Criptosistemas basados en Teoría de Grupos

Tesis Doctoral

Autor : Mª Isabel Glez. Vasco
Director : Consuelo Martínez López

Departamento de Matemáticas

Universidad de Oviedo
Julio 2003

A mi madre

Agradecimientos

Quiero, en primer lugar, agradecer la ayuda que he recibido a lo largo de estos años de mi directora de tesis, la profesora Consuelo Martínez López. Sin su permanente disponibilidad, su dedicación y su extraordinaria calidad científica y humana me hubiera sido imposible llevar a cabo el trabajo de investigación que se recoge en esta tesis doctoral. Así mismo, quiero expresar mi gratitud al profesor Santos González Jiménez, especialmente por haberme dirigido desde mis primeros pasos hacia la criptografía. Gracias a su entusiasmo por las matemáticas y a su abierta mentalidad científica he tenido a mi alcance los mejores medios para desarrollar mi trabajo. De igual modo, me gustaría dar las gracias a los investigadores del grupo de Álgebra del Departamento de Matemáticas de la Universidad de Oviedo, compañeros y amigos de los que he aprendido mucho a lo largo de estos años. Especialmente, agradezco al Dr. Policarpo Abascal sus comentarios acerca del manuscrito original de esta memoria.

Gracias también al profesor Igor Shparlinski de la Universidad Maquarie por sus enseñanzas y su apoyo constante a lo largo de estos años. Al profesor Thomas Beth, de la Universidad de Karlsruhe, le agradezco especialmente el haberme dado la oportunidad de trabajar en el grupo de Criptografía del Instituto IAKS de Karlsruhe; por su hospitalidad mis estancias allí fueron tan gratas como científicamente productivas. Quiero también manifestar mi gratitud al Dr. Rainer Steinwandt. Estoy en deuda con él por las muchas horas de estudio y discusión que ha dedicado a nuestro trabajo conjunto. Así mismo, quiero agradecer su ayuda y hospitalidad a los Drs. Martin Rötteler, Markus Grassl y Jörn Müller-Quade, asi como a Dennis Hoffheinz y los demás miembros del Intstituto IAKS.

Además, quiero dar las gracias al Dr. Juan González Meneses, de la Universidad de Sevilla, por su colaboración en los resultados relativos a grupos de trenzas. Agradezco también la ayuda y el apoyo que he recibido de los miembros del grupo de Matemática Aplicada y Computación de la Universidad Politécnica de Cataluña, en especial a Vanesa Daza, y a los Drs. Sebastián Martín y Jorge Jiménez.

No puedo olvidarme tampoco de agradecer a mis compañeros de doctorado de los departamentos de Estadística y Matemáticas de la Universidad de Oviedo su ayuda y amistad. Gracias especialmente a Ignacio Cascos y a Javier Pello, por sus sugerencias que tanto han mejorado el manuscrito original de esta Memoria.

Para terminar, gracias de corazón a mi familia y amigos por el apoyo que siempre me han brindado, por soportar mis ausencias y perdonar mi poca disponibilidad en los últimos años. Gracias en especial a mi madre y a mi hermana Camino; de ellas es todo mérito de este trabajo y de cualquier otro que realice en el futuro.

Índice general

Introducción

La criptología, ciencia que estudia la transmisión segura de información, surge de la necesidad instintiva del ser humano de proteger su privacidad y es una disciplina tan antigua como el lenguaje. Existen evidencias de la utilización de técnicas criptográficas en las civilizaciones Egipcia y Babilónica, si bien es cierto que en la antigüedad la ocultación de mensajes o esteganografía primaba sobre el cifrado.

Hasta el siglo XIX los métodos criptográficos utilizados eran fundamentalmente técnicas de sustitución alfabética, que ya comenzaron a utilizarse en el Imperio Romano. Julio César enmascaraba sus mensajes sustituyendo cada letra por la que ocupaba tres posiciones más adelante en el abecedario, procedimiento que ha pasado a la historia con el nombre de *cifrado de César*. A lo largo de la Edad Media fueron introducidas diversas variaciones en estas técnicas de sustitución, como la inserción de símbolos superfluos o redundancias para evitar que los textos claros pudieran recuperarse por medio del análisis de frecuencias. Desde el punto de vista matemático estos métodos son extremadamente sencillos; sólo involucran operaciones modulares o aplicaciones lineales acordadas por el emisor y el receptor. Esta información necesaria para cifrar y descifrar se resumía en secuencias de números o símbolos conocidos sólo por los emisores y receptores legítimos: las claves secretas.

Durante siglos el objetivo de la investigación en criptología no fue encontrar métodos criptográficos más robustos, sino construir artefactos mecánicos capaces de cifrar mensajes utilizando técnicas de sustitución más o menos sofisticadas. Entre los más célebres están el disco de Alberti (siglo XI) o las tarjetas de Cardano (siglo XVI). Por otro lado, la simplicidad de los fundamentos matemáticos subyacentes a estos procedimientos tuvo consecuencias dramáticas para varios gobernantes y estrategas. Un ejemplo en nuestra historia es el caso de Felipe II, cuyos mensajes secretos eran leídos sin dificultad por el algebrista francés Viète. El monarca español, convencido de que su procedimiento criptográfico era una *cifra indescifrable*, acusó a Vieté ante el

papa Sixto V de brujería.

A pesar de las evidentes deficiencias de los criptosistemas anteriores, hasta el siglo XIX no se plantea la utilización de matemáticas más complejas para cifrar. En ese momento empiezan a utilizarse de manera generalizada métodos de *transposición*, que construyen un mensaje cifrado permutando las letras del texto claro correspondiente. Desde el punto de vista matemático, este fue el avance más significativo en criptología hasta la primera mitad del siglo XX. Aunque hoy resulte difícil de imaginar, sustitución y transposición fueron prácticamente los únicos métodos criptográficos empleados durante las dos guerras mundiales. El sistema ADFGX empleado por los alemanes en la primera Guerra Mundial es una sencilla combinación de ambas técnicas. Después, el periodo de entreguerras fue un momento de gran actividad científica en torno a la necesidad de establecer comunicaciones militares y diplomáticas seguras. Sin embargo, nuevamente se centraron los esfuerzos en mecanizar los procesos de cifrado y descifrado, sin modificar sensiblemente los fundamentos matemáticos de las técnicas criptográficas empleadas. Ingenios como la Hagelin C-48 o la máquina Enigma permitieron el empleo generalizado de complejos métodos de sustitución y transposición, pero la escasa complejidad matemática de los cifrados que producían acabó teniendo consecuencias para su seguridad, y ambas fueron criptoanalizadas por computadoras primitivas.

En la segunda mitad del siglo pasado el desarrollo de la informática ha impulsado la investigación en criptología en mayor medida que las contiendas militares lo hicieron en el pasado. Esta investigación ha estado además dirigida a perfeccionar los métodos criptográficos a través de las herramientas matemáticas disponibles. Con el uso cada vez más extendido de las redes informáticas es cada día más difícil garantizar el derecho a la privacidad del ciudadano. Por otro lado, las técnicas criptográficas utilizadas hasta los años setenta, en las que es preciso acordar una clave secreta para la comunicación, son insuficientes en una sociedad en la que el tiempo de vigencia de la información es cada vez menor.

Impulsada por el vertiginoso desarrollo tecnológico, la criptología ha pasado por dos revoluciones en el curso de apenas cincuenta años. La primera tuvo lugar a finales de los años setenta, con la aparición de la criptografía de clave pública. En su artículo *New Directions in Cryptography*, W. Diffie y M. Hellman [32] sugerían la posibilidad de transmitir información confidencialmente utilizando únicamente canales públicos, es decir, sin necesidad de acordar claves o intercambiar información secreta a priori. Esta posibilidad pronto se materializó en propuestas factibles de métodos de cifrado seguros y rápi-

dos, basados fundamentalmente en problemas de teoría de números. La idea de Diffie y Hellman marca el nacimiento de la criptografía de clave pública, uno de los pilares que actualmente sostienen las llamadas autopistas de la información. La necesidad del empleo de matemáticas del más alto nivel en criptología es desde ese momento indiscutible.

Más difícil de precisar es el inicio de la segunda revolución, relacionada con la aparición de la computación cuántica; de hecho muchos opinan que aún está por llegar. La idea de la criptografía cuántica surge a pricipios de los setenta [98], aunque no empieza a tomar forma hasta diez años después [12, 98]. Los primeros trabajos en este campo persiguen encontrar un método eficiente para intercambiar claves utilizando las leyes de la física cuántica, por medio de la transmisión de fotones polarizados al azar en distintas direcciones. Sin embargo, no son este tipo de estudios constructivos los que producen un cambio sustancial en la criptología moderna: la verdadera revolución se produce a raíz de los algoritmos cuánticos de P. Shor para factorizar enteros y computar logaritmos discretos en tiempo polinomial [88, 89]. Estos algoritmos marcan un punto de inflexión en la historia de la criptografía, pues ponen fecha de caducidad a la mayor parte de las herramientas criptográficas actuales, que dejarán de ser seguras con la construcción del primer ordenador cuántico.

La aparición de los algoritmos de Shor ha sido un estímulo fundamental (si bien no el único) para la búsqueda de primitivas criptográficas no basadas en teoría de números. Aunque en la última década han aparecido numerosas propuestas supuestamente no vulnerables a ataques cuánticos, aún no puede decirse que se hayan encontrado alternativas realistas a los métodos criptográficos más usados en la actualidad. A lo largo de esta Memoria analizamos la mayoría de las propuestas existentes para construir sistemas de cifrado utilizando teoría de grupos. Esta teoría es una valiosa fuente potencial de herramientas criptográficas, apenas explotada aún. Estudiamos diversos métodos de cifrado, cuya seguridad se basa esencialmente en la dificultad de resolver cuestiones estrechamente ligadas a dos problemas clásicos de teoría combinatoria de grupos: *el problema de la palabra* y el *problema de la conjugación*. Señalamos los fallos de seguridad que hemos identificado en las propuestas existentes hasta la actualidad y cómo podrían solventarse. Además, sugerimos nuevas ideas de construcción que nos aproximan al diseño de herramientas de cifrado basadas en grupos seguras y eficientes.

Esta Memoria se organiza como sigue: en el Capítulo 1 introducimos las notaciones y conceptos básicos imprescindibles para comprender el resto del desarrollo. En particular, tras dar las primeras definiciones y resultados acerca de teoría de grupos y complejidad computacional, hacemos en la Sección 1.3

una breve introducción a las nociones actuales de seguridad criptográfica.

En el Capítulo 2 estudiamos en detalle diversos esquemas de cifrado basados en problemas de factorización o reescritura. La idea fundamental subyacente a estos esquemas es utilizar para cifrar factorizaciones de un conjunto dado que son difíciles de computar. Así, será sencillo determinar un elemento del conjunto a partir de su factorización, pero muy difícil encontrar la factorización que se corresponde con un elemento dado. Los elementos del conjunto son por tanto los mensajes cifrados en este tipo de esquemas, mientras que sus textos claros correspondientes se definen a partir de la factorización asociada a cada elemento.

Cabe preguntarse hasta qué punto es necesario que el conjunto utilizado en una construcción de este tipo tenga estructura de grupo. En la Sección 2.1 damos respuesta a esta pregunta, analizando la seguridad de un criptosistema basado en las llamadas *palabras Lyndon* [77] asociadas a un conjunto y razonando que nuestros ataques tienen mucho menos impacto en esquemas construidos sobre grupos [48]. A continuación, pasamos a analizar si existen construcciones seguras de este tipo basadas en grupos. En la Sección 2.2 se describe y analiza una propuesta para cifrar mensajes utilizando la dificultad del *problema de la palabra* en grupos finitamente presentados [97]. De nuevo, el *secreto* es el modo de escribir cierto elemento del grupo, su factorización a partir de una serie de palabras públicas. Demostramos que este método de cifrado es vulnerable a los llamados *ataques de reacción*. Puede decirse, por tanto, que cualquier individuo que pueda observar la actuación de un receptor autorizado es capaz de extraer mucha información acerca de las claves secretas del esquema. Explicamos con detalle cómo puede llevarse a cabo un ataque de este tipo sobre el criptosistema general basado en el problema de la palabra propuesto por Wagner y Magyarik [97]. También vemos cómo funciona nuestro ataque en el ejemplo concreto de diseño que los autores proponen [50]. Para finalizar esta sección, exponemos un ataque similar llevado a cabo por D. Hoffheinz y R. Steinwandt [58] sobre el esquema basado en grupos de Grigorchuck de Garzón y Zalcstein [37].

Aunque los criptosistemas anteriores basados en el problema de la palabra presentan problemas de seguridad, ya hay nuevas propuestas que quizá puedan en un futuro próximo inspirar la construcción de esquemas robustos. En concreto, existe un interesante diseño en esta línea que se fundamenta en ciertas secuencias de factorización de grupos llamadas *signaturas logarítmicas*: el esquema MST_1 (propuesto por S. Magliveras, D. Stinson y T. van Trung [76]). La idea de este esquema es, como antes, cifrar por medio de signaturas logarítmicas que induzcan factorizaciones difíciles de computar. Al

estudio de este método de cifrado dedicamos la última parte del Capítulo 2, deteniéndonos especialmente en el problema de selección de claves. ¿Cómo seleccionar signaturas logarímicas que induzcan factorizaciones útiles criptográficamente? Los autores del esquema proponen diversas estrategias para construir *signaturas logarítmicas salvajes*, i.e., signaturas logarítmicas para las que no existe un algoritmo polinomial que compute su factorización asociada. En esta Memoria demostramos que las propuestas de Magliveras, Stinson y van Trung para construir signaturas logarítmicas salvajes no son válidas [22].
Para alcanzar niveles aceptables de seguridad es necesario garantizar que un adversario sólo podrá vulnerar la seguridad de un criptosistema en un porcentaje muy pequeño de intentos. Puesto que el problema al que se enfrenta el adversario para romper el MST_1 es precisamente invertir la factorización inducida por cierta signatura logarítmica, es importante que dicha factorización no sea fácilmente computable para la gran mayoría de los elementos del grupo. Por tanto no basta, como exigían los autores, que las signaturas logarítmicas sean *salvajes* según su definición. Dichas secuencias no evitan los llamados *ataques por inversiones parciales*, i.e., pueden construirse algoritmos eficientes que sirvan para factorizar con respecto a una signatura logarítmica salvaje una cantidad considerable de elementos del grupo [49]. Para poder garantizar la seguridad del esquema MST_1 será necesario estudiar más a fondo cuál es el método más adecuado de selección de claves.

Nuestro siguiente paso es estudiar otro tipo de herramientas basadas en grupos cuyo diseño se inspira en la llamada *construcción de Diffie-Hellman*. La idea básica detrás de esta construcción es utilizar la dificultad del llamado *problema del logaritmo discreto* para cifrar mensajes. Resolver dicho problema implica ser capaz de, dado un elemento h perteneciente al grupo cíclico generado por un elemento g, recuperar el exponente e tal que $h = g^e$. En la Sección 3.2 describimos y comentamos los resultados conocidos acerca de los llamados *criptosistemas MOR*, basados en el problema del logaritmo discreto en ciertos grupos de automorfismos.
De manera similar, puede sustituirse en el problema anterior la exponenciación por un número natural secreto por la conjugación por un elemento secreto del grupo. De este modo es posible mimetizar el diseño del esquema de Diffie-Hellman para grupos no abelianos. Esa es la idea principal en la que se fundamentan los esquemas criptográficos basados en grupos de trenzas que estudiamos en la Sección 3.3. Dichos esquemas han sido propuestos a lo largo de los últimos cinco años y han estimulado una ferviente actividad científica a su alrededor. En los últimos años han aparecido numerosos métodos de ataque que, en mayor o menor medida, vulneran la seguridad de las

propuestas existentes, aunque cada día existen más grupos de trabajo que confían en encontrar una construcción segura. Incluimos en este apartado un breve resumen de las propuesas y criptoanálisis conocidos hasta la fecha. Para concluir, proponemos una nueva idea de construcción, basada en el llamado *teorema de Markov*.
Concluímos el Capítulo 3 con una sección dedicada al criptosistema MST_2. Dicho esquema es, al igual que el MST_1, una propuesta de S. Magliveras et al. [76] y se construye también a partir de ciertas secuencias de factorización de grupos finitos. Detallamos la descripción del esquema y realizamos un breve análisis de seguridad [49]. La estructura del criptosistema MST_2 tiene muchos elementos comunes con el resto de esquemas estudiados en este capítulo. Así, inspirados por esas similitudes, introducimos en la Sección 3.4.2 el llamado *esquema MST_2 generalizado*, justificando su utilidad como marco común para estudiar diversas herramientas criptográficas basadas en grupos [41, 42].

El último capítulo de esta Memoria se centra en cuestiones de teoría de grupos, que surgen de manera natural en un contexto criptográfico. Al estudiar en el Capítulo 2 la estructura de las claves asociadas al esquema MST_1, notamos que la eficiencia del esquema depende en gran medida del tamaño de las secuencias de factorización empleadas como claves. Es necesario, por tanto, ser capaces de construir signaturas logarítmicas que involucren a pocos elementos del grupo. Pero, ¿cómo construir signaturas logarítmicas minimales en este sentido? y ¿para qué grupos podemos asegurar que existen? Damos respuesta parcialmente a estas preguntas encontrando una cota inferior para la *longitud* de toda signatura logarítmica asociada a un grupo, y demostrando que se alcanza para los grupos resolubles, así como para los grupos simétricos y alternados y para todo grupo simple de orden menor que J_1 (el primer grupo simple esporádico de Janko) [45]. Aún es un poblema abierto determinar si es posible encontrar una signatura logarítmica de longitud mínima para cualquier grupo finito.
Para concluir el Capítulo 4 incluímos un ejemplo de signatura logarítmica salvaje (según la definición de Magliveras, Stinson y van Trung) asociada a un grupo infinito. Dicha construcción se basa en el llamado *peinado* asociado a los grupos de trenzas.

Aunque es evidente que la criptografía basada en grupos es una línea de investigación en auge, no resulta sencillo determinar con precisión qué herramientas basadas en grupos acapararán más atención a largo plazo. Los resultados expuestos en esta Memoria permiten sin embargo intuir en cierta medida qué líneas de investigación son más prometedoras. A la vista de los ataques expuestos, es posible que no aparezcan nuevos esquemas basados en

el problema de la palabra en grupos infinitos ni en el problema de la conjugación en grupos de trenzas. Por el contrario, criptosistemas MST abren un camino muy esperanzador hacia nuevos métodos de cifrado seguro, aunque aún es necesario contestar muchos interrogantes acerca de la elección adecuada de parámetros para conseguir un diseño seguro y eficiente. En cualquier caso, la criptografía basada en grupos es una línea de investigación dinámica y viva, en la que cada día surgen nuevas ideas que quizá pronto den lugar a métodos prácticos de cifrado con altas garantías de seguridad.

Capítulo 1

Preliminares

1.1. Grupos. Nociones Elementales.

Los esquemas criptográficos analizados en esta Memoria se construyen a partir de una de las estructuras algebraicas más complejas y mejor estudiadas: la estructura de grupo. Históricamente, la teoría de grupos se considera iniciada por Galois (1811-1832), aunque la idea de grupo ya aparece implícitamente en numerosas áreas de las matemáticas mucho tiempo atrás. Son sobradamente conocidas las conexiones de esta teoría con la geometría, la teoría de números, la combinatoria etc.. En menor medida se conocen el tipo de aplicaciones que nos ocupan: la construcción de criptosistemas a partir de teoría de grupos.
Recordemos, antes de empezar nuestro desarrollo, algunas nociones elementales de teoría de grupos [86].

Definición 1.1 *Llamamos grupo a un conjunto G no vacío dotado de una operación binaria $*$ que cumple:*

i. $*$ *es asociativa,*

ii. existe un elemento distinguido $e \in G$ *tal que* $e * g = g * e = g$, $\forall\, g \in G$. *El elemento* e *se dice* elemento neutro *o* identidad *del grupo.*

iii. Para cada $a \in G$, *existe* $b \in G$ *tal que* $a * b = b * a = e$. *El elemento* b *se dice* inverso de a *y se denota* a^{-1}.

Si además se tiene $a*b = b*a$ *para todo* $a, b \in G$, *el grupo* G *se dice* abeliano.

Notación: Habitualmente, denotaremos la operación binaria $*$ por $\cdot$, o con una mera yuxtaposición de elementos, refiriéndonos a ella como *producto asociado al grupo*. Además, denotaremos por e_G al elemento neutro de un grupo G.

Definición 1.2 *Dado un grupo G se llama* orden de G, *denotado por $|G|$, al cardinal del conjunto subyacente G.*

Definimos a continuación las aplicaciones entre grupos que respetan la estructura.

Definición 1.3 *Sean G y H grupos. Una aplicación $f : G \longmapsto H$ se dice* homomorfismo *si*

$$f(ab) = f(a)f(b), \quad \forall\, a, b \in G.$$

Un homomorfismo inyectivo (resp. suprayectivo) se dice monomorfismo *(resp. epimorfismo). Un homomorfismo biyectivo se dice* isomorfismo*, y si se establece de un grupo en sí mismo,* automorfismo.
El conjunto de los automorfismos de un grupo G dado tiene a su vez estructura de grupo con la operación composición de aplicaciones, y suele denotarse $Aut(G)$.

Sea G un grupo. Dado un elemento $x \in G$, la aplicación

$$f_x : G \longmapsto G$$

definida por $f_x(g) := xgx^{-1}$ es un automorfismo de G, llamado automorfismo interno. El conjunto de los automorfismos internos de un grupo G tiene estructura de grupo con la composición y suele denotarse $Inn(G)$.
Dados elementos $h, g \in G$, decimos que son *conjugados* si existe un automorfismo interno de G que transforma h en g, i.e., si existe $x \in G$ tal que $h = xgx^{-1}$. *La conjugación* es una relación de equivalencia en G, cuyas clases de equivalencia se dicen *clases de conjugación.*

Definición 1.4 *Sea G un grupo. Un subconjunto no vacío H de G se dice* subgrupo *de G si es estable respecto de la ley intena de G y tiene a su vez estructura de grupo con la restricción de la operación de G.*

Equivalentemente, un subconjunto no vacío H de G es un subgrupo si $\forall\, s, t \in H$ se tiene que $s^{-1} \in H$ y $st \in H$. Un subgrupo invariante por los automorfismos internos de G (i.e., tal que $xHx^{-1} = H$, $\forall\, x \in G$) se dice *normal o invariante*. Todo grupo G tiene al menos dos subgrupos normales, él mismo y el formado por el elemento neutro e_G. Además, estos dos subgrupos se dicen *impropios*, y cualquier otro subgrupo $\{e_G\} \lneq H \lneq G$ se dice subgrupo *propio* de G.

Notación: Escribiremos $H \leq G$ para denotar que H es un subgrupo de un grupo G. Si H es normal, se denotará $H \trianglelefteq G$. Para cualquier $X \subseteq G$ denotaremos por $\langle X \rangle_G$ (o $\langle X \rangle$, si no hay ambigüedad) al menor subgrupo de G que contiene a X, y lo llamaremos *subgrupo generado por* X. Análogamente, el menor subgrupo normal de G que contiene a X se dice *subgrupo normal generado por* X, y se denota $\langle X \rangle^G$.
Si $X = \{g\}$, para algún $g \in G$, el grupo $\langle X \rangle$ se denota $\langle g \rangle$ y se dice *cíclico*. Se dirá *orden de* g al orden del grupo generado por g, $|\langle g \rangle|$.

Definición 1.5 *Llamamos* cogrupo *(a izquda. o dcha.) de* G *a todo conjunto de la forma* gH *(*a izquierda*) ó* Hg *(*a derecha*) donde* H *es un subgrupo de* G *y* $g \in G$.
El número de cogrupos a izquierda coincide con el número de cogrupos a derecha de H *en* G *y se dice* índice *de* H *en* G *(denotado* $|G : H|$*). Notemos que* gH *(resp.* Hg*) es la clase de equivalencia del elemento* g *respecto de la relación de equivalencia*

$$g_1 \mathcal{R} g_2 \Longleftrightarrow g_1^{-1} g_2 \in H \quad (\textit{resp. }, g_2 g_1^{-1} \in H).$$

Por tanto, el grupo G *es unión disjunta de cogrupos*

$$G = \bigcup_{i=1}^{|G:H|} g_i H, \quad g_i \in G, \quad (\textit{resp.}, G = \bigcup_{i=1}^{|G:H|} H x_i, \quad x_i \in G).$$

El conjunto $\{g_i,\ i = 1, \ldots, |G : H|\}$ *(resp.* $\{x_i,\ i = 1, \ldots, |G : H|\}$*) se dice* conjunto completo de representantes a izquierda (resp. a derecha) *o* transversal a izquierda (resp. a derecha) *de* H *en* G.

La representación de G como unión disjunta de cogrupos permite demostrar el siguiente teorema.

Teorema 1.6 (de Lagrange) *Sea* G *un grupo y* $H \leq G$. *Entonces*

$$|G| = |G : H||H|,$$

en particular, si G *es finito* $|H|$ *divide a* $|G|$.

Si $H \trianglelefteq G$, para cada $g \in G$ los cogrupos a izquierda y a derecha coinciden, i.e.,

$$\forall\, g \in G, \quad gH = Hg.$$

El conjunto de clases asociado a la (única) relación de equivalencia $\mathcal{R}$ anterior, suele denotarse por G/H, y es fácil ver que tiene estructura de grupo con la operación heredada de G, i.e.,

$$g_1H * g_2H := g_1g_2H.$$

El grupo G/H se dice *grupo cociente* de G por H.

En el Capítulo 4 jugarán un papel importante los grupos sin subgrupos normales propios.

Definición 1.7 *Un grupo G se dice* simple *si no posee ningún subgrupo normal propio.*

Otra noción elemental que utilizaremos a menudo es la de *acción* de un grupo sobre un conjunto, en la que juegan un papel fundamental los llamados grupos de permutaciones.

Definición 1.8 *Sea X un conjunto cualquiera. El conjunto de biyecciones de X en sí mismo forma un grupo con la composición habitual de aplicaciones, denominado* grupo de permutaciones sobre X. *Dicho grupo suele denotarse por S_X y depende sólo del cardinal de X. Así, si X es finito y está formado por n elementos podrá denotarse alternativamente por S_n.*

Definición 1.9 *Sea G un grupo y X un conjunto. Una aplicación*

$$\phi : G \times X \longmapsto X$$

denotada por $\phi(g, x) = gx$ se dirá acción *de G sobre X si*

i. $ex = x$ para todo $x \in X$,

ii. $g(hx) = (gh)x$, para todo $g, h \in G$, $x \in X$.

Equivalentemente, si existe un homomorfismo

$$\psi : G \longmapsto S_X,$$

con $\psi(g)(x) = gx = \phi(g, x)$, ($g \in G$, $x \in X$), se dice también que G actúa sobre X. Además, se dirá que la acción es fiel *si el homomorfismo asociado, ψ, es inyectivo, o, equivalentemente si la acción de G sobre X, ϕ, cumple*

$$\phi(g, x) = x, \forall\, x \in X \iff g \in e_G.$$

A través de la noción de acción podemos ver cada grupo como un grupo de permutaciones que actúa sobre sí mismo (i.e., sobre su conjunto subyacente), sin más que considerar la acción fiel de G sobre sí mismo definida por la multiplicación a izquierda ($\phi(g,h) := gh$).

Teorema 1.10 (de Cayley) *Todo grupo G es isomorfo a un subgrupo del grupo de permutaciones $S_{|G|}$.*

Los grupos de permutaciones juegan un papel fundamental en criptografía, pues, en muchas ocasiones, las aplicaciones de cifrado no son más que permutaciones del conjunto de mensajes.

Definición 1.11 *Sea G un grupo actuando sobre un conjunto X cualquiera. Dado $x \in X$ llamamos:*

- G-órbita *(o símplemente,* órbita*) de x, al conjunto $\{gx \mid g \in G\} \subseteq X$,*
- estabilizador de x en G *al subgrupo*

$$G_x := \{g \in G \mid gx = x\} \leq G.$$

Diremos que G es transitivo *(en su acción sobre X) si sólo existe una G-órbita en X (i.e., si $\forall\, x, y \in X$, $\exists g \in G \mid gx = y$).*
Análogamente, G se dice k-transitivo ($k > 1$), si para todo par de k-tuplas distintas de elementos de X, $(x_1, \ldots, x_k), (y_1, \ldots, y_k)$, existe $g \in G$ con $gx_i = y_i$ para $i = 1, \ldots, k$. Un grupo G se dice sharply k-transitivo *si es k-transitivo y sólo la identidad fija k elementos dados de X.*

También utilizaremos en nuestro desarrollo algunas nociones elementales de teoría combinatoria de grupos. El objetivo de la teoría combinatoria de grupos es el estudio de los grupos definidos a través de presentaciones, esto es, por medio de un conjunto de generadores y un conjunto de relaciones.

Definición 1.12 *Sea X un conjunto cualquiera. Llamamos* grupo libre sobre *X al único grupo $\mathcal{F}_X$ que cumple que todo grupo G generado por los elementos de X es un cociente de $\mathcal{F}_X$.*

Los elementos del grupo $\mathcal{F}_X$ admiten una representación única como palabras reducidas (cadenas finitas de elementos sin subcadenas de la forma xx^{-1}) en $X \cup X^{-1}$.

Definición 1.13 *Sea G un grupo. Dados un subconjunto $X \subseteq G$ y un conjunto R de palabras en $X \cup X^{-1}$, diremos que el par (X, R) es una presentación de G (denotado $G = \langle X/R \rangle$) si G es isomorfo al cociente del grupo libre generado por X por el subgrupo normal engendrado en él por R[1]; en símbolos*

$$G = \langle X/R \rangle \Leftrightarrow G \backsimeq \frac{\mathcal{F}_X}{\langle R \rangle^{\mathcal{F}_X}}.$$

Llamaremos generadores a los elementos de X, relatores a los de R.
Un grupo G se dirá finitamente presentado si puede definirse mediante una presentación finita (con X y R finitos).

De manera informal, podríamos expresar la anterior definición diciendo que G es un grupo cuyo conjunto subyacente está formado por *palabras* en $X \cup X^{-1}$, que se multiplican por medio de la yuxtaposición. Dicha yuxtaposición está sujeta a las normas

$$xx^{-1} = \mathbf{e} \quad \forall\, x \in X$$

y

$$\mathbf{r} = \mathbf{e} \quad \forall\, \mathbf{r} \in R,$$

siendo $\mathbf{e}$ la *palabra vacía*, esto es, el elemento neutro de G. Si $\mathbf{w}$ es una palabra, denotaremos por w al elemento del grupo que representa.[2] Diremos que dos palabras son equivalentes si representan al mismo elemento del grupo, y lo representaremos con el símbolo $\sim$.
A menudo el conjunto de relatores R vendrá definido a través de una serie de ecuaciones en los generadores, a las que nos referiremos simplemente como relaciones. En lo que sigue, nos referiremos indistintamente a los elementos del conjunto R como relaciones o relatores.

1.2. Complejidad Computacional

Para poder comprender los conceptos básicos de criptografía, es fundamental estar familiarizado con algunas nociones elementales de computabilidad. Esto es debido a que la complejidad computacional es la medida que se emplea para cuantificar la seguridad o eficiencia de los esquemas criptográficos. Remitimos al lector interesado en una introducción más extensa al Capítulo 2 de [78].

[1] es decir; por los elementos del grupo libre definidos a partir de las palabras de R. Es frecuente identificar, abusando del lenguaje, palabras con elementos del grupo.

[2] Si la palabra consta de un único generador x, generador, palabra y elemento se denotan igual, x.

