Capítulo 6
Revoluciones en la criptografía post Turing

Nicolás Sirolli

¿Qué cambió en la criptografía después de que Turing *rompió* la máquina Enigma?

Gracias al desarrollo de la tecnología, en los últimos tiempos el intercambio de información se ha vuelto considerablemente más rápido y fácil, y también más necesario en nuestra vida cotidiana. Eso supuso nuevos desafíos para quienes necesitaran comunicarse de manera segura.

La idea que revolucionó la criptografía consistió en mostrar que dos personas pueden usar una clave pública para comunicarse de manera segura, sin necesidad de encontrarse previamente en privado. Esta idea aparece por primera vez en el protocolo de Diffie-Hellman, publicado en 1976, gracias al cual los autores recibieron recientemente el premio Turing.

En estas notas ilustraremos este gran salto con ejemplos sencillos de protocolos de criptografía basados en claves privadas y públicas.

1. Criptografía de clave privada

Los protocolos clásicos de la criptografía están basados en la utilización de una clave privada, que deben compartir las dos partes que desean comunicarse; los llamaremos Alicia y Beto. A grandes rasgos, el esquema consiste en:

- Un conjunto $\{k\}$ de claves.
- Para cada clave k, un par de funciones e_k (de encriptado) y d_k (de desencriptado), que satisfacen

$$d_k(e_k(m)) = m \qquad \text{para todo mensaje } m. \tag{1.1}$$

Es deseable que evaluar las funciones d_k y e_k sea sencillo y rápido.

Para comunicarse Alicia y Beto deben primero acordar, en privado, una clave k. Una vez hecho esto, el protocolo a seguir es el siguiente:

- Alicia encripta el mensaje m, calculando $e = e_k(m)$.
- Alicia envía a Beto el mensaje encriptado e.
- Beto desencripta el mensaje, calculando $d_k(e)$.

Gracias a (1.1) se tiene que $d_k(e) = m$. Así, Beto obtiene el mensaje m.

Asumiremos el peor escenario posible: que Miranda, quien está interesada en conocer el mensaje m, pudo interceptar la comunicación y conoce el mensaje encriptado e. Más aún, asumimos que Miranda sabe cómo utilizar la función de desencriptado d_k si conoce la clave k. Por eso, el protocolo será seguro solamente si es difícil obtener m o k a partir de $e = e_k(m)$.

A continuación daremos un ejemplo clásico de un protocolo de criptografía de clave privada.

1.1. Cifrado de Vigènere

Este cifrado, que data del siglo xvi, se destaca por su simplicidad, y por haber sido considerado *irrompible* por muchísimos años. Para formularlo, aunque no es estrictamente necesario, utilizaremos aritmética modular sencilla.

La clave que comparten secretamente Alicia y Beto es una palabra. Por ejemplo, la palabra `clave`.

Tanto para encriptar como para desencriptar, numeraremos las letras del abecedario, haciendo corresponder la `A` con el 0, la `B` con el 1, y así siguiendo.

La función de encriptado consiste en, una vez hecha esta identificación, ir sumando, letra por letra, las letras del mensaje con las letras de la clave; volviendo a repetir la clave una vez que esta se termina en caso de ser necesario.

Por ejemplo, supongamos que el mensaje que quiere enviar Alicia es "Ganó Boca". Para simplificar, omitiremos espacios y caracteres especiales, y consideraremos $m = \texttt{GANOBOCA}$. Para encriptar, Alicia realiza la suma:

$$\begin{array}{r} \texttt{GANOBOCA} \\ \texttt{CLAVECLA} \\ \hline \texttt{ILNJFQNA} \end{array}$$

Que en términos de la correspondencia entre letras y números es la suma:

$$\begin{array}{r} 6\ \ 0\ 13\ 14\ 1\ 14\ \ 2\ 0 \\ 2\ 11\ \ 0\ 21\ 4\ \ 2\ 11\ 0 \\ \hline 8\ 11\ 13\ \ 9\ 5\ 16\ 13\ 0 \end{array}$$

Notemos que no está sumando enteros, sino que está realizando la suma módulo 26, que es la cantidad de letras del abecedario. Por ejemplo, al sumar `O` y `V`, es decir al sumar 14 y 21, el resultado es `J`, es decir 9, que es congruente a $14 + 21 = 35$ módulo 26.

