inverso d existe y que es el único inverso de e en ese cuerpo $\phi(n)$.

El cifrado se realiza con posterioridad en el cuerpo público n para que cualquiera pueda usarlo. Y en dicho cuerpo n ya no se satisface que el único inverso de la clave pública e sea la clave privada d. Existe al menos otro valor diferente a d que permite descifrar lo cifrado con la clave pública. A éstas claves se les llama *Claves Privadas Parejas*.

3.4.1. Claves privadas parejas

Se ha dicho que un sistema de cifra asimétrico tiene una única clave pública, y por ende, también una única clave privada. Para el criptosistema RSA, esto ha resultado ser falso. Un ejemplo ilustrará mejor.

Ejemplo:

Tomemos una clave RSA con $p = 37$, $q = 57$, $n = 2109$, $\phi(n) = (36) \cdot (56) = 2016$, $e = 13$, $d = 1861$. Cifraremos el número 1001 con la clave pública $e = 13$ y después lo descifraremos.

Cifrado:

$$C = 1001^{13} (mód\ 2109) = 1421$$

Descifrado:

$$M = 1421^{1861} (mód\ 2109) = 1001$$

Obtenemos, naturalmente, el mensaje original. Usemos ahora los números 349, 853 y 1357 como si fueran la clave privada d:

a) Descifrado con $d' = 349 \longrightarrow M = 1421^{349}\ (mód\ 2109) = 1001.$

b) Descifrado con $d' = 853 \longrightarrow M = 1421^{853}\ (mód\ 2109) = 1001.$

c) Descifrado con $d' = 1357 \longrightarrow M = 1421^{1357}\ (mód\ 2109) = 1001.$

Es decir, para el cuerpo de cifra $n = 2109$ con clave pública $e = 13$, los números 349, 853 y 1357 son claves privadas parejas d' que cumplen la misma función que la clave privada d.

Toda clave RSA tendrá como mínimo una clave privada pareja. El número de claves privadas parejas depende fuertemente de los primos p y q.

3.5. Mensajes no cifrables

A través de la historia, muchos algoritmos criptográficos han mostrado vulnerabilidades que deben cuidarse. Algunas de éstas vulnerabilidades son, por ejemplo, claves no recomendadas que o bien no cifran la información a proteger o lo hacen de una manera predecible. Por ejemplo, en el algoritmo simétrico DES, existían claves débiles o semidébiles, que no cumplían el principio de cifra única enunciado por Shannon, y daban soluciones conocidas como soluciones falsas. Algo similar ocurre en RSA, donde se presentan mensajes no cifrables, o mejor dicho, números no cifrables.

Supongamos que se quiere cifrar un número N ($N^{clave}\ mód\ n$), con una clave que puede ser la clave pública e si se desea confidencialidad, o la clave privada d si se requiere integridad y autenticidad. El número N tomará valores desde 0 hasta $(n-1)$, siendo n el módulo. De todos los valores posibles de N, hay algunos que aunque se cifren, se envían en claro, es decir, realmente no se cifran. Dos casos obvios de números que en realidad no se cifran, son el 0 y el 1, ya que:

$$0^{clave}\ mód\ n = 0$$
$$1^{clave}\ mód\ n = 1$$

Otro valor que se transmite en claro es $(n-1)$, pues:

$$(n-1)^{clave}\ mód\ n = (n-1)$$

El siguiente ejemplo ilustra lo anterior:

Consideremos la clave RSA con $n = 493$ (producto de $p = 17$ y $q = 29$) y $e = 11$, tenemos:

$$0^{11} \ mód \ 493 = 0$$
$$1^{11} \ mód \ 493 = 1$$
$$492^{11} \ mód \ 493 = 492$$

Además de estos tres números, en RSA se tendrán siempre otros 6 números que no se cifran. Para nuestro ejemplo:

$$86^{11} \ mód \ 493 = 86$$
$$203^{11} \ mód \ 493 = 203$$
$$204^{11} \ mód \ 493 = 204$$
$$289^{11} \ mód \ 493 = 289$$
$$290^{11} \ mód \ 493 = 290$$
$$407^{11} \ mód \ 493 = 407$$

¿Cómo podemos localizar estos números no cifrables? En realidad, encontrar los números no cifrables requiere un ataque de cifrado por fuerza bruta en el espacio de los primos p y q, para verificar los valores de X que arrojan las siguientes desigualdades:

$$X^e \ (mód \ p) = X \qquad para \ \ 1 < X < (p-1)$$
$$X^e \ (mód \ q) = X \qquad para \ \ 1 < X < (q-1)$$

