Universidad de Valladolid

Facultad de Ciencias

Trabajo Fin de Grado

Grado en Matemáticas

Criptosistemas de clave pública basados en el problema de las mochilas

Autor: Mario Alonso Cardeñosa

Tutor: Juan Tena Ayuso

Índice general

1. Introducción **2**

2. Criptosistema de Merkle-Hellman **6**
2.1. Descripción del método 9

3. Variantes **13**
3.1. Método de varias iteraciones 13
3.2. Chor-Rivest 14
3.2.1. Descripción del método: 14
3.2.2. Algoritmo auxiliar para representar los mensajes: . . . 16

4. Ataques a los criptosistemas basados en el problema de las mochilas **18**
4.1. Ataque de A. Shamir 19
4.2. Ataque de baja densidad 20
4.2.1. Algoritmo 22
4.2.2. Comentarios sobre el ataque 23

5. Análogo en códigos correctores, el criptosistema de McEliece **26**
5.1. Códigos correctores de errores 27
5.2. El criptosistema de McEliece 28
5.2.1. El método 29
5.3. Pros y contras 30
5.4. Códigos de Goppa 32

Capítulo 1

Introducción

La palabra *criptografía* proviene del griego "*kryptós*", oculto, y "*graphein*", escritura. Literalmente significa "escritura oculta". Hoy en día la criptografía (o criptología) es una ciencia que trata la práctica y el estudio de técnicas para una comunicación segura entre dos partes o *usuarios* en presencia de una tercera parte o *atacante*.

La criptografía consiste en encriptar o cifrar un mensaje para que cuando sea transmitido, en caso de ser interceptado por un posible atacante, éste no sea capaz de leer el mensaje. Hay que distinguir dos tipos de criptografía: la de clave privada y la de clave pública. La primera es mucho más antigua que la segunda. El primer sistema de cifrado de clave privada que se conoce es el código de César (utilizado por el emperador romano Julio César según el testimonio del historiador romano Suetonio), en el que se sustituían unas letras por otras siguiendo un criterio determinado. El segundo tipo de criptografía data del siglo pasado. Al mismo tiempo que avanzaba la criptografía, evolucionaban las formas de cifrado y se proponían nuevos sistemas para encriptar mensajes; otra rama crecía con ella, el criptoanálisis. De igual manera que un criptógrafo se preocupa por proteger el mensaje enviado, un criptoanalista intenta recuperar el mensaje original sin conocer las claves del sistema. Cuando una persona consigue descifrar un mensaje cifrado mediante un sistema sin conocer las claves utilizadas decimos que esa persona ha *roto* dicho sistema.

En este trabajo se habla de una pequeña parte de la criptografía de clave pública, más concretamente de los criptosistemas basados en el problema de las mochilas, así como de su criptoanálisis.

En el año 1976 Diffie y Hellman publicaron el artículo fundacional para la criptografía de clave pública, "New Directions in Cryptography"[4]. En su obra daban indicaciones de cómo debería funcionar un criptosistema de clave pública y qué propiedades debería tener, aunque sin llegar a proponer uno en concreto. Pero ¿qué significa criptografía de clave pública? Hasta el momento se había utilizado la criptografía de clave privada, es decir, dos personas que querían enviarse mensajes se ponían de acuerdo en un sistema de cifrado, como por ejemplo el AES (que es el utilizado hoy en día), y elegían una clave común para comunicarse que debía ser transmitida a través de canales no cifrados. La propuesta de Diffie y Hellman da una buena manera de hacer este intercambio de forma segura.

Supongamos que un grupo de personas quiere comunicarse a través de un sistema de clave pública. Cada uno de ellos tendrá dos funciones, el usuario i por ejemplo, tendrá c_i y d_i que sirven para cifrar y descifrar mensajes, respectivamente. La función c_i se hará pública, esto es, todo el mundo podrá saber cómo cifrar un mensaje con dicha función. La función d_i en cambio, solamente será conocida por el usuario i. De esta forma, si i quiere enviar un mensaje a j solo tiene que cifrarlo con la función c_j y j lo descifrará fácilmente y de forma segura puesto que él es el único que conoce la función d_j.

Para que todo el sistema funcione de forma óptima, Diffie y Hellman enunciaron unas propiedades que cualquier criptosistema de clave pública debe cumplir:

1.- Fabricación de c_i, d_i computacionalmente simple.

2.- Cifrado computacionalmente simple.

3.- Descifrado (para el receptor legal, conociendo d_i) computacionalmente simple.

4.- Obtención de d_i conocido c_i computacionalmente imposible.

5.- Obtención del mensaje conocidos c_i y el mensaje cifrado computacionalmente imposible.

Estas funciones c se conocen como funciones de una vía. Son funciones que tienen inversa (d) pero es "imposible"de calcular. Las funciones de una vía con trampilla o con trampa son las que se utilizan habitualmente en la criptografía de clave pública. Son funciones que tienen inversa pero es "imposible"calcularla sin tener una información complementaria (la clave privada) que hace que sea muy fácil de calcular.

Aunque Diffie y Hellman no dieron ningún ejemplo concreto de estas funciones, en seguida aparecieron tres propuestas basadas en tres problemas matemáticos. Son las siguientes:

- R.S.A. (Rivest, Shamir, Adleman, 1978) [19], basado en la factorización de enteros como producto de números primos.
- ElGamal (Taher ElGamal, 1985) [5], basado en el problema del logaritmo discreto.
- Merkle-Hellman (1978) [15], basado en el problema de las mochilas.

Los criptosistemas ElGamal y RSA siguen funcionando hoy en día y son seguros. Sin embargo, el que se trata en este texto, el criptosistema de Merkle-Hellman, ya no se utiliza puesto que Adi Shamir, la S del RSA, encontró una manera de recuperar el mensaje cifrado en el año 1983 [20].

Aun así, los criptosistemas basados en el problema de las mochilas siguen siendo de interés y se sigue investigando para hallar uno que sea seguro puesto que ofrecen ciertas ventajas frente al RSA o ElGamal, como la velocidad de cifrado. En un criptosistema basado en el problema de las mochilas el tiempo necesario para cifrar un mensaje es lineal en el número de pesos del problema, dado que simplemente se está haciendo una suma. Esto hace que un criptosistema de este tipo sea mucho más rápido que el RSA, en el que el cifrado se lleva a cabo mediante una exponenciación en un cuerpo finito calculada utilizando el algoritmo de exponenciación modular, que tiene un tiempo de ejecución polinomial.

Otra ventaja muy importante de los criptosistemas basados en el problema de las mochilas frente al RSA o ElGamal es que el problema de las mochilas es un problema NP-completo, mientras que la factorización o el logaritmo discreto no lo son. Por ahora, utilizando la tecnología de la que disponemos no se puede recuperar un mensaje cifrado mediante RSA o ElGamal, sin embargo, existe una forma de computación diferente a la computación clásica que es la computación cuántica. La computación cuántica utiliza qubits en lugar de bits, lo que da lugar a nuevos algoritmos y hace que problemas intratables (para la computación clásica) sean tratables (para la computación cuántica). Entre estos problemas se encuentran tanto la descomposición en factores primos en la que se basa el RSA como el logaritmo discreto en el que se basa ElGamal. De hecho, existe un algoritmo que se puede utilizar para resolver estos problemas y romper así los dos criptosistemas. Se trata del algoritmo ideado por Shor [21] en el año 1997. El único problema es que

para hacer de la computación cuántica (y por lo tanto de este algoritmo) una realidad es necesario que se construya el computador cuántico. Ya se han diseñado modelos de computador cuántico pero su uso está muy lejos de ser extendido. Según algunas empresas de informática, dentro de unos 10 o 12 años la computación cuántica podría llegar a empresas y hogares.

El algoritmo de Shor permitiría romper el RSA y ElGamal pero no es aplicable a un caso general del problema de las mochilas. Esto hace que encontrar un nuevo criptosistema basado en dicho problema que sea seguro es una cuestión muy importante de cara al futuro de la criptografía.

En el trabajo se explica el criptosistema propuesto por Merkle y Hellman, el primero que utilizaba el problema de las mochilas como base para cifrar un mensaje. Después, en el capítulo 3, se exponen dos ataques criptoanalíticos, el primero es el ataque de Shamir para el criptosistema de Merkle y Hellman y el segundo es el ataque de baja densidad, propuesto por Lagarias y Odlyzko, y que se puede utilizar para atacar cualquier criptosistema basado en el problema de las mochilas. En el capítulo 4 se explican las dos variantes más relevantes entre las muchas surgidas. Por último, el capítulo 5 es una analogía en códigos correctores, el criptosistema de McEliece, que utiliza la misma idea principal que el criptosistema de Merkle y Hellman utilizando matrices generatrices de códigos en lugar de sucesiones.