Para desencriptar, Beto tiene que simplemente restar la palabra CLAVE al mensaje encriptado $e =$ ILNJFQNA que recibió; teniendo en cuenta también que las restas se deben realizar módulo 26.

$$\begin{array}{r} \texttt{ILNJFQNA} \\ \texttt{CLAVECLA} \\ \hline \texttt{GANOBOCA} \end{array}$$

Que en términos de la correspondencia entre letras y números es la resta:

$$\begin{array}{rrrrrrrrr} 8 & 11 & 13 & 9 & 5 & 16 & 13 & 0 \\ 2 & 11 & 0 & 21 & 4 & 2 & 11 & 0 \\ \hline 6 & 0 & 13 & 14 & 1 & 14 & 2 & 0 \end{array}$$

La ventaja de este método se basa en que una misma letra del mensaje, por ejemplo la O, aparece cifrada de dos maneras distintas: como J y como Q. Gracias a eso, Miranda no puede hacer un análisis de frecuencia de la aparición de las letras en el mensaje cifrado para intentar averiguar la clave y/o el mensaje.

Pese a eso, en el siglo xix, mucho tiempo después de su aparición, este cifrado fue roto por Kasiski (véase [1], sección 4.2).

1.2. Enigma y después

La máquina Enigma, utilizada por los alemanes durante la segunda guerra mundial, utilizaba un conjunto de claves más complejas que simples palabras, dadas por configuraciones de rotores y *plugboards*, y utilizaba para cifrar y descifrar los mensajes un dispositivo notablemente más complicado que la aritmética módulo 26.

Más allá de que el cifrado también haya sido vulnerado, esta vez más rápidamente gracias a la genialidad de Turing y su equipo, este protocolo adolece de algo en común con el de Vigenère: en ambos Alicia y Beto necesitan encontrarse secretamente para acordar una clave con la que comunicarse de manera segura después. Esto, en medio de una guerra, es desde ya nada deseable, pero no inviable.

En la actualidad, en nuestra vida cotidiana nos encontramos constantemente comunicándonos por canales inseguros, y necesitamos poder encriptar nuestra información sin necesidad de encontrarnos previamente con la otra parte, lo cual sí nos resulta inviable. Este problema es resuelto gracias a las ideas que desarrollamos a continuación.

2. Criptografía de clave pública

Alicia y Beto desean compartir un secreto sin encontrarse, utilizando un canal de comunicación que es *a priori* inseguro: Miranda podría ver toda la información que se transmiten. A primera vista su objetivo parece difícil de cumplir, pero Diffie y Hellman mostraron que es posible. A grandes rasgos, el esquema consiste en:

- Un conjunto $\{g\}$ de claves *públicas*.
- Un conjunto $\{a, b, \dots\}$ de claves *privadas*.
- Para cada par de claves privadas a, b, una par de funciones e_a, e_b (de encriptado) que satisfacen

$$e_b(e_a(g)) = e_a(e_b(g)) \qquad \text{para toda clave pública } g. \tag{2.2}$$

Para comunicarse Alicia y Beto deben primero acordar una clave pública g. Una vez hecho esto, el protocolo a seguir es el siguiente:

- Alicia elige una clave privada a, calcula $A = e_a(g)$ y se lo envía a Beto.
- Beto elige una clave privada b, calcula $B = e_b(g)$ y se lo envía a Alicia.
- Alicia calcula, en privado, $e_a(B)$.
- Beto calcula, en privado, $e_b(A)$.