Claves de 1024 bits o más, hacen computacionalmente intratable los cálculos dentro de los primos p y q si cada uno tiene al menos 512 bits. Por tanto, para esas claves no será posible encontrar los restantes números no cifrables. Las ecuaciones para calcular los números no cifrables se muestran a continuación:
La cantidad de números no cifrables σ dentro de un cuerpo n es:

$$\sigma_n = [1 + (e-1, p-1)][1 + (e-1, q-1)]$$

Los números no cifrables serán:

$$N = [q \cdot (inv(q,p)) \cdot N_p + p \cdot (inv(p,q)) \cdot N_q] \; mód \; n$$

Donde:
N_p son las soluciones de $N^e \; mód \; p = N$
N_q son las soluciones de $N^e \; mód \; q = N$

Como puede verse, el único cálculo complicado que se presenta es en las dos últimas ecuaciones, que significa atacar mediante fuerza bruta todos los valores de N candidatos a ser números no cifrables, con $1 < N < (p-1)$ para el primo p y $1 < N < (q-1)$ para el primo q.

Capítulo 4

Criptografía simétrica con números aleatorios

En este capítulo se propone un método sencillo de cifrado simétrico, utilizando números pseudoaleatorios.

4.1. Números aleatorios

Se llama número aleatorio al que se obtiene al azar, es decir, que cualquier número tenga la misma probabilidad de ser elegido y que la elección de uno no dependa de la elección de otro. Que cualquier número tenga la misma posibilidad de ser elegido nos indica que la sucesión sigue una distribución uniforme. Que la elección de uno no dependa de la elección de otro nos dice que los números son independientes.

Generar números verdaderamente aleatorios es dificil. Los métodos que se utilizan hoy en día para este proceso, apoyados por software, generan números pseudoaleatorios, es decir, números que se han elegido de forma metódica, que parecen impredecibles y que tienen las mismas propiedades estadísticas relevantes de una sucesión de números realmente aleatorios.

Casi todos los generadores de números pseudoaleatorios, toman alguna de las siguientes formas:

- Generador de recurrencia lineal de orden k.
  Este tipo de generador produce una sucesión $\{x_i\}_{i\geq 0}$ definida de forma

recursiva del siguiente modo:

$$x_{i+k} \equiv \sum_{j=1}^{k} a_{k-j} x_{i+k-j} + c \ (mód\ m), \quad 0 \leq x_i < m \tag{4.1}$$

donde $a_0, ..., a_{k-1}, c$ son enteros positivos que toman valores en $\mathbb{Z}_m = \{0, 1, 2, ..., m-1\}$, con $a_0 \neq 0$ y m un entero positivo. En el ámbito computacional, x_{i+k} puede evaluarse como

$$x_{i+k} \equiv \sum_{j=1}^{k} a_{k-j} x_{i+k-j} + c - r_i m \quad donde \tag{4.2}$$

$$r_i = \lfloor m^{-1} \sum_{j=1}^{k} a_{k-j} x_{i+k-j} + c \rfloor. \tag{4.3}$$

Si $k = 1$, el generador recibe el nombre de *generador de congruencia lineal*. Nuestra propuesta de cifrado simétrico, utilizará este método para generar la sucesión de números pseudoaleatorios.

- Generador de recurrencia lineal módulo 2.
  Una sucesión $\{b_i\}$ de $0s$ y $1s$ puede generarse utilizando la relación de recurrencia:
  $$b_i \equiv \sum_{j=1}^{k} a_j b_{i-j} \ (mód\ 2) \tag{4.4}$$
  donde $a_1, ..., a_{k-1} \in \mathbb{Z}_2 = \{0, 1\}$, $a_k = 1$ y cada b_j toma un valor en $\mathbb{Z}_2$. La expresión 4.4 es la base de los generadores de registro de desplazamiento.

- Generador de congruencia no-lineal.
  Este tipo de generador, es de la forma:
  $$x_{i+1} \equiv f(x_i) \ (mód\ m); \qquad 0 \leq x_{i+1} < m \tag{4.5}$$
  En la expresión anterior, f es una función no lineal de valor entero de x_i. Como ejemplo de tales funciones, consideremos la siguiente:
  $$f(x) = ax^{-1} + b, \qquad x \in \mathbb{Z}_m, \ x \neq 0 \tag{4.6}$$
  donde a y b son enteros positivos, $m = p \geq 5$ es un número primo y x^{-1} es el inverso multiplicativo de x módulo p, es decir, $x \cdot x^{-1} \equiv 1 \ (mód\ p)$.

4.1.1. Generador de congruencia lineal

Los generadores de números pseudoaleatorios más populares, son casos especiales del siguiente esquema, introducido por D. H. Lehmer en 1949. Deben su popularidad a la relativa facilidad con que pueden implementarse en una computadora y a la velocidad con que generan su salida.