Comenzamos definiendo las nociones de *algoritmo* y *función computable.*

Definición 1.14 *Se llama* algoritmo *a todo proceso bien definido que a partir de una serie de valores de entrada proporciona un conjunto de valores de salida. Diremos que un algoritmo es* determinístico *si el valor de entrada determina por completo al de salida. Si, por el contrario, al ejecutar el algoritmo con el mismo valor de entrada pueden obtenerse distintos valores de salida, el algoritmo se dice* probabilístico o aleatorio.
Diremos que una función f es computable si existe un algoritmo que tomando un argumento x como entrada devuelve el valor $f(x)$ como salida.

Salvo que se indique explícitamente lo contrario, en el resto de esta sección consideraremos sólo algoritmos determinísticos.

Notación: Si $\mathcal{A}$ es un algoritmo, $\mathcal{A}(x)$ denota al valor o valores de salida que proporciona $\mathcal{A}$ tras ejecutarse con valor de entrada x.

En general, los valores de entrada y salida de un algoritmo están codificados como secuencias de ceros y unos (*bits*). Supondremos sin pérdida de generalidad que cuando dichos valores son números naturales, su codificación se corresponde con su expresión en base dos. Dado un valor cualquiera x, que un cierto algoritmo toma como entrada, llamaremos *longitud* de x a la longitud de su codificación en binario. La eficiencia de un algoritmo se mide por el llamado *tiempo de ejecución*, que va a medirse en función de la longitud de su entrada.

Definición 1.15 *El* tiempo de ejecución *de un algoritmo es el máximo número de operaciones elementales (operaciones bit a bit) que realiza al ejecutarse a partir de un cierto valor de entrada.*

En general, el tiempo de ejecución de un algoritmo se expresará como una función de la longitud del valor de entrada (i.e., será una función definida sobre $\mathbb{N}$). Dichas funciones suelen expresarse a través de formulaciones asimptóticas, que permiten clasificar los algoritmos correspondientes en base a su complejidad.

Definición 1.16 *Sean f y $g : \mathbb{N} \longrightarrow \mathbb{R}$ dos funciones que toman valores siempre positivos a partir de cierto número natural. Se dice que*

- *$f = O(g)$ si existe una constante positiva c y un natural n_0 tal que $\forall n \geq n_0$*

$$0 \leq f(n) \leq cg(n).$$

- $f = \omega(g)$ *si existe una constante positiva* c *y un natural* n_0 *tal que* $\forall n \geq n_0$
$$0 \leq cg(n) \leq f(n).$$

- $f = o(g)$ *si para toda constante positiva* c *existe un natural* n_0 *tal que* $\forall n \geq n_0$
$$0 \leq f(n) < cg(n).$$

Definición 1.17 *Un algoritmo se dice* polinomial *si su tiempo de ejecución es una función* $O(g)$*, siendo* $g = n^k$*, con* k *un número natural cualquiera. Todo algoritmo que no cumpla esta condición se dirá* exponencial.
Diremos que una función es computable en tiempo polinomial *si existe un algoritmo polinomial para su evaluación, en otro caso diremos que es* computable en tiempo exponencial.

A través de este criterio basado en la complejidad computacional se establece también una clasificación de los llamados problemas de decisión,

Definición 1.18 *Llamamos problema de decisión a toda pregunta que depende de un cierto valor de partida y cuya respuesta es un valor en el conjunto* $\{0, 1\}$*. Diremos que un problema de decisión* P *pertenece a la clase..*

- *...* $\mathcal{P}$ *(o que es polinomial) si existe un algoritmo polinomial que lo resuelve, i.e., que distingue si la respuesta de* P *con valor de partida fijado es* 0 *ó* 1*.*

- *...* $\mathcal{NP}$ *si existe un algoritmo polinomial (que suele llamarse* certificado*) para verificar la solución del problema, si ésta es 1.*

- *...* $co-\mathcal{NP}$ *si existe un certificado polinomial para verificar la solución del problema, si ésta es 0.*

Diremos que un problema P *es* $\mathcal{NP}-$*completo, si está en* $\mathcal{NP}$ *y además, de existir un algoritmo polinomial que resuelva* P*, también exisitiría un algoritmo polinomial para cualquier problema de* $\mathcal{NP}$*.*

Notas:

- La mayoría de los textos reducen el estudio de la complejidad computacional a los problemas de decisión. Es habitual que otro tipo de problemas (v.g., de búsqueda) puedan formularse alternativamente como un problema de decisión.

- Claramente, $\mathcal{P} \subseteq \mathcal{NP}$ y $\mathcal{P} \subseteq co-\mathcal{NP}$. Uno de los problemas abiertos más interesantes de la computación actual es llegar a demostrar que la igualdad $\mathcal{P} = \mathcal{NP}$ es cierta, o encontrar un contraejemplo que demuestre su falsedad.

- En general, se admite que el hecho de que exista un algoritmo polinomial para resolver un problema implica que dicho problema es *abordable*. Por tanto, en criptografía se exige que para violar la seguridad de un esquema un adversario tenga que resolver problemas que no pertenecen a la clase $\mathcal{P}$.

- Los problemas de decisión pueden clasificarse, de manera similar a la expuesta, considerando algoritmos aleatorios. Dicha clasificación se hace agrupando los algoritmos según su probabilidad de error y considerando como tiempo de ejecución una cota superior de su tiempo de ejecución esperado.

- Hoy en día es necesario hablar de nuevas clases de complejidad asociadas a un modelo distinto de computación: *la computación cuántica* (ver [53, 62]). Así, puede extenderse la Definición 1.18 admitiendo la existencia de algoritmos cuánticos, introduciéndose de modo natural las clases $\mathcal{QP}$, $\mathcal{QNP}$, etc. Sabemos que $\mathcal{QP} \neq \mathcal{P}$ (pues, por ejemplo, existe un algoritmo cuántico que factoriza enteros en tiempo polinomial, mientras que todos los clásicos que se conocen son exponenciales [89, 88]). Sin embargo, también es un problema abierto verificar la igualdad $\mathcal{QP} = \mathcal{QNP}$.
A lo largo de nuestro desarrollo, salvo que se indique lo contrario, consideraremos únicamente el modelo clásico de computación. Mirando al futuro, es innegable que la computación cuántica abre un nuevo y fascinante campo de investigación en criptografía [14, 13]

1.3. Criptosistemas de Clave Pública. Nociones de seguridad.

Pasamos ya a introducir algunos conceptos básicos acerca de criptografía de clave pública. Nos centraremos en las nociones relacionadas con el cifrado de mensajes, tema central de nuestra investigación.

1.3.1. Cifrado de mensajes. Criptografía y Criptoanálisis.

El *Handbook of Applied Cryptography* [78] define la *Criptografía* como

"aquella ciencia que se ocupa de la búsqueda y mejora de técnicas para la transmisión segura de la información".

Esa definición apenas deja intuir la enorme cantidad de procesos y situaciones que modela y estudia la criptografía moderna. Las aplicaciones más evidentes, como la transmisión confidencial de información o el intercambio de claves son sólo la punta de un iceberg de esquemas y protocolos con fines tan diversos como la firma digital segura, autentificación de emisores y receptores, compartición de secretos, votación electrónica etc. Nuestro objeto principal de estudio son los *criptosistemas de clave pública*, esquemas de cifrado para los que no se requiere un intercambio de claves previo entre emisor y receptor. Este tipo de esquemas fueron introducidos a finales de los setenta [32, 85], aunque la definición formal de criptosistema de clave pública que se utiliza hoy en día es muy posterior.

Considerar fijados uno (o varios) conjuntos finitos de claves y de mensajes, así como un número natural l que actuará como parámetro de seguridad. En general, suele suponerse que existen dos conjuntos de claves (uno de claves públicas y otro de claves privadas), un único conjunto de mensajes claros (sin cifrar) y varios conjuntos de mensajes cifrados (uno por cada clave pública). Todos estos conjuntos son información pública. Además, se supone que cualquier individuo puede verificar si un elemento dado pertenece a alguno de los conjuntos anteriores [3].

Definición 1.19 *Un* criptosistema de clave *pública es una terna*

$$\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$$

de algoritmos polinomiales donde:

- $\mathcal{K}$, *el* algoritmo generador de claves, *es un algoritmo aleatorio que recibe como entrada una cadena de unos de longitud l. La salida de $\mathcal{K}$, $\mathcal{K}(1^l)$ es un par clave pública/clave privada: (p_k, s_k).*

- $\mathcal{E}$, *el* algoritmo de cifrado, *es un algoritmo aleatorio que al recibir como entrada una cadena de l unos, la cláve pública p_k y un mensaje válido m, proporciona una cadena de bits c como salida (un cifrado de m con p_k).*

- $\mathcal{D}$, *el* algoritmo de descifrado, *es un algoritmo determinístico que al recibir como entrada una cadena de l unos, la clave privada s_k y $\hat{c}$, da como salida un mensaje $\hat{m}$ (que se corresponde con el cifrado $\hat{c}$ y la clave (p_k, s_k)) o una señal de error $\perp$ (si $\hat{c}$ no es un cifrado válido con la clave correspondiente).*

[3] Sólo recientemente se ha añadido esta condición, que, en particular, admite que un adversario puede distinguir textos bien cifrados de cadenas de bits aleatorias.

Notación: Un cifrado c de un mensaje m utilizando la clave pública p_k se denota $\mathcal{E}_{p_k}(m)$. Análogamente, la salida del algoritmo de descifrado correspondiente a una entrada $(1^l, s_k, \hat{c})$ se denotará $\mathcal{D}_{s_k}(\hat{c})$.

La definición anterior es relativamente reciente (ver, por ejemplo [9]). De hecho, aún hoy se introducen esquemas de cifrado de clave pública especificando tres conjuntos; uno de claves, otro de aplicaciones de cifrado y otro de aplicaciones de descifrado. Aunque se impongan numerosas restricciones sobre dichos conjuntos, las definiciones de ese tipo están tremendamente alejadas de las implementaciones prácticas, en las que siempre existe un componente de aleatoriedad difícil de formalizar sin hablar de algoritmos aleatorios. A lo largo de esta Memoria aparecen en repetidas ocasiones especificaciones de esquemas de clave pública según el modelo *antiguo* de definición (respetando la definición original de los autores). La traducción de dichas especificaciones a la terminología actual es inmediata.

Es claro que el objetivo de un esquema de este tipo es conseguir que a través de él la transmisión de información pueda realizarse de manera segura. El *criptoanálisis* o estudio crítico de los esquemas criptográficos se encarga de investigar hasta qué punto las herramientas criptográficas garantizan los niveles esperables de seguridad. Criptografía y criptoanálisis son dos caras de una misma moneda, una *constructiva* y otra *destructiva*, que juntas constituyen la ciencia de la transmisión segura de información o *criptología*.

El criptoanálisis de un criptosistema de clave pública tiene como objetivo analizar si el cifrado de mensajes a través de dicho esquema es o no seguro. Pero, ¿qué es un cifrado seguro? Esta es una pregunta tremendamente compleja que encuentra su respuesta en las distintas nociones de seguridad criptográfica [9].

1.3.2. Seguridad Semántica

Intuitivamente parece razonable exigir que en un criptosistema de clave pública los textos cifrados no revelen información alguna acerca de los textos claros correspondientes. Esta idea se corresponde con la noción de *seguridad semántica*, definida en un principio para funciones de una vía.

Según Diffie y Hellman [32], una función f del conjunto de cadenas finitas de bits, $\{0,1\}^*$, en sí mismo se dice *de una vía* si todo $x \in \{0,1\}^*$, es *muy difícil* de computar a partir de $f(x)$. Si una función f es de una vía, una función $b : \{0,1\}^* \longmapsto \{0,1\}^*$ se dice *hard core* para f si dado cualquier $x \in \{0,1\}^*$ es *muy difícil* computar $b(x)$ a partir de $f(x)$.

Para cifrar mensajes de manera segura, las primeras propuestas sugerían utilizar una función f de una vía tal que *toda* función $b : \{0,1\}^* \longmapsto \{0,1\}^*$ sea hard-core para f. Tales funciones se dicen *semánticamente seguras*, según la definición de Goldwasser y Micali [40]. Consecuentemente, se deshechaba utilizar funciones de una vía determinísiticas, pues toda función determinísitica no es hard-core para ella misma.

Existe, sin embargo, una manera de utilizar cualquier función de una vía para cifrar de manera segura. Para ello, se construye a través de la función de una vía una componente de aleatoriedad añadida que garantice la seguridad semántica. Supongamos que $g : \{0,1\}^* \longmapsto \{0,1\}^*$ es un generador pseudoaleatorio de secuencias binarias (i.e. simula una distribución uniforme en el conjunto finito de secuencias binarias considerado). Para cifrar un mensaje m, seleccionamos primero una semilla k y construimos el cifrado c de m sumando bit a bit $g(k)$ y m.
Ahora bien, ¿cómo podemos construir un generador pseudoaleatorio g_f a partir de una función de una vía f? Blum y Micali propusieron una construcción estándar válida cuando f es una permutación [21]: simplemente, seleccionar b una función booleana que sea hard-core para f y definir la secuencia: x_0 (semilla aleatoria), $x_{i+1} := f(x_i)$. El generador g se define ahora como:

$$g_f(x_0) := b(x_0), b(x_1), \ldots.$$

Una construcción similar, aunque bastante más complicada [56], demuestra que, efectivamente, lo único necesario para construir un método de cifrado *semánticamente seguro* es una función de una vía y una función hard core asociada a ella. Finalmente, Goldreich y Levin demostraron en 1989 [39] que toda función de una vía posee, al menos, una función hard core asociada. En consecuencia, la única primitiva criptográfica imprescindible para alcanzar la seguridad semántica es la función de una vía.

Del desarrollo anterior se deduce la enorme importancia que las funciones hard core tienen en criptografía: son la piedra angular sobre la que se construye la noción menos restrictiva de seguridad criptográfica. Existen numerosos trabajos acerca de cómo contruir funciones hard core asociadas a cualquier función de una vía o a las funciones concretas más relevantes. Dentro de esta línea de investigación, se estudian también las llamadas *funciones robustas* asociadas a las funciones de una vía. Se dice que cierta función $b : \{0,1\}^* \longmapsto \{0,1\}^*$ es *robusta* para una función de una vía f si obtener $b(x)$ a partir de $f(x)$ es tan difícil como obtener x. Notar que b sólo es *hard core* si, conociendo $f(x)$, es imposible distinguir $b(x)$ de una cadena aleatoria de bits (luego ser hard-core es una condición mucho más fuerte). Las funciones ro-

bustas tienen numerosas aplicaciones, sobre todo en el diseño de implementaciones concretas de criptosistemas. Remitimos al lector interesado en estos temas al artículo monográfico [43], así como a los trabajos acerca de funciones hard core asociadas al criptosistema RSA [3, 11, 28, 35, 57], o a los estudios sobre funciones robustas de la función de Diffie-Hellman [23, 24, 46, 44, 47].
La seguridad semántica (también llamada seguridad frente a adversarios pasivos) se formaliza a través de un modelo que describe la posible actuación de un adversario que tiene acceso a un simulador del algoritmo de cifrado. El objetivo del adversario es violar la llamada propiedad de *indistinguibilidad* del esquema.

Indistinguibilidad. Sea $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ un criptosistema de clave pública. El adversario es un individuo capaz de ejecutar cualquier algoritmo polinomial probabilístico. Se ejecuta el algoritmo de generación de claves $\mathcal{K}$ y se le proporciona al adversario la clave pública p_k.
Supongamos que el adversario elige dos mensajes m_0 y m_1, que pueden cifrarse con la clave pública p_k, y le proporciona la terna (m_0, m_1, p_k) al simulador. Éste elige un bit $b \in \{0, 1\}$ (uniformemente al azar) y presenta al adversario el *reto* $\mathcal{E}_{p_k}(m_b)$, pero no el bit b.
Se dice que el esquema Π garantiza la *indistinguibilidad del cifrado* o que es *semánticamente seguro* si y sólo si el adversario no puede adivinar, con probabilidad superior[4] a $\frac{1}{2}$, si el mensaje claro correspondiente con el reto es m_0 ó m_1.

La seguridad de un esquema criptográfico puede también medirse en función de lo fácil o difícil que resulte *falsificar* mensajes cifrados a partir de información conocida.

Maleabilidad. Supongamos (en un escenario similar al anterior) que el adversario tiene acceso a la clave pública, a partir de la cual construye un algoritmo $\mathcal{M}$ llamado *de muestreo.* Este algoritmo le sirve para simular un espacio de textos claros de tamaño prefijado, es decir, tiene como salida cadenas de bits que se corresponden con mensajes no cifrados. Supongamos que el adversario recibe del simulador el *reto,* un texto cifrado $c := \mathcal{E}_{p_k}(m)$ correspondiente a un mensaje m dentro de una muestra seleccionada por $\mathcal{M}$. El objetivo del adversario es ahora encontrar n textos cifrados $c_1, \ldots, c_n$ distintos de c de modo que sus textos claros correspondientes guarden una relación conocida con m. Un esquema criptográfico se dice *no maleable* si la probabilidad de que el adversario encuentre dichos textos para el reto c no

[4]En realidad, *significativamente superior*, ver la definición formal de [9].

es significativamente mayor que dicha probabilidad para el cifrado de una cadena de bits x elegida al azar en el espacio muestral definido por $\mathcal{M}$.

1.3.3. Seguridad frente a adversarios activos.

En la sección anterior sólo hemos considerado adversarios pasivos, i.e., adversarios que no tenían más información que la de un emisor autorizado. Es claro que este modelo es insuficiente, pues en la práctica la información disponible para un intruso suele ser superior a la que tiene un emisor. Actualmente, se utiliza la siguiente nomenclatura para clarificar la actuación del adversario en las dos pruebas de seguridad descritas anteriormente. Se habla de:

- Seguridad CPA. Adversarios pasivos. Sólo tienen a su disposición un simulador del algoritmo de cifrado (como en la descripción anterior).
- Seguridad CCA. Adversarios activos. Tienen acceso a un oráculo que descifra cualquier texto, pero únicamente *antes* de recibir el *reto*.
- Seguridad CCA2. Adversarios activos. Tienen acceso a un oráculo que descifra cualquier texto *distinto del reto*, durante tiempo ilimitado.

Según la nomenclatura actual, se habla de esquemas IND-CPA, IND-CCA e IND-CCA2 para hacer referencia a los esquemas de cifrado indistinguible frente a distintos tipos de adversarios. En particular, la seguridad IND-CPA es exactamente la seguridad semántica definida por Goldwasser y Micali. De modo análogo, se definen las nociones de seguridad NM-CPA, NM-CCA y NM-CCA2, que designan a los criptosistemas no maleables frente a los distintos tipos de adversarios.
En [9] puede encontrarse una formalización rigurosa de las nociones de seguridad que hemos comentado. También, en dicho artículo se demuestran las principales relaciones entre dichas nociones. Quizá la más relevante sea el hecho de que las nociones más restrictivas, la seguridad IN-CCA2 y la NM-CCA2 son equivalentes. Precisamente esa noción es el estándar de seguridad actual. Una discusión que analiza también métodos de ataque *físicos,* no incluidos en estas nociones, puede verse en [15].

Nota: En lo anterior, se ha considerado únicamente el modelo de computación clásico (no cuántico). Quizá en un futuro se introduzcan nuevas nociones de seguridad que admitan la existencia de adversarios provistos de ordenadores cuánticos. Una excelente discusión acerca de seguridad y computación cuántica puede verse en [13, 14].

Capítulo 2

Esquemas basados en problemas de factorización

Uno de los pilares de la criptografía de clave pública es el problema de factorización de enteros, es decir, el problema de encontrar, dado un número entero N, el conjunto de números primos $p_1, \ldots, p_k$ tales que $N = p_1 \cdots p_k$. En él se basa la seguridad del criptosistema RSA, una de las herramientas criptográficas más analizadas e implementadas en la breve historia de la criptografía de clave pública. Otras muchas construcciones criptográficas con utilidades muy diversas se fundamentan también en él (ver, por ejemplo [61, 84]). Siglos de búsqueda infructuosa justifican la idea de que no existe un algoritmo eficiente capaz de resolver dicho problema. De hecho, sólo recientemente se ha encontrado un algoritmo polinomial que determine si un número entero dado es primo o compuesto [2].

Dentro del marco de la teoría de lenguajes se han propuesto diversos criptosistemas basados en problemas muy generales de factorización o reescritura. La mayoría se consideran inseguros, bien por haberse encontado métodos eficientes de ataque, o por no haberse justificado debidamente la dificultad de los problemas subyacentes. Dichas propuestas utilizan factorizaciones asociadas a las palabras de un alfabeto Σ sobre el que se imponen ciertas reglas de reescritura. Estudiaremos con cierto detalle el esquema de Siromoney y Mathew descrito en [77]. Construcciones similares pueden verse en [87, 92].

2.1. Esquema basado en palabras Lyndon

2.1.1. Descripción del esquema

Sea Σ un alfabeto totalmente ordenado por una relación $<$. Considerar el monoide Σ^*, formado por todas las palabras (cadenas finitas ordenadas de símbolos) sobre Σ con la operación yuxtaposición. Así, Σ^* puede ordenarse a través de la relación de orden en Σ de la siguiente forma (*orden lexicográfico*): dadas dos palabras $\mathbf{u}, \mathbf{v} \in \Sigma^*$, $\mathbf{u} < \mathbf{v}$ si y sólo si se da una de las siguientes condiciones:

1. existe una palabra no vacía $\mathbf{w} \in \Sigma^*$ tal que $\mathbf{uw} = \mathbf{v}$,
2. existen palabras $\mathbf{r}, \mathbf{s}, \mathbf{t} \in \Sigma^*$ y $\mathbf{a}, \mathbf{b} \in \Sigma$, con $\mathbf{a} < \mathbf{b}$ tal que $\mathbf{u} = \mathbf{ras}$ y $\mathbf{v} = \mathbf{rbt}$.

En [33] se demuestra que las palabras de Σ^* tienen una factorización única como producto de las llamadas palabras Lyndon.

Definición 2.1 *Dadas dos palabras* $\mathbf{u}, \mathbf{v} \in \Sigma^*$, *se dice que* $\mathbf{v}$ *es* conjugada *de* $\mathbf{u}$ *(o que* $\mathbf{u}$ *y* $\mathbf{v}$ *son* conjugadas*) si existen palabras* $\mathbf{x}, \mathbf{y} \in \Sigma^*$ *tales que* $\mathbf{u} = \mathbf{xy}, \mathbf{v} = \mathbf{yx}$. *Toda palabra que sea estrictamente menor que todas sus conjugadas (distintas de ella) se dice* palabra Lyndon.

Notar que, en particular, toda palabra formada por un único símbolo $\sigma \in \Sigma$ es una palabra Lyndon.

Teorema 2.2 *Sea* Σ *un alfabeto totalmente ordenado. Cualquier palabra* $\mathbf{w} \in \Sigma^*$ *admite una única factorización* $\mathbf{w} = \mathbf{w}_1 ... \mathbf{w}_m$ *tal que cada* $\mathbf{w}_i$ *es una palabra Lyndon y además*

$$\mathbf{w}_1 \geq \mathbf{w}_2 \geq \cdots \geq \mathbf{w}_m.$$

La factorización anterior suele llamarse *factorización estándar* de una palabra, y sirve de base para el esquema de Siromoney y Mathew. Necesitamos también introducir la siguiente definición

Definición 2.3 *Dado un alfabeto* Λ, *llamamos* sistema Thue *definido sobre* Λ *a cualquier subconjunto* $T \subseteq \Lambda^* \times \Lambda^*$.

Pasamos ya a describir el esquema de Siromoney y Mathew:
Sea Λ un alfabeto de cardinal mayor que Σ, $g : \Lambda^* \longmapsto \Sigma^*$ una aplicación que, en particular, hace corresponder a cada símbolo de Λ uno de $\Sigma \cup \{e\}$ (donde

e denota a la palabra vacía). Para que sea posible descifrar, es necesario imponer que para todo $\lambda_1, \lambda_2 \in \Lambda$ se cumpla $g(\lambda_1, \lambda_2) = g(\lambda_1)g(\lambda_2)$. Esto puede conseguirse definiendo g como una extensión de una aplicación de Λ en $\Sigma \cup \{e\}$.

La clave pública consta de los siguientes elementos:

- el alfabeto Λ,
- para cada $\sigma \in \Sigma$, un conjunto de palabras en Λ^*, $E_\sigma \subseteq \Lambda^*$, que se corresponden por g con palabras Lyndon en Σ^*. Dicho conjunto se elige de modo que $g(E_\sigma) \cap g(E_\tau) = \emptyset$ si $\sigma \neq \tau$. Además, si $\mathbf{u}, \mathbf{v} \in \cup_{\sigma\in\Sigma} E_\sigma$

$$|\mathbf{u}| < |\mathbf{v}| \Rightarrow g(\mathbf{u}) < g(\mathbf{v}).$$

- $T \subseteq \Lambda^* \times \Lambda^*$ un sistema Thue que cumpla la condición

$$(\mathbf{u}, \mathbf{v}) \in T \Rightarrow g(\mathbf{u}) = g(\mathbf{v}).$$

Por otro lado, la clave privada consiste en la aplicación g y el conjunto de palabras Lyndon $\cup_{\sigma\in\Sigma} g(E_\sigma)$.

Si un individuo A quiere enviar un mensaje $m = \sigma_1 \dots \sigma_n \in \Sigma^*$ a un receptor autorizado B, construye una palabra $\mathbf{w} := \mathbf{e}_1 \dots \mathbf{e}_n$ con cada $\mathbf{e}_i \in E_{\sigma_i}$ elegido de modo que para cada $1 < j \leq n$ se cumpla $|\mathbf{e}_j| < |\mathbf{e}_{j-1}|$ ó $\mathbf{e}_j = \mathbf{e}_{j-1}$ (si no existen palabras $\mathbf{e}_i$ en esas condiciones, A fragmentará m en subpalabras a las que sí pueda aplicar el método y las cifrará por separado). Después la reescribe reemplazando una subpalabra $\mathbf{u} \in \Lambda^*$ de $\mathbf{w}$ por una palabra $\mathbf{v} \in \Lambda^*$ tal que $(\mathbf{u}, \mathbf{v}) \in T$. Puede repetir este paso de reescritura un número finito de veces. Finalmente envía la palabra resultante del proceso, $\mathbf{c}$, a B.

Para descifrar, a B le basta con calcular $g(\mathbf{c})$ y luego obtener su factorización estándar (que será, por unicidad, $g(\mathbf{e}_1) \dots g(\mathbf{e}_n)$) . La antiimagen por g de cada palabra Lyndon de la factorización desvela un símbolo del texto claro (el símbolo σ tal que dicha antiimagen está en E_σ). Un observador que no disponga de la clave privada debe construir, para leer el mensaje, una palabra equivalente a $\mathbf{c}$ concatenando palabras de $\cup_{\sigma\in\Sigma} E_\sigma$ de longitud decreciente.

Es obvio que este esquema es altamente ineficiente, pues la longitud de cada texto cifrado es mucho mayor que la de del texto claro correspondiente. Además, el criptosistema de Siromoney y Mathew permite ver fácilmente qué tipo de problemas presentan los esquemas que utilizan factorizaciones basadas en reglas de reescritura.

Estos son algunos de los métodos de ataque difíciles de evitar:

- Análisis de símbolos superfluos (que no aparecen en el sistema Thue).
- Inspección de los extremos del texto cifrado.
- Análisis de la longitud del texto cifrado.

Veremos cómo se llevan a cabo en un ejemplo concreto.

2.1.2. Análisis de seguridad

En [77] los autores del esquema general proponen una construcción concreta, donde los textos claros son cadenas de bits, i. e., $\Sigma := \{0, 1\}$. Por otro lado, los textos cifrados son palabras de un alfabeto de 20 letras:

$$\Lambda := \{c_1, \ldots, c_{20}\}.$$

La clave pública está formada por dos conjuntos

$$\begin{aligned} E_0 &:= \{c_1c_2c_{16}, c_2c_3c_7c_5c_{12}c_{11}, c_4c_9c_{12}c_8c_{15}c_{19}c_7c_{20}c_{11}c_{19}\}, \\ E_1 &:= \{c_3c_1c_6c_5c_{18}, c_{18}c_2c_6c_1c_{12}c_4c_{17}c_5c_6\}, \end{aligned}$$

y el sistema Thue T viene dado por las siguientes reglas de reescritura:

$$\begin{array}{lll} c_{11}c_3 \leftrightarrow c_1c_{19}, & c_9c_{13} \leftrightarrow c_{17}c_8, & c_{18}c_1c_2 \leftrightarrow c_3c_{15}c_{19}, \\ c_2c_3c_7 \leftrightarrow c_{12}c_{15}c_1, & c_1c_5c_{12} \leftrightarrow c_3c_4c_9, & c_{15}c_{19}c_{16} \leftrightarrow c_{18}c_{13}c_{12}, \\ c_{19}c_6c_5 \leftrightarrow c_{12}c_{13}c_4, & c_5c_{18}c_3c_1 \leftrightarrow c_4c_1c_{11}c_3, & c_9c_1c_{19}c_1c_6 \leftrightarrow c_8c_{11}c_3c_7c_1, \\ c_3c_5c_1c_6c_{19}c_4c_6 \leftrightarrow c_6c_1c_4c_3c_5c_{12}c_{14}. & & \end{array}$$

Por ejemplo, un resultado posible de cifrar texto claro 0110 es la palabra

$$c_{12}c_{15}c_3c_4c_8c_{11}c_3c_7c_1c_4c_1c_1c_{12}c_{13}c_4c_3c_{15}c_{19}c_{16}, \tag{2.1}$$

(ver la Sección 6 de [77]).

Veamos ahora cómo pueden aplicarse los ataques mencionados a este ejemplo:

- **Análisis de símbolos superfluos.** El sistema Thue del ejemplo no contiene ninguna regla que involucre a los símbolos c_{10} ó c_{20}. Como además c_{10} no aparece en ninguna palabra de $E_0 \cup E_1$, dicho símbolo es superfluo y podemos ignorarlo.

En el caso de c_{20} la situación es mucho peor: existe exactamente una palabra en $E_0 \cup E_1$ que contiene a c_{20}. Por tanto, al no tener ninguna regla de reescritura para ocultar c_{20}, si este símbolo aparece en un texto cifrado podremos estar seguros de que procede de la palabra $c_4c_9c_{12}c_8c_{15}c_{19}c_7c_{20}c_{11}c_{19} \in E_0$, y de que por tanto el bit correspondiente es el 0.

- **Inspección de los extremos del texto cifrado.** Notar que para representar el bit 1 hemos de tomar una de las palabras de E_1, cuyo símbolo inicial será c_3 o c_{18}. Inspeccionando los primeros símbolos de las palabras involucradas en T, vemos que cualquier texto que comience por el bit 1 da lugar a un texto cifrado que comienza con uno de estos siete símbolos: $c_1, c_3, c_6, c_{11}, c_{15}, c_{18}$.

  Por ejemplo, el texto cifrado (2.1) comienza por el símbolo c_{12}, por lo que sabemos que el primer bit del texto claro correspondiente es el 0. Es más, puesto que sabemos que el símbolo c_{20} puede ser excluido y como uno de los elementos restantes de E_0 comienza por c_1, sabemos también que dicho 0 se ha cifrado usando la palabra $c_1c_2c_{16}$.

  De forma análoga puede actuarse para averiguar el último bit del texto claro: si el texto claro acaba en 1, el cifrado correspondiente ha de terminar con uno de los ocho símbolos: $c_1, c_2, c_3, c_6, c_7, c_{14}, c_{18}, c_{19}$.— Como en el texto cifrado de análisis (2.1) el último símbolo es c_{16}, sabemos que el texto claro correspondiente acaba en 0.

