Investigación y Tecnología Vol. 4. Nro. 1: pag.51- pag.56. Octubre 2016. ISSN 2306-0522

Una Introducción Histórica al Sistema de Criptografía RSA

A Historical Introduction to RSA cryptosystem

*** Dr. Paul A. Loomis**
Department of Mathematical and Digital Sciences
Bloomsburg University
Pennsylvania, EE.UU.
Autor de correspondencia:* ploomis@bloomu.edu

Resumen

Presentamos una introducción histórica y básica a la criptografía para aquellos que no son familiarizados con la teoría de números. Después de tratar algunos criptosistemas antiguos, vamos al problema de crear una función de encriptación cuya inversa es difícil hallar, y mostramos como la solución más conocida – el criptosistema RSA – tiene su base en resultados matemáticos del tercer siglo a.C. al presente.

Palabras clave: criptografía; fundamentos matemáticos de la criptografía; encriptación clave pública.

Abstract

We give a basic historical introduction to cryptography for those unfamiliar with number theory. After treating a few ancient cryptosystems, we move to the problem of creating an encryption function whose inverse is difficult to find, and show how the most well-known solution – the RSA cryptosystem – is based on mathematics from 300BC to the present.

***Keywords**: cryptography; mathematical foundations of cryptography; public key encryption.*

Introducción

En este articulo, describimos una breve introducción al sistema de criptografía RSA. Es decir, este artículo es una introducción a unos temas básica pero central en la teoría de números con la motivación de su aplicación a la criptografía. Si se busca tratamientos más avanzados sobre la temática, recomendamos cualquier introducción a la criptografía, por ejemplo (Coutinho, 2003), ó algunas introducciones a la teoría de números, como (Burton, 2007). Aquí queremos mostrar como el sistema más usado en la criptografía hoy tiene sus raíces en resultados matemáticos muy antiguos.

Métodos

Primeramente, un código es una función. Codificar un mensaje consiste en aplicar la función; decodificar consiste en aplicar la inversa de la función. Distinguimos entre *decodificar* y *descifrar*; decodificar es lo que hace el receptor intentado después de recibir el mensaje, y descifrar es lo que hace un *hacker* ó receptor no autorizado. Entonces, descifrar consiste en romper el código. Por siglos, saber la función de encriptación dio casi inmediatamente la función de desencriptación. Lo que buscaron Rivest, Shamir y Adleman en los años setenta era una función cuya inversa era difícil encontrar. El gran avance de los sistemas de clave pública es que se puede publicar una función de encriptación sin que alguien afuera del receptor pudiera hallar su inversa.

Empezamos históricamente con un ejemplo (demasiado) simple: una cifra de Cesar, del primer siglo a.C. Tenemos un mensaje: "VAMOS AHORA", que un emisor ***A*** quiere mandar a un receptor ***B***. En esta charla, vamos a asignar un número a cada letra del alfabeto. Simplemente, ponemos ***A***=01, ***B***=02, ***C***=03, … ***Z***=26, espacio=27. (Omitimos la ñ y los números, pero sería fácil incluirlos.) Entonces el mensaje

VAMOS AHORA

aparece como:

22 01 13 15 19 27 01 08 15 18 01

Una cifra de Cesar es una función $\boldsymbol{f\ x_i \equiv x_i + a \mod 27}$, en donde $\boldsymbol{x_1 x_2 \ldots x_n}$ es el mensaje, $\boldsymbol{a}$ es un número entero, y el número 27 es el número de caracteres encifrados. (Más adelante definimos el "***mod***"; ahora sustraemos 27 de cualquier número mayor que 27.) Por ejemplo, si $\boldsymbol{a = 11}$, nuestro mensaje es

06 12 24 26 03 11 12 19 26 02 12

ó, en letras,

FLXZCKLSZBL

Desafortunadamente, esto es fácil descifrar con un análisis de frecuencias. Por ejemplo, tal vez adivinamos que la "***L***" que aparece muchas veces es un vocal, y con este clave tenemos una entrada. Peor, si adivinamos que la cifra es uno de Cesar, solo tenemos que añadir cada número entre 1 y 27 al mensaje y elegir lo que tiene sentido.