El *método de congruencia lineal* genera una sucesión $\{x_i\}_{i\geq 0}$ definida recursivamente por:

$$x_{i+1} \equiv (a \cdot x_i + c) \ (mód \ m) \tag{4.7}$$

La elección apropiada de los parámetros que intervienen en la expresión es lo que garantiza el éxito del generador:

- El valor inicial, x_0, también llamado *semilla*.
- El valor multiplicativo: a.
- El valor aditivo: c.
- El módulo m.

Estos parámetros deben ser números enteros no negativos y cumplir que: $x_0, a, c < m$. Iniciar la generación de números con la misma semilla, produce siempre la misma secuencia de valores.

Definición. *Una sucesión $\{x_i\}$ producida por un generador de congruencia lineal (4.1) se llama una sucesión de congruencia lineal de números pseudoaleatorios y está determinada por (x_0, a, c, m). De esta sucesión, derivamos la sucesión $\{u_i\}$ de números pseudoaleatorios normalizados donde $u_i = x_i/m \in [0, 1)$ para todo $i \geq 0$.*

EJEMPLO: Consideremos la sucesión de congruencia lineal $\{x_i\}$ determinada por $(x_0, a, c, m) = (1, 5, 7, 27)$. La sucesión generada es:

$$12, 13, 18, 16, 6, 10, 3, 22, 9, 25, 24, 19, 21, 4, 0, 7, 15, 1, 12, 13, ...$$

Notemos que la sucesión tiene longitud 18, pues el número de la posición 19 es el 12 inicial. A partir de ahí, se repite toda la sucesión. Esta longitud es muy inferior a 27, que para este caso sería la longitud máxima. Es del mayor interés lograr sucesiones de longitud máxima, que bajo ciertas condiciones pueden obtenerse. El siguiente teorema establece dichas condiciones:

Teorema 10. *Un generador de congruencia lineal produce una sucesión* $\{x_i\}$ *de longitud máxima* m *si y sólo si se cumplen las siguientes condiciones:*

i) $(c, m) = 1$*;*

ii) $a \equiv 1 \pmod{p}$ *para cada divisor primo* p *de* m*;*

iii) $a \equiv 1 \pmod{4}$ *si* m *es múltiplo de 4.*

La demostración de este teorema requiere de algunos resultados de la teoría de números que son de interés en si mismos.

Lema 2. *Sea* p *un número primo y* k *un entero positivo, donde* $p^k > 2$*. Si*

$$x \equiv 1 \pmod{p^k}, \qquad x \not\equiv 1 \pmod{p^{k+1}}$$

entonces

$$x^p \equiv 1 \pmod{p^{k+1}}, \qquad x^p \not\equiv 1 \pmod{p^{k+2}}$$

Demostración. Tenemos $x = 1 + qp^k$ para algún entero q que no es múltiplo de p. Por la fórmula binomial

$$x^p = 1 + \binom{p}{1} qp^k + \ldots + \binom{p}{p-1} q^{p-1} p^{(p-1)k} + q^p p^{pk}$$

$$x^p = 1 + qp^{k+1}\left(1 + \frac{1}{p}\binom{p}{2} qp^k + \frac{1}{p}\binom{p}{3} q^2 p^{2k} + \ldots + \frac{1}{p}\binom{p}{p} q^{p-1} p^{(p-1)k}\right)$$

La cantidad en paréntesis es un entero, y, de hecho, cada término dentro del paréntesis es un múltiplo de p, excepto el primer término. Para $1 < j < p$, el coeficiente binomial $\binom{p}{j}$ es divisible por p; por tanto

$$\frac{1}{p}\binom{p}{j} q^{j-1} p^{(j-1)k}$$

es divisible por $p^{(j-1)k}$. El último término, $q^{p-1}p^{(p-1)k}$ es divisible por p, dado que $(p-1)k > 1$ cuando $p^k > 2$. Así, $x^p \equiv 1 + qp^{k+1} \pmod{p^{k+2}}$, lo que completa la prueba. □

Lema 3. *Sea* $m = p_1^{e_1} p_2^{e_2} \ldots p_t^{e_t}$ *la descomposición del módulo* m *en factores primos. La longitud* λ *del periodo de la sucesión de congruencia lineal determinada por* (x_0, a, c, m) *es el mínimo común múltiplo de las longitudes* λ_j *de los periodos de las sucesiones de congruencia lineal* $(x_0 \bmod p_j^{e_j}$*,* $a \bmod p_j^{e_j}, c \bmod p_j^{e_j}, p_j^{e_j})$*,* $1 \le j \le t$*.*