Gracias a (2.2) se tiene que $e_a(B) = e_b(A)$. De esta manera, Alicia y Beto comparten este secreto, que llamaremos C.

Como antes, asumimos que Miranda pudo interceptar la comunicación y que conoce los mensajes encriptados A y B. Además conoce g, que es público, y sabe cómo utilizar las funciones de encriptado si conoce las claves privadas a o b; por lo tanto, en tal caso podría obtener C. Por eso, al igual que antes, el protocolo será seguro solamente si es difícil obtener a o b a partir de $A = e_a(g)$ o $B = e_b(g)$.

Remarcamos que no necesariamente este tipo de protocolos vienen a reemplazar los de clave privada descriptos en la sección anterior, sino que pueden utilizarse complementariamente: por ejemplo, utilizando el secreto compartido C que se obtuvo como clave privada para un protocolo como el de Vigenère.

2.1. El protocolo de Diffie-Hellman

Tal como el cifrado de Vigenère, el protocolo de Diffie-Hellman está basado en la aritmética modular. Pero en lugar de realizar simples sumas y restas, la operación utilizada para encriptar es la exponenciación.

Concretamente, Alicia y Beto comienzan eligiendo un primo p; todos los cálculos que harán serán módulo p. Hecho esto, eligen un entero g módulo p, que no sea divisible por p. Para ser precisos, la clave pública que usarán será entonces el par (p, g). Sus claves privadas serán enteros a y b, y las funciones de encriptado las dadas por

$$e_a(x) = x^a \pmod p, \quad e_b(x) = x^b \pmod p.$$

Notemos que la propiedad (2.2) se verifica gracias a que

$$(g^a)^b \equiv (g^b)^a \pmod p.$$

Más aún, por el pequeño teorema de Fermat, que afirma que $x^{p-1} \equiv 1 \mod p$ siempre que x no sea divisible por p, a efectos de los cálculos a realizar los enteros a y b pueden ser considerados módulo $p-1$.

Por ejemplo, tomemos $p = 101$. Elegimos, por razones que se explicarán en breve, $g = 2$. Si los exponentes secretos son $a = 20$ y $b = 17$, entonces:

- Alicia calcula $A \equiv g^a \equiv 2^{20} \equiv 95 \mod 101$, y se lo envía a Beto.
- Beto calcula $B \equiv g^b \equiv 2^{17} \equiv 75 \mod 101$, y se lo envía a Alicia.
- Alicia calcula $B^a \equiv 75^{20} \equiv 36 \mod 101$.
- Beto calcula $A^b \equiv 95^{17} \equiv 36 \mod 101$.

De esta manera, Alicia y Beto comparten el secreto $C \equiv 36 \mod 101$.

2.2. El problema del logaritmo discreto

Miranda conoce el primo p. Por eso, para empezar, necesitamos que p sea *grande*. Si por el contrario, por ejemplo en el caso que consideramos arriba, p es chico, habrá solo p posibilidades para el secreto C, y Miranda puede probar uno por uno hasta dar con la clave. Por ejemplo, un primo p de 26 dígitos es grande: si Miranda puede analizar un millón de secretos C por segundo, le llevaría más de 10^{13} años analizar los p secretos posibles. Eso es demasiado tiempo: la edad del universo está estimada en 10^{10} años.

Paréntesis. Como remarcamos, es deseable que evaluar las funciones involucradas en nuestros protocolos sea sencillo y rápido. En este caso, si p es grande y a es del orden de p, entonces para calcular $A = g^a \mod p$ Alicia debe realizar del orden de p multiplicaciones. Por eso, para el cálculo sea viable, no se puede exponenciar ingenuamente; se utiliza, por ejemplo, el algoritmo de *exponenciación rápida*, que permite calcular $g^a \mod p$ realizando del orden de $\log a$ multiplicaciones (véase [1], sección 1.3.2).