Hay varios modos de hacer la cifra de Cesar un poco más difícil romper. En vez de cambiar cada número por la misma cantidad, podremos añadir una secuencia de números. Esto se denomina una cifra de palabra clave repetida. Si la palabra clave es CONDOR, vamos a añadir

03 15 14 04 15 13

repetido cuantas veces que sea necesario. Si la palabra clave tiene

números $k_1 k_2 \ldots k_n$, nuestra función es $f(x_i) \equiv x_i + k_i \pmod{27}$.

El resultado de nuestro mensaje es

22 01 13 15 19 27 01 08 15 18 01 mas
03 15 14 04 15 13 03 15 14 04 15 que es
25 16 27 19 07 13 04 23 02 22 16 ó ,

en letras,

YP SGMDWBVP

Note que ahora las "***A***" repetidas del mensaje original ahora no aparecen como letras repetidas; entonces esta cifra polialfabética es más difícil descifrar. Sin embargo, si el descifrador se puede determinar el tamaño de la palabra clave, resolver es equivalente a descifrar una serie de cifras de Cesar. Entonces sería mejor tener una palabra clave bien larga, por ejemplo un libro que ambos A y B tienen. Con los ordenadores de hoy, estos también se pueden romper. Entonces necesitamos una función cuya inversa es más difícil encontrar.

En tal sentido, quisiéramos definir unos conceptos básicos. En lo que sigue, todas las variables son números enteros. Vamos a usar p y q para representar números primos. Usamos (m, n) para indicar el menor común divisor de m y n. Decimos que dos números m y n son coprimos si $m, n = 1$. Definimos que ***m divide*** a n (escrito $m \mid n$ si existe un *x* tal que $\mathbf{n = mx}$, es decir, si n es múltiplo de m. También definimos que a **y** b **son equivalentes módulo n** (escrito $a \equiv b \pmod{n}$) si $n \mid a - b$. Entonces dos números son equivalentes mod n si y solo si sus restos después de dividir por n son iguales.

Primero, necesitaremos el Lema de Euclides, lo que apareció en los Elementos de Euclides en el tercer siglo a.C. Eso dice que si $a \mid bc$ y $a, b = 1$, entonces $a \mid c$. Un caso especial es si a es un numero primo p: si $p \mid bc$ y p no divide a b, entonces $p \mid c$. Esto hecho depende sobre el hecho que siempre se puede escribir el menor común divisor de dos números como una combinación lineal de los dos; es decir, que siempre existen x y y tal que $ax + by = (a, b)$. Este en su turno depende sobre el Principio de Buen Orden. Dado esto, si sabemos que $a \mid bc$ y $a, b = 1$, tenemos las ecuaciones $ax + by = 1$ y $az = bc$ por algunos enteros x, y, y z. Multiplicando por c, tenemos $axc + byc = c$, ó $axc + azy = c$, ó $a(xc + zy) = c$. Como $xc + zy$ es un número entero, tenemos $a \mid c$.

En el idioma de aritmética modular, el Lema de Euclides significa que podemos cancelar cuando tenemos un número coprimo con el modulus n. Es decir, si $a, n = 1$ y $ab \equiv ac \pmod{n}$, entonces $n \mid ab - ac$, y por el Lema de Euclides, $n \mid b - c$ y entonces $b \equiv c \pmod{n}$.

Necesitamos dos conceptos más: un sistema completo de restos (SCR) módulo n es un conjunto $\{a_1, \ldots, a_n\}$ tal que cada número entero es equivalente mó dulo n a uno (y solo uno) elemento a_i. El conjunto más natural seria $\{0, 1, 2, \ldots, n-1\}$, pero hay infinidades de opciones. Un sistema reducido de restos (SRR) módulo n es un conjunto tal que cada número entero *coprimo* con *n* es equivalente módulo n a uno (y solo uno) elemento del conjunto. La función φ de Euler era definido para describir el tamaño de un SRR módulo n; es decir, $\varphi(n)$ es el número de enteros entre 1 y n y coprimo con n, y el SRR expresamos como $\{a_1, \ldots, a_{\varphi(n)}\}$.