Demostración. Haciendo inducción sobre t, es suficiente probar que si m_1 y m_2 son primos relativos, la longitud λ de la sucesión de congruencia lineal determinada por los parámetros (x_0, a, c, m) es el mínimo común múltiplo de las longitudes λ_1 y λ_2 de los periodos de las sucesiones determinadas por $(x_0 \ mód \ m_1, a \ mód \ m_1, c \ mód \ m_1, m_1)$ y $(x_0 \ mód \ m_2, a \ mód \ m_2, c \ mód \ m_2, m_2)$. Notemos que si los elementos de estas tres sucesiones son respectivamente denotados por X_n, Y_n y Z_n, tenemos:

$$Y_n = X_n \ mód \ m_1 \quad y \quad Z_n = X_n \ mód \ m_2, \qquad para \ todo \ n \geq 0$$

Por tanto, dado que $(m_1, m_2) = 1$, tenemos que

$$X_n = X_k \quad si \ y \ sólo \ si \quad Y_n = Y_k \quad y \quad Z_n = Z_k. \tag{4.8}$$

Sea λ' el mínimo común múltiplo de λ_1 y λ_2, deseamos probar que $\lambda' = \lambda$. Dado que $X_n = X_{n+\lambda}$ para todos los n suficientemente grandes, tenemos $Y_n = Y_{n+\lambda}$ (por tanto λ es múltiplo de λ_1) y $Z_n = Z_{n+\lambda}$ (por tanto λ es múltiplo de λ_2), así tenemos que $\lambda \geq \lambda'$.

Por otro lado, sabemos que $Y_n = Y_{n+\lambda'}$ y $Z_n = Z_{n+\lambda'}$ para todos los n suficientemente grandes, de modo que, por (4.8), $X_n = X_{n+\lambda'}$. Esto prueba que $\lambda \leq \lambda'$. □

Ahora estamos en condiciones de probar el Teorema 10. El Lema 3 nos dice que es suficiente probar el teorema cuando m es una potencia de un número primo, porque

$$p_1^{e_1} p_2^{e_2} \ldots p_t^{e_t} = \lambda = mcm(\lambda_1, ..., \lambda_t) \leq \lambda_1, ..., \lambda_t \leq p_1^{e_1} p_2^{e_2} \ldots p_t^{e_t}$$

será verdadero si y sólo si $\lambda_j = p_j^{e_j}$ para $1 \leq j \leq t$.
Supongamos por tanto que $m = p^e$, donde p es primo y e es un entero positivo. El teorema es claramente cierto cuando $a = 1$, así que tomemos $a > 1$. El periodo puede ser de longitud m si y sólo si cada posible entero $0 \leq x < m$, ocurre sólo una vez en el periodo. Por tanto, el periodo es de longitud m si y sólo si el período de la sucesión $X_0 = 0$ es de longitud m, y estamos justificados en suponer que $X_0 = 0$. Tenemos

$$X_n = \left(\frac{a^n - 1}{a - 1}\right) c \ mód \ m. \tag{4.9}$$

]

Si c no es primo relativo con m, este valor X_n nunca será igual a 1, así que la condición (i) del teorema es necesaria. El periodo tiene longitud m si y sólo si es el valor positivo más pequeño de n para el cual $X_n = X_0 = 0$ es $n = m$. Por la expresión (4.9) y la condición (i), nuestro teorema ahora se reduce a probar el siguiente hecho:

Lema 4. *Supongamos que* $1 < a < p^k$*, con* p *primo. Si* λ *es el entero positivo más pequeño para el que:*

$$(a^\lambda - 1)/(a-1) \equiv 0 \ (\textit{mód}\ p^k)$$

entonces

$$\lambda = p^k \quad \textit{si y sólo si} \quad \begin{cases} a \equiv 1 \ (\textit{mód}\ p) & \textit{cuando } p > 2, \\ a \equiv 1 \ (\textit{mód}\ 4) & \textit{cuando } p = 2. \end{cases}$$

Demostración. Supongamos que $\lambda = p^k$. Si $a \not\equiv 1 \ (\textit{mód}\ p)$, entonces $(a^n - 1)/(a-1) \equiv 0 \ (\textit{mód}\ p^k)$ si y sólo si $a^n - 1 \equiv 0 \ (\textit{mód}\ p^k)$. La condición $a^{p^k} - 1 \equiv 0 \ (\textit{mód}\ p^k)$ implica que $a^{p^k} \equiv 1 \ (\textit{mód}\ p)$ que a su vez implica $a^{p^k} \equiv a \ (\textit{mód}\ p)$, de aquí que $a \not\equiv 1 \ (\textit{mód}\ p)$ conduce a una contradicción. Y si $p = 2$ y $a \equiv 3 \ (\textit{mód}\ 4)$, tenemos

$$(a^{2^{k-1}} - 1)/(a-1) \equiv 0 \ (\textit{mód}\ 2^k)$$

Estos argumentos muestran que en general, es necesario tener $a = 1 + qp^f$, donde $p^f > 2$ y q no es múltiplo de p, siempre que $\lambda = p^k$.