Capítulo 2

Criptosistema de Merkle-Hellman

El criptosistema de Merkle-Hellman [15] es el primero que se publicó basado en el problema de las mochilas. El problema de las mochilas es un problema clásico que fue formulado en términos de libros y mochilas, de ahí su nombre. Básicamente se trata de repartir un número total de libros entre varias mochilas de capacidad menor o igual al número total de libros y siempre llenándolas. Es un problema que se ha demostrado que pertenece a la clase de problemas NP-completos [16], es decir, no existe un algoritmo de tiempo de ejecución polinomial que resuelva el problema general. Vamos a plantear el problema formalmente:

Sea $A = \{a_1, \ldots, a_n\} \subset \mathbb{N}$ el *conjunto de pesos* del problema. Dado $S \in \mathbb{N}$, determinar, si existe, un subconjunto del conjunto de pesos tal que al sumar todos sus elementos obtengamos S. Esto es equivalente a determinar un conjunto $R = \{x_1, \ldots, x_n\}$ tal que

$$\sum_{i=1}^{n} x_i a_i = S,\ x_i \in \{0, 1\} \text{ para todo } i$$

Si se utiliza A como clave pública, el problema general es muy difícil de resolver, tanto para un potencial atacante como para el receptor legítimo (ya que tiene complejidad exponencial en n), y no se puede utilizar directamente. Hay varios estudios que han conseguido algoritmos para resolver este problema y el más rápido necesita $n2^{n/2}$ operaciones para obtener una solución, siendo n el número de pesos del problema (ver [17]).

No obstante, existen versiones fáciles del problema, casos en los que existe un algoritmo muy rápido para resolver el problema. Uno de ellos es el caso en el que los pesos forman una sucesión supercreciente y es el que utilizaron

Merkle y Hellman para su criptosistema. Lo que nos lleva a la siguiente definición.

Definición: *Se dice que una sucesión* $(a_i)_{i=1}^n$ *es supercreciente si*

$$a_i > \sum_{j=1}^{i-1} a_j, \text{ para } i = 2, \ldots, \ n$$

Ejemplo: La sucesión $\{2, 4, 8, 16, 32, 64, 128, 256\}$ forma una sucesión supercreciente puesto que $a_2 = 4 > a_1 = 2, \ a_3 = 8 > \sum_{i=1}^{2} a_i = 6, \ldots, \ a_8 = 256 > \sum_{i=1}^{7} a_i = 254$.

Si elegimos los pesos de forma que compongan una sucesión supercreciente podremos utilizar un algoritmo que resuelve el problema de las mochilas en tiempo lineal y que se describe a continuación.

Sea $A = \{a_1, \ldots, a_n\}$ el conjunto de pesos del problema que forma una sucesión supercreciente y sea S el número dado.

Algoritmo: Iniciamos el algoritmo con $x_i = 0 \ \forall i \in \{1, \ldots, n\}$ y entonces

$$x_n = 1 \text{ si y solo si } S > \sum_{i=1}^{n-1} a_i$$

Una vez determinado el valor de x_n, digamos $z \in \{0, 1\}$, nos enfrentamos a otro problema más fácil del mismo tipo, determinar $R = \{x_1, \ldots, x_{n-1}\}$ tal que

$$S - za_n = \sum_{i=1}^{n-1} x_i a_i, \ x_j \in \{0, \ 1\} \ 1 \leq i \leq n-1$$

Por lo tanto podemos encontrar fácil y rápidamente todos los valores de x_i de forma recursiva, obteniendo así el mensaje $(x_1, \ldots, x_n)$.

Ejemplo: Utilizaremos para ilustrar el algoritmo la sucesión supercreciente del ejemplo anterior, $\{a_1 = 2, 4, 8, 16, 32, 64, 128, 256 = a_8\}$. Supongamos que el mensaje a enviar es $(x_1 = 1, 0, 1, 1, 0, 1, 1, 0 = x_8)$. Entonces el emisor calcularía $\sum x_i a_i = 2 + 8 + 16 + 64 + 128 = 218$ y nos enviaría $S = 218$. Entonces nosotros aplicaríamos el algoritmo de la siguiente forma:

Para x_8:

$$\sum_{i=1}^{7} a_i = 2{+}4{+}8{+}16{+}32{+}64{+}128 = 254 > S \Rightarrow x_8 = 0;\ S := S{-}x_8 a_8 = 218;$$

Para x_7:

$$\sum_{i=1}^{6} a_i = 2{+}4{+}8{+}16{+}32{+}64 = 126 < S = 218 \Rightarrow x_7 = 1;\ S := S{-}x_7 a_7 = 90;$$

Para x_6:

$$\sum_{i=1}^{5} a_i = 2+4+8+16+32 = 62 < S = 90 \Rightarrow x_6 = 1;\ S := S - x_6 a_6 = 26;$$

Para x_5:

$$\sum_{i=1}^{4} a_i = 2+4+8+16 = 30 > S = 26 \Rightarrow x_5 = 0;\ S := S - x_5 a_5 = 26;$$

Para x_4:

$$\sum_{i=1}^{3} a_i = 2+4+8 = 14 < S = 26 \Rightarrow x_4 = 1;\ S := S - x_4 a_4 = 10;$$

Para x_3:

$$\sum_{i=1}^{2} a_i = 2+4 = 6 < S = 10 \Rightarrow x_3 = 1;\ S := S - x_3 a_3 = 2;$$

Ahora vemos que S tiene el valor exacto de uno de los pesos restantes, por lo tanto haríamos $x_2 = 0$ y $x_1 = 1$. Hemos obtenido el mensaje original $(1, 0, 1, 1, 0, 1, 1, 0)$.

El uso de una sucesión supercreciente como clave pública haría que fuese muy fácil para el receptor descifrar el mensaje, pero también haría que fuese muy fácil para un potencial atacante. La idea de muchos de los criptosistemas basados en el problema de las mochilas es empezar con una sucesión de pesos $b_1, \ldots, b_n$ que haga el problema muy fácil de resolver (por ejemplo

una sucesión supercreciente) y transformarla en otra, $a_1, \ldots, a_n$, que es la que se publica y aparentemente no tiene ninguna estructura. Igualmente, tiene que haber una manera de volver a la sucesión original que sea fácil de implementar para el diseñador del sistema y le permita regresar al problema fácil para obtener el mensaje.

La forma usual para hacer esto es la multiplicación modular utilizada por Merkle y Hellman.

2.1. Descripción del método

En esta sección se explica el método básico de Merkle y Hellman. El receptor, que construye el sistema para que otras personas le envíen información, necesita una serie de elementos que tiene que precomputar para que sea posible el cifrado y descifrado. Empieza por construir una sucesión de pesos supercreciente $b_1, \ldots, b_n$, asimismo, elige dos enteros positivos M y W tales que

$$M > \sum_{i=1}^{n} b_i \text{ y } mcd(M, W) = 1,$$

Más adelante discutiremos qué número n y qué tamaño de los parámetros es el adecuado.

Entonces el usuario calcula

$$a'_i \equiv b_i W \bmod M,\ 0 < a'_i < M.$$

Nótese que no podemos tener $a'_i = 0$ puesto que $mcd(M, W) = 1$ y $M \gneq b_i$. Ahora el diseñador elige una permutación aleatoria σ de $\{1, \ldots, n\}$ y define

$$a_i = a'_{\sigma(i)},\ 1 \leq i \leq n.$$

Los coeficientes a_i se publican, pero M, W, σ y los b_j se mantienen en secreto. Ya tenemos todo preparado para cifrar.

Cifrado: Una persona que quiere enviar un mensaje a este receptor tendría que escribirlo en código binario, obteniendo $(x_1, \ldots, x_n)$ y calcular:

$$S = \sum_{i=0}^{n} x_i a_i,$$

que es lo que enviará.

Descifrado: Para el descifrado hay que calcular previamente W^{-1}, el inverso multiplicativo de W módulo M. Una vez hecho esto, el receptor (que conoce M y W) calcula:

$$C \equiv SW^{-1} \bmod M,\ 0 \leq C < M.$$

Se tiene la siguiente sucesión de equivalencias:

$$C \equiv SW^{-1} \bmod M$$

Sustituyendo S por su valor,

$$C \equiv \sum_{i=1}^{n} x_i a_i W^{-1} \bmod M$$

$$\equiv \sum_{i=1}^{n} x_i a'_{\sigma(i)} W^{-1} \bmod M$$

$$\equiv \sum_{i=1}^{n} x_i b_{\sigma(i)} \bmod M.$$

La última equivalencia es debido a que $a'_i \equiv b_i W \bmod M$, entonces $a'_{\sigma(i)} \equiv b_{\sigma(i)} W \bmod M$ y por lo tanto $a'_{\sigma(i)} W^{-1} \equiv b_{\sigma(i)} \bmod M$

Ahora, como $M > \sum b_i$, la condición $0 \leq C < M$ implica que

$$C = \sum_{i=1}^{n} x_i b_{\sigma(i)}.$$

Ahora resolvemos el problema fácil con la sucesión supercreciente que habíamos elegido y obtenemos $(x_{\sigma(1)}, \ldots, x_{\sigma(n)})$ puesto que los b_i, $i = 1, \ldots, n$ están permutados. Aplicamos la permutación inversa a σ, σ^{-1} y obtenemos finalmente los valores del mensaje original $(x_1, \ldots, x_n)$.