Eso nos lleva a otro modo de hacer un código; si tenemos nuestro mensaje

22 01 13 15 19 27 01 08 15 18 01

Elegiremos un número primo más que 27; digamos 29. Esta vez nuestra función de encriptación consiste en elevar los números a una potencia y reducirlos módulo 29. Elegimos $\boldsymbol{e} = 11$ (es necesario elegir uno coprimo con 28; pronto veremos porque) y ponemos $\boldsymbol{f\ x_i\ \equiv x_i^e (mod\ 29)}$. Ahora nuestro mensaje es

06 01 04 21 27 11 01 03 21 19 01

Decodificar el mensaje parece difícil; no podemos tomar la raíz ***11-ésima*** después de que los resultados fueron reducidos. Pero un resultado del año 1640 nos ayuda: el Teorema Pequeño de Fermat: si $\boldsymbol{p}$ no divide a $\boldsymbol{a}$, entonces $\boldsymbol{a^{p-1} \equiv 1\ (mod\ p)}$. Como era su costumbre, Fermat nunca publicó una demostración de su teorema, pero Euler lo hizo en 1736. Su demostración usó el Lema de Euclides y el concepto de la SRR: sea $\{\boldsymbol{1, 2, \dots, p-1}\}$ una SRR módulo $\boldsymbol{p}$. Sea que $\boldsymbol{p}$ no divide a $\boldsymbol{a}$, y pensamos en el conjunto $\boldsymbol{a, 2a, \dots\ p-1\ a}$. Queremos demostrar que esto también es una SRR módulo $\boldsymbol{p}$. Como hay $\boldsymbol{p-1}$ elementos, solo es necesario demostrar que todos son distintos módulo $\boldsymbol{p}$. Entonces supongamos que dos elementos $\boldsymbol{ia}$ y $\boldsymbol{ja}$ del conjunto son equivalentes módulo $\boldsymbol{p}$: $\boldsymbol{ia \equiv ja\ (mod\ p)}$ con $\boldsymbol{1 \le i, j \le p-1.}$ Como $\boldsymbol{a, p\ = 1}$, $\boldsymbol{i \equiv j\ \ mod\ p}$, ó $\boldsymbol{p|i-j.}$ Pero $\boldsymbol{i-j\ \le\ p-1,}$ y entonces $\boldsymbol{p|i-j}$ hace que $\boldsymbol{i = j,}$ y nuestros dos elementos actualmente son el mismo. Entonces, nuestros dos conjuntos son idénticos cuando los reducimos módulo $\boldsymbol{p}$, y luego sus productos también son equivalentes módulo $\boldsymbol{p}$. Es decir,

$$\boldsymbol{1 \cdot 2 \cdot \dots \cdot\ p-1\ \equiv a \cdot 2a \cdot \dots \cdot\ p-1\ \ a\ (mod\ p)}$$

o, reagrupando,

$$\boldsymbol{1 \cdot 2 \cdot \dots \cdot\ p-1\ \equiv 1 \cdot 2 \cdot \dots \cdot\ p-1\ \ a^{p-1}\ (mod\ p)}$$

Pero todos los números $\boldsymbol{1}$ hasta $\boldsymbol{p-1}$ son coprimos con $\boldsymbol{p}$ y entonces los cancelamos, dejando solo $\boldsymbol{1 \equiv a^{p-1}\ (mod\ p)}$.

Esto significa que también $\boldsymbol{a^{k(p-1)} \equiv 1\ (mod\ p)}$ y entonces $\boldsymbol{a^{k\ p-1\ +1} \equiv a\ (mod\ p)}$ para cada entero positivo $\boldsymbol{k}$. Entonces si el receptor $\boldsymbol{B}$ encuentra una $\boldsymbol{d}$ tal que $\boldsymbol{de = k\ p-1\ +1}$ (es decir $\boldsymbol{de \equiv 1\ (mod\ p-1)}$) se puede recuperar los números que el emisor $\boldsymbol{A}$ ha encifrado. En nuestro ejemplo, tenemos que resolver $\boldsymbol{11d \equiv 1\ \ mod\ 28}$ (note que aquí requerimos que 11 y 28 son coprimos), lo que resuelve rápidamente como $\boldsymbol{d \equiv 23\ (mod\ 28)}$. Entonces, $\boldsymbol{B}$ recibe

06 01 04 21 27 11 01 03 21 19 01

se eleva todos los números a 23, reduce mod 29, y tiene

22 01 13 15 19 27 01 08 15 18 01

cuál era el mensaje original.