Sólo resta demostrar que esta condición es suficiente para hacer $\lambda = p^k$. Aplicando repetidamente el Lema 2, tenemos

$$a^{p^g} \equiv 1 \ (\textit{mód}\ p^{f+g}), \qquad a^{p^g} \not\equiv 1 \ (\textit{mód}\ p^{f+g+1})$$

para todo $g \geq 0$, y por lo tanto

$$\begin{aligned} (a^{p^g} - 1)/(a-1) &\equiv 0 \ (\textit{mód}\ p^g), \\ (a^{p^g} - 1)/(a-1) &\not\equiv 0 \ (\textit{mód}\ p^{g+1}). \end{aligned} \tag{4.10}$$

En particular, $(a^{p^k} - 1)/(a-1) \equiv 0 \ (\textit{mód}\ p^k)$. Ahora, la sucesión de congruencia lineal $(0, a, 1, p^k)$ tiene $X_n = (a^n - 1)/(a-1) \ (\textit{mód}\ p^k)$, por lo tanto, tiene un periodo de longitud λ, esto es, $X_n = 0$ si y sólo si n es múltiplo de λ. De aquí que p^k es múltiplo de λ. Esto sólo puede ocurrir si $\lambda = p^g$ para algún g, y las expresiones en (4.10) implican que $\lambda = p^k$, lo que completa la prueba. □

4.1.2. ¿Cómo funciona el generador?

Un generador de congruencia lineal sigue la recurrencia:

$$x_{i+1} \equiv (a \cdot x_i + c) \ (mód\ m) \tag{4.11}$$

Ahora que ya conocemos las condiciones que generan una longitud máxima, analizemos a detalle el funcionamiento del generador con un ejemplo.

EJEMPLO: Sean $x_0 = 3$, $a = 7$, $c = 5$ y $m = 27$.

La primera iteración produce: $26 \equiv (7 \cdot 3 + 5) \ (mód\ 27)$.

La segunda iteración, tomando ahora 26 como semilla: $25 \equiv (7{\cdot}26{+}5) \ (mód\ 27)$.

La semilla es ahora 25, lo que genera: $18 \equiv (7 \cdot 25 + 5) \ (mód\ 27)$.

Iterando sucesivamente, obtenemos la sucesión completa:

$$26, 25, 18, 23, 4, 6, 20, 10, 21, 17, 16, 9, 14, 22, 24, 11, 1, 12, 8, 7, 0, 5, 13, 15, 2, 19, 3$$

El hecho de tener una sucesión de longitud máxima implica que se cumplen los criterios del Teorema 10, cuestión fácilmente verificable:

i) $(c, m) = 1$, que en este caso se cumple claramente: $(5, 27) = 1$.

ii) $a \equiv 1 \ (mód\ p)$, para cada divisor primo p de m. El único divisor primo de 27 es 3, que también verifica: $7 \equiv 1 \ (mód\ 3)$.

Tomemos ahora la expresión lineal $a \cdot x + c$. Vista geométricamente, es claro que dicha expresión define una recta con pendiente a y ordenada al origen c. Para nuestro ejemplo, la ecuación de dicha recta sería: $y = 7x + 5$. Las siguientes gráficas muestran la recta, en principio, cerca del origen. La segunda gráfica muestra la misma recta con una escala modificada.