- **Análisis de la longitud del texto cifrado.** Los propios autores hacen notar en [77] que la longitud del texto cifrado siempre puede acotarse en función de la longitud del texto claro y las reglas de reescritura. Si las reglas de reescritura (como ocurre en el ejemplo) sólo establecen equivalencias entre palabras de igual longitud, la longitud del texto cifrado no cambia a partir del primer paso del algoritmo de cifrado. Esto puede revelar una cantidad importante de información.

  En el ejemplo que nos ocupa, los conjuntos E_0 y E_1 sólo contienen palabras de longitud 3, 5, 6, 9 y 10. De las posibles 490 particiones de la longitud del texto cifrado (2.1) sólo 6 pueden expresarse a partir exclusivamente de los números 3, 5, 6, 9 y 10. Otras tres pueden excluirse si recordamos que el símbolo c_{20} es superfluo (y por tanto, la palabra de longitud 10 también). Es más, debido a la restricción $|\mathbf{e}_j| \leq |\mathbf{e}_{j-1}|$ las tres particiones que quedan nos bastan para determinar el texto claro

completamente:

$$\begin{array}{rcllcrcl} 19 & = & 9+5+5 & & \rightarrow & m & = & 111 \\ 19 & = & 6+5+5+3 & & \rightarrow & m & = & 0110 \\ 19 & = & 5+5+3+3+3 & & \rightarrow & m & = & 11000 \end{array}$$

Como ya habíamos razonado que el primer bit del texto claro era un 0, es claro que el texto claro correspondiente al cifrado (2.1) es el 0110.

En resumen, está claro que el criptosistema de Siromoney y Mathew no puede considerarse seguro (a menos que se especifique cómo elegir claves adecuadas que eviten los ataques mencionados). Sin embargo la idea de utilizar reglas de reescritura para cifrar ha seguido explorándose en criptografía, aunque utilizando como base estructuras más complejas. En concreto, los ataques anteriores no son tan efectivos en construcciones que utilizan, en lugar de reglas de reescritura, relaciones en un grupo finitamente presentado. En la siguiente sección estudiaremos en detalle este tipo de esquemas.

2.2. Esquemas basados en el problema de la palabra

Durante los años ochenta se propusieron algunos criptosistemas basados en el problema de la palabra en grupos finitamente presentados:

Problema de la palabra: *¿Es posible construir un algoritmo que decida, dada una presentación finita $\langle X/R\rangle$ de un grupo G y una palabra* $\mathbf{w}$ *en $X \cup X^{-1}$, cuándo* $\mathbf{w}$ *representa el elemento neutro de G, esto es, cuándo pertenece a la clausura normal de R en el grupo libre sobre X?*

Diremos que un grupo finitamente presentado $G = \langle X/R\rangle$ tiene problema de la palabra resoluble si existe un algoritmo para decidir cuándo una palabra en los generadores representa al elemento neutro. Puede probarse que tener problema de la palabra resoluble es una propiedad intrínseca del grupo i.e., no depende de la presentación finita que se considere.

Analizaremos en detalle dos criptosistemas basados en este problema que están inspirados en la misma construcción general, propuesta por Wagner y Magyarik en 1984 [97].

2.2.1. La construcción general de Wagner y Magyarik

Sea G un grupo, del cual conocemos una presentación finita $\langle X/R\rangle$ y para el cual el problema de la palabra es *muy difícil* (al menos exigiremos que no

se conozca ningún algoritmo polinomial que sirva para decidir si una palabra representa al elemento neutro de G).
Supongamos también que existe un cociente de G, $\tilde{G}$, que podemos construir fácilmente ampliando el conjunto R y para el cual el problema de la palabra sí puede resolverse en tiempo polinomial.
Sea ahora $W(\Sigma)$ un conjunto de palabras en $X \cup X^{-1}$, biyectivo con el alfabeto Σ empleado para la comunicación. Dicho conjunto, $W(\Sigma) = \{\mathbf{w}_\sigma \mid \sigma \in \Sigma\}$, estará formado por palabras que representen distintos elementos de G y que, además, no 'colisionen' en $\tilde{G}$ (esto es, estas palabras representan también elementos distintos en dicho cociente).

La clave pública del esquema está formada por la presentación mencionada de G, $\langle X/R\rangle$ y el conjunto $W(\Sigma)$. El grupo $\tilde{G}$, o, equivalentemente, un conjunto finito de relaciones S tal que $\tilde{G} = \langle X/R \cup S\rangle$, constituye la clave privada del esquema.

El cifrado de un símbolo $\sigma \in \Sigma$ se realiza reescribiendo la palabra $\mathbf{w}_\sigma$, es decir, construyendo a través de las relaciones de R una palabra $\mathbf{c}$ equivalente a $\mathbf{w}_\sigma$. Para descifrar, simplemente hay que identificar la palabra de $W(\Sigma)$ que es equivalente a $\mathbf{c}$; un receptor autorizado podrá hacerlo usando un algoritmo que resuelva el problema de la palabra en $\tilde{G}$, pues dos palabras equivalentes en G siguen siéndolo en cualquier cociente.

Para que un esquema de este tipo pueda considerarse seguro, es necesario al menos garantizar que es robusto ante los siguientes ataques:

- **Ataque directo al problema matemático de base:** puesto que para descifrar sólo es necesario decidir a cuál de las palabras públicas es equivalente una palabra dada, este problema ha de ser suficientemente difícil en G. Para ello no basta que el problema de la palabra en G sea difícil.

- **Construcción de una clave privada alternativa:** un adversario debe ser incapaz de construir un cociente $\hat{G}$ de G para el cual exista un algoritmo eficiente para el problema de la palabra y en el que las palabras de $W(\Sigma)$ no colisionen (pues distinguir palabras en dicho cociente basta para descifrar).

Esquema general de cifrado basado en el problema de la palabra:

- Clave pública: $G = \langle X/R \rangle, \quad W(\Sigma)$.
- Clave privada: S (equivalentemente, $\tilde{G} = \langle X/R \cup S \rangle$).

- *A quiere enviar un mensaje a B, para ello cifra cada símbolo σ escribiendo una palabra $\mathbf{c}_\sigma$, equivalente a $\mathbf{w}_\sigma$ en G (paso de reescritura).*
- *B descifra cada palabra $\mathbf{c}$ recibida, obteniendo la palabra $\mathbf{w}_\sigma$, equivalente a $\mathbf{c}$ en $\tilde{G}$ (y en G) y recuperando así el símbolo transmitido σ.*
- *Un observador que no dispone de la clave privada ha de decidir qué palabra de $W(\Sigma)$ es equivalente a $\mathbf{c}$ para obtener σ.*

Es claro que no es fácil garantizar que los ataques mencionados no tengan éxito ante una elección concreta de claves. Sin embargo, es sencillo ver que los ataques de 2.1.2 son ahora fáciles de evitar: por ejemplo, en este caso no hay símbolos superfluos, todos los generadores aparecen en al menos la relación trivial $x_i x_i^{-1} = e$. También por la intervención de los inversos de los generadores es fácil elegir las claves de modo que ni los extremos del texto cifrado ni su longitud revelen información alguna. Más adelante haremos un análisis más profundo de seguridad que nos permitirá ver que, a pesar de que esta idea mejora considerablemente la construcción basada en palabras Lyndon, es difícil que inspire la construción de un criptosistema seguro.

En cuanto a la eficiencia del esquema, aun suponiendo que la generación de claves y el proceso de cifrado puedan realizarse de manera eficiente, quedaría por resolver el problema de la expansión del texto cifrado: al cifrar transformamos cada símbolo del alfabeto en una palabra de longitud arbitraria, que seguramente se representará por una cadena de bits al menos polinomial en dicha longitud. Este problema podría solventarse, al menos en parte, asociando k palabras públicas a k cadenas de símbolos (en lugar

de a símbolos individuales), pero entonces la generación de claves podría complicarse mucho, y además el tamaño de la clave pública aumentaría considerablemente.

El ejemplo de Wagner y Magyarik

En [97], los autores proponen un método concreto de llevar a cabo la construcción anterior. En particular, sugieren la elección de un grupo G para la clave pública que quede definido a partir de un conjunto finito de generadores $X = \{x_1, \ldots, x_n\}$ sujetos exclusivamente a relaciones de tres tipos:

- $T1$. $x_i x_j x_k x_l = x_l x_k x_j x_i, \quad x_i, x_j, x_k, x_l \in X$,
- $T2$. $x_i x_j x_k = x_k x_j x_i, \quad x_i, x_j, x_k \in X$,
- $T3$. $x_i x_j x_k = x_j x_k x_i, \quad x_i, x_j, x_k \in X$.

Es fácil ver que, en ese caso, las relaciones de la presentación pueden hacerse triviales añadiendo un nuevo conjunto de relaciones, S, formado por ecuaciones de estos tipos:

- $S1$. $x = y, \quad x, y \in X \cup X^{-1}$,
- $S2$. $x = e, \quad x \in X \cup X^{-1}$,
- $S3$. $x_i x_j = x_j x_i, \quad x_i, x_j \in X$.

En general no será necesario que las relaciones de S involucren a todos los generadores, si bien el tamaño de S dependerá, como es lógico, de la elección del grupo G. Imponiendo que los generadores cumplan las relaciones de S puede construirse fácilmente un cociente $\tilde{G}$ de G que pueda presentarse a través de un subconjunto de X y una serie de relaciones de tipo $S3$. Nótese que como dichas relaciones no tienen por qué involucrar a todos los generadores del grupo, $\tilde{G}$ no es necesariamente abeliano. Este cociente podrá utilizarse como clave privada, pues existe un algoritmo polinomial para resolver el problema de la palabra en este tipo de grupos.
El conjunto de palabras públicas $W(\Sigma)$ ha de construirse ahora de forma que en $\tilde{G}$ sigan representando distintos elementos. Además se intentará que colisionen en la mayoría de los cocientes de G construidos añadiendo relaciones de los tipos $S1, S2$ y $S3$ (que un adversario puede construir fácilmente).

Es relativamente fácil dar ejemplos concretos de cómo llevar a cabo la construcción anterior. Veamos por ejemplo este sencillo método (c.f. [97]):
Dado un conjunto finito X,

1. Separar un conjunto pequeño de pares de elementos de X, que denotaremos $\mathcal{P}$.

2. Escribir un conjunto R de relaciones de tipo $T1$, $T2$ y $T3$ que puedan reducirse a la trivial añadiendo un conjunto S de relaciones de tipo $S1, S2$ y $S3$. Ninguna de las relaciones de S implicará que los dos elementos de un par de $\mathcal{P}$ conmuten.

3. Procurar además al construir S que el mayor número de pares que no pertenezcan a $\mathcal{P}$ conmuten en el cociente $\tilde{G}$.

4. El conjunto de palabras públicas $W(\Sigma)$ se construirá de modo tal que si hiciésemos conmutar los dos elementos de alguno de los pares de $\mathcal{P}$, todas sus palabras serían equivalentes a la palabra vacía.

Existen dudas razonables acerca de la seguridad de este esquema, pues ni siquiera se conoce la complejidad del problema de la palabra en el grupo G. En cualquier caso, al dar este ejemplo los autores sólo pretendían demostrar la viabilidad de su construcción general, y en ningún momento argumentan que este esquema sea seguro o eficiente.

2.2.2. El esquema de Garzón y Zalcstein

Varios años después de la aparición del trabajo de Wagner y Magyarik, Garzón y Zalcstein propusieron en [37] un esquema inspirado en su construcción general que utiliza grupos de Grigorchuk como claves privadas. Nótese que estrictamente hablando, el esquema de Garzón y Zalcstein no es un caso particular del esquema descrito en 2.2.1, pues ahora los grupos involucrados no son todos finitamente presentados.

Los grupos de Grigorchuk son grupos de permutaciones del conjunto de caminos infinitos que descienden desde la raiz de un árbol binario completo e infinito. Todos pueden generarse con sólo cuatro elementos, pero la mayoría no admiten presentaciones finitas.

Definición 2.4 *Sea Γ el conjunto de todos los caminos infinitos descendentes que empiezan en la raíz del árbol binario infinito completo.*
Sea χ una secuencia ternaria infinita.
Definimos a partir de χ una matriz de tres filas e infinitas columnas

$$\mathcal{M}_\chi = \begin{pmatrix} U \\ V \\ W \end{pmatrix}.$$

Las entradas de $\mathcal{M}_\chi$ *son símbolos* S *o* I*, que indican respectivamente, como veremos, cambio o invariabilidad. Cada columna* j *de la matriz* $\mathcal{M}_\chi$ *se define de la siguiente forma:*

$$\begin{pmatrix} u_j \\ v_j \\ w_j \end{pmatrix} \text{ es}$$

$$\begin{pmatrix} S \\ S \\ I \end{pmatrix} \text{ si } \chi_j = 0, \quad \begin{pmatrix} S \\ I \\ S \end{pmatrix} \text{ si } \chi_j = 1 \text{ y } \begin{pmatrix} I \\ S \\ S \end{pmatrix} \text{ si } \chi_j = 2.$$

Dado un camino $\gamma \in \Gamma$*, lo identificaremos con una secuencia binaria* $(\gamma_i)_{i\geq 1}$*, donde el* 0 *y el* 1 *simbolizan respectivamente giros a izquierda y derecha en el descenso por el árbol. En lo sucesivo, dado* $x \in \{0,1\}$ *denotamos por* $\bar{x}$ *a* $x + 1 \bmod 2$.

El grupo de Grigorchuk asociado a χ*,* G_χ*, es el subgrupo de* S_Γ *generado por las permutaciones* a, b_χ, c_χ *y* d_χ*, que se definen a partir de la matriz* $\mathcal{M}_\chi$:

- $a(\gamma) = (a(\gamma)_i)_{i\geq 1}$ *con* $a(\gamma)_i = \begin{cases} \bar{\gamma}_i & \text{si } i = 1, \\ \gamma_i & \text{en otro caso.} \end{cases}$
- $b_\chi(\gamma) = (b_\chi(\gamma)_i)_{i\geq 1}$ *con*

$$b_\chi(\gamma)_i = \begin{cases} \gamma_i & \text{si } i \leq k \text{ siendo } \gamma_k \text{ el primer } 1 \text{ de la secuencia } \gamma, \\ \bar{\gamma}_i & \text{si } u_{i-1} = S \text{ e } i > k, \\ \gamma_i & \text{si } u_{i-1} = I \text{ e } i > k. \end{cases}$$

- $c_\chi(\gamma) = (c_\chi(\gamma)_i)_{i\geq 1}$ *con*

$$c_\chi(\gamma)_i = \begin{cases} \gamma_i & \text{si } i \leq k \text{ siendo } \gamma_k \text{ el primer } 1 \text{ de la secuencia } \gamma, \\ \bar{\gamma}_i & \text{si } v_{i-1} = S \text{ e } i > k, \\ \gamma_i & \text{si } v_{i-1} = I \text{ e } i > k. \end{cases}$$

- $d_\chi(\gamma) = (d_\chi(\gamma)_i)_{i\geq 1}$ *con*

$$d_\chi(\gamma)_i = \begin{cases} \gamma_i & \text{si } i \leq k \text{ siendo } \gamma_k \text{ el primer } 1 \text{ de la secuencia } \gamma, \\ \bar{\gamma}_i & \text{si } w_{i-1} = S \text{ e } i > k, \\ \gamma_i & \text{si } w_{i-1} = I \text{ e } i > k. \end{cases}$$

Se puede demostrar (ver [37]) que si la secuencia χ tiene ciertas propiedades de complejidad, el correspondiente grupo no es finitamente presentado y el problema de la palabra asociado es polinomial. De hecho, dado un número natural N es posible construir un algoritmo eficiente que decida si una palabra

w de longitud menor o igual que N representa a la identidad en el grupo de Grigorchuk G_χ.

Una vez fijada una secuencia χ adecuada, los autores proponen construir las claves (siguiendo el diseño de Wagner y Magyarik) de la siguiente forma:

- Clave pública: el grupo $G = \langle \{a, b, c, d\}/R \rangle$, donde R es un conjunto finito de relaciones que son satisfechas por a, b_χ, c_χ y d_χ tal que no se conozca ningún algoritmo polinomial para resolver el problema de la palabra en G. Un conjunto de palabras $W(\Sigma)$, que representen elementos distintos en G_χ.
- Clave privada: la secuencia χ. Conociéndola es posible resolver el problema de la palabra en G_χ en tiempo polinomial.

La seguridad de este esquema es tremendamente difícil de probar. Desde luego, sería necesario demostrar que el problema de la palabra en G es lo suficientemente complicado, pero no suficiente. Es importante además tener en cuenta cómo se realiza el proceso de descifrado. Si usamos la solución para el problema de la palabra que proponen los autores en [37], sólo sería necesario conocer una parte (finita) de la secuencia χ para ser capaces de descifrar textos de una longitud determinada. Como existe sólo un número polinomial de secuencias finitas ternarias susceptibles de permitir descifrar, en principio una búsqueda exhaustiva de dicha 'clave parcial' es posible. Sin embargo, los autores aseguran que un ataque de este tipo no es un peligro real, pues un adversario no tiene manera de comprobar en tiempo polinomial si una secuencia ternaria finita es parte de χ. Esto se debe a que un par texto llano-texto cifrado no determina de manera única la clave privada, luego una secuencia de tamaño k que sirva para transformar un cierto texto llano en otro cifrado puede sin embargo no ser parte de la clave privada. Veremos que, al razonar de esta forma, los autores olvidaron que aunque no pueda obtenerse la clave privada, a menudo es posible construir una clave alternativa que sirva para descifrar (como ya mencionábamos en 2.2.1).

2.2.3. Análisis de seguridad

Ataques de reacción al esquema de Wagner y Magyarik

A la hora de analizar la seguridad del esquema general basado en el problema de la palabra nos centramos en su funcionamiento en la práctica, pues un ataque al problema matemático subyacente es inviable en el caso general. Así, demostramos que el esquema de Wagner y Magyarik es vulnerable a

los llamados *ataques de reacción*, i.e., es posible recuperar la clave privada observando la reacción de un receptor legítimo ante ciertos mensajes. Este tipo de ataques fueron introducidos y utilizados por primera vez por Hall, Goldberg y Schneider [55], contra los esquemas de McEliece y Atjai-Dwork.

Supongamos, por simplicidad, que el alfabeto de la comunicación Σ es binario. El conjunto público $W(\Sigma)$ contiene entonces sólo dos palabras: $\mathbf{w}_0\,, \mathbf{w}_1$, que representan respectivamente a los bits 0 y 1. Supondremos también que las palabras $\mathbf{w}_0\mathbf{w}_1$ y $\mathbf{w}_1\mathbf{w}_0$ representan elementos distintos en G y $\widetilde{G}$.
La capacidad de observación del adversario se modela suponiendo que tiene acceso a un oráculo $\mathcal{O}$ — que puede verse como una aplicación

$$\mathcal{O} : \{X \cup X^{-1}\}^* \longrightarrow \{0, 1\}$$

— tal que dada una palabra $\mathbf{w} \in \{X \cup X^{-1}\}^*$, $\mathcal{O}(\mathbf{w}) = 1$ si $\mathbf{w}$ es un texto cifrado correctamente (i. e., si $\mathbf{w} \sim \mathbf{w}_i$ para algún $i \in \{0, 1\}$,) y $\mathcal{O}(\mathbf{w}) = 0$ en otro caso. Es decir, nuestra hipótesis fundamental es que el adversario pueda distinguir textos cifrados de palabras que no lo son.
También hemos de suponer que existe un conjunto de palabras $A \subseteq \{X \cup X^{-1}\}^*$ en el que es factible realizar una búsqueda exhaustiva. Además, partiremos de la hipótesis de que el subconjunto

$$\bar{S} = \{\mathbf{a} \in A \mid \mathbf{a} \sim e \text{ en } \widetilde{G}\}$$

permite construir un conjunto $\bar{\mathcal{S}}$ tal que $\langle X/\bar{\mathcal{S}}\rangle$ es una presentación de $\widetilde{G}$ (o bien de otro cociente de G que sirva de clave privada alternativa).
El objetivo de un adversario será encontrar $\bar{S}$ por medio del oráculo $\mathcal{O}$. Esto podrá hacerlo efectuando una búsqueda exhaustiva en A; para cada $\mathbf{a} \in A$ el adversario hace, a lo sumo, 2 llamadas al oráculo $\mathcal{O}$ para decidir si $\mathbf{a} \in \bar{S}$:

i. $\mathbf{a}\mathbf{w}_0$:

- Si $\mathcal{O}(\mathbf{a}\mathbf{w}_0) = 0$, claramente $\mathbf{a} \notin \bar{S}$.
- Si $\mathcal{O}(\mathbf{a}\mathbf{w}_0) = 1$, ó $\mathbf{a} \in \bar{S}$ o bien se tiene, en $\widetilde{G}$, $\mathbf{a}\mathbf{w}_0 \sim \mathbf{w}_1$ (y por tanto $\mathbf{a} \notin \bar{S}$). Para distinguir entre estas dos posibilidades, el adversario puede hacer una segunda llamada al oráculo:

ii. $\mathbf{w}_0\mathbf{a}$:

- Si $\mathcal{O}(\mathbf{w}_0\mathbf{a}) = 0$, claramente $\mathbf{a} \notin \bar{S}$.

- Si $\mathcal{O}(\mathbf{w}_0\mathbf{a}) = 1$, entonces ó $\mathbf{a} \in \bar{S}$ o bien se tiene, en $\widetilde{G}$, $\mathbf{w}_0\mathbf{a} \sim \mathbf{w}_1$ (y $\mathbf{a} \notin \bar{S}$). En este último caso ($\mathbf{a} \notin \bar{S}$) se sigue que $\mathbf{w}_0\mathbf{a}\mathbf{w}_0 \sim \mathbf{w}_1\mathbf{w}_0$. Sin embargo, la llamada anterior (i) revela que $\mathbf{a} \notin \bar{S}$ sólo ocurre si $\mathbf{a}\mathbf{w}_0 \sim \mathbf{w}_1$, i. e., $\mathbf{w}_0\mathbf{a}\mathbf{w}_0 \sim \mathbf{w}_0\mathbf{w}_1$—lo que contradice $\mathbf{w}_0\mathbf{w}_1 \not\sim \mathbf{w}_1\mathbf{w}_0$. En resumen, no puede ser que $\mathcal{O}(\mathbf{w}_0\mathbf{a}) = 1$ y $\mathbf{a} \notin \bar{S}$, y por tanto $\mathcal{O}(\mathbf{w}_0\mathbf{a}) = 1$ implica $\mathbf{a} \in \bar{S}$.

Este ataque, descrito en términos de la construcción general de 2.2.1, se materializa en la práctica en un algoritmo rápido y sencillo cuando se emplea contra construcciones más concretas sobre las que se dispone de información adicional. Veamos por ejemplo cómo funciona con la construcción concreta propuesta por Wagner y Magyarik (2.2.1).

Criptoanálisis del ejemplo de Wagner y Magyarik (2.2.1). El adversario ha de realizar el algoritmo de búsqueda exhaustiva descrito anteriormente tres veces, una por cada tipo de relaciones (S1), (S2) y (S3). Para encontrar relaciones del tipo (S1) comienza buscando en el conjunto $A_1 = X$ (de tamaño n). Con los relatores que identifique, construye un conjunto de palabras (de hecho, generadores) $\bar{S}_1$, que representan al neutro en $\widetilde{G}$. Llamemos X_2 al conjunto de generadores restante $X_2 = X \setminus \bar{S}_1$. El adversario efectúa a continuación su búsqueda en el conjunto

$$A_2 = \{x_i x_j^{-1} \mid x_i \neq x_j \text{ y } x_i, x_j \in X_2 \cup X_2^{-1}\}$$

(de tamaño $\mathrm{O}(n^2)$) con el objetivo de identificar relaciones del tipo (S2). Esta segunda búsqueda da lugar a un segundo conjunto $\bar{S}_2$ de palabras que representan al neutro en $\widetilde{G}$. Para buscar ahora relaciones del tipo (S3), basta restringirse al conjunto de palabras en aquellos generadores que no han sido identificados como superfluos hasta ahora. Denotamos ese conjunto de generadores (contenido en X_2) por X_3. La última búsqueda exhaustiva que realiza el adversario recorre el conjunto

$$A_3 = \{x_i x_j x_i^{-1} x_j^{-1} \mid x_i \neq x_j \text{ y } x_i, x_j \in X_3\}$$

(de tamaño $\mathrm{O}(n^2)$) y resulta en un conjunto $\bar{S}_3$ de palabras que representan al neutro de $\widetilde{G}$.
El adversario construye ahora el conjunto objetivo $\bar{S} = \bar{S}_1 \cup \bar{S}_2 \cup \bar{S}_3$. Es fácil comprobar que, de hecho, $\langle X/R \cup \bar{S} \rangle$ es una presentación del cociente secreto.

Criptoanálisis del esquema de Garzón y Zalcstein (2.2.2). El esquema basado en grupos de Grigorchuk propuesto por Garzón y Zalcstein en [37]

es vulnerable a un tipo de análisis muy similar al que hemos descrito. Dicho ataque es obra de Steinwandt y Hofheinz [91]. Estrictamente hablando no puede decirse que sea un caso particular del ataque general descrito en esta sección, pues éste fue formulado en términos de grupos finitamente presentados. Además, este criptoanálisis explota las características especiales de los grupos escogidos y el hecho de que el adversario sabe que la clave privada es un grupo de Grigorchuck. Observar también que en este caso no es necesaria interacción alguna con un receptor autorizado (puede prescindirse del oráculo). De todas formas la idea subyacente es muy similar: se persigue encontrar un cociente como clave privada válida comprobando que en él las palabras de $W(\Sigma)$ no colisionen y que las relaciones públicas se cumplan. Este cociente no tiene por qué coincidir con la clave privada que poseen los receptores autorizados (por eso no es necesario el oráculo), pero sirve igualmente para descifrar.

Recordemos que para todo número natural N, conocidos los $\lceil log_2 N \rceil$ primeros elementos de cualquier secuencia ternaria χ, es posible construir un algoritmo eficiente (que denotaremos $\mathcal{A}_\chi$) que decida si una palabra $\mathbf{w}$ de longitud menor o igual que N representa a la identidad en el grupo de Grigorchuk G_χ. Esa es la razón por la que en el esquema de Garzón y Zalcstein basta conocer cierta subsecuencia inicial de la clave privada χ para descifrar. Veamos entonces cómo puede aprovechar un adversario esta circunstancia: Sea M el máximo de las longitudes de las palabras del conjunto público de relatores R. Sea N el máximo entre M y la longitud de $\mathbf{w}_0^{-1}\mathbf{w}_1$. Asumimos que $\lceil log_2 N \rceil$ es un número lo bastante pequeño como para analizar todas las secuencias ternarias de esa longitud. Denotemos por $\bar{\chi}$ a la subsecuencia inicial de longitud $\lceil log_2 N \rceil$ de la clave privada χ. Por lo anterior, si dos grupos de Grigorchuk G_1 y G_2 están definidos por dos secuencias χ_1 y χ_2 cuyos $\lceil log_2 N \rceil$ primeros elementos coinciden, un algoritmo que decida qué palabras de longitud menor que N representan al neutro en G_1 resuelve el mismo problema en G_2. Por tanto, si un adversario identifica $\bar{\chi}$, será capaz de descifrar cualquier texto. Para ello, puede utilizar el algoritmo $\mathcal{A}_\Gamma$ asociado a cualquier grupo de Grigorchuck G_Γ, con Γ una prolongación de $\bar{\chi}$. El adversario realiza su ataque siguiendo los siguientes pasos:

- busca en el conjunto de secuencias ternarias de longitud $\lceil log_2 N \rceil$, una secuencia ξ tal que, en los grupos de Grigorchuk definidos por secuencias cuyos $\lceil log_2 N \rceil$ primeros elementos coinciden con ξ, cada palabra de R represente al elemento neutro y $\mathbf{w}_0$ no sea equivalente con $\mathbf{w}_1$.

  Obviamente, el adversario encontrará ξ en esas condiciones (pues al menos $\bar{\chi}$ las cumple y está en su conjunto de búsqueda).

- utiliza el algoritmo $\mathcal{A}_{\Upsilon_0}$ para descifrar, donde Υ_0 es la secuencia que resulta al concatenar ξ con una secuencia infinita de ceros. Obviamente G_{Υ_0} es un cociente válido como clave privada para el esquema.

A la vista de los dos criptoanálisis descritos en esta sección, es difícil imaginar una realización concreta del esquema de Wagner y Magyarik que garantice un mínimo nivel de seguridad. De cualquier forma, quizá la manera de llegar a construcciones seguras sea evitar los diseños muy generales y explotar la estructura de los grupos concretos utilizados en cada esquema. Con esta filosofía se diseñó el criptosistema MST_1, que estudiaremos con detalle en la siguiente sección.

2.3. El criptosistema MST_1

A finales de los ochenta se publicó un esquema de clave privada [69] que abría paso a un nuevo tipo de herramientas criptográficas basadas en grupos. Magliveras y otros iniciaban con ese esquema una nueva línea de investigación cuyo objetivo era explorar las aplicaciones en criptografía de ciertas factorizaciones de grupos de permutaciones finitos. El esquema que proponían [69], llamado PGM, fue exhaustivamente estudiado [72, 71, 73, 74, 75] y atrajo la atención de diseñadores y criptoanalistas durante varios años. Hoy sabemos que este esquema tiene en la práctica diversos inconvenientes, derivados en su mayoría de la utilización de la representación habitual de grupos de permutaciones. Solventando esas pequeñas deficiencias, hace cuatro años se propuso una variante del PGM; el esquema TST [60], más eficiente y con mejores propiedades de seguridad.

En lo que se refiere a clave pública, Magliveras, Stinson y van Trung han propuesto recientemente dos esquemas de cifrado llamados MST_1 y MST_2 [76]. En esta sección describiremos y analizaremos el esquema MST_1 con cierto detalle. En el capítulo siguiente haremos un estudio similar del esquema MST_2. Una extensa introducción a estos esquemas puede leerse en [17, 31, 70].

2.3.1. Descripción del esquema

El criptosistema MST_1 se construye a partir de las llamadas *signaturas logarítmicas,* secuencias de factorización definidas para grupos de permutaciones finitos. Recordar que un grupo finito G se dice grupo de permutaciones de grado $n \in \mathbb{N}$ si es isomorfo a un subgrupo del grupo simétrico S_n. El teorema de Cayley permite interpretar cualquier grupo finito como un grupo de permutaciones de grado igual a su orden.

Definición 2.5 *Sea G un grupo de permutaciones de grado n. Para $s \in \mathbb{N}_0$, sea $\alpha = [\alpha_1, \ldots, \alpha_s]$ una secuencia tal que α_i $(1 \leq i \leq s)$ es una secuencia de elementos de S_n, $\alpha_i = [\alpha_{i0}, \ldots, \alpha_{ir_i-1}]$, con $r_i \in \mathbb{N}_0$ $(0 \leq j < r_i)$.*
Decimos que α es una signatura logarítmica *de G si $|G| = r_1 \cdots r_s$, y cada elemento $g \in G$ se representa de manera única como un producto*

$$g = \alpha_{1j_1} \cdots \alpha_{sj_s}$$

con $\alpha_{ij_i} \in \alpha_i$ $(1 \leq i \leq s)$.
Si $\alpha = [\alpha_1, \ldots, \alpha_s]$ es una signatura logarítmica, llamamos bloques *de α a las secuencias $\alpha_i = [\alpha_{i0}, \ldots, \alpha_{ir_i-1}]$ $(1 \leq i \leq s)$. El vector $r = (r_1, \ldots, r_s)$ se dirá* tipo *de α.*
Denotamos por $\Lambda(G)$ al conjunto de todas las signaturas logarítmicas de G (o simplemente por Λ si no hay ambigüedad posible).