Ejemplo: Para entender mejor todo el proceso de cifrado y descifrado vamos a hacer un ejemplo. Tomaremos como sucesión supercreciente la siguiente sucesión: $\{b_1, \ldots, b_7\} = \{2, 4, 10, 17, 36, 67, 140\}$. Ahora elegimos $M = 281$ y $W = 92$. Se cumplen las condiciones para M y W, $mcd(M, W) = 1$ y $M > \sum b_i = 273$. Ahora calculamos los a'_i, $i = 1, \ldots, 7$.

$$a'_1 = 2 \cdot 92 \equiv 184 \ mod\ 281;\ a'_2 = 4 \cdot 92 \equiv 97 \ mod\ 281$$
$$a'_3 = 10 \cdot 92 \equiv 77 \ mod\ 281;\ a'_4 = 17 \cdot 92 \equiv 159 \ mod\ 281$$
$$a'_5 = 36 \cdot 92 \equiv 221 \ mod\ 281;\ a'_6 = 67 \cdot 92 \equiv 263 \ mod\ 281$$
$$a'_7 = 140 \cdot 92 \equiv 235 \ mod\ 281$$

Ya tenemos el conjunto $\{a'_1, \ldots, a'_7\} = \{184, 97, 77, 159, 221, 263, 235\}$. Ahora elegimos una permutación $\sigma \in S_7$, por ejemplo elegimos $\sigma = (135)(724)(6)$ cuya permutación inversa es $\sigma^{-1} = (6)(274)(315)$. Aplicamos la permutación como se ha explicado ($a_i = a'_{\sigma(i)}$) para obtener el conjunto de pesos que publicaremos. Tenemos por lo tanto $\{a_1, \ldots, a_7\} = \{77, 159, 221, 235, 184, 263, 97\}$. Por último, calculamos el inverso de W, $W^{-1} \equiv 168 \ mod\ 281$. Ya tenemos todo preparado para publicar los pesos y empezar a recibir mensajes.

Vamos ahora con el ejemplo de cifrado y descifrado. Supongamos que alguien nos quiere enviar un mensaje $(x_1, \ldots, x_7) = (1, 0, 1, 1, 0, 1, 0)$, entonces el emisor computaría y nos enviaría

$$S = \sum_{i=1}^{7} x_i a_i = 77 + 221 + 235 + 263 = 796$$

Entonces nosotros recibiríamos S y calcularíamos

$$C = SW^{-1} = 133728 \equiv 253 \ mod\ 281$$

Ahora tenemos

$$253 = x_1 b_{\sigma(1)} + \cdots + x_7 b_{\sigma(7)}$$

O lo que es lo mismo,

$$253 = x_5 b_1 + x_7 b_2 + x_1 b_3 + x_2 b_4 + x_3 b_5 + x_6 b_6 + x_4 b_7$$

Por lo tanto, al resolver este problema de las mochilas con sucesión supercreciente mediante el algoritmo explicado anteriormente, vamos a obtener el mensaje original permutado, esto es, obtendremos $(x_5, x_7, x_1, x_2, x_3, x_6, x_4) = (0, 0, 1, 0, 1, 1, 1)$. Ahora aplicamos la permutación σ^{-1} y obtenemos el mensaje original $(x_1, \ldots, x_7) = (1, 0, 1, 1, 0, 1, 0)$.

Parámetros: Los parámetros recomendados por Merkle y Hellman en su publicación [15] para un criptosistema seguro son los siguientes: $n = 100$, $b_1 \approx 2^n$, $b_j > \sum_{i=0}^{j-1} b_i$, $2 \leq j \leq n = 100$ y $b_n \approx 2^{2n}$

Nótese que si algunos b_i (y por lo tanto M) son muy grandes, la sucesión sería ineficiente, puesto que 100 bits de información serían codificados en $\log_2 M$ bits. De igual manera, es peligroso dejar que alguno de los b_i sea muy pequeño, por ejemplo, si $b_1 = 1$ entonces $a_j = W$ para algún j y como el sistema es mucho más fácil de romper conocido M o W, se podría intentar cada a_j como un posible W y comprometer el sistema.

Por ello la mejor configuración de parámetros es la propuesta inicialmente (teniendo en cuenta que el sistema ya no es seguro).

Se podría elegir $b_i = c2^{i-1}$, $1 \leq i \leq n$ para un cierto $c \approx 2^n$. Esto resulta en una sucesión válida en el sentido de que cumple las condiciones que se piden (ser supercreciente) pero resulta en un sistema muy inseguro. La razón es que para $1 \leq j \leq n-1$ tendríamos que $a'_{j+1} = 2a'_j$ o $a'_{j+1} = 2a'_j - M$, dependiendo de si $2a'_j < M$ ó $2a'_j > M$ y ocurriendo cada caso aproximadamente $n/2$ veces. Por lo tanto, computando $a_i - 2a_j$ para $1 \leq i, j \leq n$ obtendríamos unas $n/2$ veces $-M$ entre las n^2 restas, pudiendo deducir cuál es el mensaje y rompiendo el sistema.

Se han propuesto más posibilidades para los parámetros, como por ejemplo tomar $b_i = 2^{i-1}$ y escondiendo esta estructura con una doble multiplicación modular, como se explica en [9], aunque ninguna propuesta es segura ya en vista del ataque de Shamir que se explica en el capítulo 4.

El criptosistema de Merkle y Hellman era muy prometedor cuando apareció debido a su bajo coste computacional y por lo tanto era preferido antes que su principal competidor, el RSA. Así fue hasta que Adi Shamir, en el año 1983, rompió el sistema de Merkle y Hellman [20]. En su artículo publicaba un algoritmo de tiempo polinomial que permitía obtener el mensaje en claro a partir del mensaje cifrado y los pesos públicos del criptosistema $\{a_1, \ldots, a_n\}$.

Capítulo 3

Variantes

Las dos variantes aquí mencionadas no son ni mucho menos las únicas surgidas ni tampoco las más recientes, pero son la más relacionada con el método original de Merkle y Hellman y la que más tiempo estuvo sin ser rota. Desde el año 2009 hasta la fecha se han propuesto más de 5 nuevos criptosistemas basados en el problema de las mochilas, tales como [10] o [24], pero todas las propuestas han sido rotas a los pocos meses de aparecer, además de que muchas son muy parecidas entre sí o son mejoras de una propuesta anterior.

3.1. Método de varias iteraciones

Para hacer el método de Merkle-Hellman más seguro (e invulnerable al ataque de A. Shamir, como él mismo dijo en su publicación [20]), se puede utilizar el método iterado de h pasos de Merkle-Hellman, aunque este método fue roto por Adleman [1] un año después de que Shamir rompiera el método básico. Se trata simplemente de repetir las operaciones hechas antes, volviendo a cifrar el mensaje cifrado con unos nuevos valores. Más concretamente:

Supongamos que un usuario ha construido su sistema Merkle-Hellman. Sean entonces $M_1 = M$, $W_1 = W$, $a_i^{(0)} = b_i$ y $a_i^{(1)} = a'_i$. De esta forma se construye una sucesión de módulos, multiplicadores y combinaciones de pesos eligiendo sucesivamente M_k y W_k positivos tales que

$$mcd(M_k, W_k) = 1,\ M_k > \sum_{i=1}^{n} a_i^{(k-1)}$$

y definiendo

$$a_i^{(k)} \equiv a_i^{(k-1)} W_k \ mod\ M_k$$

.

Así, repetimos el cifrado del mensaje k veces. Este sistema es invulnerable al ataque de Shamir precisamente porque dicho ataque no recupera necesariamente los M y W originales.

Sin embargo, no es invulnerable a otros ataques propuestos con posterioridad a la publicación de este criptosistema, tales como el ataque de baja densidad, propuesto por Lagarias y Odlyzko, o el ya mencionado ataque directo de Adleman.

3.2. Chor-Rivest

El criptosistema Chor-Rivest [3] es otro método basado en el problema de las mochilas pero que difiere del de Merkle-Hellman en el hecho de que no convierte una sucesión secreta que resuelve el problema fácilmente en una aparentemente aleatoria como hacía el método explicado en el capítulo 2. En esta variante el papel principal lo juega la exponenciación en cuerpos finitos para los cuales el problema del logaritmo discreto es fácil de resolver (por ejemplo con el algoritmo de Pohlig-Hellman [18]).