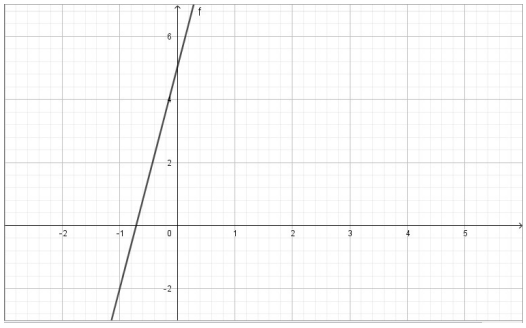

Figura 4.1: Cerca del origen

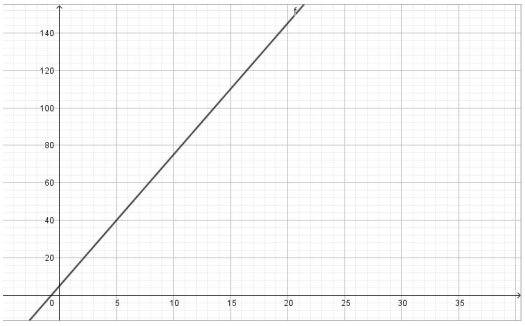

Figura 4.2: Escala modificada

Podemos darnos cuenta de que, para cada valor de x en el eje de abcisas, obtenemos un valor asociado en el eje de ordenadas. Sin embargo, para nuestros propósitos, es del mayor interés limitar los valores de x a un cierto intervalo $[a, b]$, y que desde luego, obtengamos sobre el eje de ordenadas la proyección bajo la función, de dicho intervalo $[a, b]$. ¿Cómo lograrlo? Ese es el propósito del módulo. Una vez que se obtiene un valor de la expresión lineal, al aplicarse el módulo se obtiene un residuo que se "proyecta"sobre el eje de ordenadas en correspondencia con el intervalo de interés $[a, b]$

4.2. Propuesta criptográfica

Se utilizará el *generador de congruencia lineal* ya comentado para producir números pseudoaleatorios que nos permitan un cifrado simétrico sencillo.

Sean $x_0 = 1$, $a = 7$, $c = 13$ y $m = 27$. Los parámetros anteriores, de acuerdo al Teorema 10, generan una sucesión de números pseudoaleatorios de longitud máxima, 27 en este caso. Se utilizará dicha sucesión de números para hacerla corresponder con el conjunto {0,1,...,26}, posiciones ordenadas de los caracteres del alfabeto en claro. Se ha escrito un programa Python que sigue la recurrencia lineal del generador para producir la sucesión pseudoaleatoria, al tiempo que realiza el cifrado/descifrado de mensajes. El código se incluye en el siguiente apartado.

La sucesión generada con los parámetros anteriores es:

$$20, 18, 4, 14, 3, 7, 8, 15, 10, 2, 0, 13, 23, 12, 16, 17, 24, 19, 11, 9, 22, 5, 21, 25, 26, 6, 1$$

Haciendo corresponder esta lista pseudoaleatoria con el conjunto {0,1,...,26}, tenemos:

0	20
1	18
2	4
3	14
4	3
5	7
6	8
7	15
8	10
9	2
10	0
11	13
12	23
13	12
14	16
15	17
16	24
17	19
18	11
19	9
20	22
21	5
22	21
23	25
24	26
25	6
26	1

Figura 4.3: Lista pseudoaleatoria

que genera la siguiente dispersión de puntos para el cifrado:

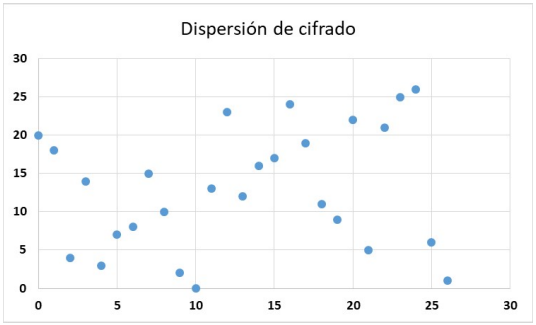

Figura 4.4: Dispersión de cifrado

La correspondencia *inversa* es la siguiente:

20	0
18	1
4	2
14	3
3	4
7	5
8	6
15	7
10	8
2	9
0	10
13	11
23	12
12	13
16	14
17	15
24	16
19	17
11	18
9	19
22	20
5	21
21	22
25	23
26	24
6	25
1	26

Figura 4.5: Correspondencia inversa de la lista pseudoaleatoria

que a su vez, genera la siguiente dispersión de descifrado:

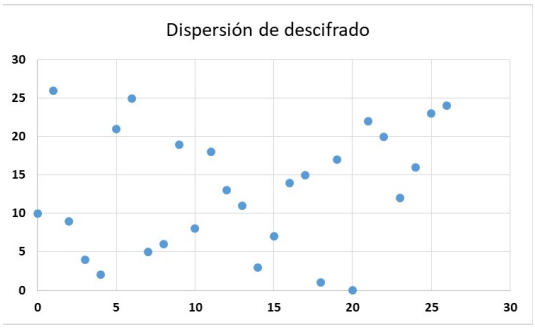

Figura 4.6: Dispersión de cifrado

La correspondencia entre el alfabeto en claro y el alfabeto de cifra es lo que nos permite el cifrado/descifrado.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
20	**18**	**4**	**14**	**3**	**7**	**8**	**15**	**10**	**2**	**0**	**13**	**23**	**12**	**16**	**17**	**24**	**19**	**11**	**9**	**22**	**5**	**21**	**25**	**26**	**6**	**1**
T	R	E	Ñ	D	H	I	O	K	C	A	N	W	M	P	Q	X	S	L	J	V	F	U	Y	Z	G	B

Ejemplo:
Mensaje en claro: CIFRADOSIMETRICOALEATORIO
Mensaje cifrado: EKHLTÑQJKWDVLKEQTNDTVQLKQ

La clave para este sistema de cifrado simétrico, viene dada desde luego por los parámetros (x_0, a, c, m) del generador de congruencia lineal. El destinatario del mensaje debe conocer estos parámetros para poder *descifrar* el mensaje que está recibiendo.