Notación: Denotaremos por $\alpha_i \cdot \alpha_j$ a la secuencia

$$[\alpha_{i0}\alpha_{j0}, \alpha_{i0}\alpha_{j1}, \ldots, \alpha_{i0}\alpha_{jr_j-1}, \ldots, \alpha_{ir_i-1}\alpha_{j0}, \ldots, \alpha_{ir_i-1}\alpha_{jr_j-1}].$$

A menudo se hace referencia a dicha secuencia llamándola *bloque producto* de α_i por α_j.

La dificultad de computar eficientemente la factorización de la Definición 2.5 es la base del criptosistema MST_1. Antes de definir el esquema, necesitamos introducir ciertas aplicaciones que pueden construirse a partir de cualquier signatura logarítmica.

Sea G un grupo de permutaciones finito y $\alpha = [\alpha_1, \ldots, \alpha_s]$ una signatura logarítmica de G, de tipo $r = (r_1, \ldots, r_s)$ y con $\alpha_i = [\alpha_{i0}, \ldots, \alpha_{ir_i-1}]$ $(1 \leq i \leq s)$. Para cualquier número natural m, denotamos por $\mathbb{Z}_m$ al conjunto de enteros $\{0, \ldots, m-1\}$. A partir de α podemos definir las siguientes aplicaciones:

$$\begin{array}{rccc} \lambda: & \mathbb{Z}_{r_1} \times \cdots \times \mathbb{Z}_{r_s} & \longrightarrow & \mathbb{Z}_{|G|} \\ & (m_1, \ldots, m_s) & \longmapsto & \sum_{i=1}^{s} \left(m_i \cdot \prod_{j=1}^{i-1} r_j \right) \end{array}$$

y

$$\begin{array}{rccc} \Theta_\alpha: & \mathbb{Z}_{r_1} \times \cdots \times \mathbb{Z}_{r_s} & \longrightarrow & G \\ & (m_1, \ldots, m_s) & \longmapsto & \alpha_{1m_1} \cdots \alpha_{sm_s}. \end{array}$$

Es inmediato comprobar que ambas aplicaciones son biyectivas, y a su vez nos permiten construir una tercera biyección

$$\begin{array}{rccc} \breve{\alpha}: & \mathbb{Z}_{|G|} & \longrightarrow & G \\ & m & \longmapsto & (\Theta_\alpha \lambda^{-1})(m) = \Theta_\alpha\left(\lambda^{-1}(m)\right), \end{array}$$

que esencialmente representa la factorización de los elementos del grupo inducida por α:

$$\breve{\alpha} : \mathbb{Z}_{|G|} \xrightarrow{\lambda^{-1}} \mathbb{Z}_{r_1} \times \cdots \times \mathbb{Z}_{r_s} \xrightarrow{\Theta_\alpha} G.$$

El punto clave para la seguridad del MST_1 será el hecho de que para cierta signatura logarítmica α, la inversa de $\breve{\alpha}$ no pueda computarse eficientemente.

Definición 2.6 *Sea G un grupo de permutaciones de grado n y α una signatura logarítmica de G. Decimos que α es* mansa *si $\breve{\alpha}^{-1}$ puede computarse en tiempo polinomial (en n), en otro caso decimos que es* salvaje. *Si $\breve{\alpha}^{-1}$ es computable en tiempo $\mathcal{O}(n^2)$, diremos que α es* supermansa.

Notas:

- Estrictamente hablando, las nociones de *mansa* y *salvaje* deberían introducirse para *familias* de grupos y *familias* de signaturas logarítmicas asociadas (en otro caso, no tiene sentido hablar de complejidad polinomial en n). Nos ajustamos sin embargo a las definiciones y notaciones originales de los autores, pues en lo sucesivo resulta evidente cuál es el significado asimptótico de los resultados (i.e., cuales son las familias de grupos y signaturas logarítmicas involucradas).

- Es habitual restringirse en este contexto a signaturas logarítmicas formadas exclusivamente por elementos del grupo G que interesa factorizar. Como se razona en [31], tal restricción no supone pérdida alguna de generalidad. Aunque nosotros no hacemos explícitamente esta restricción, cabe resaltar que las signaturas logarítmicas que se usan en el MST_1 son de este tipo.

Para estudiar la factorización inducida por una signatura logarítmica α la aplicación asociada relevante es $\breve{\alpha}$, por ello se introduce la siguiente definición.

Definición 2.7 *Sea G un grupo de permutaciones de grado n y α, β, dos signaturas logarítmicas de G. Diremos que α y β son* equivalentes *si $\breve{\alpha} = \breve{\beta}$.*

Veamos ya algún ejemplo concreto de signatura logarítmica. Una construcción estándar es la siguiente: sea G un grupo de permutaciones de grado n y considérese una cadena de subgrupos

$$G = G_0 > G_1 > \ldots > G_s = \{1\}.$$

Sea ahora $\alpha = [\alpha_i \mid i = 1, \ldots, s]$, una secuencia formada por subsecuencias $\alpha_i = [\alpha_{ij} \mid j = 0, \ldots, r_i - 1]$ de modo tal que para $i = 1, \ldots, s$, los elementos de α_i constituyen un sistema completo de representantes a izquierda de G_{i-1} módulo G_i. En estas condiciones, es fácil ver que α es una signatura

logarítmica mansa de G (ver, por ejemplo [74]). Las signaturas logarítmicas construidas de este modo reciben el nombre de *l-transversales exactas*, y aquellas construidas de modo análogo, pero tomando representantes de clase a derecha, se llaman *r-transversales exactas*. De manera similar puede construirse una signatura logarítmica construyendo cada bloque como un conjunto completo de representantes de clase a izquierda o a derecha (indistintamente). Tales signaturas logarítmicas se dicen *transversales exactas mixtas*. Si una signatura logarítmica se construye según alguno de los métodos anteriores, se dice que es *transversal exacta*; el conjunto de todas las transversales exactas de un grupo G se denota $\mathcal{E}(G)$.

Otra forma de construir una signatura logarítmica es modificar los bloques de otra conocida adecuadamente. Por ejemplo, dada una signatura logarítmica de un grupo de permutaciones G de grado n, α, podemos transformarla en otra usando $s+1$ permutaciones: $t_1, \ldots, t_{s-1} \in S_n$ y $t_0, t_s \in G$. Así, construímos a partir de α la secuencia $\beta = [\beta_1, \ldots, \beta_s]$, con bloques β_i, $1 \leq i \leq s$, definidos por $\beta_i := t_{i-1}^{-1}\alpha_i t_i$. Es claro que β es también una signatura logarítmica de G. Si además $t_0 = t_s = 1$, se dice que β es un *sandwich* de α. La siguiente proposición aparece en [70]:

Proposición 2.8 *Dos signaturas logarítmicas de un grupo G y del mismo tipo r son equivalentes si y sólo si una es un sandwich de la otra.*

Demostración: Sea G un grupo de permutaciones de grado n, α y β dos signaturas logarítmicas de tipo $r = (r_1, \ldots, r_s)$;

$$\alpha = [\alpha_1, \ldots, \alpha_s] \text{ con } \alpha_i = [\alpha_{i0}, \ldots, \alpha_{ir_i-1}]$$

y

$$\beta = [\beta_1, \ldots, \beta_s] \text{ con } \beta_i = [\beta_{i0}, \ldots, \beta_{ir_i-1}].$$

Supongamos que β es un sandwich de α, esto es, que existen $t_1, \ldots, t_{s-1} \in S_n$ tales que, para $i = 1, \ldots, s$, $\beta_i = t_{i-1}^{-1}\alpha_i t_i$. Es obvio que α y β son equivalentes, pues para toda s-tupla $(j_1, \ldots, j_s)$ con $0 \leq j_i \leq r_i - 1$, se tiene

$$\begin{aligned}
\Theta_\alpha(j_1, \ldots, j_s) &= \alpha_{1j_1} \cdots \alpha_{sj_s}, \\
&= \beta_{1j_1} t_1^{-1} t_1 \beta_{2j_2} t_2^{-1} \cdots t_{s-2}\beta_{s-1} t_{s-1}^{-1} t_{s-1}\beta_s, \\
&= \beta_{1j_1} \cdots \beta_{sj_s}, \\
&= \Theta_\beta(j_1, \ldots, j_s).
\end{aligned}$$

Por tanto,

$$\forall\, m \in \mathbb{Z}_{|G|}, \quad \breve{\alpha}(m) = \Theta_\alpha(\lambda^{-1}(m)) = \Theta_\beta(\lambda^{-1}(m)) = \breve{\beta}(m),$$

i.e, α y β son equivalentes.

Recíprocamente, si α y β son equivalentes, para toda s-tupla $(j_1, \ldots, j_s)$, con $0 \leq j_i \leq r_i - 1$, se tiene

$$\Theta_\alpha(j_1, \ldots, j_s) = \Theta_\beta(j_1, \ldots, j_s) \text{ i.e.,} \quad \alpha_{1j_1} \cdots \alpha_{sj_s} = \beta_{1j_1} \cdots \beta_{sj_s}.$$

En particular,

$$\alpha_{10}\alpha_{20} \cdots \alpha_{s-10}\alpha_{sj} = \beta_{10}\beta_{20} \cdots \beta_{s-10}\beta_{sj},$$

para todo $0 \leq j \leq r_s - 1$.
Sea $t_{s-1} = (\alpha_{10}\alpha_{20} \cdots \alpha_{s-10})^{-1}\beta_{10}\beta_{20} \cdots \beta_{s-10}$ claramente $\alpha_{sj} = t_{s-1}\beta_{sj}$. Así, es claro que $\alpha_s = t_{s-1}\beta_s$.
Análogamente, es fácil ver que $\beta_{s-1} = t_{s-2}^{-1}\alpha_{s-1}t_{s-1}$, siendo

$$t_{s-2} = (\alpha_{10}\alpha_{20} \cdots \alpha_{s-20})^{-1}\beta_{10}\beta_{20} \cdots \beta_{s-20}.$$

Reiterando este proceso, se demuestra que β es un sandwich de α, concluyéndose así la demostración. □

Las anteriores definiciones nos permiten dar una pequeña clasificación en el conjunto $\Lambda(G)$:

Definición 2.9 *Sea G un grupo de permutaciones de grado n. Una signatura logarítmica de G se dice*

- transversal, *si es sandwich de una signatura logarítmica transversal exacta,*
- no transversal, *si no es transversal,*
- totalmente no transversal, *si ninguno de sus bloques es un cogrupo de un subgrupo H no trivial de G,*
- totalmente aperiódica, *si cada bloque es un conjunto aperiódico de G, i. e., no es unión de cogrupos de un subgrupo no trivial de G,*
- permutablemente transversal, *si permutando sus bloques puede construirse una signatura logarítmica transversal.*

Los conjuntos de signaturas logarítmicas transversales, no transversales, totalmente no transversales, totalmente aperiódicas y permutablemente transversales de G se denotan, respectivamente, $\mathcal{T}(G), \mathcal{NT}(G), \mathcal{TNT}(G), \mathcal{TA}(G)$

y $\mathcal{PT}(G)$. Si no hay ambigüedad posible, simplemente escribiremos $\mathcal{T}$, $\mathcal{NT}$, $\mathcal{TNT}$, $\mathcal{TA}$ y $\mathcal{PT}$.

Las signaturas logarítmicas de un grupo pueden usarse para generar permutaciones en el conjunto de enteros $\{1, \ldots, |G|\}$, pues fijada cualquier $\eta \in \Lambda$, podemos considerar la permutación $\hat{\alpha} = \breve{\eta}^{-1}\breve{\alpha} \in S_{|G|}$, para toda $\alpha \in \Lambda$. De hecho, según el Teorema 6.5 de [74] dada cualquier $\eta \in \mathcal{T}$, el conjunto

$$\mathcal{T}_\eta = \{\hat{\alpha} = \breve{\eta}^{-1}\breve{\alpha} : \alpha \in \mathcal{T}\}$$

casi siempre genera el grupo simétrico $S_{|G|}$. La importancia de este resultado para el MST_1 es enome, pues justifica que exista un conjunto lo bastante grande de claves privadas posibles.

Con las definiciones anteriores, ya estamos en condiciones de dar una descripción del esquema.

Criptosistema MST1:

Sea G un grupo de permutaciones de grado n, $\eta \in \Lambda$ una signatura logarítmica de referencia fijada (supermansa).

- Clave pública: G, η y un par de signaturas logarítmicas (α, β) con α salvaje y β mansa.
- Clave privada: $\{\theta_1, \ldots, \theta_k\} \subseteq \mathcal{T}$ tales que $\hat{\beta}^{-1}\hat{\alpha} = \hat{\theta}_1 \cdots \hat{\theta}_k$.

  - *B quiere enviar a A* $m \in \mathbb{Z}_{|G|}$, *para ello le envía el mensaje cifrado* $\mathbf{c} = \hat{\beta}^{-1}\hat{\alpha}(m)$.
  - *A recupera el mensaje m usando la clave privada, por ser*

$$m = \hat{\alpha}^{-1}\hat{\beta}(\mathbf{c}) = \hat{\theta}_k^{-1} \cdots \hat{\theta}_1^{-1}(\mathbf{c}).$$

La seguridad de este esquema se fundamenta en la dificultad de resolver los siguientes problemas:

- dada una signatura logarítmica salvaje β, computar $\hat{\beta}^{-1}$,
- dada una signatura logarítmica salvaje β, encontrar un conjunto de signaturas logarítmicas transversales $\{\varrho_1, \ldots, \varrho_k\}$ tales que $\beta = \varrho_1 \cdots \varrho_k$.

Nótese que si el segundo problema se resolviese, se dispondría también de una solución para el primero. Como ocurre a menudo en criptografía, el esquema MST_1 se construye suponiendo que para cierta elección de claves (en concreto, de α), hay evidencias suficientes de que los anteriores problemas son *muy* difíciles, aunque no se disponga de una demostración formal que garantice su dureza. Haremos un análisis profundo de la seguridad del MST_1, cuyos resultados indican que el punto más delicado del diseño es elegir signaturas logarítmicas que puedan considerarse salvajes.

2.3.2. Análisis de seguridad

El primer punto que resulta interesante investigar es el tamaño de claves que puede esperarse en una implementación del MST_1. A raíz de la definición de signatura logarítmica parece claro que tanto la clave pública como la clave privada del esquema están formadas por grandes cantidades de información. Veremos cómo puede afectar este hecho a una implementación concreta del esquema.

Longitud de las claves del MST_1

Supongamos fijado un grupo de permutaciones G, ¿qué podemos decir del tamaño de la clave pública $(\alpha, \beta) \in \Lambda \times \Lambda$? Comenzamos por definir la noción de longitud de una signatura logarítmica:

Definición 2.10 *Sea G un grupo de permutaciones de grado n, $\alpha = [\alpha_1, \ldots, \alpha_s]$ cualquier signatura logarítmica de G. Llamamos* l*ongitud de α al entero $\ell(\alpha) = \sum_{i=1}^{s} r_i$, donde $|\alpha_i| = r_i$, $1 \leq i \leq s$.*

Nota: Sea G un grupo de permutaciones de grado n y orden $|G| = \prod_{j=1}^{t} p_j^{a_j}$, con $p_1, \ldots, p_t$ números primos distintos. Para toda $\alpha \in \Lambda$ se tiene

$$\ell(\alpha) \geq \sum_{j=1}^{t} a_j p_j.$$

En efecto, por ser $|G| = \prod_{j=1}^{t} p_j^{a_j}$, se tiene, para $1 \leq i \leq s, \quad r_i = \prod_{j=1}^{t} p_j^{a_{ij}}$ con $\sum_{i=1}^{s} a_{ij} = a_j$. Así, $r_i \geq \sum_{j=1}^{t} a_{ij} p_j, \quad 1 \leq i \leq s$, de donde se sigue la acotación deseada.

En el Capítulo 4 demostramos que dicha cota se alcanza en diversas familias de grupos. Para algunos grupos damos demostraciones constructivas que permiten obtener signaturas logarítmicas de *longitud mínima*. En pro de la eficiencia del esquema, es fundamental que las signaturas logarítmicas involucradas en el MST_1 sean, además de mansas o salvajes según corresponda, de longitud mínima. Si utilizamos al implementar el MST_1 (como sugieren los autores) grupos de orden 48!, el número de elementos que componen la clave pública es $\ell(\alpha) + \ell(\beta) \geq 2 \cdot 679$. Representando cada elemento de una signatura logarítmica con $\log_2(48!)$ bits, tendríamos una clave pública de más de 2^{18} bits.— un tamaño mucho mayor que los que actualmente se manejan en clave pública (alrededor de 2^{12} bits).

Aun olvidando este problema que únicamente atañe a la eficiencia del esquema la elección incorrecta de claves puede acarrear serios problemas de seguridad. En concreto, las formas de elegir signaturas logarítmicas propuestas en [76] no garantizan, como veremos, niveles aceptables de seguridad criptográfica.

Acerca de la elección de claves seguras

A la vista de la definición de signatura logarítmica salvaje no podemos esperar dar con una forma algorítmica de comprobar cuándo una signatura logarítmica dada es salvaje. Se debe sin embargo exigir que algunos resultados teóricos respalden el hecho de que una signatura logarítmica sea considerada salvaje. En [76] los autores parten de la hipótesis de trabajo de que las signaturas logarítmicas totalmente no transversales son salvajes. Sin embargo, demostramos que utilizar signaturas logarítmicas de $\mathcal{TNT}$ no garantiza en modo alguno que la implementación MST_1 resultante sea segura.

Signaturas logarítmicas mansas en $\mathcal{TNT}$. Veamos algunos argumentos que demuestran que la elección de claves sugerida en [76] no es acertada. En concreto, demostraremos que las signaturas logarítmicas totalmente no transversales pueden ser mansas. Nuestra argumentación utiliza el hecho de que la diferencia entre una signatura logarítmica totalmente no transversal y otra transversal puede ser extremadamente local.

La idea que utilizamos es la siguiente: supongamos que $\alpha = [\alpha_1, \ldots, \alpha_s]$ es una signatura logarítmica transversal exacta asociada a un grupo finito de

permutaciones G con $\alpha_i \neq G$ para todo $i \in \{1, \ldots, s\}$. Sabemos entonces que existe un índice $i \in \{1, \ldots, s\}$, tal que el bloque α_i es un subgrupo no trivial de G. Supongamos que sólo el bloque α_i hace que $\alpha \in \mathcal{E}(G)$ no sea totalmente no transversal (es decir, suponemos que ninguno de los demás bloques es un cogrupo de un subgrupo propio de G).
Sea ahora $\gamma = [\gamma_1, \ldots, \gamma_t] \in \mathcal{TNT}(\alpha_i)$ donde $1 < |\gamma_j| < |\alpha_i|$ $(1 \leq j \leq t)$. Entonces

$$\tilde{\alpha} := [\alpha_1, \ldots, \alpha_{i-1}, \gamma_1, \ldots, \gamma_t, \alpha_{i+1}, \ldots, \alpha_s]$$

está en $\mathcal{TNT}(G)$, pero obviamente, si $|\alpha_i|$ es pequeño, computar la factorización de cualquier $g \in G$ con respecto a $\tilde{\alpha}$ es fácil:

1. Tras *fusionar* los bloques $\gamma_1, \ldots, \gamma_t$ en un único bloque α_i, determinamos la factorización de g con respecto a α.

2. Sustituimos cada α_{ij_i} del producto anterior por su factorización con respecto a γ. Si $|\alpha_i|$ es *suficientemente pequeño*, factorizar con respecto a γ puede hacerse por búsqueda exhaustiva.

Con esta idea, se demuestra el siguiente resultado:

Proposición 2.11 *Para todo $n > 5$, pueden encontrarse signaturas logarítmicas de longitud mínima para el grupo simétrico S_n y alternado A_n que sean a la vez totalmente no transversales y mansas.*

Demostración: Sea $\theta := [\theta_1, \theta_2, \theta_3]$ la secuencia formada por los bloques:

$$\begin{aligned}
\theta_1 &:= [(2,3,4), (1,2,4,5,3), (1,2,3,5,4), (1,5)(2,4), (2,5,3)], \\
\theta_2 &:= [\mathrm{id}, (1,2)(4,5), (1,3,2,4,5), (1,3)(2,5)], \\
\theta_3 &:= [\mathrm{id}, (2,3)(4,5), (1,4,2,5,3)].
\end{aligned}$$

Puede verificarse que θ es una signatura logarítmica de longitud mínima perteneciente a $\mathcal{TNT}(A_5)$. Además, θ es mansa. [1] Procedemos por inducción en n, esto es, dada una signatura logarítmica de longitud mínima, mansa y perteneciente a $\mathcal{TNT}(A_{n-1})$, $\theta = [\theta_1, \ldots, \theta_s]$, construimos una signatura logarítmica de longitud mínima y mansa perteneciente a $\mathcal{TNT}(A_n)$, para $n > 5$.

[1] es obvio que podemos calcular la factorización asociada a θ con un algoritmo de búsqueda exhaustiva, de complejidad fija

Supongamos que n se factoriza como $n = p_1 \cdots p_k$, con p_i, $i = 1, \ldots, k$, números primos no necesariamente distintos. Considerar la secuencia de bloques $[\beta_1, \ldots, \beta_k]$ definida de la siguiente forma:

$$\beta_1 \;:=\; \big[e, \beta_{1,1}, \ldots, \beta_{1,p_1-2}, (n-2, p_1-1, n)\big]$$

donde $\beta_{1,j} := (j+1, j, n) \quad (1 \leq j \leq p_1 - 2)$, y tomar, para $i = 2, \ldots, k$

$$\beta_i := \big[e, \beta_{i,1}, \ldots, \beta_{i,p_i-2}, \beta_{i,p_i-1}\big],$$

donde para $j = 1, \ldots p_i - 2$

$$\beta_{i,j} := \big((j+1)p_1 \cdots p_{i-1}, jp_1 \cdots p_{i-1}, n\big) \circ \prod_{1 \leq l < p_1 \cdots p_{i-1}} \big((j+1)p_1 \cdots p_{i-1}, jp_1 \cdots p_{i-1} + l, l\big)$$

y

$$\beta_{i,p_i-1} := \big(n-1, (p_i-1)p_1 \cdots p_{i-1}, n\big) \circ \prod_{1 \leq l < p_1 \cdots p_{i-1}} \big(p_1 \cdots p_i, (p_i-1)p_1 \cdots p_{i-1} + l, l\big).$$

Es fácil comprobar que $\alpha = [\theta_1, \ldots, \theta_s, \beta_1, \ldots, \beta_k]$ es una signatura logarítmica de A_n. Recordar para ello que que $A_n = \bigcup_{i=1}^{n} A_{n-1} g_i$, donde para $i = 1, \ldots, n$, g_i es cualquier permutación par tal que $g_i(i) = n$. Observar ahora que cada producto de la forma $\beta_{1j_1} \cdots \beta_{kj_k}$ con $\beta_{ij_i} \in \beta_i$ define una permutación par distinta que asigna a cada $i \in \{1, \ldots, n\}$ la letra n.
Además α tiene por construcción longitud mínima. Para verificar que es también totalmente no transversal, basta comprobar que ninguno de los bloques $\beta_1, \ldots, \beta_k$ es un cogrupo asociado a un subgrupo propio de A_n. Esa comprobación es inmediata, pues todos los bloques contienen a la identidad y ninguno es un grupo:

- β_1: como $|\beta_1| = p_1$ es primo y salvo la identidad, todos los elementos de β_1 son tres ciclos, si β_1 fuese un grupo se tendría $|\beta_1| = 3$ y $\beta_{1,1} = (2, 1, n)$ lo que contradice $n - 2 > 3$.

- β_i $(i = 2, \ldots, k)$: si $p_i \neq 2$ ó $i < k$, la permutación $\beta_{i,1}$ asigna $p_1 \cdots p_{i-1}$ a n, y ninguna permutación de β_i asigna n a $p_1 \cdots p_{i-1}$.

  Si $p_i = 2$ y además $i = k$, necesariamente $p_1 \cdots p_{i-1} \geq 3$ y $\beta_{k,1}(2) = p_1 \cdots p_{i-1} + 1$. Sin embargo, $\beta_{k,1}(p_1 \cdots p_{i-1} + 1) = 1$, i. e., $\beta_{k,1}^2 \neq e$, de donde $\beta_k = [e, \beta_{k,1}]$ no puede ser un grupo.

Veamos ahora que es mansa. Es claro que cada bloque β_i es un sistema completo de representantes de A_i módulo A_{i-1}, luego $\beta := [A_5, \beta_1, \beta_2 \ldots, \beta_k]$ es una signatura logarítmica transversal exacta de A_n. Por tanto, β es mansa y como α se construye sustituyendo los $5!/2 = 60$ elementos del primer bloque de β por la signatura logarítmica θ de A_5, demostramos que α también es mansa.

Para construir una signatura logarítmica de longitud mínima mansa y en $\mathcal{TNT}(S_n)$, basta añadir el bloque $[\mathrm{id}, (1,2)^n(1,\ldots,n)]$ a una signatura logarítmica de longitud mínima y mansa perteneciente a $\mathcal{TNT}(A_n)$. □

La observación anterior no resulta un inconveniente importante para el futuro del MST_1, ya que existen modos de identificar cuándo una signatura logarítmica α puede modificarse fácilmente para construir una transversal que ayude a computar la factorización inducida por α. A continuación exponemos uno de los métodos posibles. De él pueden deducirse pautas para identificar claves inseguras, aunque no podemos asegurar que éstas basten para seleccionar claves con garantías totales de seguridad.

Sea G un grupo de permutaciones de grado n, $\alpha = [\alpha_1, \ldots, \alpha_s]$ una signatura logarítmica de G de tipo $r = (r_1, \ldots, r_s)$. Para $1 \leq i \leq j \leq s$ definimos los valores $M^{\alpha}_{i,j} \in \{0, 1\}$ del modo siguiente:

$$M^{\alpha}_{i,j} := \begin{cases} 1, & \text{si } \alpha_i \cup \cdots \cup \alpha_j \text{ genera un subgrupo de orden } r_i \cdots r_j \\ 0, & \text{en otro caso.} \end{cases}$$

Para calcular los valores $M^{\alpha}_{i,j}$ puede utilizarse el algoritmo descrito en la Tesis de J. Cusack (ver Sección 3.5 de [31]). Este algoritmo utiliza simplemente el hecho de que para valores pequeños de la suma $r_i + \cdots + r_j$, el orden del subgrupo generado por $\alpha_i \cup \cdots \cup \alpha_j$ puede calcularse de manera eficiente. En particular, si $\ell(\alpha)$ no es muy grande, los elementos $M^{\alpha}_{i,j}$ $(1 \leq i \leq j \leq s)$ se calculan fácilmente. La siguiente proposición aparece también en [31]:

Proposición 2.12 *Sea $\alpha = [\alpha_1, \ldots, \alpha_s]$ una signatura logarítmica de un grupo de permutaciones G de grado n. Entonces $\alpha \in \mathcal{E}(G)$ si y sólo si puede encontrarse una secuencia de pares*

$$(i_1, j_1), \ldots, (i_s, j_s) \in \{1, \ldots, s\} \times \{1, \ldots, s\}$$

tales que $M^{\alpha}_{i_k, j_k} = 1$ para $1 \leq k \leq s$. Además $i_1 = j_1$, y para todo $1 < k \leq s$ se tiene $(i_k, j_k) \in \{(i_{k-1} - 1, j_{k-1}), (i_{k-1}, j_{k-1} + 1)\}$.

Demostración: Supongamos que $\alpha \in \mathcal{E}(G)$, y sea

$$G = G_0 > G_1 > \ldots > G_s = \{\mathrm{id}\}$$

la cadena de subgrupos a partir de la cual se construye α. Recordar entonces que los bloques de α se han ido formando tomando clases completas de representantes a derecha o izquierda de distintos subgrupos, comenzando el proceso por el subgrupo G_{s-1} de G. Así, alguno de los bloques de α está compuesto por los elementos del grupo G_{s-1}. Sea i_1 el índice de dicho bloque. Es claro entonces que que $M^{\alpha}_{i_1,i_1} = 1$. Si $s = 1$, ya hemos terminado. Si no, ha de darse uno de los casos siguientes:

- $i_1 < s$ y los elementos del bloque α_{i_1+1} forman un conjunto completo de representantes a derecha de α_{i_1} en el subgrupo generado por $\alpha_{i_1} \cup \alpha_{i_1+1}$ (que será G_{s-2});
- $1 < i_1$ y α_{i_1-1} es un conjunto completo de representantes a izquierda de α_{i_1} en el subgrupo generado por $\alpha_{i_1-1} \cup \alpha_{i_1}$ (que será G_{s-2}).

Haciendo ahora $(i_2, j_2) := (i_1, i_1+1)$ (resp. $(i_2, j_2) := (i_1-1, i_1)$), se tiene que $M^{a}_{i_2,j_2} = 1$, como queríamos demostrar. Si $s = 2$ hemos terminado. Si $s > 2$ repetimos el argumento anterior con el grupo G_{s-2}. Notar que los elementos de ese grupo son exactamente los de $\alpha_{i_2} \cdot \alpha_{j_2}$. Ahora o bien los elementos de α_{j_2+1} forman un conjunto completo de representantes a derecha o los de α_{i_2-1} son un conjunto completo de representantes a izquierda de G_{s-3} módulo G_{s-2}. Según el caso, definiremos el par (i_3, j_3). Razonando de esta forma, se construye la secuencia deseada $(i_1, j_1), \ldots, (i_s, j_s)$.