Este criptosistema fue publicado en el año 1988 y no se descubrió una forma de romperlo hasta el año 1998 cuando Vaudenay publicó un artículo [23] en el que propone algoritmos operativos de tiempo de ejecución polinomial para recuperar mensajes cifrados por este método.

3.2.1. Descripción del método:

Al igual que en el criptosistema de Merkle-Hellman, el usuario que construye el sistema necesita una serie de elementos que debe computar a priori para que sea posible cifrar y descifrar mensajes.

1.- El usuario elige un cuerpo finito $\mathbb{F}_q = \mathbb{F}_{p^k}$ con $1 < k < p$ y tal que el problema del logaritmo discreto sea tratable en $\mathbb{F}_q^*$ (por ejemplo tomando q de forma que $q-1$ tenga solamente factores primos pequeños, ver [18]). Para representar el cuerpo $\mathbb{F}_q$ el constructor del sistema elige un polinomio irreducible aleatorio $f(X)$ de grado k con coeficientes en $\mathbb{F}_p$. Los elementos de $\mathbb{F}_q$ pueden ser representados por polinomios de grado menor que k con coeficientes en $\mathbb{F}_p$.

2.- Se elige un elemento α aleatorio primitivo en $\mathbb{F}_q$, es decir, que engendre todo $\mathbb{F}_q^*$. Nótese que el propio α es también un polinomio $g(X)$ de grado menor que k con coeficientes en $\mathbb{F}_p$.

3.- A continuación, el usuario debe computar, para todo $i = 0, \ldots, p-1$, los logaritmos $a_i = \log_\alpha(X + i)$. Esto es realizable gracias a la condición impuesta de que el problema logarítmico sea fácil en $\mathbb{F}_q^*$.

4.- Ahora, se selecciona $1 < d < q-1$ y una permutación $\sigma \in S_p$ y se computa $u_i \equiv a_{\sigma(i)} + d \ mod \ q-1$.

Ya estamos en condiciones de dar la clave pública: $u_0, \ldots, u_{p-1}, q$.

Nota sobre los mensajes: Un mensaje debe ser un número $0 \leq M < \binom{p}{k}$. El mensaje se representa (mediante un algoritmo auxiliar que se explica más adelante) como mensaje binario $(m_0, \ldots, m_{p-1})$ de longitud p y peso de Hamming k. Esto último quiere decir que exactamente k de los m_i son iguales a 1.

Cifrado: El emisor del mensaje calcula y envía

$$c = \sum_{i=0}^{p-1} m_i u_i \ mod \ q$$

.

Descifrado: El receptor calcula

$$C = c - kd$$

Y después computa

$$\alpha^C (= g(x)^C) = \alpha^{(\sum m_i u_i) - kd} = \alpha^{\sum m_i (a_{\sigma(i)} + d) - kd} =$$

$$= \alpha^{\sum m_i a_{\sigma(i)}} \alpha^{(\sum m_i) d - kd} = \alpha^{\sum m_i a_{\sigma(i)}} =$$

$$= \prod_{i=0}^{p-1} (\alpha^{a_{\sigma(i)}}) m_i$$

Y sustituyendo los logaritmos precomputados obtenemos:

$$\alpha^C = \prod_{i=0}^{p-1} (x + \sigma(i))^{m_i}$$

Esto quiere decir que

$$\alpha^C \equiv \prod_{i=0}^{p-1} (x + \sigma(i))^{m_i} \ mod \ f(X)$$

El polinomio en el miembro derecho es un polinomio mónico de grado exactamente k.

Ahora añadimos un múltiplo de $f(X)$ tal que $a(X) = \alpha^C + \beta f(X)$ sea mónico, $\beta \in \mathbb{F}_q$. Dado que tanto $a(X)$ como $\prod(x + \sigma(i))^{m_i}$ son mónicos y ambos tienen el mismo grado, tenemos que

$$a(X) = \prod_{i=0}^{p-1}(x + \sigma(i))^{m_i}$$

.

Por lo tanto, $m_i = 1$, $0 \leq i \leq p-1$, si y sólo si $-\sigma(i)$ es una raíz de $a(X)$. De esta forma el constructor del sistema puede recuperar todos los m_i del mensaje y después aplicar el inverso de la permutación σ obteniendo así las posiciones de los coeficientes 1 del mensaje.

3.2.2. Algoritmo auxiliar para representar los mensajes:

Existe un algoritmo recursivo que identifica un número M, $1 \leq M \leq \binom{p}{k}$, con una cadena de caracteres binarios $m_0, m_1 \ldots, m_{p-1}$ de longitud p y peso k haciendo uso de la fórmula:

$$\binom{p}{k} = \binom{p-1}{k} + \binom{p-1}{k-1}$$

Algoritmo: Si $M \geq \binom{p-1}{k}$, ponemos $m_{p-1} = 1$ y definimos $M' = M - \binom{p-1}{k}$. Este nuevo valor estará entre 1 y $\binom{p-1}{k-1}$ y se puede identificar con una cadena binaria $m_0, m_1, \ldots, m_{p-2}$ de longitud $p-1$ y peso de Hamming $k-1$ utilizando este mismo algoritmo.

Si por el contrario $M < \binom{p-1}{k}$, ponemos $m_{p-1} = 0$ y podemos identificar M con una cadena binaria $m_0, m_1, \ldots, m_{p-2}$ de longitud $p-1$ y peso de Hamming k utilizando de nuevo el algoritmo.

Ejemplo: Para este ejemplo vamos a utilizar $p = 7$ y $k = 4$. Por lo tanto, utilizando el algoritmo podemos identificar un número M, $1 \leq M \leq \binom{p}{k} = \binom{7}{4} = 35$ con una cadena binaria de 7 caracteres $(m_0, \ldots, m_6)$. Por ejemplo, tomemos $M = 27$.

$$p = 7,\ k = 4;\ M = 27 > \binom{6}{4} = 15 \Rightarrow m_6 = 1;\ M := M - 15 = 12$$

Y podemos identificar el nuevo M con una cadena binaria de 6 caracteres con $p = 6$ y $k = 3$.

$$p = 6,\ k = 3;\ M = 12 < \binom{5}{3} = 20 \Rightarrow m_5 = 0;$$

Y podemos identificar M con una cadena binaria de 5 caracteres y peso de Hamming 3.

$$p = 5,\ k = 3;\ M = 12 > \binom{4}{3} = 8 \Rightarrow m_4 = 1;\ M := M - 8 = 4$$

$$p = 4,\ k = 2;\ M = 8 > \binom{3}{2} = 3 \Rightarrow m_3 = 1;\ M := M - 3 = 1$$

$$p = 3,\ k = 1;\ M = 1 < \binom{3}{1} = 3 \Rightarrow m_2 = 0;$$

$$p = 2,\ k = 1;\ M = 1 < \binom{2}{1} = 2 \Rightarrow m_1 = 0;$$

$$p = 1,\ k = 1;\ M = 1 \leq \binom{1}{1} = 1 \Rightarrow m_0 = 1;$$

Por lo tanto, siguiendo el algoritmo hemos obtenido $M \equiv (1, 0, 0, 1, 1, 0, 1)$ que efectivamente tiene 7 caracteres y peso de Hamming 4.

Capítulo 4

Ataques a los criptosistemas basados en el problema de las mochilas

Existen ataques específicos para criptosistemas concretos, como el de Shamir para el criptosistema de Merkle-Hellman, pero también existen otros ataques criptoanalíticos que se pueden aplicar a cualquier criptosistema (basado en el problema de las mochilas). Vamos a explicar en este capítulo de un ataque de cada uno de los dos tipos de criptoanálisis mencionados.

El primero de ellos es el ataque de Shamir contra el criptosistema de Merkle y Hellman, que fue propuesto en el año 1983 por Adi Shamir [20], uno de los diseñadores del RSA.

El segundo ataque es el que se conoce como ataque de baja densidad, debido a que es efectivo en criptosistemas de baja densidad, parámetro que explicaremos más adelante. También se conoce como ataque L^3 porque se basa en un algoritmo creado por Lenstra, Lenstra y Lovász [12].

Estos dos ataques no son los únicos que existen pero son los dos principales ataques criptoanalíticos contra los criptosistemas basados en el problema de las mochilas. Hay muchos ataques, entre ellos el de Shamir, que utilizan la aproximación diofántica como técnica para resolver el problema de las mochilas. La aproximación diofántica es una rama de la teoría de números que trata la aproximación de números reales mediante números racionales.