4.3. Código Python

En este apartado se incluye el código Python que genera la sucesión de números aleatorios y que realiza el cifrado/descifrado de mensajes, utilizando desde luego la sucesión pseudoaleatoria generada.

```
# Este programa genera números pseudoaleatorios utilizando el
    Método
# de congruencia lineal:
# x[n+1] = (a·x[n] + c) (mód m)

# Variables globales
alfabeto_en_claro =['A','B','C','D','E','F','G','H','I','J','K',
    'L','M','N','Ñ','O','P','Q','R','S','T','U','V','W','X','Y','
    Z']
alfabeto_cifrado = []
lista = []

# Rutina para cifrar
def cifrado(mensaje_en_claro):
    texto_cifrado = ""
    for j in mensaje_en_claro:
        pos=alfabeto_en_claro.index(j) # Capturar posición del
            caracter en el alfabeto_en_claro
        caracter=alfabeto_cifrado[pos] # Tomar el caracter de
            esa posicion en el alfabeto_cifrado
        texto_cifrado = texto_cifrado + caracter
```

```
    return texto_cifrado

# Rutina para descifrar
def descifrado(mensaje_cifrado):
    texto_en_claro = ""
    for j in mensaje_cifrado:
        pos=alfabeto_cifrado.index(j)   # Capturar posición del
            caracter en el alfabeto_cifrado
        caracter=alfabeto_en_claro[pos] # Tomar el caracter de
            esa posicion en el alfabeto_en_claro
        texto_en_claro = texto_en_claro + caracter
    return texto_en_claro

def genera_aleatorios():
   a = 7                  # Constante multiplicativa
   c = 13                 # Constante aditiva
   m = 27                 # Módulo
   xn = 1                 # Semilla
   i = 0
   while i < m:
      xnn = (a*xn + c) %m
      lista.append(xnn)          # Se almacena el número generado
      alfabeto_cifrado.append(alfabeto_en_claro[xnn])   # Se
          almacena el caracter cifrado
      i = i + 1
      xn = xnn                               # Se modifica la semilla

genera_aleatorios()
print ("=============================================================")
print ("Recurrencia utilizada: x[n+1] = (a·x[n] + c) (mód m)")
print ("Con: a = 7;    c = 13    y    m = 27.")
print ("=============================================================")
print("Sucesión pseudoaleatoria generada:")
print(lista)
print ("")
print ("Alfabeto en claro:")
print(alfabeto_en_claro)
print ("")
print ("Alfabeto cifrado:")
print(alfabeto_cifrado)
print ("")
op=1
while op !='3':
   print ("[ 1 ] Cifrar.")
   print ("[ 2 ] Descifrar.")
```

```
    print ("[ 3 ] Salir.")
    op=input("Escribe opción: ")
    if op == '1':
        texto_en_claro=input("Mensaje a cifrar: ")
        texto_cifrado=cifrado(texto_en_claro.upper())
        print("El mensaje cifrado es: ",texto_cifrado)
    elif op == '2':
        texto_cifrado=input("Mensaje a descifrar: ")
        texto_descifrado=descifrado(texto_cifrado.upper())
        print("El mensaje en claro es: ",texto_descifrado)
```

Conclusiones

Consideramos de particular interés que se conozcan las herramientas criptográficas actuales, tanto por los usuarios legos en la materia como por los especialistas, ello conducirá sin duda a que los usuarios se preocupen de la seguridad de sus datos personales y sensibles, en tanto que a los especialistas, principalmente los matemáticos, los motivará, quizá, a orientar sus investigaciones a este campo, lo que desde luego redundará en un mayor desarrollo de la criptografía en general y en la invención de nuevos algoritmos, tan necesarios en este campo.

A pesar de su longevidad, pues RSA surgió en 1977, se ha mantenido como el estándar criptográfico asimétrico más utilizado. En este sentido, el propósito de esta memoria fue dar a conocer los fundamentos matemáticos del algoritmo así como su funcionamiento a la hora de implementarlo.

Aumentar la longitud de la clave a través del tiempo ha sido suficiente para que el algoritmo haya sobrevivido hasta nuestros días.

Los ataques a los que ha sido sometido RSA durante su ya larga vida, se han revelado cada vez más sofisticados y en particular, los ataques por canal lateral han logrado vulnerar la seguridad de este y otros algoritmos. Sin embargo, las recomendaciones vertidas por los investigadores e implementadas por los fabricantes le han permitido "sobrevivir".