Para ver el recíproco, supongamos que existen pares $(i_1, j_1), \ldots, (i_s, j_s)$ según el enunciado de la proposición. Por la definición de M^{α}, sabemos que los elementos de α_{i_1} forman un grupo. Ahora, si $(i_2, j_2) = (i_1-1, i_1)$, las clases $g\alpha_{i_1}$ $(g \in \alpha_{i_2})$ deben ser diferentes dos a dos, pues por ser α una signatura logarítmica se tiene:

$$\alpha_{i_1-1} \cdot \alpha_{i_1} = \biguplus_{g \in \alpha_{i_1-1}} g\alpha_{i_1}.$$

Es más, al ser $M^{\alpha}_{i_1-1,i_1} = 1$ sabemos que los elementos de $\alpha_{i_1-1} \cdot \alpha_{i_1}$ también forman un grupo. Análogamente al caso anterior si $(i_2, j_2) = (i_1, i_1+1)$, sabemos que los elementos de α_{i_1+1} forman un conjunto completo de representantes a derecha de $\alpha_{i_1} \cdot \alpha_{i_1+1}$ módulo α_{i_1}. Repitiendo este proceso se va obteniendo una cadena de subgrupos de G a partir de la cual puede construirse α. Así, α es una signatura logarítmica transversal exacta, como queríamos demostrar. □

A raíz de este resultado, es claro que si α es una signatura logarítmica transversal exacta de G, al escribir los valores $M^{\alpha}_{i,j}$ en una tabla aparecerá una *escalera de unos* a partir del elemento $M^{\alpha}_{i_1,j_1}$ y hasta la esquina superior derecha de la tabla: $M^{\alpha}_{i_s,j_s} = M^{\alpha}_{s,1}$. Veamos algunos ejemplos:

Ejemplo 2.1 *Construimos una signatura logartímica para S_5 de la siguiente forma: elegimos los elementos del primer bloque, α_1, de manera que formen un conjunto completo de representantes a izquierda de S_5 módulo S_4 :*

$$\alpha_1 := [e, (1,5), (2,5), (3,5), (4,5)].$$

Los elementos del cuarto bloque, α_4, se eligen de modo que formen un conjunto completo de representantes a derecha de S_4 módulo A_4, $\alpha_4 := [e, (1,2,3,4)]$, y los de α_2 de modo que formen un conjunto completo de representantes a izquierda de A_4 módulo A_3,

$$\alpha_2 := [e, (1,2)(3,4), (1,3,4), (2,3,4)].$$

Así, construimos el bloque α_3 de α tomando todas las permutaciones del grupo A_3 (en cualquier orden). Está claro que $\alpha = [\alpha_1, \alpha_2, \alpha_3, \alpha_4]$ es una signatura logarítmica transversal exacta de S_5, construida a partir de la cadena de subgrupos

$$S_5 > S_4 > A_4 > A_3 > \{e\}.$$

La tabla siguiente contiene los valores $M^{\alpha}_{i,j}$, y en ella se ve muy bien la escalera de unos *que mencionábamos antes:*

$i \backslash j$	1	2	3	4
1	0	0	0	**1**
2		0	**1**	**1**
3			**1**	0
4				0

Por otra parte, si observamos que en la tabla correspondiente a una cierta signatura logarítmica α aparece una *escalera de unos* desde un elemento diagonal hasta la esquina superior derecha, sabemos que α es transversal exacta. Esta observación puede ayudarnos a ver cuándo una signatura logarítmica totalmente no transversal puede transformarse en una signatura logarítmica transversal exacta:

Ejemplo 2.2 *La signatura logarítmica* $\alpha = [\alpha_1, \ldots, \alpha_7]$ *pertenece a* $\mathcal{TNT}(S_7)$*:*

$$\begin{aligned}
\alpha_1 &:= [e, (1,7), (2,7), (3,7), (4,7), (5,7), (6,7)],\\
\alpha_2 &:= [e, (1,3)(2,6), (1,5)(4,6)],\\
\alpha_3 &:= [e, (1,2,6)],\\
\alpha_4 &:= [e, (1,5), (2,5), (3,5), (4,5)],\\
\alpha_5 &:= [e, (1,3,4,2)],\\
\alpha_6 &:= [e, (1,2), (1,2,4), (1,4)],\\
\alpha_7 &:= [e, (1,3), (2,3)].
\end{aligned}$$

La tabla de valores $M^{\alpha}_{i,j}$ *asociada es la siguiente:*

$_i\diagdown^j$	1	2	3	4	5	6	7
1	0	0	0	0	0	0	**1**
2		0	0	0	0	0	**1**
3			0	0	0	0	0
4				0	0	0	**1**
5					0	0	**1**
6						0	0
7							0

Veamos ahora cómo transformar α *en una signatura logarítmica que sea transversal exacta. Como* $M^{\alpha}_{2,7} = 1$*, podemos* eliminar *la entrada* $M^{\alpha}_{3,7} = 0$ *sustituyendo los bloques* α_2 *y* α_3 *de* α *por un bloque* $\tilde{\alpha}_2 := [\alpha_{2i}\alpha_{3j} : 0 \leq i \leq 2, 0 \leq j \leq 1]$ *de longitud* $|\alpha_2| \cdot |\alpha_3| = 6$*. De manera análoga, para que en la tabla no aparezcan los valores* $M^{\alpha}_{6,7} = 0$ *y* $M^{\alpha}_{7,7} = 0$*, reemplazamos los bloques* α_5, α_6*, y* α_7 *por un único bloque*

$$\tilde{\alpha}_4 := [\alpha_{5i}\alpha_{6j}\alpha_{7k} : 0 \leq i \leq 1, 0 \leq j \leq 3, 0 \leq k \leq 2]$$

de 24 *elementos. Permutando los elementos de* $\tilde{\alpha}_2$ *y* $\tilde{\alpha}_4$ *de modo adecuado, la signatura logarítmica resultante* $\tilde{\alpha} := [\tilde{\alpha}_1, \ldots, \tilde{\alpha}_4]$ *(con* $\tilde{\alpha}_1 := \alpha_1$*,* $\tilde{\alpha}_3 := \alpha_4$*) es equivalente a* α *y transversal exacta. Su tabla de valores* $M^{\tilde{\alpha}}_{i,j}$ *es:*

$_i\diagdown^j$	1	2	3	4
1	0	0	0	**1**
2		0	0	**1**
3			0	**1**
4				**1**

Así, si queremos que un adversario no pueda transformar una signatura logarítmica totalmente no transversal en una transversal exacta equivalente, hemos de evitar utilizar signaturas logarítmicas para las que procesos similares al del ejemplo anterior sean posibles. Representar la tabla correspondiente puede servir, por ejemplo, para identificar y rechazar signaturas logarítmicas que sean a la vez totalmente no transversales y mansas.

Signaturas logarítmicas permutablemente transversales y mansas. Otro procedimiento que analizamos es la consideración de sandwiches de signaturas logarítmicas permutablemente transversales como signaturas logarítmicas salvajes. Esta sugerencia (aparecida en la tesis de J. Cusack [31]) propone construir una signatura logarítmica salvaje η de un grupo G abeliano de la siguiente forma:

1. A partir de una cadena de subgrupos

$$G = G_0 > G_1 > \ldots > G_s = \{e\}, \tag{2.2}$$

construir una signatura logarítmica transversal exacta

$$\alpha = [\alpha_1, \ldots, \alpha_s] \in \mathcal{E}(G).$$

2. Transformar α en una signatura logarítmica transversal $\beta = [\beta_1, \ldots, \beta_s]$ haciendo un sandwith de α con un cierto $t := (t_0, \ldots, t_s) \in G^s$ (donde $t_0 = t_s = e$).

3. Transformar β en otra signatura logarítmica $\gamma \in \Lambda(G)$ permutando sus bloques según cierta $\sigma \in S_s$ i.e.,$\gamma := [\beta_{\sigma^{-1}(1)}, \ldots, \beta_{\sigma^{-1}(s)}]$.

4. Hacer un sandwich de γ con un cierto $u := (u_0, \ldots, u_s) \in G^s$ ($u_0 = u_s = e$). Obtener así una nueva signatura logarítmica de G,

$$\eta = [\eta_1, \ldots, \eta_s].$$

Como G es abeliano, η puede escribirse como

$$\eta = [g_1\alpha_{\sigma^{-1}(1)}, \ldots, g_s\alpha_{\sigma^{-1}(s)}],$$

para ciertos $g_i \in G, (1 \leq i \leq s)$ que verifican $g_1 \cdots g_s = 1$.
La propuesta concreta de [31] es utilizar η como clave pública y α como privada, siendo por tanto σ y $(g_1, \ldots, g_1)$ información secreta.

Sin embargo, sabemos que por construcción alguno de los bloques α_{i_1} de α está compuesto exactamente por los elementos del sugbrupo G_{s-1} de G. Consecuentemente η contiene el bloque

$$\eta_{\sigma(i_1)} = g_{\sigma(i_1)}\alpha_{i_1}$$

y podemos intentar recuperar el bloque α_{i_1} identificando un índice $1 \leq j_1 \leq s$ tal que $\eta_{j_1} \cdot \eta_{j_1 0}^{-1}$ sea un grupo (donde, como siempre, $\eta_{j_1 0}$ denota el primer elemento de η_{j_1}). Una vez hayamos detectado el índice $j_1 = \sigma(i_1)$, nuestro objetivo es identificar el bloque de η que se corresponde con la transversal α_{i_2} de G_{s-2} módulo G_{s-1} usada en la construcción de α. Perseguimos por tanto el bloque

$$\eta_{\sigma(i_2)} = g_{\sigma(i_2)}\alpha_{i_2}.$$

Sabemos que al multiplicar $\eta_{\sigma(i_2)}$ por $\eta_{\sigma(i_2)0}^{-1}$ obtenemos una transversal de G_{s-2} módulo G_{s-1} (en concreto, $h\alpha_{i_2}$ con un cierto $h \in G_{s-2}$). Por tanto, buscamos un índice $1 \leq j_2 \leq s$ tal que $\eta_{j_2} \cdot \eta_{j_2 0}^{-1}$ sea un sistema completo de representantes de $G_{s-1} = \eta_{j_1} \cdot \eta_{j_1 0}^{-1}$ en el grupo que generan G_{s-1} y $\eta_{j_2} \cdot \eta_{j_2 0}^{-1}$. Equivalentemente, podemos comprobar si el grupo generado por $\eta_{j_1} \cdot \eta_{j_1 0}^{-1}$ y $\eta_{j_2} \cdot \eta_{j_2 0}^{-1}$ tiene orden $|\eta_{j_1}| \cdot |\eta_{j_2}|$.
Una vez hemos identificado el índice correcto $j_2 = \sigma(i_2)$, conocemos en particular el grupo $G_{s-2} = \eta_{j_1}\eta_{j_1 0}^{-1} \cdot \eta_{j_2}\eta_{j_2 0}^{-1}$, y si $s > 2$ simplemente reiteramos el proceso: para ello buscamos el bloque de η asociado a la transversal α_{i_3} de G_{s-3} módulo G_{s-2} usada en la construcción de α. Así, comprobamos para qué índice $1 \leq j_3 \leq s$ el grupo generado por $\eta_{j_1} \cdot \eta_{j_1 0}^{-1}$, $\eta_{j_2}^{-1} \cdot \eta_{j_2 0}^{-1}$, y $\eta_{j_3} \cdot \eta_{j_3 0}^{-1}$ es de orden $|\eta_{j_1}| \cdot |\eta_{j_2}| \cdot |\eta_{j_3}|$. Continuando de esta forma, podemos recuperar la cadena de subgrupos (2.2) completa. Es más, cada transversal $\alpha'_{i_k} := \eta_{\sigma(i_k)} \cdot \eta_{\sigma(i_k)0}^{-1}$ $(1 \leq k \leq s)$ de G_{s-k+1} en G_{s-k} que hemos recuperado coincide con α_{i_k} o es múltiplo suyo por un elemento de G_{s-k}.
Para factorizar un $g \in G$ dado con respecto η, empezamos por factorizar $g \cdot \prod_{k=1}^{s} \eta_{k0}^{-1}$ con respecto a la signatura logarítmica transversal exacta $\alpha' := [\alpha'_{i_1}, \ldots, \alpha'_{i_s}]$ para obtener una expresión de la forma:

$$g \cdot \prod_{k=1}^{s} \eta_{k0}^{-1} = \alpha'_{i_1 l_1} \cdots \alpha'_{i_s l_s} \quad (\text{with } \alpha'_{i_k l_k} \in \alpha'_{i_k}).$$

Luego la factorización con respecto a η se obtiene multiplicando cada $\alpha'_{i_k l_k}$ por el correspondiente $\eta_{\sigma(i_k)0}$ $(1 \leq k \leq s)$:

$$g = \prod_{k=1}^{s} \underbrace{\left(\eta_{j_k 0} \cdot \alpha'_{i_k l_k}\right)}_{\in \eta_{j_k}} \quad (\text{donde } j_k = \sigma(i_k)).$$

Por supuesto, no existe ninguna razón por la que este método de ataque deba funcionar (podemos, por ejemplo elegir j_1 incorrectamente si varios bloques de α son grupos y equivocarnos ya en el primer paso). También, si en algún momento no podemos seguir con el proceso, podemos volver atrás un paso y verificar si nos hemos equivocado en la correspondiente elección. Hemos de dejar claro que con este método de *vuelta atrás* la complejidad del algoritmo de ataque es exponencial, si bien es probable que en la mayoría de los casos sea muy efectivo.

Además de que, como hemos visto, resulte muy difícil determinar cuándo ciertas signaturas logarítmicas pueden considerarse salvajes, la propia Definición 2.6 de *salvaje* es más que cuestionable. Aunque no exista un algoritmo polinomial para computar la factorización asociada a una signatura logarítmica, es posible que la mayoría de los elementos del grupo puedan factorizarse fácilmente, con lo que el esquema construido con dicha signatura logarítmica sería inseguro. Veremos algunos ejemplos de esta situación.

Ataques por inversiones parciales. Una situación sin duda indeseable es que partes de la signatura logarítmica de G que consideramos salvaje constituyan una signatura logarítmica mansa de un cierto subgrupo H de G, especialmente si el índice de H en G es pequeño. Si por ejemplo $|G : H| < 2^{80}$, el esquema se consideraría criptográficamente inseguro, pues un adversario puede factorizar de manera eficiente *demasiados* elementos de G (todo $g \in H$).

Ejemplo 2.3 *Ilustramos este problema usando la signatura logarítmica* $\alpha = [\alpha_1, \alpha_2] \in \mathcal{TNT}(A_5)$ *que aparece como ejemplo en [74]:*

$$\begin{array}{lccccc}
\alpha_1 = [& e, & (1,5,2), & (1,3)(2,4), & (2,3,4), & (2,5,4), \\
 & (1,3,5,2,4), & (2,3)(4,5), & (1,5,4,3,2), & (1,5,3,4,2), & (3,4,5), \\
 & (2,5,3), & (3,5,4), & (2,5)(3,4), & (2,3,5), & (1,3,2,4,5)], \\
\alpha_2 = [& (1,3)(4,5), & (1,4)(3,5) & e, & (1,3,5)]. &
\end{array}$$

Definiendo $\alpha_1' = [e, (3,4,5), (3,5,4)] \subset \alpha_1$ *y* $\alpha_2' = \alpha_2$, *obtenemos una signatura logarítmica transversal exacta del subgrupo* $\langle (1,3,5), (3,5,4) \rangle$ *de* A_5, *que es isomorfo a* A_4.
Usando $\alpha' := [\alpha_1', \alpha_2']$ *un adversario puede factorizar el 20 % de los elementos de* A_5 *con respecto a* $\alpha \in \mathcal{TNT}(A_5)$.

Una situación parecida se da cuando partes de la signatura logarítmica supuestamente salvaje pueden *sumergirse* en una signatura logarítmica mansa:

Ejemplo 2.4 *La siguiente signatura logarítmica de S_4 es totalmente no transversal: $\alpha = [\alpha_1, \alpha_2, \alpha_3]$, con*

$$\begin{array}{l} \alpha_1 = [\ \ e,\ \ (1,4,3),\ \ \ \ (3,4)], \\ \alpha_2 = [\ \ e,\ \ (1,2,4)], \\ \alpha_3 = [\ \ e,\ \ \ (1,2),\ \ \ (1,3)(2,4),\ \ (1,3,2,4)] \end{array}$$

(esto puede verificarse con un sistema de álgebra computacional, como el MAGMA *[25]).*
Considerar ahora la signatura logarítmica $\alpha^* = [\alpha_1^*, \alpha_2^*, \alpha_3^*] \in \mathcal{E}(S_4)$, donde:

$$\begin{array}{l} \alpha_1^* = [\ \ e,\ \ (1,4,3),\ \ \ (3,4),\ \ \ (2,4,3)], \\ \alpha_2^* = [\ \ e,\ \ (1,2,4),\ \ (1,4,2)], \\ \alpha_3^* = [\ \ e,\ \ \ (1,2)]. \end{array}$$

Es claro que $\alpha_1 \subset \alpha_1^*$ y $\alpha_2 \subset \alpha_2^*$. Es más, los bloques α_3 y α_3^* tienen dos elementos en común. Así, pueden factorizarse $3 \cdot 2 \cdot 2 = 12$ elementos de S_4, (i. e., el 50 % de los elementos del grupo) con respecto a $\alpha \in \mathcal{TNT}(S_4)$ usando el algoritmo de factorización asociado a $\alpha^* \in \mathcal{E}(S_4)$.

Los ataques expuestos en esta sección no demuestran que no existan implementaciones seguras del MST_1. Nuestros resultados prueban que los criterios propuestos hasta la fecha para elegir signaturas logarítmicas que induzcan factorizaciones difíciles no llevan a criptosistemas seguros, lo que no excluye que en un futuro pueda darse un método de generación de claves que garantice la seguridad del esquema. Para ello será necesario dar una definición de signatura logarítmica salvaje que formalice la idea de que la factorización inducida ha de ser difícil de calcular salvo, a lo sumo, para una cantidad despreciable de elementos del grupo. Después, será necesario obtener algoritmos eficientes de construcción de signaturas logarítmicas con estas características. El esquema MST_1 explota con gran elegancia la situación en la que los elementos de un grupo finito son difíciles de factorizar respecto a un conjunto dado; sólo es necesario encontrar los conjuntos adecuados de factorización para conseguir una construcción segura.

Capítulo 3

Esquemas inspirados en la construcción de Diffie-Hellman

3.1. El problema del logartimo discreto y el problema de Diffie-Hellman

Al igual que el problema de factorización de enteros, el llamado *problema del logaritmo discreto* ha jugado un papel fundamental en la construcción de los cimientos de la criptografía de clave pública. Su formulación es sencilla:

Problema del Logaritmo Discreto: sea G un grupo, $g \in G$ un elemento de orden n. Dado $h \in \langle g \rangle$, encontrar $k \in \{0, \ldots, n-1\}$ tal que $h = g^k$.

Para que dicho problema pueda utilizarse en criptografía, suele requerirse que el grupo G involucrado cumpla al menos dos condiciones:

- la ley de grupo ha de ser fácil de implementar,
- no debe conocerse un algoritmo polinomial que resuelva el problema del logaritmo discreto en G.

Algunos grupos que reúnen estas condiciones son:

- el grupo multiplicativo $\mathbb{F}_{p^m}^*$, asociado al cuerpo finito $\mathbb{F}_{p^m}$ de característica prima p, (m cualquier número natural),
- el grupo de puntos de una curva elíptica sobre un cuerpo finito,
- el grupo de unidades $\mathbb{Z}_n^*$, donde n es un entero compuesto,
- el jacobiano de una curva hiperelíptica definida sobre un cuerpo finito.

Entre los ejemplos anteriores, destacamos los dos primeros; de hecho, muchos esquemas criptográficos basados en el logaritmo discreto emplean el grupo multiplicativo $\mathbb{F}_p^*$, con p primo, o bien el grupo asociado a un cuerpo finito de característica dos. Además, la mayor parte de esas herramientas criptográficas no se fundamentan exactamente en el problema del logaritmo discreto, sino en un problema relacionado con él: el llamado problema de Diffie-Hellman.

Problema de Diffie-Hellman (búsqueda): dados un número entero n, un generador g del grupo cíclico multiplicativo de orden n y dos elementos g^a y g^b ($a, b \in \{1, \cdots, n-1\}$) encontrar el elemento g^{ab}.

Nota: En la literatura, a menudo se llama al problema anterior *problema de Diffie-Hellman generalizado*, denominando entonces *problema de Diffie-Hellman* al caso particular planteado sobre el grupo multiplicativo $\mathbb{Z}_p^*$, con p primo. En este problema se basan el esquema de intercambio de claves del mismo nombre y el criptosistema ElGamal [34]; dos herramientas clásicas que han sido analizadas en profundidad tanto teóricamente como desde el punto de vista práctico.

Intercambio de claves de Diffie-Hellman:

Sea p un número primo g un generador del grupo $\mathbb{Z}_p^*$.
Los individuos A y B quieren intercambiar una clave secreta para comunicarse. Para ello actúan según los siguientes pasos:

- *A elige un entero $a \in \{1, \ldots, p-2\}$, y envía a B el elemento g^a.*
- *De modo análogo, B elige un entero $b \in \{1, \ldots, p-2\}$ y envía a A el elemento g^b.*
- *A y B computan la clave común $K = g^{ba} = g^{ab}$.*

A partir de esta idea de generación de claves, se propone (con los mismos parámetros) el criptosistema de clave pública ElGamal:

Criptosistema ElGamal:

- Clave pública: (p, g, g^a).
- Clave privada: $a \in \{1, \ldots, p-2\}$.

Supongamos que B quiere enviar un mensaje $m \in \mathbb{Z}_p^*$ a A :

- Cifrado: *B elige un elemento $b \in \{1, \ldots, p-2\}$, y envía a A el elemento g^b, y el producto mg^{ab}.*
- Descifrado: *A recibe un par $(c, d) \in \mathbb{Z}_p^* \times \mathbb{Z}_p^*$. Usando su clave privada, recupera el mensaje $m = (c^a)^{-1}d$.*

Nota: Obviamente, todos los productos indicados en las descripciones anteriores se consideran en $\mathbb{Z}_p^*$.

Aunque, como decíamos, estos esquemas se diseñaron tomando como base el problema de Diffie-Hellman, su seguridad depende de un problema más sencillo. En [96], se demuestra que la seguridad semántica del criptosistema ElGamal es equivalente a la dificultad del llamado *problema de decisión de Diffie-Hellman*:

Problema Diffie-Hellman (decisión): sea G un grupo no abeliano, Dada una tupla (g, g^a, g^b, c) con $g, c \in G$, y a, $b \in \{1, \ldots, ord(g) - 1\}$, decidir si $c = g^{ab}$.

Por otro lado, es claro que si el problema del logaritmo discreto puede resolverse en G, lo mismo ocurre con el problema de Diffie-Hellman (decisión y búsqueda). El recíproco no es cierto en general. Sin embargo, es habitual aceptar que el problema de Diffie-Hellman es difícil de resolver en aquellos grupos con problema del logaritmo discreto difícil.

En algunos casos puede estudiarse el problema de factorización de enteros a partir de ciertos casos particulares del problema de Diffie-Hellman. Para un estudio detallado de las relaciones entre dichos problemas, ver, por ejemplo, el Capítulo 3 de [78]. Por todo esto, no es de extrañar que las técnicas matemáticas de diseño y criptoanálisis de este tipo de herramientas criptográficas sean esencialmente de teoría de números. Sólo recientemente se han propuesto esquemas de este tipo en el que la estructura del grupo subyacente (y no sólo su orden) juega un papel fundamental en el funcionamiento del esquema criptográfico. Estudiaremos algunos de estos esquemas con cierto detalle.

3.2. Los criptosistemas MOR

En el Crypto 2001 (ver [82]), Paeng y otros propusieron un nuevo criptosistema de clave pública basado en el problema del logaritmo discreto en el grupo de automorfismos internos de un grupo finito no abeliano. Más adelante, los mismos autores generalizaron su propuesta inicial en [83]. Comencemos por describir el esquema propuesto en [82].

3.2.1. El criptosistema MOR básico

Sea G un grupo no abeliano con centro no trivial. Sea g un elemento de G e $Inn(g)$ el automorfismo interno del grupo asociado al elemento g, i.e.

$$\begin{array}{rccc} Inn(g): & G & \longmapsto & G \\ & x & \longmapsto & gxg^{-1}. \end{array}$$

Cada clave privada será un elemento $a \in \{1, \ldots, ord(g)-1\}$, y su clave pública correspondiente vendrá dada por una especificación de $Inn(g)$ e $Inn(g^a)$ dada por sus imágenes en un conjunto de generadores de G.

Criptosistema MOR:

- Clave pública: $Inn(g), Inn(g^a)$.
- Clave privada: $a \in \{1, \ldots, ord(g) - 1\}$.

- *B quiere enviar un mensaje $m \in G$ a A, para ello:*
  1. *elige $b \in \{1, \ldots, ord(g) - 1\}$ al azar y computa $(Inn(g^a))^b$,*
  2. *computa $E := Inn(g^{ab})(m) = (Inn(g^a))^b(m)$,*
  3. *computa $\varphi := Inn(g)^b$,*
  4. *envía el par (E, φ) a A.*
- *A descifra el mensaje recibido computando $m = \varphi^{-a}(E)$.*

Claramente, la seguridad de este esquema se basa en el problema de Diffie-Hellman asociado al grupo $Inn(G)$.

3.2.2. El criptosistema MOR generalizado

Poco después de la propuesta del MOR básico, sus autores introdujeron una generalización del mismo. Su objetivo era obtener nuevos esquemas asociados a su diseño original con distintas propiedades de seguridad y eficiencia. El esquema MOR generalizado involucra a dos grupos G y G', finitos y no necesariamente abelianos que se relacionan por medio de un homomorfismo

$$\phi : G \longmapsto Aut(G').$$

La base del esquema es el problema del logaritmo discreto en $Aut(G')$. Como antes, las claves se construyen a partir de un elemento $g \in G$ y un número natural $a \in \{1, \ldots, ord(g) - 1\}$.

Criptosistema MOR generalizado:

- Clave pública: $\phi(g), \phi(g)^a$.
- Clave privada: $a \in \{1, \ldots, ord(g) - 1\}$.

- *B quiere enviar un mensaje $m \in G'$ a A, realiza los siguientes pasos:*
  1. *elige $b \in \{1 \ldots, ord(g) - 1\}$ y computa $(\phi(g^a))^b$,*
  2. *computa $E := \phi(g)^{ab}(m) = (\phi(g)^a)^b(m) \in G'$,*
  3. *computa $\varphi := \phi(g)^b$,*
  4. *envía el par (E, φ) a A.*
- *A descifra el mensaje recibido computando $m = \varphi^{-a}(E)$.*

Notas:

- El esquema MOR básico es un caso particular de la construcción general: si fijado un grupo G tomamos $G' = \frac{G}{Z(G)}$ y $\phi(g) = Inn(g)$, el criptosistema resultante es claramente el esquema MOR básico.

- También el criptosistema ElGamal es un caso particular del MOR generalizado: tomar $G := \mathbb{Z}_p$ (grupo aditivo) y $G' := G$. En ese caso, $Aut(G') = \mathbb{Z}_p^*$ y $\phi(x)(m) = mg^x$, donde $g := \phi(1)$.

- Otros ejemplos de esta construcción, utilizando grupos lineales, son estudiados en detalle en [83].

3.2.3. Análisis de seguridad

El esquema MOR generalizado no ha sido demasiado estudiado en términos de seguridad, quizá por ser una propuesta muy general que da poca información al criptoanalista. No ocurre igual con el MOR básico, pues en [82] se dan sugerencias muy concretas de implementación que han permitido analizar el comportamiento del esquema en la práctica.

En el PKC 2003, Christian Tobias presentó un criptoanálisis del esquema MOR básico [95]. Su ataque se centra fundamentalmente en una elección del grupo G sugerida por los autores. Estos proponen en [82] tomar como grupo base G un producto semidirecto $SL_2(\mathbb{Z}_p) \times_\theta Z_p$, con

$$\theta : Inn \circ \theta_1 : \mathbb{Z}_p \longmapsto Aut(SL_2(\mathbb{Z}_p)),$$

y θ_1 un monomorfismo de $\mathbb{Z}_p$ en un subgrupo de $SL_2(\mathbb{Z}_p)$ (más detalles pueden verse en [82]).
Entre otros posibles fallos de seguridad, el análisis de Tobias demuestra que si el exponente b queda fijado para el cifrado de varios mensajes, el esquema es vulnerable a ataques por texto conocido (IND-CCA1): si un adversario conoce un par texto cifrado-texto claro, podrá descifrar cualquier texto cifrado con el exponente b. También se demuestra que si una sola entrada del texto claro (que es una matriz dos por dos) es conocida por el adversario, éste puede recuperar su totalidad a partir del texto cifrado. El autor del criptoanálisis propone por tanto usar algún método de *relleno o padding* y elegir los exponentes de cifrado de manera independiente y uniformemente al azar. De todas formas, serán necesarios nuevos resultados para determinar qué elecciones de parámetros conducen a una implementación del esquema MOR básico con garantías de seguridad.

3.3. Criptografía y grupos de trenzas

Las herramientas criptográficas basadas en grupos no abelianos que hemos estudiado hasta ahora son propuestas muy analizadas desde el punto de vista teórico, aunque apenas se han explorado sus propiedades en la práctica . Estudiamos ahora otro tipo de esquemas diseñados con un espíritu eminentemente práctico. Dichas construcciones surgen a raíz de la propuesta de Anshel, Anshel y Goldfeld [6, 8], que supuso el principio de una nueva dirección en criptografía: los esquemas basados en grupos de trenzas. Esta línea de investigación es hoy una de las más activas dentro de la criptografía de clave pública, no sólo desde el punto de vista teórico sino también en lo que se refiere a implementaciones prácticas.

Recordemos algunas nociones elementales de grupos de trenzas antes de detenernos en sus aplicaciones a la criptografía.

Definición 3.1 *Sea $n \in \mathbb{N}$, llamamos* grupo de trenzas de n cuerdas*, B_n, al grupo dado por la presentación finita $\langle X/R \rangle$, donde*

$$X := \{\sigma_1, \ldots, \sigma_{n-1}\},$$

$$R := \{\sigma_i\sigma_j(\sigma_j\sigma_i)^{-1}\ i,j \leq n-1, |i-j| \geq 2\} \cup$$
$$\{\sigma_i\sigma_j\sigma_i(\sigma_j\sigma_i\sigma_j)^{-1}\ i,j \leq N-1, |i-j| = 1\}.$$

Los elementos $\sigma_i, \quad i = 1, \ldots, n-1$ *se llaman* generadores de Artin.

La presentación $\langle \tilde{X}/\tilde{R}\rangle$ con $\tilde{X} := \{a_{ts}, \quad N \geq t > s \geq 1\}$ y

$$\tilde{R} := \{a_{ts}a_{rq} = a_{rq}a_{ts} \text{ si } (t-r)(t-q)(s-r)(s-q) > 0\} \cup$$
$$\{a_{ts}a_{sr} = a_{tr}a_{ts} = a_{sr}a_{tr} \quad \forall t,s,r \text{ con } n \geq t > s > r \geq 1\}$$

define también al grupo de trenzas de n cuerdas. Los generadores de $\tilde{X}$ reciben el nombre de *generadores de banda* y admiten la siguiente expresión en función de los de Artin:

$$a_{ts} = (\sigma_{t-1}\sigma_{t-2}\ldots\sigma_{s+1})\sigma_s(\sigma_{s+1}^{-1}\ldots\sigma_{t-2}^{-1}\sigma_{t-1}^{-1}) \quad \forall\, t,s \text{ con } n \geq t > s \geq 1.$$

Habitualmente se llama *trenzas* a los elementos del grupo B_n, haciendo referencia a su interpretación geométrica. Cada uno se ellos puede verse como una estructura de n cuerdas entre dos barras paralelas horizontales. La cuerda i-ésima parte de la posición i señalada en la barra superior y llega, posiblemente cruzando a otras cuerdas, a una cierta posición j señalada en la barra inferior. A cada posición $1, \ldots, n$ de la barra inferior llega exactamente una de las cuerdas. Dicha estructura suele llamarse el *diagrama* de la trenza. Según esta representación, el i-ésimo generador de Artin, σ_i, se representa por el diagrama en el que todas las cuerdas unen la posición j de la barra superior con la j de la barra inferior, salvo las cuerdas que salen de las posiciones i e $i+1$, que acaban respectivamente en las posiciones $i+1$ e i. La cuerda que sale de la posición $i+1$ cruza por delante a la que sale de la posición i.

El diagrama del producto de dos trenzas ab se construye 'apilando' los diagramas asociados a las trenzas a y b. Según esto, es fácil ver que la representación geométrica de un generador de banda a_{ts} es la trenza en la que las cuerdas que nacen en las posiciones s y t se intercambian, la que nace en t cruzando por delante a la que nace en s y ésta a su vez pasando por delante de todas las demás.

Esta representación resulta muy útil a la hora de interpretar intuitivamente la estructura del grupo. Por ejemplo, fijado un conjunto de generadores, dos palabras en los mismos serán equivalentes si y sólo si sus diagramas coinciden (o uno es deformación continua del otro). Esta idea es la que en su día inspiró la primera solución para el problema de la palabra en grupos de trenzas, que comentaremos en el Capítulo 4.

Otra herramienta de gran utilidad a la hora de manejar grupos de trenzas es el epimorfismo entre B_n y S_n que sugiere la presentación a través de los generadores de Artin. Nótese que añadiendo a dicha presentación la relación $\sigma_i^2 = e$, $\forall\, i$, obtenemos una presentación de S_n, por lo que la identificación

$$\sigma_i \longrightarrow \tau_i = (i, i+1)$$

define un epimorfismo.

Pero, ¿qué hace estos grupos especialmente atractivos desde un punto de vista criptográfico? Entre otras, las siguientes propiedades:

- El problema de la palabra es resoluble en tiempo polinomial. De hecho, podemos construir una aplicación F del conjunto de las palabras en los generadores en sí mismo cumpliendo que dos palabras son equivalentes si y sólo si sus imágenes por F coinciden. Toda aplicación con esa propiedad recibe el nombre de forma canónica. Existen varias formas canónicas para grupos de trenzas, que además pueden computarse en tiempo polinomial. En concreto, la llamada *forma canónica de Birman, Ko y Lee [19]* permite codificar cada trenza de n cuerdas a través de un número natural y un número finito de permutaciones. Con esta codificación puede además implementarse la ley de grupo de manera eficiente.