4.1. Ataque de A. Shamir

Vamos a exponer una versión sencilla del ataque de Shamir, el original es más complicado y muy enrevesado. Consistía en encontrar, analizando el conjunto de pesos $\{a_i,\ i = 1, \ldots, n\}$, un par M, W tal que $Wa_i\ mod\ M$ sea una sucesión supercreciente y tal que $\sum a_i < M$. Puesto que nos enfrentaríamos a un criptosistema real, sabemos que existe al menos un par que cumple estas condiciones (el elegido por el usuario que ha contruido el sistema) y por lo tanto el problema tiene solución. Sin embargo, ni el algoritmo proporcionado por Shamir ni el que explicamos aquí tiene por qué encontrar el mismo par M, W utilizado para la construcción del sistema ni la misma sucesión supercreciente original $\{b_1, \ldots, b_n\}$.

El ataque: Supongamos que tenemos a_i, $i = 1, \ldots, n$ pesos públicos construidos como en la sección 2.1. Sea $U \equiv W^{-1}\ mod\ M$, $0 < U < M$. Tenemos entonces que

$$a_i \equiv b_{\sigma(i)}W\ mod\ M$$

o lo que es lo mismo,

$$b_{\sigma(i)} \equiv a_iU\ mod\ M$$

Es decir, existen ciertos $k_i \in \mathbb{Z}$ para el cual se cumple que $a_iU - k_iM = b_{\sigma(i)}$, $i = 1, \ldots, n$. Dividiendo esta igualdad por a_iM obtenemos

$$\frac{U}{M} - \frac{k_i}{a_i} = \frac{b_{\sigma(i)}}{a_iM}$$

Esto quiere decir que los dos sumandos del primer miembro de la igualdad están muy próximos entre sí, porque

$$\frac{b_{\sigma(i)}}{a_iM} = \frac{b_{\sigma(i)}}{b_{\sigma(i)}WM} = \frac{1}{WM} \approx 0 \text{ puesto que } W \geq 1 \text{ y } M > \sum a_i \approx n2^{2n}$$

.

De todos los parámetros que intervienen en esta igualdad solamente conocemos los a_i, que son públicos, y el problema ahora es extraer información del resto. Intentaremos calcular los k_i para obtener una aproximación de U/M y construir una sucesión supercreciente. En lugar de hacer una análisis para $i = 1, \ldots, n$, Shamir dice en [20] que es suficiente con utilizar las igualdades anteriores para $i = 1, \ldots, 5$, (aunque dependiendo del número de pesos del criptosistema podrían ser necesarias más o menos igualdades). Sabemos por cómo se construye el sistema que $b_1, \ldots, b_5 \lesssim 2^n$ y que cada a_i, k_i y M es del orden de 2^{2n}. Sea $i_j = \sigma^{-1}(j)$, entonces tenemos:

$$\left|\frac{U}{M} - \frac{k_{i_j}}{a_{i_j}}\right| \lesssim \frac{2^n}{2^{4n}} = 2^{-3n},\ 1 \leq j \leq 5$$

Y si restamos el término $j = 1$ del resto queda

$$\left|\frac{k_{i_j}}{a_{i_j}} - \frac{k_{i_1}}{a_{i_1}}\right| \lesssim 2^{-3n},\ 2 \leq j \leq 5$$

Ahora, multiplicando por $a_{i_j}a_{i_1}$ obtenemos

$$\left|k_{i_j}a_{i_1} - k_{i_1}a_{i_j}\right| \lesssim 2^{n},\ 2 \leq j \leq 5$$

Esta última desigualdad muestra que los a_{i_j} y k_{i_j} son muy particulares dado que son del orden de 2^{2n} y por tanto el producto $a_{i_1}k_{i_j}$ es del orden de 2^{4n} y que la diferencia de dos términos tales sea del orden de 2^n requiere una estructura especial y de hecho (en la mayoría de los casos) los k_{i_j} quedan determinados por estas desigualdades.

Una vez encontrados los k_{i_j} tendremos una aproximación muy buena para el cociente U/M que nos permitirá construir un par (U', M') con U'/M' próximo a U/M y tal que los pesos $c_i \equiv a_iU'\ mod\ M'$, $0 < c_i < M'$, $1 \leq i \leq n$ forman una sucesión supercreciente y $\sum c_i < M'$.

La cuestión ahora es cómo determinar en la práctica los k_{i_j}. Shamir se dio cuenta de que se pueden recuperar en tiempo polinomial aplicando el teorema de Lenstra para el problema de programación entera en un número finito de variables [13]. Gracias a este teorema se consigue obtener los k_{i_j}, permitiendo completar la construcción de los pesos c_i, $1 \leq i \leq n$ y la recuperación del mensaje.

Como ya se ha dicho antes, U', M'(construidos a partir de los a_i) y por lo tanto los c_i construidos a partir de U' y M' no tienen por qué coincidir con los parámetros utilizados para la construcción del sistema.

El ataque de Shamir, aunque hizo que no se pudiese utilizar el criptosistema de Merkle y Hellman, no evitó que se siguiera trabajando para encontrar un criptosistema basado en el problema de las mochilas seguro, puesto que las ventajas que ofrecen (como el tiempo de ejecución lineal para el cifrado del mensaje o la resistencia a la computación cuántica) son grandes. Es por ello que surgieron otras propuestas, algunas de ellas expuestas en el capítulo anterior.

4.2. Ataque de baja densidad

El ataque de baja densidad es muy sencillo de explicar y de implementar, y aunque funciona en gran parte gracias al algoritmo L^3 desarrollado en

[12], la idea de aplicar dicho algoritmo en este contexto se debe a Lagarias y Odlyzko [11].

De lo primero de lo que debemos hablar es de la densidad de un criptosistema basado en el problema de las mochilas.

Definición: Dado un conjunto de pesos a_i, $i = 1, \ldots, n$ que a partir de ahora denotaremos $\bar{a} = (a_1, \ldots, a_n)$ definimos la densidad del problema como

$$d(\bar{a}) = \frac{n}{\text{máx}\{\log_2 a_i,\, 1 \leq i \leq n\}}$$

La densidad de un conjunto de pesos mide la tasa de información del criptosistema. Más concretamente,

$$d(\bar{a}) \approxeq \frac{\text{Número de bits en el mensaje en claro}}{\text{Media de número de bits en el mensaje cifrado}}$$

Como es de esperar, el ataque es efectivo cuando la densidad del problema es baja, en concreto cuando es menor que 0.645. Hablaremos sobre esta cota más adelante.

Para comprender el funcionamiento del ataque es necesario saber algunos conceptos previos sobre retículos, en particular, sobre retículos enteros.

Definición: *Un retículo $L \subset \mathbb{R}^n$ es un subgrupo aditivo de $\mathbb{R}^n$ que contiene n vectores linealmente independientes* $\mathbf{v_1}, \ldots, \mathbf{v_n} \in \mathbb{R}^n$. *Una base de L es un conjunto* $\{\mathbf{u_1}, \ldots, \mathbf{u_n}\}$ *tal que* $R = \mathbb{R}\mathbf{u_1} \oplus \cdots \oplus \mathbb{R}\mathbf{u_n}$ *y la representamos mediante una matriz $n \times n$*

$$U = \begin{bmatrix} \mathbf{u_1} \\ \cdot \\ \cdot \\ \cdot \\ \mathbf{u_n} \end{bmatrix}$$

Definición: *Un retículo entero R es un subgrupo aditivo de $\mathbb{Z}^n$ que contiene n vectores linealmente independientes* $\mathbf{v_1}, \ldots, \mathbf{v_n} \in \mathbb{Z}^n$. *Una base de R es un conjunto* $\{\mathbf{u_1}, \ldots, \mathbf{u_n}\}$ *tal que* $R = \mathbb{Z}\mathbf{u_1} \oplus \cdots \oplus \mathbb{Z}\mathbf{u_n}$ *y la representamos mediante una matriz $n \times n$*

$$U = \begin{bmatrix} \mathbf{u_1} \\ \cdot \\ \cdot \\ \cdot \\ \mathbf{u_n} \end{bmatrix}$$

Un concepto importante relacionado con los retículos enteros es el de vector corto. Se entiende por corto un vector tal que su norma euclídea es *pequeña*. A priori parece que es un problema muy difícil de resolver, dada un base de R encontrar $v \in R$ tal que $\|\mathbf{x}\| \geq \|\mathbf{v}\|\ \forall \mathbf{x} \in R$, aunque no se ha probado que lo sea.

Otro concepto importante es el de base reducida. No vamos a dar una definición explícita de base reducida dado que existen varias. Una propiedad de la que haremos uso es que los vectores de una base reducida son cortos, en particular, en [12] se expone que:

Lema: *Dada U base reducida de R un retículo entero n-dimensional. Entonces tenemos lo siguiente:*

$$\|\mathbf{u_1}\|^2 \leq 2^{n-1} \min_{\mathbf{x} \in R, \mathbf{x} \neq 0} \|\mathbf{x}\|^2$$

Esta cota se puede mejorar, pero lo importante es que Lenstra, Lenstra y Lovász encontraron un algoritmo de tiempo de ejecución polinomial que dada una base arbitraria de un retículo n-dimensional produce una base reducida. Es en esto en lo que se va a basar el ataque, por eso en algunos libros se describe este ataque como ataque L^3. Según lo explicado en [11], el algoritmo L^3 no sólo tiene una cota buena (y mejorable) para la longitud del vector corto de la base reducida, sino que en la práctica es muy rápido y normalmente produce un vector más corto que lo garantizado por la cota del lema anterior.