Siguiendo, necesitamos que g sea tal que los valores posibles de la función $x \mapsto g^x \mod p$ sean muchos. De lo contrario, tendríamos como recién un conjunto acotado de secretos C posibles, lo cual no es deseable. Una de las bondades de la aritmética modular es que para todo p siempre existirá un g tal que el conjunto de valores es tan grande como es posible: tiene $p-1$ elementos. En otras palabras, el grupo de unidades de los enteros no nulos módulo p es *cíclico*.

Supongamos entonces que elegimos p y g "correctamente". Miranda conoce g, A y B, y sabe que $g^a \equiv A \mod p$. Por lo tanto, para obtener $C \equiv B^a \mod p$ le basta con despejar a de la ecuación $A \equiv g^a \mod p$. Es decir, tiene que calcular el logaritmo de A en base g. Tratándose de cálculos módulo p, lo llamamos *logaritmo discreto*.

Por lo observado arriba, utilizar la *fuerza bruta*, esto es, recorrer los $p-1$ valores posibles de a hasta obtener uno que satisfaga $g^a \equiv A \mod p$, no es viable si p es grande.

Tal como hicieran Kasiski con el cifrado de Vigenère y Turing con la máquina Enigma, Shanks mostró que el problema del algoritmo discreto se

puede atacar con algo mejor que la fuerza bruta. Su algoritmo para calcular el logaritmo de A en base g, llamado *Pasos de bebé, pasos de gigante* (véase [1], sección 2.7), consiste en:

- Tomar $n = \lfloor 1 + \sqrt{p-1} \rfloor$.
- *Pasos de bebé*: calcular la lista

$$1, g, g^2, g^3, \ldots, g^n \pmod{p}.$$

- *Pasos de gigante*: calcular la lista

$$A, g^{-n}A, g^{-2n}A, g^{-3n}A, \ldots, g^{-n^2}A \pmod{p}.$$

- Hallar una *colisión* entre ambas listas: esto es, un par de valores i, j tales que $g^i = g^{-jn}A$.
- Devolver $a = i + jn$.

Notemos que para calcular ambas listas necesitamos n multiplicaciones, por lo que en total serán necesarias del orden de $2\sqrt{p}$ multiplicaciones: muchas menos que las p que necesita la fuerza bruta.

En el algoritmo está implícita la afirmación de que existe una colisión entre ambas listas: ese fue el descubrimiento de Shanks. La demostración de este hecho excede los límites del presente trabajo.

Epílogo. El problema del logaritmo discreto excede a la aritmética modular: puede ser planteado en un grupo abeliano G cualquiera. El algoritmo de Shanks, como puede observarse, funciona en este contexto general.

Si bien mejora al de fuerza bruta, la cantidad de operaciones que necesita el algoritmo de Shanks sigue siendo *exponencial* en el logaritmo de p. Hay otros algoritmos, exclusivos de la aritmética modular, que son sensiblemente más rápidos: por ejemplo, el *cálculo de índices* resulta subexponencial (véase [1], sección 3.8). Por eso resultan de interés para la criptografía las *curvas elípticas* sobre cuerpos finitos, ya que son grupos abelianos cuya estructura es lo suficientemente complicada como para que hasta ahora, eligiendo los parámetros de manera adecuada, el algoritmo de Shanks sea el ataque más rápido que se conoce.

Breve biografía del autor

Nicolás Sirolli se graduó como licenciado en Ciencias Matemáticas y como doctor en Matemática por la Facultad de Ciencias Exactas y Naturales de la UBA. Actualmente reviste como profesor adjunto de la misma facultad y como investigador CONICET del Instituto IMAS. Se dedica a la investigación en teoría de números y algunas de sus conexiones con la criptografía.

Referencias

[1] J. Hoffstein, J. Pipher y J. H. Silverman. *An introduction to mathematical cryptography*. 2nd. Undergraduate Texts in Mathematics. Springer, 2014.