- Su estructura está muy bien estudiada, existen además varios problemas en la teoría de trenzas de aceptada dificultad. Por ejemplo, tras más de veinte años de búsqueda no se ha encontrado ningún algoritmo polinomial que resuelva el problema de la conjugación para $n > 6$. De hecho, los algoritmos construidos hasta la fecha con tal fin son exponenciales [19, 36].

A continuación, repasamos brevemente las principales herramientas criptográficas construidas utilizando grupos de trenzas.

3.3.1. El esquema de Anshel, Anshel y Goldfeld

En [6], Anshel, Anshel y Goldfeld propusieron un esquema de intercambio de claves algebraico descrito en términos muy generales. Desarrollaron con cierto detalle un caso particular de dicho esquema, que utiliza el problema de la conjugación múltiple en grupos de trenzas.

Problema de la conjugación múltiple: sea s un número natural. Dadas dos s-tuplas de elementos de un grupo G, $(a_1, \ldots, a_s), (b_1, \ldots, b_s)$ encontrar un elemento $x \in G$ tal que $b_i = x^{-1} a_i x$, $\forall\, i$.

Relacionado con éste, puede formularse el siguiente problema:

Problema de la conjugación múltiple - caso exhaustivo: sea s un número natural. Dadas dos s-tuplas de elementos de un grupo G, $(a_1, \ldots, a_s)$, $(b_1, \ldots, b_s)$ encontrar *todos* los $x \in G$ tal que $b_i = x^{-1}a_i x$, $\forall\, i$.

El principal argumento que los autores esgrimen para justificar la dificultad de los anteriores problemas es la dificultad del problema de la conjugación en grupos de trenzas.

Problema de la conjugación: sea G un grupo no abeliano y considerar dos elementos $g, h \in G$ conjugados en G. Hallar $b \in G$ tal que $h = bgb^{-1}$.

Claramente, un algoritmo que resuelva el problema de la conjugación en B_n también sirve para solucionar el problema de la conjugación múltiple, aunque todo apunta a que el recíproco de esta afirmación es falso. Más adelante comentaremos las mejores soluciones que hoy se conocen para ambos.

Una de las hipótesis básicas del esquema de Anshel, Anshel y Goldfeld es que exista una aplicación pública $\mathcal{K}$ (a la que llaman *extractor de clave*) que asigne a cada elemento del grupo un elemento dentro de un conjunto posible de claves. Tratándose del grupo de trenzas, una forma canónica computable en tiempo polinomial puede hacer las veces de dicho extractor (si las claves son elementos del grupo).

A continuación describimos la versión del esquema de Anshel, Anshel y Goldfeld que utiliza grupos de trenzas, un análisis mucho más detallado de la misma y varios ejemplos de implementaciones concretas pueden verse en la patente [5].

Intercambio de claves AAG:

Considerar disponibles en un directorio público un número natural n y dos conjuntos de palabras , X_A y X_B, en los generadores de Artin del grupo B_n,

$$X_A = \{\mathbf{s}_1, \ldots, \mathbf{s}_m\} \text{ y } X_B = \{\mathbf{t}_1, \ldots, \mathbf{t}_l\}.$$

Los sujetos A y B, que pretenden intercambiar una clave, perteneciente a un conjunto K. Disponen para ello de una aplicación $\mathcal{K} : B_n \longrightarrow \mathsf{K}$ (el *extractor de clave*).
A y B siguen los siguientes pasos:

1. A elige un elemento secreto $a \in G_A = \langle X_A \rangle_{B_n}$. Después computa, reescribe y transmite a B palabras representando a

   $$a^{-1}t_1a, \ldots, a^{-1}t_la.$$

   Análogamente, B elige un elemento secreto $b \in G_B = \langle X_B \rangle_G$. Después computa, reescribe y transmite a A palabras representando a:

   $$b^{-1}s_1b, \ldots, b^{-1}s_mb.$$

2. Usando la expresión de $a \in \langle X_A \rangle_{B_n}$ en función de los generadores del conjunto X_A, A computa una palabra representando a $b^{-1}ab$.

   De modo análogo, B computa una palabra representando a $a^{-1}ba$.

3. A y B computan palabras representantes del elemento $C := [a, b]$.

4. A y B pueden ahora utilizar como clave común $\mathcal{K}(C)$

Una de las propiedades indeseables de este esquema es el hecho de que los papeles de A y B no son totalmente simétricos (pues al construir el elemento K cada individuo realiza una operación distinta). Este es un inconveniente grande, pues los protocolos de intercambio de claves buscan ante todo que las partes involucradas estén en igualdad de condiciones.

Existen además algunos problemas de seguridad en las implementaciones propuestas hasta hoy. En la más reciente [4], se propone una forma de llevar a la práctica el esquema de Anshel, Anshel y Goldfeld para grupos de trenzas, sugeriéndose utilizar un extractor de claves basado en la llamada *representación coloreada de Bureau* del grupo de trenzas asociado. Dicha propuesta es hoy considerada insegura a raíz del trabajo de Lee y Lee. En [67] dichos autores propusieron dos ataques distintos al esquema:

- **Ataque al extractor de claves:** utilizando técnicas de álgebra lineal, Lee y Lee demuestran que a partir de las permutaciones inducidas por las claves secretas a y b es posible recuperar una parte considerable de la clave intercambiada resolviendo un problema de conjugación múltiple (caso exhaustivo) en $GL_n(\mathbb{F}_p)$.

  La parte restante de la clave puede obtenerse resolviendo un problema análogo en S_n. El problema de la conjugación múltiple para el caso no exhaustivo se resuelve en tiempo polinomial en S_n y $GL_n(\mathbb{F}_p)$. Aunque no se conoce la complejidad del caso exhaustivo, los autores razonan que no hay evidencias para argumentar que ésta sea alta. Para prevenir este ataque sería por tanto necesario demostrar que una cierta elección de parámetros conduce a problemas de conjugación múltiple exhaustivos difíciles (en S_n o $GL_n(\mathbb{F}_p)$).

- **Ataque al problema matemático:** los autores proponen un algoritmo para el problema de la conjugación múltiple en grupos de trenzas, muy rápido en ciertos casos particulares. Aunque dicho algoritmo no es polinomial, su existencia basta para considerar este problema *demasiado sencillo* para basar en él la seguridad de herramientas criptográficas. Además, el trabajo de Lee y Lee demuestra que el problema de la conjugación múltiple es mucho más fácil de resolver que el problema de la conjugación.

Inspirándose en el ataque al problema matemático de Lee y Lee, Hofheinz y Steinwandt propusieron en el PKC 2003 (ver [91]), un ataque definitivo:

- **Ataque heurístico de Hofheinz y Steinwandt**: partiendo del algoritmo de [67] para el problema de la conjugación, los autores construyen

un algoritmo heurístico que no resuelve el problema de la conjugación en el caso general, pero sí funciona en los casos involucrados en las implementaciones propuestas del esquema de Anshel, Anshel y Goldfeld. Hasta la fecha no existe una nueva propuesta de elección de parámetros que resista este ataque en la práctica.

3.3.2. El criptosistema de Ko-Lee-Cheon-Han-Kan-Park

A partir de un problema inspirado en el de la conjugación en grupos de trenzas, conocido ahora como *problema de Ko-Lee* o *problema de la conjugación de Diffie-Hellman*, se proponen en el Crypto del 2000 [63] un sistema de intercambio de claves y un criptosistema asociado. Aunque esta construcción es independiente de la que acabamos de ver, puede demostrarse (ver [7]) que ambos son un caso particular del protocolo algebraico general de intercambio de claves propuesto por Anshel, Anshel y Goldfeld en [6]. Sin embargo, las propiedades de seguridad de estos esquemas son significativamente mejores que las de su predecesor.

Comencemos por enunciar el problema matemático en el que se basan estos esquemas:

Problema de Ko-Lee: sea G un grupo no abeliano, G_1 y G_2 dos subgrupos de G tales que $g_1 g_2 = g_2 g_1$, para todo $g_1 \in G_1, g_2 \in G_2$. Dada una terna (u, aua^{-1}, bub^{-1}), con $u \in G, a \in G_1$ y $b \in G_2$, calcular el elemento $baua^{-1}b^{-1}$.

El problema de Ko-Lee en grupos de trenzas se aplicará considerando los siguientes subgrupos:

Definición 3.2 *Sea n un número natural cualquiera, l y r dos números naturales que suman n. Llamaremos l-subgrupo a izquierda del grupo de trenzas de n cuerdas al subgrupo que generan los $l-1$ primeros generadores de Artin. Lo denotaremos LB_l. De manera análoga, llamaremos r-subgrupo a izquierda del grupo de trenzas de n cuerdas (denotado RB_r) al subgrupo que generan los $r-1$ últimos generadores de Artin. Así,*

$$LB_l := \langle \sigma_1, \ldots, \sigma_{l-1} \rangle_{B_n}, \quad RB_r := \langle \sigma_{n-r+1}, \ldots, \sigma_{n-1} \rangle_{B_n}.$$

Nótese que por ser $r + l = n$, se tiene $n - r + 1 = l + 1$, de donde cada elemento de LB_l conmuta con todos los de RB_r.

El principal argumento que los autores esgrimían para justificar la dificultad del problema de Ko-Lee, es la dificultad del llamado *problema de la conjugación generalizado* en grupos de trenzas:

Problema de la conjugación generalizado: sea G un grupo no abeliano y $H \leq G$. Considerar dos elementos $g, h \in G$ que son conjugados por un elemento de H. Encontrar $b \in H$ tal que $h = bgb^{-1}$.

Los autores del esquema partían de la suposición de que dicho problema es muy difícil para $G := B_n$ y $H := B_m$, con $m \leq n$. De no ser así, es claro que tampoco el problema de Ko-Lee sería difícil para $G_1 := LB_l$ y $G_2 := RB_r$: notar que LB_l es isomorfo a B_l y recuperar el elemento $a \in G_1$ conocidos u y aua^{-1} es exactamente resolver el problema de la conjugación generalizado.

Esquema de intercambio de claves de Ko-Lee-Cheon-Han-Khan-Park:

Sea n un número natural, l y r dos naturales que suman n. Se hacen públicos estos números, al igual que un elemento $x \in B_n$. Supongamos que dos individuos A y B quieren intercambiar una clave secreta para comunicarse. Para ello actúan según los siguientes pasos:

- *A elige un elemento secreto $a \in LB_l$, computa, reescribe y envía a B una palabra equivalente a x^a.*
- *De modo análogo, B elige un elemento secreto $b \in RB_r$, computa, reescribe y envía a A una palabra equivalente a x^b.*
- *A y B computan palabras equivalentes a*

$$y := x^{ba} = x^{ab}.$$

- *A y B obtienen a partir del elemento y su clave privada.*

En principio, los autores no especifican cómo determinar la información concreta que constituye la clave privada intercambiada. De su descripción se

deduce que cualquier forma canónica del correspondiente grupo de trenzas puede utilizarse como extractor de clave.

A partir del protocolo anterior se construye el criptosistema asociado, de modo análogo a cómo ElGamal se construye a partir del esquema de Diffie-Hellman.

Criptosistema de clave pública de Ko-Lee y otros:

- Clave pública: (r, l, x, y, H) donde: r, l son dos números naturales, $x \in B_{r+l}$. El elemento $y \in B_{r+l}$ se obtiene conjugando x por un elemento secreto $a \in LB_l$ que constituye la clave privada. H es una función hash adecuada definida de B_{l+r} en el espacio de mensajes, que suponemos es $\{0,1\}^k$.

- Clave privada: $a \in LB_l$.

Supongamos que A quiere enviar un mensaje $m \in \{0,1\}^k$ a B.

- Cifrado: *A elige un elemento secreto $b \in RB_r$, computa, reescribe y envía a B una palabra equivalente a x^b, y la cadena de bits que resulta al sumar (bit a bit, módulo dos) m y $H(y^b)$.*
- Descifrado: *B dispone de la clave privada, y recibe de A un par (c, d), con $c \in B_{r+l}$ y $d \in \{0,1\}^k$. Recupera m sumando bit a bit módulo dos las secuencias $H(c^a)$ y d.*

Por supuesto, elegir cuidadosamente la trenza x es crucial a la hora de garantizar la seguridad del esquema. También han de imponerse ciertas condiciones de seguridad en la función hash H. En [63], los autores discuten diversas estrategias para elegir estos parámetros.

No nos extendemos en la discusión de cómo pueden elegirse los parámetros del esquema anterior pues ésta pierde sentido a la vista de los últimos ataques descubiertos. A continuación comentamos varios tipos de ataques muy recientes que aparentemente son muy difíciles de evitar con una elección correcta de los parámetros del esquema:

- **Ataque heurístico de Hofheinz y Steinwandt:** el algoritmo heurístico de [91] sirve también en este caso para obtener las claves secretas de cualquiera de las implementaciones propuestas de este esquema. Eso basta para que no pueda considerarse seguro, pero es posible que existan razones teóricas aun más convincentes:

- **Ataque de Cheon y Jung al problema de Ko-Lee:** recientemente ha aparecido (ver [27]) un algoritmo que resuelve el problema de Ko-Lee en tiempo polinomial. Para ello utilizan una representación de las trenzas distinta a la habitual; la llamada representación de Lawrence-Krammer, y luego resuelven el problema en un subconjunto acotado del álgebra de matrices de polinomios en $\mathbb{Z}[t, q]$. Si se verifica la corrección de este algoritmo, su existencia haría muy difícil modificar el esquema de Ko y otros para garantizar niveles aceptables de seguridad.

- **Ataque de Lee y Park al problema de la conjugación generalizado:** en el Eurocrypt 2003 va a presentarse [65] un nuevo criptoanálisis del esquema de Ko y otros. Este criptoanálisis utiliza un algoritmo que resuelve el problema de la conjugación generalizado para el caso $G := B_n, H := B_m$, $m \leq n$, usando la representación coloreada de Bureau (ver [54]). El algoritmo que proponen para romper el esquema es heurístico, una especie de búsqueda exhaustiva selectiva en la línea del ataque de Hofheinz y Steinwandt. Sin embargo, este último podía aplicarse para parámetros n y m mucho mayores. Por otra parte, el ataque de Lee y Park es efectivo contra la implementación del esquema de Ko y otros detallada en [26], que no puede atacarse con el método de Hofheinz y Steinwandt.

3.3.3. Otras construcciones

Repasaremos algo más superficialmente otras propuestas de herramientas criptográficas basadas en grupos de trenzas:

Generación pseudoaleatoria

En el Crypto 2001 [64] Lee, Lee y Hahn propusieron un método de generación de secuencias pseudoaleatorias basado en el problema de decisión asociado al problema de Ko-Lee, i.e.:

Problema de Ko-Lee (decisión): sea G un grupo no abeliano, G_1 y G_2 dos subgrupos de G tales que $g_1 g_2 = g_2 g_1$, para todo $g_1 \in G_1, g_2 \in G_2$. Dada una tupla $(u, aua^{-1}, bub^{-1}, cuc^{-1})$, con $u, c \in G, a \in G_1$ y $b \in G_2$, decidir si $c = ba$.

Poco después (en el Eurocrypt 2002 [38]), Gennaro y Micciancio presentaron un algoritmo polinomial que resuelve con probabilidad muy alta el problema de base. Consecuentemente, el generador pseudoaleatorio propuesto es una herramienta insegura.

Distribución autentificada de claves en grupo

Muy recientemente ha sido propuesto un esquema de distribución autentificada de claves en grupo [66] basada en grupos de trenzas. El objetivo de tal herramienta es hacer posible que un grupo de individuos construya una clave común sin que sea posible que uno de ellos determine por sí sólo la clave. Otras propiedades que (según los autores) tiene este esquema, es el evitar que un adversario suplante a algún individuo del grupo y también que la participación en el protocolo en el pasado proporcione información acerca de claves que se construyan en el futuro.

Puesto que la base del esquema es el problema de Ko-Lee, su seguridad es más que cuestionable a la vista de los ataques que mencionamos en la sección anterior.

Esquemas de autenticación

Recientemente ha aparecido también una propuesta para construir esquemas de autenticación de entidades usando grupos de trenzas (ver [90]). Un esquema de autenticación de entidades es una herramienta criptográfica que permite a un individuo que trabaja en red verificar la identidad de aquellos

con los que interacciona. Una definición formal de este tipo de esquemas se encuentra en [10].

Los autores proponen tres esquemas que utilizan grupos de trenzas, uno basado en problema de Ko-Lee, otro en el problema de la conjugación (búsqueda) y un tercero basado también en el problema de conjugación (búsqueda) y en el llamado problema de extracción de raíces:

Problema de extracción de raíces: sea $g \in G$ un elemento del que se sabe es una e-potencia de un elemento en G, siendo $e \in \mathbb{Z}$ conocido. Encontar $k \in G$ tal que $g = k^e$.

Será necesario estudiar estos esquemas a fondo antes de poder extraer conclusiones acerca de su seguridad, aunque en principio parece razonable pensar que la seguridad de los dos primeros esquemas propuestos en [90] se vea también comprometida por los ataques de [91, 27, 65].

En cuanto al tercer esquema, no existe una manera clara de resolver el problema de extracción de raíces con los algoritmos de ataque a otros esquemas de trenzas. Aun así, tal como el esquema se describe actualmente, un adversario que sea capaz de resolver el problema de conjugación (búsqueda) puede suplantar con éxito a un usuario autorizado con probabilidad muy superior a la que los autores estiman. En conclusión, los tres esquemas habrán de ser revisados y modificados considerando los nuevos algoritmos para el problema de la conjugación en grupos de trenzas.

A continuación, resumimos en un cuadro el *estado del arte* en criptografía de clave pública basada en grupos de trenzas.

Esquema	***Problema***	***Criptoanálisis***
- AAG 1999, [6, 5, 4]	- Conjugación múltiple	- Eurocrypt 2002 [67]
- KLCHKP Crypto 2002 [63]	- Ko-Lee (búsqueda)	-PKC 2003 [58] - [27] - Eurocrypt 2003 [65]
- LLH Crypto 2001 [64]	- Ko-Lee (decisión)	- Eurocrypt 2002 [38]
- LLL [66]	- Ko-Lee	- [58] ? - [27] ? - [65] ?
- SDG [90]	- Ko-Lee - conjugación - extracción de raíces	- [58], [27] [65] ? - [65] ? - ?

Criptografía y grupos de trenzas: estado del arte

Las interrogaciones simbolizan el hecho de que, aunque el artículo referenciado probablemente rompe el esquema en cuestión, dicho criptoanálisis no ha sido publicado explícitamente.

3.3.4. Trenzas y criptografía: nuevas construcciones a través de secuencias de Markov

A la vista del cuadro anterior, es claro que si los resultados de [65] son correctos, no tiene demasiado sentido construir nuevas herramientas basadas en el problema de la conjugación generalizado. Más aún, el hecho de que la mayoría de los expertos esperen que a corto plazo se disponga de un algoritmo polinomial para el problema de la conjugación en grupos de trenzas (ver [19, 36]), justifica la desconfianza de la comunidad científica hacia los esquemas propuestos hasta ahora.
Sin embargo, el robusto armazón teórico disponible permite reutilizar las propuestas anteriores tomando como base, en lugar de grupos de trenzas, otros grupos de similares características. Otra alternativa prometedora es utilizar en grupos de trenzas problemas más difíciles que el de la conjugación. En esta sección esbozamos algunas ideas para utilizar en criptografía la teoría desarrollada alrededor de un teorema fundamental en el estudio de nudos y trenzas: el *teorema de Markov*. Nuestra intención no es proponer un nuevo esquema de clave pública, sino señalar, en términos muy abstractos, una posible dirección de trabajo en este campo. La idea que exponemos a continuación surge de la colaboración con J. González-Meneses, D. Hofheinz y R. Steinwandt.

El teorema de Markov

Una de las razones por la que la teoría de grupos de trenzas está enormemente desarrollada es su estrecha relación con la teoría de nudos. Recordar que un nudo es una curva lisa simple y cerrada. A lo largo de esta sección hablamos sólo de nudos y trenzas orientados. La orientación de las trenzas se considera siempre suponiendo que en el diagrama plano las cuerdas se dibujan de abajo a arriba.

Cualquier nudo puede representarse como la *clausura* de una trenza (i.e., si vemos el diagrama de la trenza en el espacio, podemos unir 'por detrás' del plano donde está el diagrama cada extremo i de la barra superior con el extremo i de la barra inferior, obteniendo así un nudo orientado). Este resultado es el famoso *teorema de Alexander* [81]. Nótese que a veces la clausura de una trenza no es un único nudo, sino varios nudos que forman una *cadena*.

Así, dos trenzas que se correspondan con el mismo elemento del grupo originan al cerrarse el mismo nudo o cadena. Sin embargo, también es posible que trenzas distintas originen el mismo nudo o la misma cadena (es el caso, por ejemplo, de las trenzas conjugadas). En los años treinta Markov planteaba

la siguiente pregunta : ¿Podemos decidir cuándo dos trenzas originarán el mismo nudo al cerrarse? Ya en los setenta, Joan Birman dio una respuesta afirmativa a esta pregunta, demostrando el llamado *teorema de Markov* en [18].

Para enunciar el teorema necesitamos introducir ciertas transformaciones de trenzas llamadas *movimientos de Markov*.

Definición 3.3

- *Se llama* movimiento de Markov de tipo 1 (M_1) *a la conjugación de una trenza* $b \in B_n$ *por otra trenza* $a \in B_n$,

$$b \xrightarrow{M_1} aba^{-1}.$$

- *Se llama* movimiento de Markov de tipo 2 (M_2) *a la multiplicación en* B_{n+1} *de una trenza* $b \in B_n$ *por el generador de Artin* $\sigma_n \in B_{n+1}$ *o por su inverso,*

$$b \xrightarrow{M_2} b\sigma, \text{ donde } \sigma = \sigma_n^{\pm 1}.$$

El teorema de Markov establece que dos trenzas originan al cerrarse el mismo nudo o cadena si y sólo si es posible transformar una en la otra realizando un número finito de movimientos de Markov:

Teorema 3.4 (Teorema de Markov) *Sean* b *y* b' *dos trenzas (no necesariamente del mismo número de cuerdas). Las clausuras de* b *y* b' *representan el mismo nudo o cadena si y sólo si* b *puede transformarse en* b' *por medio de un número finito de movimientos de Markov, es decir, si existe una secuencia*

$$b = b_0 \longrightarrow b_1 \longrightarrow \cdots \longrightarrow b_{m-1} \longrightarrow b_m = b'$$

tal que para $i = 0, \ldots, m-1$, *la trenza* b_{i+1} *es el resultado de aplicar a* b_i *un movimiento de Markov.*

Definición 3.5 *Llamaremos* secuencia de Markov *a cualquier secuencia finita de movimientos de Markov.*

Notas:

- El teorema de Markov sólo está demostrado considerando nudos y enlaces orientados.

- El teorema de Markov no resuelve el problema de clasificación de nudos y enlaces a partir de la clasificación de las trenzas: hoy en día no existe ningún método para decidir si dos trenzas son *equivalentes Markov*, i.e., si existe una secuencia de Markov finita que transforme a una en otra.

- Ser *equivalentes Markov* es una relación de equivalencia y por el resultado anterior dos trenzas son equivalentes Markov si y sólo si generan el mismo nudo orientado o cadena orientada.

Es claro que un algoritmo que, dadas dos trenzas b y b' equivalentes Markov, calculase una secuencia de Markov para transformar b en b', podría usarse para decidir si dos trenzas son o no equivalentes Markov. Notar que además el problema de recuperar tal secuencia generaliza al problema de la conjugación en grupos de trenzas.

Por lo anterior, es razonable plantear la posibilidad de construir un esquema de intercambio de claves de la siguiente forma:
Sean M_A y M_B dos listas de secuencias de Markov con la siguiente propiedad: cada secuencia de M_A conmuta con toda secuencia de M_B, es decir, para toda $M_{\mathsf{a}} \in M_A$ y $M_{\mathsf{b}} \in M_B$, las secuencias de Markov $M_{\mathsf{a}}M_{\mathsf{b}}$ y $M_{\mathsf{b}}M_{\mathsf{a}}$ producen exactamente la misma transformación en cualquier trenza. Sea $b \in B_n$ una trenza. Definimos el siguiente esquema de intercambio de claves entre dos individuos A y B.

Intercambio de claves basado en el teorema de Markov:

- *A elige una secuencia secreta M_{a} de la lista M_A y envía a B la trenza b_1 que resulta de aplicar M_{a} a b.*
- *De modo análogo, B elige una secuencia secreta M_{b} de la lista M_B y envía a A la trenza b_2 que resulta de aplicar M_{b} a b.*
- *A y B conocen la trenza K, obtenida al aplicar M_{b} a b_1 o M_{a} a b_2, y utilizan dicha información común para fijar una clave secreta.*

Notas:

- A partir de este esquema de intercambio de claves, podría definirse un criptosistema de clave pública (de modo análogo a como el criptosistema ElGamal se construye a partir del intercambio de claves de Diffie–Hellman).

- El esquema de intercambio de claves de Ko, Lee y otros 3.3.2 es un caso particular de esta construcción siendo M_A y M_B movimientos de Markov de tipo 1 involucrando elementos de RB_r y LB_l respectivamente.

No es en absoluto trivial diseñar los conjuntos M_A y M_B de modo que la construcción resultante sea, al menos teóricamente, más segura que que la de Ko, Lee y otros. Definir con precisión un esquema con esas características es por tanto un problema abierto. En el Capítulo 4 discutiremos otra idea reciente y aún no investigada para construir herramientas criptográficas a partir de grupos de trenzas.

3.4. El criptosistema MST_2

En el Capítulo 2, Sección 2.3, estudiábamos con cierto detalle el esquema MST_1, una de las herramientas propuestas por Magliveras y otros en [76]. En esta sección haremos un análisis similar del MST_2, otro criptosistema de clave pública presentado en dicho trabajo que se inspira en la construcción de Diffie-Hellman.

A partir del esquema MST_2 introducimos una construcción general de un criptosistema de clave pública que generaliza a muchos de los esquemas que hemos visto en la sección anterior. Dicha generalización no sólo permite estudiar simultáneamente las propiedades de los esquemas conocidos, sino que puede además inspirar la búsqueda de nuevas herramientas criptográficas basadas en grupos.

3.4.1. El criptosistema MST_2 original

El criptosistema MST_2 es una generalización del criptosistema ElGamal que puede construirse a partir de grupos finitos no abelianos. En concreto, las piezas fundamentales de este esquema son, al igual que en el MST_1, ciertas secuencias que inducen factorizaciones en grupos de permutaciones finitos.

Definición 3.6 *Sea G un grupo de permutaciones de grado n. Para $s \in \mathbb{N}_0$, sea $\alpha = [\alpha_1, \ldots, \alpha_s]$ una secuencia tal que α_i $(1 \leq i \leq s)$ es a su vez una secuencia $\alpha_i = [\alpha_{i0}, \ldots, \alpha_{ir-1}]$, con $r \in \mathbb{N}_0$ y $\alpha_{ij} \in S_n$ $(0 \leq j < r)$. Se dice que α es una* $[s, r]-$malla *de G si todo $g \in G$ puede expresarse como un producto*

$$g = \alpha_{1j_1} \cdots \alpha_{sj_s},$$

con $\alpha_{ij_i} \in \alpha_i$ $(1 \leq i \leq s)$, y además si cada elemento $g \in G$ admite $a_g \in \mathbb{N}_0$ factorizaciones de esa forma distintas, la distribución de probabilidad definida sobre el grupo G por

$$\mathcal{P}_\alpha(g) := \frac{a_g}{s^r}$$

es aproximadamente *uniforme.*

La definición anterior no es demasiado precisa, pues no indica qué distribuciones de probabilidad pueden considerarse *aproximadamente uniformes*. En [76] los autores proponen admitir como aproximadamente uniformes aquellas distribuciones que pasen un test estándar de uniformidad (como el de Kolmogorov–Smirnov).

De modo análogo a lo que ocurría en el caso de las signaturas logarítmicas, podemos asociar a cada malla α una biyección $\breve{\alpha} : \mathbb{Z}_{|s|^r} \longrightarrow G$ (ver 2.3).
En este caso, la seguridad del MST_2 se estudirará bajo la hipótesis de trabajo de que invertir dichas biyecciones es muy difícil, i.e., supondremos que el siguiente problema es intratable:

Problema de factorización con respecto a una $[s, r]-$***malla***: dado G un grupo de permutaciones de grado n, un elemento cualquiera $g \in G$ y una $[s, r]-$malla α, encontrar una factorización de g de la forma

$$g = \alpha_{1j_1} \cdots \alpha_{sj_s},$$

con $\alpha_{ij_i} \in \alpha_i$.

Aunque en [31] se demuestra que este problema es, en su planteamiento más general, al menos tan difícil como el problema del logaritmo discreto, demostraremos que en muchos casos puede resolverse de manera eficiente.
Ya estamos en condiciones de ver la descripción del esquema MST_2.

Criptosistema MST_2 :

Sea G un grupo de permutaciones de grado n, H un grupo que constituye el espacio de mensajes.

- Clave privada: Un epimorfismo $f : G \longrightarrow H$.

- Clave pública: G, H y dos secuencias α y β, donde α es una $[s, r]-$malla de G elegida con la condición de que la secuencia $\beta = f(\alpha)$ sea también una $[s, r]-$malla para H.

  - *B quiere enviar* $h \in H$ *a A, para ello realiza los siguientes pasos:*

    i. elige al azar un entero $R \in \mathbb{Z}_{r^s}$,

    ii. computa $y_1 := \breve{\alpha}(R)$, $y_2 := \breve{\beta}(R)$ *e* $y_3 := hy_2$,

    iii. envía $y := (y_1, y_3)$ *a A.*

  - *A recupera el mensaje* h, *calculando* $y_2 = f(y_1)$, *y obteniendo así* $h = y_3 y_2^{-1}$.

Así, además de la hipótesis de trabajo que mencionábamos antes, es necesario imponer que un adversario no pueda encontrar, a partir de la información pública, un epimorfismo $f' : G \longrightarrow H$ tal que $f'(\alpha) = \beta$. Podemos también hacer otras observaciones acerca de la seguridad del esquema.

Análisis de seguridad del MST_2 original

El primer punto que queremos resaltar es la necesidad de introducir los conceptos de malla mansa y malla salvaje para disponer de un marco teórico adecuado. Tal distinción es fundamental, pues está claro que existen mallas cuyas factorizaciones asociadas son sencillas de computar (tomar, por ejemplo, una signatura logarítmica mansa con bloques del mismo tamaño). Como

en el caso de las signaturas logarítmicas, la definición de *salvaje* debería formalizar la idea de que la factorización inducida es *muy difícil de calcular, salvo para una cantidad despreciable de elementos.*

Aunque en [76] no se precisa cómo elegir los elementos para una implementación del MST_2, los autores indican que quizá una elección adecuada sea tomar $H = G$ y $f : G \longrightarrow H$ la conjugación por un elemento g del grupo, esto es, $\beta_{ij} = g\alpha_{ij}g^{-1}$. En ese caso, para romper el esquema es necesario encontrar un elemento $g' \in G$ tal que

$$\beta_{ij} = g'\alpha_{ij}g'^{-1}, \text{ para } i = 1, \ldots, s,\ j = 1, \ldots, r.$$

Nuevamente el adversario se enfrenta a un sistema de ecuaciones de conjugación. La búsqueda de soluciones de este sistema de ecuaciones puede analizarse desde el siguiente punto de vista. Asumamos que es sencillo calcular elementos $u_{ij} \in G$ tales que $\beta_{ij} = \alpha_{ij}^{u_{ij}}$. Si denotamos por $C_G(x)$ al centralizador en G del elemento x, es claro que dados $x, y, z \in G$ tales que $yxy^{-1} = z$, se tiene $\{w \in G \mid wxw^{-1} = z\} = C_G(x)y$. Así, el elemento g' que buscamos está en

$$\bigcap_{i,j} C_G(\alpha_{ij})u_{ij}.$$

No existen resultados que indiquen, de modo general, cuándo computar un elemento en esa intersección es sencillo. Sin embargo, sí hay resultados parciales (ver de nuevo [76]) que indican qué grupos no deben utilizarse para alcanzar un mínimo nivel de seguridad.