Con estos conceptos en mente el ataque es muy fácil de entender. Se explica como aparece expuesto en [11].

4.2.1. Algoritmo

Queremos encontrar una solución (el mensaje) al problema de las mochilas siguiente: Dado S y un conjunto de pesos $a_1, \ldots, a_n \in \mathbb{N}$ encontrar un mensaje $\bar{\mathbf{x}} = (x_1, \ldots, x_n) \in \mathbb{Z}_2^n$ de forma que $\sum x_i a_i = S$. Entonces utilizamos el algoritmo siguiente:

Paso 1: Formamos un retículo entero de dimensión $n+1$ con base

$$U = \begin{pmatrix} 1 & 0 & \ldots & 0 & -a_1 \\ 0 & 1 & \ldots & 0 & -a_2 \\ & & \ddots & & \\ 0 & 0 & \ldots & 1 & -a_n \\ 0 & 0 & \ldots & 0 & S \end{pmatrix}$$

Denotamos las filas de U como $\mathbf{u_1}, \ldots, \mathbf{u_{n+1}}$. Si existe $\bar{\mathbf{x}}$ que soluciona el problema (en nuestro caso sabemos que sí puesto que es un mensaje cifrado que habríamos interceptado) entonces se tiene que

$$\sum_{i=1}^{n} x_i \mathbf{u_i} + \mathbf{u_{n+1}} = (x_1, x_2, \ldots, x_n, 0)$$

Como los x_i, $i = 1, \ldots, n$ son o bien ceros o bien unos, este vector $(x_1, x_2, \ldots, x_n, 0)$ es muy corto, $\|(x_1, x_2, \ldots, x_n, 0)\|_2 \leq \sqrt{n}$.

Paso 2: Encontrar una base reducida $\mathbf{w_1}, \ldots, \mathbf{w_{n+1}}$ del retículo que acabamos de formar utilizando para ello el algoritmo L^3 ([12]).

Paso 3: Comprobar si alguno de los vectores "cortos"hallados, $\mathbf{w_i}, 1 \leq i \leq n+1$ es tal que la coordenada $n+1$ es cero y el resto son o bien cero o bien una constante λ, para cierto λ.

Si lo encontramos, comprobar si el vector $\frac{1}{\lambda}\mathbf{w_i}$ es una solución del problema. Si lo es entonces salimos del algoritmo, si no, procedemos con el paso 4.

Paso 4: Repetir los pasos 1, 2 y 3 reemplazando S por $\sum a_i - S$. Si así obtenemos una solución $\{x_i\}_{i=0}^n$ de este nuevo problema, entonces $\{x'_i\}_{i=0}^n$ definido por $x'_i = 1 - x_i, 1 \leq i \leq n$ será solución del problema original.

Lagarias y Odlyzko garantizan en [11] que si la densidad de los pesos del problema de las mochilas enfrentado es menor que 0.645 el algoritmo descrito produce una solución válida, pero si no se cumple la condición sobre la densidad puede ocurrir que el algoritmo funcione, aunque lo más probable es que no lo haga.

4.2.2. Comentarios sobre el ataque

¿Por qué funciona el ataque? Para entender mejor qué hace que el algoritmo funcione hay que pensar un poco. Para empezar, si los pesos propuestos $a_i, i = 1, \ldots, n$ son muy grandes, entonces todos los vectores que se encuentran en el retículo entero que generamos deberán ser largos (en el sentido de que su norma será grande) y por lo tanto el vector $(x_1, x_2, \ldots, x_n, 0)$ que obtenemos debería ser de los más cortos, puesto que está compuesto solamente de unos y ceros, y podría ser el más corto. Es posible demostrar un resultado en este sentido.

Lema: *Dados $a_i, i = 1, \ldots, n$ pesos aleatorios con $a_i \approx 2^{\beta n}, 1 \leq i \leq n,\ \beta >$ 1,54725 entonces el vector $(x_1, x_2, \ldots, x_n, 0)$ es el más corto del retículo.*

La prueba de este resultado se encuentra en [11] así como un análisis sobre la constante β, y una prueba simplificada se expone en [7].

El tiempo de ejecución de los pasos 1 y 3 del algoritmo es despreciable, por lo que el tiempo de ejecución total del algoritmo es simplemente dos veces el tiempo de ejecución del algoritmo L^3, que es de orden polinomial (se da una cota más precisa en [12]).

Otra cuestión que queda pendiente es la de la densidad. Hasta ahora no hemos hablado en ningún momento de la densidad del problema y en qué afecta a nuestro ataque. No hay nada que nos garantice que al aplicar el algoritmo a un problema en concreto vayamos a obtener una solución, y es aquí donde entra en juego la densidad del conjunto de pesos.

Lagarias y Odlyzko enuncian en [11] una serie de teoremas en los que dan cotas cada vez mayores para la densidad de un conjunto de pesos de forma que el algoritmo obtiene una solución al problema de las mochilas. La mejor versión propuesta es la siguiente:

Teorema: *Sea $B > 2^{(\frac{1}{2}+\beta)n^2}$ para un cierto $\beta > 0$. Entonces si denotamos por K el número de conjuntos de pesos para los que el algoritmo encuentra una solución y tales que $1 \leq a_i \leq B$, $i = 1, \ldots, n$, se tiene que*

$$K = B^n + O(B^{n - c_3(\beta) + \frac{4 \log n}{n}})$$

Donde $c_3(\beta) = \frac{2\beta}{1+2\beta} > 0$

Lo enunciado en el teorema significa que para cualquier β positivo fijado se puede resolver el problema para casi todo $\bar{a} = (a_1, \ldots, a_n)$ tal que $d(\bar{a}) < (0{,}5 + \beta)^{-1} n^{-1}$. Esta cota está asegurada (y se puede mejorar, aunque no demasiado, ver [11]), pero es posible que un problema con densidad mayor sea susceptible de ser atacado con éxito mediante este ataque. Esto se debe a que la cota que alcanzamos en este teorema es una cota basada en el peor de los casos, pero como los propios autores del ataque explican en su publicación, en la práctica y basándose en la experiencia se puede poner una cota mejor que viene dada por el siguiente teorema.

Teorema: *Sea* **e** *un vector compuesto por ceros y unos tal que $\sum e_i \leq n/2$. Entonces, si $B = 2^{\beta n}$ para cualquier constante $\beta >$ 1,54725, denotando por*

$K(B, \mathbf{e})$ el número de conjuntos de pesos para los que $\mathbf{e}$ es el vector no nulo de norma euclídea más pequeña y tales que $1 \leq a_i \leq B$, $i = 1, \ldots, n$, se tiene que

$$K = B^n + O(B^{n-c_1(\beta)}(\log B)^2)$$

Donde $c_1(\beta) = 1 - 1{,}54725/\beta > 0$.

Este teorema dice que para cualquier β cumpliendo las condiciones, se puede resolver el problema para casi todo $\bar{a} = (a_1, \ldots, a_n)$ cumpliendo las hipótesis del teorema. Si $B = 2^{\beta n}$ entonces la densidad $d(\bar{a}) = \beta^{-1}$ por lo tanto el teorema es aplicable a conjuntos de pesos de densidad menor que $\beta^{-1} \approxeq 0{,}645$. Como ya se ha mencionado, un análisis sobre la cota para β se da en [11].

Capítulo 5

Análogo en códigos correctores, el criptosistema de McEliece

Presentamos el criptosistema de McEliece [14] como análogo al de Merkle y Hellman [15] porque ambos se basan en la misma idea. No es un análogo realmente, puesto que ninguno de los componentes del sistema (cifrado, descifrado, construcción de claves y construcción del sistema) se lleva a cabo de forma análoga a la de los elementos del criptosistema de Merkle y Hellman. Aun así, decimos que es un análogo porque los dos se basan en la idea de un problema fácil que se resuelve de manera rápida y sencilla y que se modifica para parecer un problema arbitrario.

Así como en el caso del criptosistema de Merkle y Hellman utilizábamos la sucesión supercreciente para el problema de las mochilas, en el criptosistema de McEliece el usuario construye un código de Goppa (fácil de decodificar) y lo modifica para que parezca un código corrector sin estructura reconocible para el que no se conoce un algoritmo eficaz de decodificación.

Cabe destacar que el criptosistema de McEliece aún no ha sido roto y ofrece unos niveles de seguridad muy buenos con los parámetros adecuados. Por si fuera poco, el problema de decodificar un código arbitrario (en el cual este sistema basa su seguridad) es un problema NP-completo, al igual que la versión general del problema de las mochilas. Este resultado se demuestra en [2].