Este sistema, aunque es menos transparente que el anterior, también es vulnerable al análisis de frecuencias. Esto se puede corregir, cambiando el tamaño de los bloques, mandando el mensaje como

060 104 212 711 010 321 190 100

y usando un el mismo estrategia con un número primo mayor que 711.

Todo esto requiere que $\boldsymbol{A}$ y $\boldsymbol{B}$ comunican antes en una forma segura sobre ***p, d***, y ***e, y*** con la fuerza de computación disponible hoy, también es fácil romper. Entonces necesitamos algo más sutil. La solución es

usar un módulo compuesto. Por eso, necesitamos un teorema más.

Recordamos que fue Euler quien era el primero publicar una demostración del Teorema Pequeño de Fermat, y entonces es justo que fuera él quien también lo generalizó para módulos compuestos. Entonces, la Generalización de Euler del Teorema Pequeño de Fermat dice que si $a, n = 1$, entonces $a^{\varphi(n)} \equiv 1 \ (mod\ n)$. La demostración es muy parecido de lo del anterior: empezamos con un SRR módulo n $\{a_1, a_2, \dots, a_{\varphi\ n}\}$. Dejamos como un ejercicio para demostrar que si $a, b = a, c = 1$, entonces también $a, bc = 1$. Dado esto, ahora sea a coprimo con n. Tenemos $aa_1, aa_2, \dots, aa_{\varphi\ n}$ también es un SRR módulo n. Entonces los dos conjuntos son idénticos mó dulo n, y los productos son equivalentes módulo n. Llegamos a

$$a_1 \cdot a_2 \cdot \dots \cdot a_{\varphi(n)} \equiv aa_1 \cdot aa_2 \cdot \dots \cdot aa_{\varphi\ n} \ (mod\ n)$$

y reagrupándolo, a

$$a_1 \cdot a_2 \cdot \dots \cdot a_{\varphi(n)} \equiv a_1 \cdot a_2 \cdot \dots \cdot a_{\varphi(n)} \cdot a^{\varphi(n)} \ (mod\ n)$$

Pero otra vez todos los a_i son coprimos con n, y por el Lema de Euclides, se puede cancelarlos, dejando el resultado: $1 \equiv a^{\varphi(n)} \ (mod\ n)$.

Ahora haremos una función de encriptación más, con $n = pq$, en donde p y q son números primos distintos. Es claro que $\varphi\ 1 = 1$, y si $n = \prod_{i=1}^{r} p_i^{e_i}$ es la descomposición de n en números primos distintos, se puede mostrar que $\varphi\ n = \prod_{i=1}^{r}(p_i^{e_i} - p_i^{e_i-1})$. Para nosotros, sería suficiente saber que $\varphi\ pq = p-1\ (q-1)$.

Entonces seguimos más ó menos como hicimos antes: elegimos dos números primos, digamos $p = 41$ y $q = 53$, y entonces $n = 2173$ y $\varphi\ n = 2080$. Elegimos una potencia de encriptación que es coprimo con 2080, por ejemplo $e = 31$. El emisor A puede empezar con un mensaje

060 104 212 711 010 321 190 100

elevar todos los bloques a 31, reducir módulo 2173, y mandar

1278 0403 0265 0618 0625 0022 0972 1658

La generalización de Euler dice que si a es coprimo con 2173, entonces $a^{2080} \equiv 1 \ (mod\ 2173)$ y $a^{2080k+1} \equiv a \ mod\ 2173$ para cada entero positivo k. Entonces para decodificar, tenemos que encontrar un d tal que $31d = 2080k + 1$ ó $31d \equiv 1 \ (mod\ 2080)$ (otra vez, es por eso que tenemos que elegir un e coprimo con 2180.) Con el algoritmo euclideano, eso no es difícil, y llegamos a $d = 671$. Por lo tanto, el receptor B, sabiendo n y d, eleva todos los bloques a 671, y tiene el mensaje original.