También es importante resaltar que la dificultad del problema de factorización con respecto a una $[s, r]-$malla depende, (como ocurre en el caso del problema del logaritmo discreto), de la representación de los elementos del grupo que se emplee. Si representamos G como un grupo de permutaciones y consideramos, como antes, $H = G$ y $f = Inn(g)$, con $g \in G$, el esquema es claramente inseguro: si disponemos de una representación de los elementos de las mallas públicas α y β como producto de ciclos disjuntos, el sistema de ecuaciones de conjugación resultante

$$\alpha_{i0}^g = \beta_{i0}, \ldots, \alpha_{ir-1}^g = \beta_{ir-1} \quad (1 \leq i \leq s),$$

proporciona muchísima información acerca del elemento g. Recordar que dado un producto de ciclos disjuntos $\tau = c_1 \cdots c_t$ se tiene $\tau^g = c_1^g \cdots c_t^g$, y por cada ciclo $(i_1, \ldots, i_l)$ podemos obtener información sobre cómo g actúa sobre varias letras

$$(i_1, \ldots, i_l)^g = (g(i_1), \ldots, g(i_l)).$$

Así, simplemente observando los elementos de α y β se puede extraer muchísima información acerca del elemento g. Veamos un ejemplo:

Ejemplo 3.1 *Considerar las secuencias*

$$\begin{array}{lcccc} \alpha_1 := [& (1,2,4,5,3), & (1,4,5), & (2,3)(4,5), & (1,3,4,5,2) \ ], \\ \alpha_2 := [& (1,4,5), & (1,3)(2,5), & (1,3,5,4,2), & (1,3,2,4,5) \ ], \\ \alpha_3 := [& (3,4,5), & (1,3,4), & (1,4,3), & \mathrm{id} \ ]. \end{array}$$

La secuencia $\alpha := [\alpha_1, \alpha_2, \alpha_3]$ *es una* $[3,4]$*-malla del grupo* A_5*. Si conjugamos la malla* α *con el elemento* $g = (1,4,2,3,5) \in A_5$ *obtenemos una* $[3,4]$*-malla* $\beta = [\beta_1, \beta_2, \beta_3]$*:*

$$\begin{array}{lcccc} \beta_1 = [& (1,5,4,3,2), & (1,4,2), & (1,2)(3,5), & (1,3,4,5,2) \ ], \\ \beta_2 = [& (1,4,2), & (1,3)(4,5), & (1,2,3,4,5), & (1,4,5,3,2) \ ], \\ \beta_3 = [& (1,5,2), & (2,4,5), & (2,5,4), & \mathrm{id} \ ]. \end{array}$$

Comparando los primeros elementos de α_3 *y* β_3 *es fácil ver que* g *transforma el conjunto* $\{3,4,5\}$ *en el conjunto* $\{1,5,2\}$*. Análogamente, mirando los segundos elementos de los bloques* α_3 *y* β_3 *tenemos que* $g(\{1,3,4\}) = \{2,4,5\}$*. Por tanto,* $g(5) \in \{1,5,2\} \setminus \{2,4,5\}$ *luego* $g(5) = 1$*. De modo similar, como* $g(1) \in \{2,4,5\} \setminus \{1,5,2\}$*, se tiene* $g(1) = 4$*. Fijémonos ahora en los segundos elementos de los bloques* α_2 *y* β_2*. Como* $g((1,3)) \in \{(1,3),(4,5)\}$ *y además* $g(1) = 4$*, tenemos que* $g((1,3)) = (4,5)$*. Por tanto,* $g(3) = 5$*. Usando las ecuaciones* $g((1,3)(2,5)) = (1,3)(4,5)$ *y* $g((1,3)) = (4,5)$ *se sigue que* $g((2,5)) = (1,3)$ *y como* $g(5) = 1$ *es claro que* $g(2) = 3$*. Así, un adversario puede conocer las imágenes por* g *de* 1*,* 2*,* 3*, y* 5*, por lo que sabe que sólo cabe la posibilidad de que* $g = (1,4,2,3,5)$*.*

Ya hemos mencionado que el criptosistema ElGamal es un caso particular del esquema anterior. De hecho, ElGamal es el único ejemplo que dieron de los autores de cómo llevar el MST_2 a la práctica. Con la intención de buscar nuevos ejemplos, investigamos las características comunes del MST_2 con otros esquemas basados en grupos conocidos, identificando cierto paralelismo en su estructura. Así, extendiendo los conceptos de factorización que aparecen en [76], introducimos una versión generalizada del esquema MST_2 y demostramos que varios criptosistemas conocidos pueden verse como casos particulares de la misma. Esta formalización proporciona un marco adecuado para estudiar simultáneamente varias herramientas criptográficas basadas en teoría de grupos.

3.4.2. El esquema MST_2 generalizado

Comenzamos por generalizar las secuencias de factorización involucradas en los esquemas MST_1 y MST_2. Para ello, sustituimos en la definición de dichas secuencias la acción regular de un grupo sobre sí mismo por la acción de un semigrupo con unidad (no necesariamente finito) sobre el conjunto que se pretende factorizar. Nótese que no imponemos que los conjuntos o secuencias involucradas sean finitos.

Definición 3.7 *Sea $M = (M, \cdot, 1)$ un semigrupo con unidad que actúa sobre un conjunto T por medio de la acción*

$$\begin{array}{rccc} \circ: & M \times T & \longrightarrow & T \\ & (m,t) & \longmapsto & m \circ t \end{array} ,$$

y Λ^* un conjunto contable totalmente ordenado. Definir $\Lambda := \Lambda^* \cup \{\top\}$ con $\top \notin \Lambda^*$ y considerar en Λ el orden extendido según el cual $\top$ es mayor que cualquier elemento de Λ^*.
Sea $\alpha = [\alpha_\lambda]_{\lambda\in\Lambda}$ una secuencia tal que:

- *para cada $\lambda \in \Lambda^*$, $\alpha_\lambda := [\alpha_{\lambda i}]_{i\in I_\lambda}$ es una secuencia contable de elementos de M, apareciendo la unidad en todos salvo a lo sumo un número finito de los* bloques *α_λ;*
- *el bloque $\alpha_\top := [\alpha_{\top i}]_{i\in I_\top}$ es una secuencia contable de elementos de T.*

*Se dice que α es una $(M, \circ, \Lambda)$-*cubierta *de T (o simplemente, una* cubierta *de T) si todo $t \in T$ puede expresarse como $t = m \circ \alpha_{\top i}$, para algún $\alpha_{\top i} \in \alpha_\top$ y algún elemento m de M que se factoriza como un producto $\prod_{\lambda\in\Lambda^*} \alpha_{\lambda i}$, donde $\alpha_{\lambda i} \in \alpha_\lambda$ y sólo un número finito de factores son distintos de la unidad.*

Para hacer referencia a dicha factorización escribiremos

$$t = \bigcirc_{\lambda\in\Lambda} \alpha_{\lambda i},$$

donde $\bigcirc_{\lambda\in\Lambda} \alpha_{\lambda i}$ denota aplicación iterativa de $\circ$.
En lo que sigue, cuando $\Lambda^* = \{1, \ldots, n\} \subseteq \mathbb{N}$ con el orden natural, elegiremos $\top := n + 1$, de modo que la cubierta tenga forma $\alpha = [\alpha_1, \ldots, \alpha_{n+1}]$.

Definición 3.8 *Sea* $\alpha := [[\alpha_{\lambda i}]_{i\in I_\lambda}]_{\lambda\in\Lambda}$ *una* $(M, \circ, \Lambda)$*-cubierta de un conjunto* T*, y para cada* $t \in T$ *considerar el conjunto*

$$F(t) := \Big\{[i_\lambda]_{\lambda\in\Lambda} \in \prod_{\lambda\in\Lambda} I_\lambda | t = \bigcirc_{\lambda\in\Lambda} \alpha_{\lambda i_\lambda}\Big\},$$

que caracteriza las factorizaciones de t *respecto a la cubierta* α*. Se dice que* α *es una*

- signatura logarítmica *si* $\mathrm{card}(F(t)) = 1$ *para todo* $t \in T$*, i.e., si todo elemento de* T *admite exactamente una factorización respecto a* α*.*
- malla *si se cumple que*
  - *para todo* $t \in T$*,* $F(t)$ *es infinito, o*
  - *todos los conjuntos* $F(t)$ *son finitos y además, si* T *es infinito, de igual cardinal, mientras que si* T *es finito, la distribución de probabilidad*
    $$[P_t = |F(t)|/|T| : t \in T]$$
    es aproximadamente uniforme (en el sentido de [76]).
- $[\mathrm{card}(\Lambda), r]$-malla *si* α *es una malla y todos los conjuntos de índices* I_λ*,* $\lambda \in \Lambda$*, son del mismo cardinal* r*.*

Es fácil ver que las nociones signatura logarítmica y $[\mathrm{card}(\Lambda), r]$-malla usadas en la construcción original del MST_1 y MST_2 son casos particulares de la definición anterior(ver las definiciones 3.6, 2.5 ó el artículo original [76]).

Nota: Sea $M = T := G$ un grupo finito, y denotemos por $\circ$ la acción de G sobre sí mismo por multiplicación a izquierda (acción regular de G). Tomando $\Lambda := \{1, \ldots, s\}$ con $s \in \mathbb{N}$ e imponiendo que cada I_λ, $\lambda \in \Lambda$, sea finito, se obtienen las nociones originales de cubierta, signatura logarítmica y $[\mathrm{card}(\Lambda), r]$-malla definidas en [76].

Del mismo modo, se pueden definir las nociones de *mansa* y *salvaje* para las factorizaciones de 3.7.

Definición 3.9 *Decimos que una* $(M, \circ, \Lambda)$*-cubierta* $\alpha = [\alpha_\lambda]_{\lambda\in\Lambda}$ *de un conjunto* T *es* mansa *si existe un algoritmo polinomial que con entrada* $t \in T$ *computa elementos* $\alpha_{\lambda i} \in \alpha_\lambda$*,* $\lambda \in \Lambda$*, tales que* $t = \bigcirc_{\lambda\in\Lambda} \alpha_{\lambda i}$*. Una cubierta no mansa se dice* salvaje*.*

Nuevamente, el hecho de que una cubierta α sea salvaje según esta definición no implica que factorizar respecto a ella sea un proceso de una vía: incluso aunque no se conozca ningún algoritmo eficiente para factorizar elementos arbitrarios de T respecto a la cubierta α puede ser que un subconjunto de T de tamaño no despreciable pueda factorizarse eficientemente; i.e., pueden ser efectivos los que llamábamos *ataques por inversión parcial*.

Introducimos a continuación el esquema MST_2 generalizado. En lo sucesivo, cada semigrupo M considerado será en particular un grupo finitamente presentado y en las factorizaciones consideradas (que siempre serán mallas) el conjunto de índices Λ será finito.

MST_2 generalizado

Sea $M = G = \langle X; R\rangle$ un grupo finitamente presentado actuando sobre dos conjuntos T y T'. Denotamos las acciones de G sobre T y T' por $\circ$ y $\bullet$, respectivamente.
Sea ahora $\alpha = [\alpha_\lambda]_{1\leq\lambda\leq s}$ una malla de T con $s \in \mathbb{N}$ y con bloques $\alpha_\lambda = [\alpha_{\lambda i}]_{i\in I_\lambda}$, $1 \leq \lambda \leq s$. Se define una aplicación

$$\begin{array}{rccc} \breve{\alpha}: & I_1 \times \cdots \times I_s & \longrightarrow & T \\ & (r_1, \ldots, r_s) & \longmapsto & \bigcirc_{\lambda=1}^{s} \alpha_{\lambda r_\lambda} \end{array}.$$

Considerar $f : T \longrightarrow T'$ una G-aplicación (i. e. $f(g \circ t) = g \bullet f(t)$ para todo $t \in T$, $g \in G$) tal que $\beta := [\alpha_1, \ldots, \alpha_{s-1}, f(\alpha_s)]$ sea una malla de T'. De modo análogo al caso anterior, se define la aplicación asociada

$$\breve{\beta} : I_1 \times \cdots \times I_s \longrightarrow T'.$$

También es necesario considerar un conjunto $\mathcal{T}$ que sirva de espacio de mensajes, y una aplicación (pública)

$$\tau : T' \longrightarrow S_{\mathcal{T}}$$

que asocie a cada $t' \in T'$ una permutación en $\mathcal{T}$.
Con esta notación se formula el esquema MST_2 generalizado.

Criptosistema MST_2 generalizado:

- **Clave pública:** $\breve{\alpha}$ y $\breve{\beta}$.
- **Clave privada:** la G-aplicación f.
- **Cifrado:** B pretende enviar un mensaje $m \in \mathcal{T}$ a A, para ello
  1. elige $(r_1, \ldots, r_s) \in I_1 \times \cdots \times I_s$ al azar,
  2. computa $y_1 := \breve{\alpha}(r_1, \ldots, r_s) \in T$, $y_2 := \breve{\beta}(r_1, \ldots, r_s) \in T'$ y
  3. envía el par $(y_1, \tau(y_2)(m)) \in T \times \mathcal{T}$ a A.
- **Descifrado:** De $y_2 = f(y_1)$, A obtiene

$$m = \tau(y_2)^{-1}(\tau(y_2)(m)).$$

Definición 3.10 *Diremos que un criptosistema de clave pública es un esquema $G - MST_2$ si es un caso particular del criptosistema MST_2 generalizado.*

Veamos algunos ejemplos de criptosistemas $G - MST_2$.

El esquema MST_2 es un esquema $G - MST_2$. Sea $f : G \longrightarrow H$ un epimorfismo de grupos. Sea $\circ$ la acción regular de G sobre sí mismo (i.e., G actúa sobre $T := G$ por multiplicación a izquierda) y considerar la acción de G sobre $T' := H$ vía $g \bullet h := f(g) \cdot h$ (el producto ordinario de $f(g)$ con $h \in H$). Notar que, en particular, f es una G-aplicación. Tomar $\mathcal{T} := H$ y $\tau : H \longrightarrow S_H$ definida por $\tau(h)(x) := x \cdot h$. Aplicar ahora estos parámetros al esquema general que acabamos de definir, considerando que, para hacer públicas las aplicaciones $\breve{\alpha}$, $\breve{\beta}$, el individuo A hará públicas las secuencias $\alpha = [\alpha_1, \ldots, \alpha_s]$ y $f(\alpha) = [f(\alpha_1), \ldots, f(\alpha_s)]$. Como la Definición 3.7 extiende la dada en [76], el esquema que se obtiene fijando estos elementos no es otro

que el esquema original MST_2 definido en [76]. Nótese que de lo anterior se deduce que el criptosistema ELGamal (que era un caso particular del MST_2 original) es también un esquema $G - MST_2$.

El criptosistema de Ko 3.3.2 es un esquema $G - MST_2$. Sea B_n el grupo de trenzas de $n = l + r$ cuerdas para algún $n \in \mathbb{N}$ adecuado, y denotemos por LB_l (resp. RB_r) el subgrupo de B_n generado por los $l - 1$ primeros (resp. $r - 1$ últimos) generadores de Artin.
Fijados $x \in B_n$ y $a \in LB_l$ adecuados (ver [63]), el grupo $G := RB_r$ actúa sobre los conjuntos $T := GxG^{-1}$ y $T' := aTa^{-1}$ por conjugación.
Además, la aplicación $f : T \longrightarrow T'$, $t \mapsto ata^{-1}$ es una G-aplicación, y puede comprobarse fácilmente que la secuencia $\alpha := [\alpha_1, \alpha_2]$, con $\alpha_1 := [b]_{b \in RB_r}$ y $\alpha_2 := [x]$, es una malla de T
En efecto, dado cualquier $t = b_0 x b_0^{-1} \in T$, para $b \in G$ se tiene $b \circ x = t$ si y sólo si $b^{-1}b_0 \in C_G(x)$ (i. e. $b^{-1}b_0 x = xb^{-1}b_0$), de donde se sigue que cada elemento de T admite exactamente $\mathrm{card}(C_G(x))$ factorizaciones respecto a α. De modo análogo puede verificarse que la secuencia $\beta := [\beta_1, \beta_2]$, con $\beta_1 := [b]_{b \in RB_r}$ y $\beta_2 := [f(x)]$, es una malla de T'.
Por último, se toma como espacio de mensajes $\mathcal{T}$ el conjunto de secuencias binarias de cierta longitud fijada k, y a través de una función hash ideal $h : B_n \longrightarrow \mathcal{T}$ se define

$$\begin{array}{rccc} \tau : & T' & \longrightarrow & S_{\mathcal{T}} \\ & b & \longmapsto & (m \mapsto h(b) \oplus m) \end{array},$$

donde $\oplus$ denota la operación habitual 'XOR' entre secuencias binarias.
Con estos elementos se obtiene, a partir del esquema MST_2 generalizado, una formulación del criptosistema de [63]:

- Clave pública: $\breve{\alpha}$ (descrita por x) y $\breve{\beta}$ (descrita por $f(x) = axa^{-1}$)
- Clave privada: f (que queda determinada por a)
- **Cifrado:** para enviar un mensaje $m \in \{0,1\}^k$ a A, B realiza los siguientes pasos:
  1. elige $b \in RB_r$ al azar,
  2. computa $y_1 = \breve{\alpha}(b, 1) = bxb^{-1}$, $y_2 = \breve{\beta}(b, 1) = baxa^{-1}b^{-1}$ y
  3. envía $(y_1, h(y_2) \oplus m)$ a A.
- **Descifrado:** A computa $y_2 = ay_1a^{-1}$ y obtiene

$$m = h(y_2) \oplus (h(y_2) \oplus m).$$

En definitiva; el esquema de [63] basado en grupos braid es un caso particular del MST_2 generalizado.

Los criptosistemas MOR 3.2 son esquemas $G - MST_2$ Sean K, K' grupos finitos y $\varphi : K \longrightarrow \mathrm{Aut}(K')$ un homomorfismo de grupos adecuado (ver [83]). Fijado un elemento $g \in K$, se define G como el subgrupo de $\mathrm{Aut}(K')$ generado por $\Phi := \varphi(g)$. Sea ahora $\Gamma = [\gamma_i]_{i\in I}$ una secuencia de generadores de K'. Se define el conjunto $T := G(\Gamma)$, esto es,

$$T := \big\{\Psi(\Gamma) = [\Psi(\gamma_i)]_{i\in I} | \Psi \in G\big\}.$$

Fijado un número entero a (secreto), se define el conjunto $T' := G^a(\Gamma)$, i.e.,

$$T' := \big\{\Psi^a(\Gamma) | \Psi \in G\big\}.$$

El grupo G actúa sobre el conjunto T mediante $\Phi^k \circ \Phi^i(\Gamma) := \Phi^{k+i}(\Gamma)$ y sobre T' a través de $\Phi^k \bullet (\Phi^a)^i(\Gamma) := (\Phi^a)^{i+k}(\Gamma)$.
Es claro que la aplicación $f : T \longrightarrow T', \Phi^i(\Gamma) \longmapsto (\Phi^i)^a(\Gamma)$ es una G-aplicación.
Tomar ahora como espacio de mensajes $\mathcal{T}$ el grupo K', y definir la aplicación $\tau : T' \longrightarrow S_{\mathcal{T}}$ como

$$\tau(\Phi^{ak}(\Gamma))(x) := \Phi^{ak}(x).$$

Sea $m = p_1 \cdots p_{s-1}$ el orden de Φ. Sea $m_1 = 1$, para $1 < \lambda \leq s-1$ se define

$$m_\lambda := \prod_{i=1}^{\lambda-1} p_i.$$

Considerar ahora la secuencia $\alpha := [\alpha_1, \ldots, \alpha_s]$ donde $\alpha_\lambda := [\Phi^{im_\lambda}]_{0\leq i\leq p_\lambda - 1}$ para $1 \leq \lambda \leq s-1$ y $\alpha_s := [\Gamma]$. Es fácil ver que α es una malla para T. De modo análogo, $\beta := [\alpha_1, \ldots, \alpha_{s-1}, f(\alpha_s)]$ es una malla para T'. Las aplicaciones correspondientes $\breve{\alpha}$ y $\breve{\beta}$ puede hacerse públicas revelando las secuencias $\alpha = [\alpha_1, \ldots, \alpha_s]$ y $f(\alpha) = [f(\alpha_1), \ldots, f(\alpha_s)]$.

El criptosistema que resulta del planteamiento anterior puede describirse como sigue:

- **Clave pública:** α y $f(\alpha)$, dadas por Φ y Φ^a. En efecto, para todo r menor que el orden de Φ, existen $r_1, \ldots, r_{s-1}$ tales que $r = \sum_{i=1}^{s-1} r_i m_i$. Ahora, $\breve{\alpha}(r_1, \ldots, r_{s-1}, 1) = \Phi^{r_1 m_1} \cdots \Phi^{r_{s-1} m_{s-1}}(\Gamma) = \Phi^{\sum_{i=1}^{s-1} r_i m_i}(\Gamma)$, que obviamente puede computarse a partir de Φ. Razónese análogamente para $f(\alpha)$.

- **Clave privada:** f (definido por a.)
- **Cifrado:** Se asume que se dispone de un algoritmo eficiente para expresar un elemento dado $x \in K'$ como una palabra en los generadores de Γ(ver [82, 83]). Así, pueden computarse de manera inmediata las permutaciones $\tau(\Phi^{ak}(\Gamma))$.

  Para enviar un mensaje m a A, el individuo B

  1. elige al azar enteros r_λ con $0 \leq r_\lambda \leq p_\lambda - 1$ $(1 \leq \lambda \leq s-1)$
  2. computa
  $$y_1 = \breve{\alpha}(r_1, \ldots, r_{s-1}, 1) = \Phi^b(\Gamma),$$
  $$y_2 = \breve{\beta}(r_1, \ldots, r_{s-1}, 1) = \Phi^{ab}(\Gamma),$$
  donde $b = \sum_{i=1}^{s-1} r_i m_i$,
  3. envía $(y_1, \Phi^{ab}(m))$ a A.
- **Descifrado:** A partir de $f(y_1) = \Phi^{ab}(\Gamma)$ A obtiene
$$m = \Phi^{-ab}(\Phi^{ab}(m)).$$

En definitiva, el criptosistema obtenido es el esquema MOR generalizado descrito en [83], por lo que los esquemas MOR [82, 83] pueden también verse como ejemplos del MST_2 generalizado.

Esquemas $G - MST_2$: análisis formal de seguridad

La seguridad del esquema $G - MST_2$ se basa en la dificultad de computar la componente y_2 de un texto cifrado a partir de la información pública. Como $\breve{\beta}$ es conocida para cualquier adversario, la secuencia $r_1, \ldots, r_s$ debe ser difícil de deducir a partir de y_1; en particular α debe ser *salvaje* según la Definición 3.9. Ya hemos razonado que además es necesario imponer a α otras condiciones que garanticen que factorizar la gran mayoría de los elementos respecto a α sea muy difícil. Obviamente es también necesario que la segunda componente del cifrado, $\tau(y_2)(m)$, no proporcione información significativa acerca del mensaje m. Estas cuestiones son difíciles de abordar debido a la generalidad de la definición del esquema, sin embargo, este marco resulta especialmente adecuado para realizar un análisis formal de seguridad de los esquemas $G - MST_2$, estudiando cómo se trasladan los resultados conocidos para algunos criptosistemas a otros esquemas $G - MST_2$.

Para empezar, veremos que la maleabilidad del criptosistema ElGamal es un problema que comparten muchos de los esquemas $G - MST_2$. Es muy sencillo

ver que el criptosistema ElGamal es maleable: dado un mensaje $m \in \mathcal{T}$ y su texto cifrado correspondiente $c := (y_1, m \cdot y_2)$ $(= (y_1, \tau(y_2)(m))$, un adversario que desconoce m siempre puede construir un cifrado válido c' del mensaje $m'm$, para cualquier mensaje $m' \in \mathcal{T}$ conocido. Para ello, sólo ha de multiplicar la segunda componente de c por m', obteniendo $c' := (y_1, m' \cdot m \cdot y_2)$.

En la notación del MST_2 generalizado, el adversario explota la igualdad $\tau(y_2)(m' \cdot m) = m' \cdot \tau(y_2)(m)$, (donde obviamente juega un papel importante el hecho de que $\mathcal{T}$ tenga estructura de grupo). Observar ahora que en los esquemas $G - MST_2$ vistos la aplicación τ también se define a través de una estructura de grupo en $\mathcal{T}$ (para el esquema de Ko y otros, basta identificar $\mathcal{T}$ con $(\mathbb{Z}/2\mathbb{Z})^k$). Ahora, dados $m, m' \in \mathcal{T}$ dos mensajes cualesquiera de los conjuntos adecuados, se tienen las siguientes igualdades:

ElGamal:	$\tau(y_2)(m' \cdot m)$	$=$	$m' \cdot \tau(y_2)(m)$,
$\boldsymbol{MST_2}$**:**	$\tau(y_2)(m' \cdot m)$	$=$	$m' \cdot \tau(y_2)(m)$,
Ko y otros:	$\tau(y_2)(m' \oplus m)$	$=$	$m' \oplus \tau(y_2)(m)$,
MOR:	$\tau(y_2)(m' \cdot m)$	$=$	$\tau(y_2)(m') \cdot \tau(y_2)(m)$.

Es decir, para todos estos esquemas es posible construir un cifrado del producto $m' \cdot m$ si se conoce m' y los cifrados correspondientes a m y m'. En particular, en el esquema MOR y si m no es el neutro del grupo, el adversario puede calcular un cifrado de m^2, mientras que en los demás ejemplos puede cifrar el producto $m' \cdot m$ para cualquier $m' \in \mathcal{T}$ conocido. En otras palabras, todos los esquemas $G - MST_2$ que acabamos de describir son inseguros según la noción NM-CPA (y, consecuentemente, también según NM-CCA1, NM-CCA2 e IND-CPA2). Cabe sin embargo la posibilidad de que otros esquemas $G - MST_2$ no presenten esta vulnerabilidad, para lo que seguramente será necesario imponer en la construcción que τ no sea compatible con la ley de grupo de $\mathcal{T}$, como ocurre en los ejemplos anteriores.

Con respecto a otras nociones de seguridad, es fácil ver que el esquema MST_2 original es también inseguro en el sentido de IND-CPA (no cumpliendo, por tanto, ninguno de los requerimientos de seguridad considerados estándares actualmente). Afortunadamente, el ataque con texto claro conocido que vulnera la indistinguibilidad del MST_2 no se traslada fácilmente a otros esquemas $G - MST_2$. Recordemos que el cifrado MST_2 de un mensaje m (elemento de un grupo finito H) viene dado por un par $(g, mf(g))$, donde g es un elemento de un grupo finito G y f es un isomorfismo entre G y H. Supongamos que un adversario selecciona dos mensajes m_0 y m_1, que son permutaciones de distinta paridad, y obtiene el cifrado $c = (g, m_b f(g))$ de uno de ellos. El

reto IND-CPA consiste ahora para él en decidir si el mensaje cifrado corresponde a m_0 o m_1. Para acertar correctamente, el adversario sólo tiene que estudiar la paridad de las componentes del cifrado, según indica la tabla siguiente:

m_0	m_1	g	$m_b f(g)$	m_b ?
par	impar	par	par	m_0
par	impar	impar	impar	m_0
par	impar	par	impar	m_1
par	impar	impar	par	m_1

Así, está claro que el esquema MST_2 original no es semánticamente seguro (IND-CPA). Sin embargo, este tipo de ataque no se traslada a otros esquemas $G - MST_2$, pues hace uso de la noción de paridad (específica de grupos de permutaciones). De hecho, se ha demostrado por ejemplo que la seguridad semántica del criptosistema ElGamal es equivalente a la dificultad del *problema de decisión de Diffie-Hellman* (ver [96]).

Desde un punto de vista constructivo, sería interesante diseñar una modificación del $G - MST_2$ que garantizase la seguridad de los criptosistemas correspondientes en cualquiera de las nociones estándar. Una construcción general de ese tipo ha sido propuesta por Cramer y Shoup [29, 30]: a partir de cualquier grupo abeliano y en condiciones muy generales, puede por tanto construirse un esquema IND-CCA2. Resta por tanto saber si tal construcción es factible a partir de grupos no abelianos. Estudiar desde este punto de vista posibles modificaciones del esquema MST_2 generalizado es sin duda un buen punto de partida.

Capítulo 4

Signaturas logarítmicas

En [69], S. Magliveras define ciertas factorizaciones de grupos finitos, llamadas signaturas logarímicas, que constituyen la base del criptosistema MST_1 (ver Definición 2.5). En el Capítulo 3 de esta Memoria hemos introducido una generalización de esta definición que, en particular, extiende la noción de signatura logarítmica a grupos infinitos. A continuación estudiamos algunas cuestiones acerca de signaturas logarítmicas que aparecen de modo natural al investigar sus aplicaciones criptográficas. Al hacer el análisis de seguridad del esquema MST_1 ya nos enfrentamos a preguntas del tipo ¿cuál es la longitud mínima de una signatura logarítmica de un cierto grupo G? o ¿cómo pueden construirse signaturas logarítmicas que induzcan factorizaciones *difíciles*? Volvemos ahora sobre algunas de estas cuestiones, aunque abstrayéndonos, en la medida de lo posible, del contexto criptográfico.

Este capítulo consta de dos partes bien diferenciadas. En la Sección 4.1, exploramos la posibilidad de construir signaturas logarítmicas de *longitud mínima* para varias familias de grupos finitos. Por otra parte, en la Sección 4.2 construimos explícitamente una signatura logarítmica para la que, bajo la hipótesis de que cierto problema de teoría de nudos es muy complejo, no existe un algoritmo polinomial que compute su factorización inducida. Así, dicha signatura logarítmica es *salvaje* según la Definición 3.9. Además del indiscutible valor que este tipo de resultados tienen en la práctica, desde el punto de vista teórico constituyen problemas de gran interés en sí mismos.

4.1. Signaturas logarítmicas de longitud mínima

En el análisis de seguridad del esquema MST_1 (ver 2.3.2), señalábamos la necesidad de disponer de signaturas logarítmicas *cortas* para que las claves del esquema tuviesen un tamaño razonable. Pero, ¿para qué grupos podemos asegurar que existen signaturas logarítmicas de longitud mínima? y, ¿cómo podemos construirlas? A lo largo de esta sección trataremos de dar una respuesta lo más general posible a estas preguntas.

Manejaremos la definición original de signatura logarítmica (2.5), y no la definición más general de (3.8), pues las factorizaciones que nos interesan van asociadas a grupos finitos. Así, a lo largo de esta sección consideraremos que una signatura logarítmica de un grupo de permutaciones G de grado n y orden $|G| = r_1 \cdots r_s$, es una secuencia $\alpha = [\alpha_1, \ldots, \alpha_s]$ tal que cada elemento $g \in G$ se representa de manera única como un producto

$$g = \alpha_{1j_1} \cdots \alpha_{sj_s},$$

con $\alpha_{ij_i} \in \alpha_i, 1 \leq i \leq s$. Cada bloque α_i de α es a su vez una secuencia en S_n de longitud r_i. Escribiremos además α^G para indicar que α es una signatura logarítmica del grupo G.

