

# El criptosistema de Merkle-Hellman

Anna Piezrak

Diego Kiedanski

Tobias Winkler

Criptografía y Teoría de Códigos

Universidad Complutense de Madrid

May 23, 2017

## 1 Introducción

Cuando enviamos mensajes por cualquier canal de comunicación siempre nos exponemos al riesgo de que sean leídos por terceras personas antes de llegar a su destino. Supongamos que queremos mandar el mensaje

$$M = \text{COMPLUTENSE}$$

a nuestro compañero Bob. Para evitar que Eva, una tercera persona que de alguna manera ha conseguido interceptar  $M$  en su camino, sea capaz de entender lo que le queremos decir a Bob, le podemos mandar una versión *cifrada* de  $M$ , por ejemplo

$$M' = \text{DPNQMVUFOTF}.$$

Sin embargo, si hacemos eso tenemos que asegurarnos de que el destinatario Bob es capaz de *descifrarlas* y sería ideal si él fuese la única persona con esa capacidad.

Para ese fin ciframos  $M$  de tal manera que el resultado  $M'$  dependa de una clave  $K$ . Más formal, podemos pensar en el cifrado como una función

$$e_K : \mathbb{M} \rightarrow \mathbb{M}'$$

que depende de  $K$ . Aquí  $\mathbb{M}$  y  $\mathbb{M}'$  serían los espacios de todos los mensajes y mensajes cifrados posibles, respectivamente. En el ejemplo anterior  $e_K$  es la función definida por

$$e_K(m_1 \dots m_n) = m_1 + K \dots m_n + K$$

donde la adición de una letra  $m$  más un número entero  $K$  significa avanzar  $m$  por  $K$  posiciones en el alfabeto (posiblemente volviendo de Z a A) y la clave que elegimos fue  $K = 1$ .

Solo las personas que disponen de  $K$  serán capaces de entender, es decir descifrar, el mensaje original  $M$ . Igual que antes, podemos considerar este proceso como otra función

$$d_K : \mathbb{M}' \rightarrow \mathbb{M}$$

nuevamente dependiente de  $K$ . Entonces, en el ejemplo  $d_K$  sería dado por

$$e_K(m_1 \dots m_n) = m_1 - K \dots m_n - K.$$

¡Ahora solo nos tenemos que preocupar de que Bob sepa  $K$  y ya estaremos listos para mandarle todos los mensajes que queramos!

## Cifrado asimétrico

¿Pero cómo nos podemos comunicar con Bob para intercambiar  $K$ ? En realidad, eso es equivalente a enviar un mensaje con el contenido  $K$  y si ese mensaje es interceptado, entonces la comunicación ya no es segura.

Para evitar este problema se han inventado los *cifrados asimétricos*. Usan por lo menos dos claves, una clave *secreta*  $S$  y una clave *pública*  $P$ . Para que le podamos enviar un mensaje a Bob necesitamos saber su clave pública  $P$  y él va a tener que usar su clave privada  $S$  para descifrarla. Con la notación de antes, el cifrado es una función  $e_P$  dependiente de  $P$  y el descifrado es  $d_S$  que depende de  $S$ . El cifrado debe ser tal que sea muy difícil (en la práctica imposible) invertir  $e_P$  solo conociendo la clave pública  $P$ .

## 2 El cifrado de Merkle-Hellman

Es un ejemplo de cifrados asimétricos. Antes de explicar como funciona tenemos que introducir un poco de teoría.

### El problema de la mochila

Dado un conjunto finito  $M \subset \mathbb{Z}$  y un límite  $L \in \mathbb{Z}$ , el *problema de la mochila* consiste en encontrar un subconjunto  $S \subseteq M$  que maximice  $R := \sum_{s \in S} s$  bajo la restricción que  $R \leq L$ .

Un problema relacionado es el *problema de la suma de subconjuntos*. Aquí la pregunta es: ¿Existe  $S \subseteq M$  tal que  $\sum_{s \in S} s = L$ ? Ese problema se puede considerar como un caso especial de él de la mochila y sigue siendo NP-completo. A continuación vamos a referirnos a este problema cuando decimos 'el problema de la mochila'.

En casos especiales, el problema de la mochila es fácil de resolver como vamos a ver enseguida.

**Definición** Sea  $M = \{m_1, \dots, m_n\} \subset \mathbb{Z}$  finito y supongamos, sin pérdida de generalidad, que  $m_{i+1} \geq m_i$  para  $i = 1, \dots, n-1$ . Si  $M$  verifica que

$$m_{i+1} \geq \sum_{k=1}^i m_k$$

entonces  $M$  se llama *mochila supercreciente*.