Hay un detalle sutil pendiente: la generalización de Euler requiere que a sea coprimo con n; note que el bloque 212, como es múltiplo de 53, no es. Entonces 212^{2080} no es equivalente a 1 módulo 2173, pero con una doble aplicación cuidadosa del Teorema de Fermat, se puede mostrar que $212^{2080k+1} \equiv 212 \ (mod\ 2173)$ para cada entero positivo **k**. Sucede con cada numero entre 0 y 2173.

Pero ahora hay un detalle mucho más importante. Note que A tiene que saber *e* y $\boldsymbol{n}$ para mandar un mensaje, pero no tiene que saber *p* ni *q*. Entonces el par $(\boldsymbol{n}, \boldsymbol{e})$ puede ser público, y cualquier persona puede encifrar un mensaje. Por eso se denomina método de clave pública, de que RSA, nombrado por sus tres inventores, es el ejemplo más famoso. El par $(\boldsymbol{n}, \boldsymbol{e})$ se denomina la clave pública, y el par $(\boldsymbol{n}, \boldsymbol{d})$ el clave privado. Note que para descifrar el código, uno tiene que saber $\boldsymbol{p}$ y $\boldsymbol{q}$, ó por lo menos $\boldsymbol{\varphi}(\boldsymbol{n})$. Entonces romper el código es igual que factorizar 2173. Eso no es difícil, pero ¿Qué pasa si usamos números primos muchos más grandes? Con un ordenador, es rápido multiplicar números grandes, pero factorizarlos queda difícil. Es decir, hasta ahora, no existen algoritmos que pueden factorizar un número de 200 digitos en corto tiempo. En fin, llegamos en una función (ó acción) cuya inversa es difícil hacer

Conclusiones

Solo hemos llegado a un punto de salida para hablar sobre el método RSA y la idea de criptografía de clave pública. La historia de los métodos de factorizar números grandes es larga y fascinante, y uno tiene que tomar esos métodos en cuenta cuando se elige los números primos $\boldsymbol{p}$ y $\boldsymbol{q}$. También una elección no cuidadosa de la potencia de encriptación $\boldsymbol{e}$ deja que el código es vulnerable. Hemos sido un poco informales en la pre-codificación, convertir letras en números y separar el mensaje en bloques. Esto también tiene que hacer con cuidado, para que se pueda decodificar los bloques en modo único y para que no sean vulnerables a un análisis de frecuencias. Tampoco hemos hablado de esquemas de relleno, con que se puede disfrazar un mensaje aún más. Para concluir, hay muchas cuestiones de implementación que hemos dejado pendientes. Sin embargo, esperamos que este artículo dé la esencia del tópico y estimule interés en estos resultados número-teoréticos.

Agradecimientos

Quisiera agradecer a Bloomsburg University, por el año de sabático que me dio la libertad de visitar La Paz, y a dos carreras de la Facultad de Ciencias Puras y Naturales de la Universidad Mayor de San Andrés: a la Carrera de Matemáticas, por alojarme y darme la oportunidad de dictar un curso sobre la teoría de números, y a la Carrera de Informática, por la invitación de impartir una conferencia en el Congreso Nacional de Ciencias de Computación Bolivia (CCBOL) y de entregar este articulo. Finalmente, quisiera agradecer a Rubén Hilare Quispe por su ayuda con el castellano.

Referencias

Burton, D. (2007). *Elementary Number Theory, Sixth Edition:* McGraw-Hill, New York.

Coutinho, S. (2003). *Números Enteros y Criptografía RSA:* Instituto de Matemáticas y Ciencias Afines, Lima.

Presentado: La Paz, 16 de Septiembre de 2016.

Aceptado: La Paz, 10 de Octubre de 2016