Explicaremos a continuación el criptosistema de McEliece, pero antes hay que dar unas nociones básicas sobre códigos correctores de errores.

5.1. Códigos correctores de errores

Los códigos correctores de errores se utilizan para realizar transmisiones de datos de forma "segura". A diferencia de en criptografía, el término "seguro"se refiere a la protección frente a los errores que pudieran ser causados en el mensaje por un posible ruido en el canal. Si no se utilizase la codificación, entre emisor y receptor solo estaría el canal, y el ruido podría modificar la información enviada sin posibilidad de recuperar la original.

La idea consiste en modificar el mensaje original añadiendo información redundante (codificación) y que mediante esta información añadida el receptor pueda corregir los posibles errores, siempre que no sea un número excesivo de ellos, y recuperar el mensaje original (decodificación). Un ejemplo simple es el siguiente:

Ejemplo: Queremos enviar la palabra 'sí' = 1 o 'no' = 0. Entonces en lugar de enviar 0 enviamos 00000, de manera que si el ruido modifica el mensaje y el receptor obtiene 10010 podrá ver que se 'parece'más a 00000 que a 11111. Está claro que no es un sistema bueno puesto que el mensaje original podría ser 11111 y que los errores se hubieran producido en las posiciones segunda, tercera y quinta, pero sirve muy bien para ilustrar la idea de introducir información redundante.

Definición: Fijado $\mathbb{F}_q$, un *código lineal corrector de errores*, $\mathcal{C}$, de longitud n y dimensión k no es otra cosa que un subespacio de dimensión k dentro de $\mathbb{F}_q^n$. Los elementos del código, que son vectores de longitud n y componentes en $\mathbb{F}_q$, reciben el nombre de *palabras código*.

Una de las formas más sencillas de determinar un código es mediante su matriz generatriz, $\mathcal{G}$, que tiene k filas y n columnas. Las k filas son palabras código que forman una base del subespacio que es el código.

A partir de la matriz $\mathcal{G}$ podemos codificar mensajes $\mathbf{m} \in \mathbb{F}_q^k$ obteniendo la palabra código $\mathbf{c} = \mathbf{m}\mathcal{G}$. Nótese que $\mathbf{c}$ tiene n componentes mientras que $\mathbf{m}$ tiene $k \leq n$. Es decir, al codificar un mensaje se aumenta su tamaño, lo que concuerda con la idea de añadir información redundante.

Los códigos tienen asociada otra matriz, $\mathcal{H}$, de dimensiones $(n-k) \times k$ y rango máximo llamada matriz de control. Se puede utilizar para determinar qué elementos de $\mathbb{F}_q^n$ son palabras código. Esta matriz es la matriz generatriz

del espacio ortogonal a $\mathcal{C}$ (para el producto escalar usual, $\mathbf{x} \cdot \mathbf{y} = \sum x_i y_i$) y por lo tanto se cumple que $\mathcal{H}\mathbf{c}^t = \mathbf{0}$ para todo $\mathbf{c} \in \mathcal{C}$.

Otra noción importante ya mencionada en este trabajo es la de peso (de Hamming) de un vector, definido como el número de componentes no nulas de dicho vector. Se representa por $peso(\cdot)$ o $wt(\cdot)$. Es muy fácil comprobar que esta aplicación es una norma en $\mathbb{F}_q^n$ y por lo tanto induce una distancia $d(\mathbf{x}, \mathbf{y}) = peso(\mathbf{x} - \mathbf{y})$.

El tercer parámetro que define un código, además de la longitud y la dimensión, es la distancia mínima, d, definida por:

$$d = \text{mín}\{d(\mathbf{x} - \mathbf{y}) | \mathbf{x}, \mathbf{y} \in \mathcal{C},\ \mathbf{x} \neq \mathbf{y}\} = \text{mín}\{peso(\mathbf{x}) | \mathbf{x} \in \mathcal{C},\ \mathbf{x} \neq \mathbf{0}\}$$

La capacidad correctora de un código está muy relacionada con su distancia mínima, como describe el siguiente lema.

Lema: La capacidad correctora t de un código lineal corrector de errores viene dada por la fórmula

$$t = \lfloor \frac{d-1}{2} \rfloor$$

Donde d es la distancia mínima del código.

La idea es que, suponiendo una serie de errores producidos dentro de una palabra, si no son suficientes para hacer que esa palabra esté cerca de un elemento del código diferente del original, podemos recuperar la palabra código enviada originalmente.

Con estos conocimientos basta para comprender el criptosistema ideado por McEliece, que se explica a continuación.

5.2. El criptosistema de McEliece

El criptosistema de McEliece (1978,[14]) fue el primero en utilizar la teoría de códigos correctores de errores en criptografía. En el momento de su aparición rivalizó con el RSA, ElGamal y el criptosistema de Merkle y Hellman, pero debido al tamaño de las claves, que era mucho mayor que el de las de los otros criptosistemas; la investigación se centró en los tres últimos.

Sin embargo, el criptosistema de McEliece y en general los basados en teoría de códigos son mucho más rápidos a la hora de encriptar y desencriptar, al igual que el criptosistema de Merkle y Hellman. Además, el criptosistema

de McEliece en particular ofrece otras ventajas, como seguridad y versatilidad que comentaremos más adelante.

La idea del criptosistema es construir un código corrector lineal de fácil decodificación (en nuestro caso un código de Goppa) que tiene a $\mathcal{G}$ por matriz generatriz. Después se transforma $\mathcal{G}$ en $\mathcal{G}'$, matriz generatriz de otro código corrector sin estructura reconocible y por lo tanto un código para el que no se conoce un algoritmo de decodificación eficiente.

5.2.1. El método

Vamos ya con la explicación de las cuatro partes del criptosistema de McEliece, generación de clave privada, clave pública, cifrado y descifrado.

Generación de la clave privada: Para crear una clave privada el usuario debe construir un código corrector binario de Goppa $\mathcal{C}$, que será la clave privada. Se cumple que para cada polinomio irreducible $g(X)$ de grado t sobre $\mathbb{F}_{2^m}$ existe siempre un código binario irreducible de Goppa de longitud $n = 2^m$, dimensión $k \geq n - mt$ y que corrige a lo sumo t errores.

Por lo tanto la matriz generatriz del código, $\mathcal{G}$, que se obtiene fácilmente a partir del polinomio $g(X)$, tiene k filas, n columnas y es de rango máximo.

Generación de la clave pública: El usuario calcula y publica

$$\mathcal{G}' = S\mathcal{G}P$$

Donde S es una matriz regular de tamaño $k \times k$ y P es una matriz de permutación de tamaño $n \times n$.

$\mathcal{G}'$ es la matriz generatriz de un código arbitrario $\mathcal{C}'$ para el que no se conoce algoritmo eficiente de decodificación.

Cifrado: Cuando un emisor quiere enviar un mensaje $\mathbf{m} = (m_1, \ldots, m_k)$ debe calcular y enviar

$$\mathbf{c} = (c_1, \ldots, c_n) = \mathbf{m}\mathcal{G}' + \mathbf{e}$$

Donde $\mathbf{e} = (e_1, \ldots, e_n)$ es un vector de error de peso de Hamming t escogido aleatoriamente por el emisor. Además de cifrar el mensaje, se le añade un error que será corregido en el descifrado.

Descifrado: El usuario recibe $\mathbf{c}$ y calcula

$$\mathbf{c}P^{-1} = \mathbf{m}S\mathcal{G}PP^{-1} + \mathbf{e}P^{-1} = (\mathbf{m}S)\mathcal{G} + \mathbf{e}P^{-1}$$

La matriz P es de permutación, es decir, tiene un 1 en cada columna y fila y todo el resto de coeficientes de la matriz son ceros. Además, $P^{-1} = P^t$, por lo que P^{-1} también es una matriz de permutación y consecuentemente $\mathbf{e}P^{-1}$ no es más que una permutación del error original introducido por el emisor y tiene el mismo peso de Hamming que dicho error.

Utilizando el algoritmo de decodificación del código $\mathcal{C}$ el usuario puede corregir esos t errores y recuperar $\mathbf{m}S$. Por último, el usuario obtiene el mensaje original invirtiendo la matriz S, $\mathbf{m} = (\mathbf{m}S)S^{-1}$.

5.3. Pros y contras

Existen principalmente dos inconvenientes fundamentales en el criptosistema de McEliece, pero cada vez son menos importantes debido a los estudios teóricos y a los avances de la tecnología hoy en día.

Expansión de los datos: Debido a que los mensajes de k bits se codifican en cadenas de n bits, que son las que realmente circulan por el canal, la tasa de información del sistema es baja. La tasa de información del sistema viene dada por el cociente k/n e indica la proporción entre la cantidad de bits necesarios para enviar un mensaje y el propio mensaje.