Dado el poder de cómputo necesario para descifrar una clave RSA de 2048 bits o de tamaño superior, se hace imperativo utilizar otros esquemas de cifrado, y hace inviable utilizar RSA en el cifrado de "pequeños dispositivos", tales como las smartcards, lugar que ya está siendo ocupado por la criptografía de curvas elípticas, que ofrece una seguridad similar a la de RSA con

tamaños de clave mucho menores.

Sin embargo, cabe señalar que el esquema de trabajo de la criptografía asimétrica sigue siendo el mismo: *dos claves*, una pública, una privada, una para cifrar, la otra para descifrar, y en ese sentido, no hay en el horizonte propuestas de cifrado novedosas o distintas a lo que actualmente utilizamos.

Bibliografía

[1] Apostol, Tom M. *Introducción a la teoría analítica de números*. Reverté, España, 2002.

[2] Boneh, Dan. «*Twenty years of attacks on the RSA cryptosystem*». Notices on the AMS, Vol. 46(2), pp 203-213, USA, 1999.

[3] Burnett, Steve and Paine, Stephen. *RSA Security's official guide to cryptography*. McGraw-Hill, USA, 2001.

[4] Diffie, Whitfield; Hellman, Martin «*New directions in Cryptography*». IEEE Transactions on Information Theory, Vol. IT-22, USA, 1976.

[5] Durán Díaz, José Raúl. «*Números primos especiales y sus aplicaciones criptográficas*». Escuela Técnica Superior de Ingenieros de Telecomunicación: Tesis Doctoral. España, 2003.

[6] Durfee, Glenn. «*Cryptanalysis of RSA using algebraic and lattice methods*». Stanford University: Ph.D. Thesis. USA, 2002.

[7] Genkin, Daniel; Shamir, Adi; Tromer, Eran. «*RSA key extraction via Low-Bandwith Acoustic Cryptanalysis*». Advances in Cryptology - CRYPTO 2014. Springer, USA, 2014.

[8] Glen, Amy. «*On the period length of pseudoramdom number sequences*». The University of Adelaide: Ph.D. Thesis. Australia, 2002.

[9] González Gutiérrez, Francisco José. *Apuntes de Matemática Discreta*. Universidad de Cádiz, España, 2004.

[10] Hoffstein, Jeffrey; Pipher, Jill; Silverman, Joseph H. *An introduction to mathematical cryptography*. Springer, USA, 2008.

[11] Hull, T. E.; Dobell, A. R.. «*Random Number Generators*». Society for Industrial and Applied Mathematics Review. Vol. 4, No. 3. USA, 1962.

[12] Jonhsonbaugh, Richard. *Matemáticas discretas*. Pearson Educación, México, 2005.

[13] Katzenbeisser, Stefan. *Recent advances in RSA cryptography*. Springer Science+Business Media, USA, 2001.

[14] Knuth, Donald E. *The art of computer programming. Vol II Seminumerical algorithms* . Addison Wesley Longman, USA, 1998.

[15] Lezaun Iturralde, Mikel. «*Las matemáticas de lo secreto*». Revista de Didáctica de las Matemáticas, Vol. 74, pp. 7-13, España, 2010.

[16] Muñoz Muñoz, Alfonso y Ramió Aguirre, Jorge. *Cifrado de las comunicaciones digitales. De la cifra clásica al algoritmo RSA*. 0xWORD, España, 2013.

[17] Niven, Iván y Zuckerman, Herbert S. *Introducción a la teoría de los números*. 2a. edición, Limusa, México D.F., 1976.

[18] Paar, Christof and Pelzl, Jan. *Understanding cryptography*. Springer, Germany, 2010.

[19] Pérez Seguí, María Luisa. *Teoría de Números*. UNAM. Instituto de Matemáticas, México, 2004.

[20] Preeta, M.; Nithya, M. «*A study and performance analysis of RSA algorithm*». International Journal of Computer Science and Mobile Computing, Vol. 2, pp. 126-139, USA, 2013.

[21] Saranya, Vinothini, Vasumathi. «*A study on RSA algorithm for Cryptography*». International Journal of Computer Science and Information Technologies, Vol. 5(4), pp. 5708-5709, USA, 2014.

[22] Shannon, Claude E. «*A mathematical theory of communication*». Bell System Technical Journal, Vol. 27, pp. 379-423, USA, 1948.

[23] Shannon, Claude E. «*Communication theory of secrecy systems*». Bell System Technical Journal, Vol. 28-4, pp. 656-715, USA, 1949.

[24] Silverman, Joseph H. *A friendly introduction to Number Theory.* Pearson, USA, 2012.