### Ejemplo

- $M := \{3, 4, 11, 42\}$  es una mochila supercreciente porque  $4 > 3$ ,  $11 > 3 + 4 = 7$  y  $42 > 3 + 4 + 11 = 18$ .
- Para  $n \in \mathbb{N}$ ,  $M := \{2^0, 2^1, \dots, 2^n\}$  también es una mochila supercreciente ya que todo  $i \in \mathbb{N}$  verifica que

$$2^{i+1} - 1 = \sum_{k=0}^i 2^k.$$

Observamos que si  $M = \{m_1, \dots, m_n\}$  es supercreciente y  $m_n \leq L$ , entonces si  $M$  tiene una solución  $S$ , es necesario que  $m_n \in S$  porque si no, no podremos alcanzar  $L$  ya que  $m_n$  es mayor que la suma de todos los demás elementos de  $M$ . De esta observación podemos concluir el siguiente algoritmo:

```

1: procedure RESOLVERMOCHILASUPERCRECIENTE( $\{m_1, \dots, m_n\}, L$ )
2:    $S \leftarrow \emptyset$ 
3:   for  $i = n, \dots, 1$  do
4:     if  $m_i \leq L$  then
5:        $S \leftarrow S \cup \{m_i\}$ 
6:        $L \leftarrow L - m_i$ 
7:   return  $S$ 

```

El algoritmo asuma que  $M = \{m_1, \dots, m_n\}$  es una mochila supercreciente y que los  $m_i$  están en orden ascendente. Su complejidad de tiempo es  $O(n)$  (lineal en  $n$ ) porque consiste de un solo bucle de exactamente  $n$  iteraciones.

## Cifrar y la clave pública

Para cifrar un bloque  $B = b_1 \dots b_n$  de  $n$  bits tomamos una mochila supercreciente  $M = \{m_1, \dots, m_n\}$  de longitud  $n$ . Entonces elegimos dos números  $q, r \in \mathbb{Z}$  tal que  $q > \sum_{i=1}^n m_i$ ,  $r > 1$  y el máximo común divisor de  $q$  y  $r$  sea 1. Llamamos *modulo* a  $q$  y *multiplicador* a  $r$ . Calculamos

$$M' = \{m'_1, \dots, m'_n\} = \{rm_1 \bmod q, \dots, rm_n \bmod q\}.$$

Es importante entender que  $M'$  en general ya no es supercreciente debido a las operaciones de modulo. Obtenemos el bloque cifrado  $B'$  sumando aquellos elementos de  $M'$  cuyos índices corresponden a los bits que valen uno en nuestro bloque  $B$ , es decir hacemos la suma

$$B' = \sum_{i=1}^n b_i m'_i.$$

$B'$  es un solo número en  $\mathbb{Z}$ .

**Ejemplo** Supongamos que queremos cifrar el número 157. Su representación binaria es  $B = 10011101$ . Entonces, como explicado antes, tenemos que elegir una mochila supercreciente de longitud 8, por ejemplo

$$M = \{1, 3, 6, 11, 27, 53, 111, 213\}$$

y los números  $r$  y  $q$ . Como la suma de todos los elementos en  $M$  es igual a 425, podemos tomar  $q = 499$  y  $r = 101$ . Como 101 es primo,  $\text{mcd}(q, r) = 1$ . La mochila  $M'$  resultante sería

$$M' = \{101, 303, 107, 113, 232, 363, 233, 56\}$$

que claramente no es supercreciente. Ya podemos calcular el bloque cifrado:

$$B' = 101 + 113 + 232 + 363 + 56 = 865$$

Para cifrar solo se necesita la mochila transformada  $M'$ , por eso la clave pública es  $K_P = M'$ .

## Descifrar y la clave secreta

Como da igual si primero sumamos y después tomamos modulo o al revés, la siguiente equivalencia es correcta:

$$B' = \sum_{i=1}^n b_i(rm_i \mod q) \equiv r \left( \sum_{i=1}^n b_i m_i \right) \mod q$$

y por lo tanto

$$\sum_{i=1}^n b_i m_i \equiv r^{-1} \cdot B' \mod q.$$

$r^{-1}$  existe porque  $\text{mcd}(r, q) = 1$ . Ahora para averiguar los  $b_i$  solo tenemos que resolver el problema de mochila con  $M$  supercreciente y eso se puede hacer muy rápido como hemos visto antes.

**Ejemplo** Tenemos  $r^{-1} = 101^{-1} = 84 \mod 499$  lo que se puede calcular eficientemente con el algoritmo extendido de Euclides. Entonces,

$$r^{-1} \cdot B' = 84 \cdot 865 \equiv 305 \mod 499.$$

Lo único que falta es llamar el algoritmo `RESOLVERMOCHILASUPERCRECIENTE(M, 305)` que nos da el resultado  $305 = 1 + 11 + 27 + 53 + 213$ . Estos números de la mochila  $M$  corresponden precisamente al bloque original 10011101.

Para descifrar se necesita  $r^{-1}$ ,  $q$  y la mochila original  $M$ , por lo tanto tenemos la clave secreta  $K_S = (M, r^{-1}, q)$ .

## 3 Implementación

Hemos implementado el sistema de Merkle-Hellman en Python (usando la librería Sage???). Existen los programas de consola **generate-keys**, **encrypt** y **decrypt** que se utilizan como se explica a continuación:

- **generate-keys [k]**: Crea dos archivos *privkey* y *pubkey* que forman una pareja de claves. Se van a generar claves (mochilas) de longitud  $16k$ . El valor por defecto de  $k$  es ???.

- **encrypt message key:** Genera el archivo *message\_encrypted* que se obtiene cifrando el mensaje *message* con la clave pública *key*.
- **decrypt message key:** Genera el archivo *message\_decrypted* que es el resultado de descifrar el mensaje *message* con la clave secreta *key*.

Aparte de esto también creamos el programa **brute-force** que, dado un mensaje cifrado y la clave pública, intenta reconstruir el mensaje original resolviendo el problema de la mochila *no supercreciente* tras probar todas las posibles combinaciones. Este programa solo sirve para evaluar la seguridad y no forma parte de criptosistema.