En el ejemplo original del propio McEliece, se utiliza un código de Goppa de parámetros $(n = 1024, k = 524, d = 101)$. Para este código la tasa de información es bastante baja, aproximadamente 0,512, lo cual puede considerarse no muy eficiente puesto que para transmitir un mensaje se necesita enviar casi el doble de información.

No obstante, empleando técnicas diferentes a la original se puede mejorar el sistema para que tenga una tasa de información mayor. Por ejemplo, en [22], Sun consigue alcanzar tasas cercanas a $0,8$.

Tamaño de las claves: La clave pública del sistema es la matriz $\mathcal{G}'$, que en el caso clásico es binaria y de dimensiones $k \times n$, por lo que ocupa $nk/8$ bytes. Para un código de Goppa, la distancia mínima d es $2t + 1$ siendo t el número máximo de errores que puede corregir el código. A continuación se presentan los tamaños de G' para tres posibles elecciones de código:

- $(n = 1024, k = 524, d = 101)$: Tamaño de $\mathcal{G}' \approx 67$KB (el código corrige hasta 50 errores).
- $(n = 2048, k = 1278, d = 141)$: Tamaño de $\mathcal{G}' \approx 327$KB (el código corrige hasta 70 errores).
- $(n = 4096, k = 2056, d = 341)$: Tamaño de $\mathcal{G}' \approx 1$MB (el código corrige hasta 170 errores).

La diferencia con los criptosistemas más comunes como el RSA es muy grande, ya que los tamaños de clave de dicho sistema, que es el principal hoy en día, son 256, 512 y 1024 bytes para módulos de tamaño 1024, 2048 y 4096, respectivamente.

Por otra parte, podemos destacar 2 ventajas principales frente a los criptosistemas utilizados hoy en día.

Velocidad: El criptosistema de McEliece es mucho más rápido que cualquier otro criptosistema que necesite realizar operaciones de exponenciación modular, tales como ElGamal o RSA. El proceso de cifrado en el sistema original de McEliece requiere

$$\frac{k}{2}n + t$$

operaciones además del coste de generar el vector de errores $\mathbf{e}$.

Para el descifrado, en [6] se demuestra que son suficientes $O(ntm^2)$ operaciones para un código de Goppa binario irreducible de parámetros $(n = 2^m, k, d)$. Como comparativa, para cifrar y descifrar sobre el código de Goppa $(1024, 524, 101)$ se necesitan 2^{18} y 2^{22} operaciones binarias básicas, respectivamente, mientras que las mismas operaciones para el RSA con módulo de 1024 bits requieren ambas 2^{30} operaciones binarias.

Sin embargo, no es más rápido que un criptosistema basado en el problema de las mochilas. El criptosistema de Merkle y Hellman con $n = 100$ solamente necesita hacer 100 sumas para el cifrado y una multiplicación modular más 100 restas para el descifrado, además del coste de hacer una permutación de 100 elementos. El coste de cifrado de un criptosistema de las mochilas siempre va a ser más bajo que el del sistema de McEliece, pero el coste de descifrado no necesariamente será menor, depende del criptosistema utilizado.

Seguridad: Como ya se ha mencionado, el problema sobre el que basa su seguridad el criptosistema de McEliece es el problema de decodificación de un código corrector de errores arbitrario. Este problema pertenece a la clase

de problemas NP-completos (demostración en [2]), por lo que. en general, no existe un algoritmo eficiente (de tiempo polinomial) que pueda ser empleado para recuperar un mensaje cifrado simplemente decodificando sobre el código arbitrario $\mathcal{C}'$.

El hecho de que este problema sea NP-completo hace que el sistema sea resistente a un posible ataque criptoanalítico que hiciera uso del algoritmo ideado por Shor para la criptografía cuántica [21].

Además, el criptosistema de McEliece no ha sido roto todavía, y los ataques presentados contra él no son definitivos, dado que el trabajo a realizar por el atacante es demasiado elevado para llevar a cabo con éxito un ataque en un tiempo razonable.

5.4. Códigos de Goppa

Esta familia de códigos fue introducida por V.D.Goppa en 1970 [8]. Se conocen como códigos de Goppa clásicos, debido a que él mismo introdujo posteriormente los códigos algebrogeométricos, que se conocen también con el nombre de códigos de Goppa.

Estos códigos son los utilizados originalmente por McEliece en su criptosistema por tres razones principales:

- La cota inferior de la distancia mínima del código (d) es fácil de calcular($d = 2t + 1$).
- Conocer el polinomio generador del código permite poder corregir errores de forma eficiente. Gracias a ésto, el usuario que construye el sistema será capaz de descifrar los mensajes que le sean enviados con un coste computacional aceptable.
- Si no se conoce el polinomio generador o la estructura del código de Goppa, no existe ningún algoritmo eficiente para la corrección de errores, lo cual dificulta mucho el ataque de un criptoanalista que pretenda recuperar un mensaje enviado sin tener la clave (el polinomio de Goppa que genera nuestro código).

Bibliografía

[1] Leonard M Adleman. On breaking the iterated merkle-hellman public-key cryptosystem. In *Advances in Cryptology*, pages 303–308. Springer, 1983.

[2] Elwyn R Berlekamp, Robert J McEliece, and Henk CA Van Tilborg. On the inherent intractability of certain coding problems. *IEEE Transactions on Information Theory*, 24(3):384–386, 1978.

[3] Benny Chor and Ronald L Rivest. A knapsack-type public key cryptosystem based on arithmetic in finite fields. *Information Theory, IEEE Transactions on*, 34(5):901–909, 1988.

[4] Whitfield Diffie and Martin E Hellman. New directions in cryptography. *Information Theory, IEEE Transactions on*, 22(6):644–654, 1976.

[5] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Advances in Cryptology*, pages 10–18. Springer, 1985.

[6] Daniela Engelbert, Raphael Overbeck, and Arthur Schmidt. A summary of mceliece-type cryptosystems and their security. *Journal of Mathematical Cryptology*, 1(2):151, 2007.

[7] Alan M Frieze. On the lagarias-odlyzko algorithm for the subset sum problem. *SIAM Journal on Computing*, 15(2):536–539, 1986.

[8] Valerii Denisovich Goppa. A new class of linear correcting codes. *Problemy Peredachi Informatsii*, 6(3):24–30, 1970.

[9] PS Henry. Fast decryption algorithm for the knapsack cryptographic system. *Bell System Technical Journal*, 60(5):767–773, 1981.

[10] Xiao Ping Ji, Hai Bin Zhang, Bo Ying Wu, and Guang Yu Li. An extended knapsack public key cryptosystem. *Applied Mechanics and Materials*, 441:678–681, 2014.

[11] Jeffrey C Lagarias and Andrew M Odlyzko. Solving low-density subset sum problems. *Journal of the ACM (JACM)*, 32(1):229–246, 1985.

[12] Arjen Klaas Lenstra, Hendrik Willem Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

[13] Hendrik W Lenstra Jr. Integer programming with a fixed number of variables. *Mathematics of operations research*, 8(4):538–548, 1983.

[14] Robert J McEliece. A public-key cryptosystem based on algebraic coding theory. *DSN progress report*, 42(44):114–116, 1978.

[15] Ralph Merkle and Martin E Hellman. Hiding information and signatures in trapdoor knapsacks. *Information Theory, IEEE Transactions on*, 24(5):525–530, 1978.

[16] R Garey Michael and S Johnson David. Computers and intractability: a guide to the theory of np-completeness. *WH Freeman & Co., San Francisco*, 1979.

[17] Andrew M Odlyzko. The rise and fall of knapsack cryptosystems. *Cryptology and computational number theory*, 42:75–88, 1990.

[18] Stephen C Pohlig and Martin E Hellman. An improved algorithm for computing logarithms over gf(p) and its cryptographic significance (corresp.). *Information Theory, IEEE Transactions on*, 24(1):106–110, 1978.

[19] Ronald L Rivest, Adi Shamir, and Len Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[20] Adi Shamir. A polynomial time algorithm for breaking the basic merkle-hellman cryptosystem. In *Advances in Cryptology*, pages 279–288. Springer, 1983.

[21] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM journal on computing*, 26(5):1484–1509, 1997.

[22] Hung-Min Sun. Enhancing the security of the mceliece public-key cryptosystem. *J. Inf. Sci. Eng.*, 16(6):799–812, 2000.

[23] Serge Vaudenay. Cryptanalysis of the chor-rivest cryptosystem. In *Advances in Cryptology CRYPTO'98*, pages 243–256. Springer, 1998.

[24] Weidong Zhang, Baocang Wang, and Yupu Hu. A new knapsack public-key cryptosystem. In *Information Assurance and Security, 2009. IAS'09. Fifth International Conference on*, volume 2, pages 53–56. IEEE, 2009.