Recordar además que, de acuerdo con la Definición 2.5, llamábamos *longitud de* α al entero $\ell(\alpha) = \sum_{i=1}^{s} r_i$. En 2.3.2 razonábamos que si $|G| = \prod_{j=1}^{t} p_j^{a_j}$, con $p_1, \ldots, p_t$ números primos distintos, entonces $\ell(\alpha) \geq \sum_{j=1}^{t} a_j p_j$. A partir de ahora denotaremos por $\mathcal{B}(G)$ a la cota inferior para la longitud de las signaturas logarítmicas de un grupo G de orden $\prod_{j=1}^{t} p_j^{a_j}$, i.e.,

$$\mathcal{B}(G) := \sum_{j=1}^{t} a_j p_j.$$

Nuestro objetivo es estudiar para qué grupos finitos existen signaturas logarítmicas de longitud mínima. Algunos ejemplos triviales son la signatura logarítmica *vacía* $\alpha^{\{e\}} := []$ del grupo formado por un único elemento $\{e\}$ o la signatura logarítmica $\alpha^{C_p} := [[g, g^2, \ldots, g^p]]$ del grupo cíclico de orden primo $C_p = \langle g \rangle$.

4.1.1. Primeros ejemplos

Grupos abelianos. Es muy sencillo ver que para todo grupo abeliano finito existe una signatura logarítmica de longitud mínima. Sea H un grupo

cíclico de orden p^m, para algún primo p, y a cualquier elemento que genere H. Claramente,

$$\alpha = [\alpha_1, \ldots, \alpha_m] \text{ con } \alpha_i = [e_H, a^{p^{i-1}}, \ldots, a^{(p-1)p^{i-1}}],\ i = 1, \ldots, m$$

es una signatura logarítmica de H de longitud $m \cdot p = \mathcal{B}(H)$.
Ahora, si G es un grupo abeliano, basta yuxtaponer signaturas logarítmicas de longitud mínima asociadas a los sumandos cíclicos correspondientes a la descomposición de G según sus divisores elementales para construir una signatura logarítmica de G de longitud $\mathcal{B}(G)$.

Nota: A partir de signaturas logarítmicas de longitudes l_1 y l_2 para grupos G_1 y G_2, podemos construir una signatura logarítmica de longitud $l_1 + l_2$ para cualquier extensión de G_1 por G_2. En particular, si disponemos de una signatura logarítmica de longitud mínima para dos grupos G_1 y G_2 podemos construir una signatura logarítmica de longitud mínima para cualquier extensión de G_1 por G_2.

Cuando consideramos grupos abelianos, la construcción que veremos a continuación para grupos resolubles nos proporciona, en general, signaturas logarítmicas distintas a las se obtienen por el método que acabamos de ver.

Grupos resolubles. Sea G un grupo resoluble de orden $\prod_{i=1}^{k} p_i$ donde $p_1, \ldots, p_k$ son números primos no necesariamente distintos. Por ser G resoluble, podemos encontrar una serie de composición para G:

$$\gamma :\equiv G = G_0 > G_1 > \ldots > G_k = \{e_G\}$$

donde cada índice $|G_{i-1} : G_i|$ es primo y $\prod_{i=1}^{k} |G_{i-1} : G_i| = \prod_{i=1}^{k} p_i$. Es obvio que cualquier signatura logarítmica transversal exacta asociada a γ tiene longitud mínima $\left(\sum_{i=1}^{k} p_i\right)$.

Grupo alternado. Grupo simétrico. En la demostración de la Proposición 2.11 construíamos explícitamente signaturas logarítmicas de longitud mínima para los grupos A_n y S_n, para $n > 5$. Puede hacerse una demostración alternativa utilizando también inducción en n, veámoslo:
El resultado es obvio para $n \leq 4$, por ser S_n resoluble en esos casos. Supongamos ahora que hemos construido una signatura logarítmica

$$\alpha^{n-1} = [\alpha_1^{n-1}, \ldots, \alpha_m^{n-1}]$$

para S_{n-1} de longitud $\mathcal{B}(S_{n-1})$. Notar que eligiendo elementos $g_i \in S_n$ tales que $g_i(i) = n$, $i = 1, \ldots, n$ tenemos

$$S_n = \bigcup_{i=1}^{n} S_{n-1} g_i,$$

(donde gh denota a la permutación construida aplicando primero h y después g). Ahora, si n se factoriza como un producto $k = p_1 \cdots p_r$, con $p_1, \ldots, p_r$ números primos no necesariamente distintos, construiremos r nuevos bloques, $\alpha_1, \ldots, \alpha_r$ para completar α^{n-1}. La longitud de cada bloque $|\alpha_i|$ será p_i y además, será posible, para $1 \leq i \leq n$, construir una permutación g_i multiplicando un elemento de cada bloque tal que $g_i(i) = n$.
Para ello, tomar

$$\alpha_1 := \{\mathrm{id}, (1, n), (2, n), \ldots, (p_1 - 1, n)\}.$$

Después, construir el bloque α_2 de modo que, a través de sus elementos, sea posible asignar a cada uno de los puntos $i \in \{p_1, 2p_1, \ldots, (p_2 - 1)p_1\}$ el punto n, y a los puntos j, $p_1 < j < p_1 p_2$, un punto en el conjunto $\{1, \ldots, p_1 - 1\}$. Tomar, por ejemplo

$$\begin{aligned} \alpha_2 := [\quad & \mathrm{id}, \\ & (n, p_1) \cdot \textstyle\prod_{i=1}^{p_1 - 1} (i, p_1 + i), \\ & (n, 2p_1) \cdot \textstyle\prod_{i=1}^{p_1 - 1} (i, 2p_1 + i), \\ & \vdots \\ & (n, (p_2 - 1)p_1) \cdot \textstyle\prod_{i=1}^{p_1 - 1} (i, (p_2 - 1)p_1 + i)]. \end{aligned}$$

Así, la secuencia $[\alpha_1, \alpha_2]$ nos permite construir permutaciones $g_i \in S_n$ tales que $g_i(i) = n$ para $1 \leq i \leq p_1 p_2 - 1$. Definiendo $\alpha_3, \ldots, \alpha_{r-1}$ de modo análogo, concluímos construyendo el bloque α_r de modo que por medio de alguna permutación suya pueda asignarse a cada punto

$$\{p_1 p_2 \cdots p_{r-1}, 2p_1 p_2 \cdots p_{r-1}, \ldots, (p_r - 1)p_1 \cdots p_{r-1}\}$$

el punto n directamente, y a cualquier otro punto j, $p_1 \cdots p_{r-1} < j < n$, un punto del conjunto $\{1, \ldots, p_1 p_2 \cdots p_{r-1} - 1\}$. Por ejemplo:

$$\begin{aligned} \alpha_r := [\quad & \mathrm{id}, \\ & (k, p_1 p_2 \cdots p_{r-1}) \cdot \textstyle\prod_{i=1}^{p_1 p_2 \cdots p_{r-1} - 1} (i, p_1 p_2 \cdots p_{r-1} + i), \\ & (k, 2p_1 p_2 \cdots p_{r-1}) \cdot \textstyle\prod_{i=1}^{p_1 p_2 \cdots p_{r-1} - 1} (i, 2p_1 p_2 \cdots p_{r-1} + i), \\ & \vdots \\ & (k, (p_r - 1)p_1 p_2 \cdots p_{r-1}) \cdot \textstyle\prod_{i=1}^{p_1 p_2 \cdots p_{r-1} - 1} (i, (p_r - 1)p_1 p_2 \cdots p_{r-1} + i)]. \end{aligned}$$

Es fácil comprobar que $\alpha^n := [\alpha_1^{n-1}, \ldots, \alpha_m^{n-1}, \alpha_1, \ldots, \alpha_r]$ es una signatura logarítmica de S_n de longitud $\mathcal{B}(S_n)$.

El resultado análogo para el grupo alternado es obra de S. Magliveras y aparece publicado en [70]. Esta demostración también utiliza un proceso inductivo de construcción:
Si suponemos construida una signatura logarítmica de longitud mínima β^{n-1} para el grupo A_{n-1}, podemos considerar una factorización de A_n como producto conjuntista de un subgrupo H por A_{n-1}. Si n es impar, H es el grupo generado por la permutación $\pi = (1, 2, \ldots, n)$. Si $n = 2m$, $m \in \mathbb{N}$, H es el grupo generado por

$$\sigma = (1, 2, \ldots, m)(m+1, m+2, \ldots, n)$$

y

$$\tau = (1, m+1)(2, m+2) \cdots (m, n).$$

En cualquier caso, H es resoluble y por tanto posee una signatura logarítmica de longitud mínima β^H. Ahora, $\beta^n := [\beta^H, \beta^{n-1}]$ es una signatura logarítmica de A_n de longitud $\mathcal{B}(A_n)$.

Los resultados anteriores relativos a los grupos A_n y S_n siguen una técnica que explota la estructura de los grupos de permutaciones: se identifica un punto i cuyo estabilizador G_i pueda factorizarse por medio de una signatura logarítmica de longitud mínima. Después, se busca un conjunto completo de representantes de G módulo G_i adecuado, cuyos elementos muevan i a cualquier punto j tal que exista $g \in G$ con $g(i) = j$.

Notas:

- Observar que en los ejemplos anteriores se repite el siguiente método de construcción: se factoriza el grupo de base en partes disjuntas y luego se construye una signatura logarítmica de longitud mínima para G yuxtaponiendo secuencias de factorización adecuadas de esas partes disjuntas. Por ejemplo, sabemos que la cota se alcanza si disponemos de una cadena de subgrupos de G de modo que al sumar los índices asociados (de cada subgrupo en el siguiente) se obtenga exactamente $\mathcal{B}(G)$.
Por otro lado, puede darse el caso de que las partes disjuntas que se consideran sean subgrupos. La forma más general de formalizar la descomposición de un grupo en subgrupos disjuntos es el llamado producto de Zappa-Szép [79]. Este producto generaliza los productos directo y semidirecto de grupos. Además, el producto de Zappa-Szép formaliza otras situaciones, como el caso de que un grupo G pueda escribirse como producto (conjuntista) de dos subgrupos propios no normales.

4.1.2. El producto de Zappa-Szép

Definimos a continuación el producto de Zappa-Szép, razonando luego que si un grupo G es producto de Zappa-Szép de dos subgrupos suyos, podemos yuxtaponer signaturas logarítmicas de dichos subgrupos para construir una signatura logarítmica para G. Como antes, si las signaturas logarítmicas de los subgrupos son de longitud mínima, también lo es la signatura logarítmica de G construida.

Definición 4.1 *Sean K y H dos grupos finitos. Un par de aplicaciones $\kappa : H \times K \longrightarrow K$, $\hbar : K \times H \longrightarrow H$, se dice* par de acciones automórficamente ligadas para (K, H), *si se cumplen las siguientes condiciones:*

1. *la aplicación*
$$\begin{array}{rccc} \psi_\kappa : & H & \longrightarrow & S_K \\ & h & \longmapsto & \kappa_h(\cdot) := \kappa(h, \cdot) \end{array}$$
es un homomorfismo de grupos,

2. *la aplicación*
$$\begin{array}{rccc} \phi_\hbar : & K & \longrightarrow & S_H \\ & k & \longmapsto & \hbar_k(\cdot) := \hbar(k, \cdot) \end{array}$$
es un anti-homomorfismo de grupos, esto es, $\hbar_{e_K} = \mathrm{id}_H$ y
$\forall\, k_1, k_2 \in K :\ \hbar_{k_1}\hbar_{k_2} = \hbar_{k_2 k_1}$,

3. $\forall\, k_1, k_2 \in K, h \in H :\ \kappa_h(k_1 k_2) = \kappa_h(k_1)\kappa_{\hbar_{k_1}(h)}(k_2)$, *y*

4. $\forall\, k_1, k_2 \in K, h \in H :\ \hbar_k(h_1 h_2) = \hbar_{\kappa_{h_2}(k)}(h_1)\hbar_k(h_2)$.

Definición 4.2 *Sean ahora K y H grupos y $(\kappa, \hbar)$ un par de acciones automórficamente ligadas para (K, H). El conjunto $K \times H$ puede dotarse de estructura de grupo considerando el producto*

$$(k_1, h_1) \cdot (k_2, h_2) := (k_1 \kappa_{h_1}(k_2), \hbar_{k_2}(h_1) h_2),$$

siendo entonces el elemento unidad (e_K, e_H). Dicho grupo recibe el nombre de producto ligado o producto de Zappa-Szép de K y H, *y se denota $K \times_{(\kappa,\hbar)} H$.*

Es claro que $K \times \{e_H\}$ y $\{e_K\} \times H$ son subgrupos de $K \times_{(\kappa,\hbar)} H$, isomorfos respectivamente a K y H. Además, si $\psi_\kappa \equiv \mathrm{id}_K$ entonces $\{e_K\} \times H$ es un subgrupo normal de $K \times_{(\kappa,\hbar)} H$, y por tanto el producto de Zappa-Szép es en particular un producto semidirecto. Análogamente, si $\phi_\hbar \equiv e_H$. Si se dan ambas condiciones a la vez, $K \times_{(\kappa,\hbar)} H$ es producto directo de $K \times \{e_H\}$ y $\{e_K\} \times H$.

Veamos ahora que a partir de signaturas logarítmicas de longitud mínima para H y G pueden construirse signaturas logarímicas para $K \times_{(\kappa,\hbar)} H$.

Proposición 4.3 *Sea G un producto de Zappa-Szép de dos grupos finitos K y H, y suponer que existen signaturas logarítmicas α^K de K y α^H de H tales que $l(\alpha^K) = \mathcal{B}(K)$ y $l(\alpha^H) = \mathcal{B}(H)$. Entonces existe una signatura logarítmica α^G de G con $l(\alpha^G) = \mathcal{B}(G)$.*

Demostración: Sean

$$\alpha^K = [\alpha_1^K, \ldots, \alpha_s^K], \quad \alpha_i^K = [\alpha_{i0}^K, \ldots, \alpha_{ir_i-1}^K] \;\; i = 1, \ldots, s$$

y

$$\alpha^H = [\alpha_1^H, \ldots, \alpha_t^H], \quad \alpha_i^H = [\alpha_{i0}^H, \ldots, \alpha_{il_i-1}^H] \;\; i = 1, \ldots, t$$

donde $s, t \in \mathbb{N}_0$.
Claramente

$$\alpha^G = [\alpha_1^G, \ldots, \alpha_s^G, \alpha_{s+1}^G, \ldots, \alpha_{s+t}^G]$$

con

$$\begin{array}{rcll} \alpha_i^G & = & [(\alpha_{i0}^K, e_H), \ldots, (\alpha_{ir_i-1}^K, e_H)] & i = 1, \ldots, s, \\ \alpha_{s+i}^G & = & [(e_K, \alpha_{i0}^H), \ldots, (e_K, \alpha_{il_i-1}^H)] & i = 1, \ldots, t, \end{array}$$

es una signatura logarítmica de G de longitud mínima. □

De manera similar es fácil ver que, por ejemplo, un grupo G tiene una signatura logarítmica de longitud mínima si posee una serie subnormal cuyos factores cumplen esa condición. Así, por ejemplo, sabemos que existen signaturas logarítmicas de longitud mínima para los grupos *casi resolubles*. Un grupo es *casi resoluble* si posee una serie subnormal cuyos factores son grupos de Möbius. Utilizar el razonamiento anterior y el hecho de que los grupos de Möbius son todos resolubles, excepto A_5 (para el cual sabemos que existen signaturas logarímicas de longitud mínima).

Nota: Por lo anterior, es claro que si existieran grupos para los que no hay signaturas logarítmicas de longitud mínima, el grupo G de menor orden en estas condiciones sería simple (pues de tener un subgrupo normal H podríamos construir una signatura logarítmica de longitud mínima para G yuxtaponiendo signaturas logarítmicas de H y G/H).

4.1.3. Grupos simples

La nota anterior justifica sobradamente el interés que tiene estudiar las factorizaciones de grupos simples en este contexto: para demostrar que existe una signatura logarítmica de longitud mínima para todo grupo finito, bastaría demostrar el resultado para todo grupo simple.

Aunque los grupos simples finitos son objetos matemáticos tremendamente complejos, están relativamente bien estudiados. El teorema de clasificación de los grupos finitos (ver [52]) es uno de los grandes resultados matemáticos de nuestro siglo. Dicho teorema establece la existencia de (exactamente) 18 familias infinitas de grupos simples: los grupos cíclicos de orden primo, los grupos alternados y las 16 clases de grupos de Lie. Además, existen 26 grupos simples esporádicos.

Se han realizado numerosos estudios acerca de la factorización de grupos simples finitos como producto de dos subgrupos propios (ver, por ejemplo el artículo monográfico [68]). Las factorizaciones que nosotros buscamos son, desde luego, mucho más generales: los elementos de cada bloque de una signatura logarítmica forman a priori un conjunto sin estructura. Sin embargo, es claro que podemos utilizar las factorizaciones de grupos simples conocidas para construir signaturas logarítmicas de longitud mínima.

Notación: En lo sucesivo, consideraremos que el producto $(\sigma\pi) \in S_n$ de dos permutaciones $\sigma, \pi \in S_n$ es la permutación que asigna cada $i \in \{1, \ldots, n\}$ a $\pi(\sigma(i))$. Esta notación viene impuesta por el uso de los sistemas de álgebra computacional GAP [93] y Magma [25].

Factorizaciones de algunos grupos simples de Lie. En [59] Holt y Rowley estudian qué grupos pueden descomponerse como producto de sus subgrupos de Sylow. En concreto, tratan de responder a esta pregunta: *dado un grupo G, cuyo orden $|G|$ tiene exactamente r divisores primos $p_1, \ldots, p_r$, ¿es posible elegir subgrupos $P_i \in Syl_{p_i}(G)$, $i = 1, \ldots, r$ tales que $G = P_1 \cdots P_r$?* El producto que se considera es conjuntista, por lo que puede verse como un producto de Zappa-Szép de varios factores.
Como los subgrupos de Sylow son resolubles, es claro que un grupo que admita dicha factorización posee signaturas logarítmicas de longitud mínima. Así, a partir de los resultados de Holt y Rowley se demuestra que para todo q potencia de primo, los grupos simples

- $\mathrm{PSL}_2(q), q \leq 3$, y
- $\mathrm{PSL}_3(q)$ para $q \not\equiv 1 \bmod 3$,

poseen signaturas logarítmicas de longitud mínima. Concretamente, en [59] se demuestra que pueden elegirse subgrupos de Sylow de los grupos $\mathrm{PSL}_2(q)$, $q \leq 3$ y $\mathrm{PGL}_3(q)$ ($q \not\equiv 1 \bmod 3$), de modo que dichos grupos se factoricen como un producto de sus subgrupos de Sylow elegidos. Así, para $q \not\equiv 1 \bmod 3$, el resultado se sigue para los grupos simples $\mathrm{PSL}_3(q) \simeq \mathrm{PGL}_3(q)$.

Nota: Este resultado permite construir signaturas logarítmicas de longitud mínima para otros grupos no simples:

- los grupos proyectivos lineares de mock $\mathrm{PML}_2(q)$ (donde q es una potencia par de un primo impar),
- los grupos generales lineales $\mathrm{GL}_2(q), q \geq 3$.

En efecto, como $\mathrm{PSL}_2(q)$ es un subgrupo de $\mathrm{PML}_2(q)$ de índice dos (ver el Capítulo XI de [20] y [1]), $\mathrm{PML}_2(q)$ es una extensión de $\mathrm{PSL}_2(q)$ por un grupo cíclico de orden dos. Razonamos análogamente, por ser $\mathrm{GL}_2(q)$ una extensión cíclica de $\mathrm{SL}_2(q)$, y $\mathrm{PSL}_2(q)$ isomorfo al cociente de $\mathrm{SL}(2,q)$ por su centro.

Hemos comprobado que también algunos grupos $\mathrm{PSL}_3(q)$, con $q \equiv 1 \bmod 3$ pueden factorizarse como producto de algunos de sus subgrupos de Sylow.

P_2	$=$	$\langle (2,15)(3,10,12,7)(4,11,21,13)(5,17)(6,8,20,9)(14,18,16,19),$ $(2,5)(3,19,8,13)(4,10,14,20)(6,21,7,16)(9,11,12,18)(15,17),$ $(2,5)(3,8)(4,16)(6,20)(7,10)(9,12)(14,21)(15,17),$ $(3,14)(4,8)(6,19)(7,13)(9,21)(10,11)(12,16)(18,20)\rangle \leq \mathrm{PSL}_3(4)$
P_3	$=$	$\langle (1,12,5)(2,8,7)(3,13,15)(4,10,19)(6,14,18)(9,16,17),$ $(1,5,12)(2,13,9)(3,17,7)(6,14,18)(8,15,16)$ $(11,21,20)\rangle \leq \mathrm{PSL}_3(4)$
P_5	$=$	$\langle (1,5,17,2,15)(3,21,4,8,6)(7,11,12,9,19)$ $(13,16,20,14,18)\rangle \leq \mathrm{PSL}_3(4)$
P_7	$=$	$\langle (1,20,18,10,8,12,19)(2,13,6,4,17,15,3)$ $(5,21,16,9,14,11,7)\rangle \leq \mathrm{PSL}_3(4)$

Cuadro 4.1: factorización de $\mathrm{PSL}_3(4) = P_7P_3P_2P_5$

Por ejemplo, representando $\mathrm{PSL}_3(4)$ como el subgrupo de S_{21} generado por

$$\alpha = (4,11,20)(5,15,17)(6,16,18)(7,14,13)(8,12,9)(10,21,19) \quad \text{y}$$
$$\beta = (1,8,21,16,15,3,2)(4,10,20,18,17,9,7)(5,12,11,14,19,13,6),$$

(ver [99])y eligiendo los subgrupos de Sylow P_2, P_3, P_5, P_7 descritos en el Cuadro 4.1, podemos expresar $\mathrm{PSL}_3(4)$ como un producto $\mathrm{PSL}_3(4) = P_7P_3P_2P_5$. Dicha factorización puede comprobarse usando un sistema de álgebra computacional.

Pueden utilizarse factorizaciones similares para construir signaturas logarítmicas de longitud mínima para los grupos $\mathrm{PSU}_4(2)$ y $\mathrm{PSU}_3(4)$. Representando $\mathrm{PSU}_4(2)$ como el subgrupo de S_{45} generado por

$$\begin{aligned}
\gamma &= (2,5,3)(4,12,7)(6,17,10)(8,21,14)(9,19,13)(11,29,20)(16,30,25)\\
&\quad (18,28,27)(22,26,32)(23,36,31)(33,35,39)(34,37,41)(40,42,43) \quad \text{y}\\
\delta &= (1,2,4,8,15,24)(3,6,11,21,31,37)(5,9,16,14,23,34)\\
&\quad (7,13,22,33,40,20)(10,18,25)(12,17,26,35,42,30)(19,28,29)\\
&\quad (32,38,43)(36,41,45)(39,44),
\end{aligned}$$

ver [99] y eligiendo los subgrupos de Sylow P_2, P_3, P_5 del Cuadro 4.2, podemos expresar $\mathrm{PSU}_4(2)$ como el producto $\mathrm{PSU}_4(2) = P_3P_2P_5$.

El mismo método puede aplicarse con éxito al grupo $\mathrm{PSU}_3(4)$. Usando su representación [99] como el subgrupo de S_{65} generado por

$$\begin{aligned}
\epsilon &= (2,9,50,12,61,38,14,3,15,4,27,63,52,23,6)\\
&\quad (5,32,21,10,53,33,18,11,55,45,22,47,30,16,8)\\
&\quad (7,62,39,29,51,25,60,17,43,42,49,44,28,34,36)\\
&\quad (13,48,57,59,46,41,24,20,37,54,58,35,40,19,26)(56,64,65)\text{y}\\
\zeta &= (1,2,4,8,18,39,26,48,59,44,22,21,6,15,31)\\
&\quad (3,7,10,23,33,47,27,16,34,9,20,41,61,40,58)\\
&\quad (5,12,17,37,55,54,60,38,32,56,28,52,63,62,64)\\
&\quad (11,25,43,45,51,46,49,50,57,65,13,14,29,19,42)(24,35,36),
\end{aligned}$$

factorizamos $\mathrm{PSU}_3(4)$ como un producto $P_{13}P_5P_2P_3$, donde P_{13}, P_5, P_2 y P_3 son sus subgrupos de Sylow especificados en el Cuadro 4.3.

Aunque este método de factorización ha sido muy útil para demostrar la existencia de signaturas logarítmicas de longitud mínima de algunos grupos, es necesario resaltar que, hasta la fecha, desconocemos si puede aplicarse con éxito a otros grupos simples de Lie. Sin embargo, sabemos que existen grupos para los que el método no puede aplicarse, ya que en [59] Holt and Rowley demuestran que el grupo $PSU_3(3)$ no puede factorizarse como producto de sus subgrupos de Sylow.

P_2	$=$	$\langle (1,30,45,23)(2,28,3,42)(4,17,16,19)(5,11)(6,9,26,35)$ $(7,43,20,27)(8,31,41,12)(10,44,39,18)(13,40,32,38)$ $(14,36,37,29)(15,24,34,21)(22,33)$, $(1,15,17,6)(2,44,31,13)(3,32,12,18)(4,40,23,39)(5,22)(7,20)$ $(8,9,42,24)(10,30,38,16)(11,33)(14,37)(19,34,45,26)$ $(21,28,35,41)(27,29)(36,43)$, $(1,21,17,35)(2,10,31,38)(3,40,12,39)(4,32,23,18)(5,33)$ $(6,41,15,28)(7,20)(8,26,42,34)(9,19,24,45)(11,22)$ $(13,16,44,30)(14,37)(27,36)(29,43)\rangle \leq \mathrm{PSU}_4(2)$
P_3	$=$	$\langle (1,27,3,19,16,12,11,34,25)(2,32,26,17,8,21,33,24,36)$ $(4,30,22,37,43,45,10,35,31)(5,41,29)$ $(6,20,44,28,13,14,40,38,23)(7,42,18)(9,39,15)$, $(2,44,34)(3,40,24)(4,22,30)(5,42,41)(6,32,12)(7,18,39)$ $(8,25,28)(9,15,29)(10,31,35)(13,20,38)(14,27,17)(16,33,23)$ $(21,36,26)(37,45,43)\rangle \leq \mathrm{PSU}_4(2)$
P_5	$=$	$\langle (1,30,40,34,20)(2,42,21,18,37)(3,23,10,13,7)(4,35,15,27,17)$ $(5,11,33,22,25)(6,14,45,28,9)(8,38,26,36,19)(12,16,24,32,43)$ $(29,31,41,39,44)\rangle \leq \mathrm{PSU}_4(2)$

Cuadro 4.2: factorización $\mathrm{PSU}_4(2) = P_3P_2P_5$

Grupos de Mathieu. Los cinco grupos de Mathieu son grupos esporádicos simples construidos hace poco más de un siglo. Es habitual representarlos como grupos de autormorfismos de ciertos sistemas de Steiner, aunque nosotros utilizaremos su representación como simetrías de ciertas geometrías proyectivas. A través de esa representación utilizamos para los grupos de Mathieu una técnica similar a la que usamos para construir signaturas logarítmicas de longitud mínima para el grupo simétrico y alternado; descomponer cada grupo en producto de dos factores haciendo un cociente del mismo por el estabilizador de un punto adecuado.

I. $\mathbf{M_{11}}$ y $\mathbf{M_{12}}$. Empezamos por describir una construcción de los grupos de Mathieu M_{11} y M_{12} a través de la geometría proyectiva $\mathrm{PG}(2,\mathbb{F}_{11})$.

$$
\begin{aligned}
P_2 \quad = \quad & \langle (1,40,29,21)(2,23,15,20)(3,49,34,12)(4,27,46,54) \\
& (5,50,30,26)(6,32,47,38)(7,35,65,25)(8,41,62,13) \\
& (9,44,36,52)(10,28,45,56)(11,53,59,17)(14,61,24,63) \\
& (16,19,43,37)(22,60,48,31)(33,39,58,55)(42,51,64,57), \\
& (1,27,24,6)(2,8,22,37)(3,51,52,33)(4,21,38,61)(5,25,45,11) \\
& (7,50,53,56)(9,64,49,39)(10,59,30,35)(12,55,36,42) \\
& (13,20,43,31)(14,47,29,54)(15,62,48,19)(16,60,41,23) \\
& (17,28,65,26)(32,63,46,40)(34,57,44,58), \\
& (1,2,14,48)(3,30,44,45)(4,16,32,13)(5,52,10,34)(6,62,54,37) \\
& (7,42,17,39)(8,27,19,47)(9,28,12,50)(11,58,35,51) \\
& (15,24,22,29)(20,63,60,21)(23,61,31,40)(25,57,59,33) \\
& (26,36,56,49)(38,41,46,43)(53,55,65,64), \\
& (1,57,14,33)(2,35,48,11)(3,6,44,54)(4,12,32,9)(5,37,10,62) \\
& (7,31,17,23)(8,30,19,45)(13,26,16,56)(15,25,22,59) \\
& (20,65,60,53)(21,42,63,39)(24,58,29,51)(27,34,47,52) \\
& (28,41,50,43)(36,46,49,38)(40,64,61,55)\rangle \leq \mathrm{PSU}_3(4)
\end{aligned}
$$

$$
\begin{aligned}
P_3 \quad = \quad & \langle (1,21,22)(2,14,40)(3,62,42)(4,65,49)(5,41,25)(6,33,56) \\
& (7,9,32)(8,55,44)(10,13,11)(12,38,53)(15,29,61)(16,59,30) \\
& (17,36,46)(19,39,34)(23,31,60)(24,63,48)(26,47,51) \\
& (27,57,28)(35,45,43)(37,64,52)(50,54,58)\rangle \leq \mathrm{PSU}_3(4)
\end{aligned}
$$

$$
\begin{aligned}
P_5 \quad = \quad & \langle (1,40,52,42,30)(2,51,60,38,8)(3,55,10,24,63) \\
& (4,37,22,33,23)(5,29,21,44,64)(6,53,43,28,9)(7,13,26,49,27) \\
& (12,54,65,41,50)(14,61,34,39,45)(15,57,31,32,62) \\
& (16,56,36,47,17)(19,48,58,20,46), \\
& (1,47,54,5,15)(2,51,60,38,8)(3,53,49,34,22)(4,24,9,13,14) \\
& (6,26,61,37,63)(7,45,23,10,28)(11,59,25,35,18) \\
& (12,64,62,30,36)(16,41,21,31,52)(17,65,29,57,40) \\
& (19,58,46,48,20)(27,39,33,55,43) \\
& (32,42,56,50,44)\rangle \leq \mathrm{PSU}_3(4)
\end{aligned}
$$

$$
\begin{aligned}
P_{13} \quad = \quad & \langle (1,3,48,30,53,6,58,18,60,56,16,55,38) \\
& (2,65,21,61,13,46,45,19,10,59,42,8,64) \\
& (4,54,14,50,27,44,26,15,22,34,25,29,32) \\
& (5,47,35,28,39,51,31,37,36,9,62,33,20) \\
& (7,49,24,43,40,57,23,63,17,52,12,41,11)\rangle \leq \mathrm{PSU}_3(4)
\end{aligned}
$$

Cuadro 4.3: factorización de $\mathrm{PSU}_3(4) = P_{13}P_5P_2P_3$

Denotamos por

$$P = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, \infty\}$$

al conjunto de puntos de dicha geometría. Esta representación de $\mathrm{PG}(2, \mathbb{F}_{11})$ no es la habitual, pero será especialmente útil en nuestro desarrollo.

Representaremos M_{11} y M_{12} como grupos de permutaciones de los puntos de la geometría $\mathrm{PG}(2, \mathbb{F}_{11})$. Para ello, considerar las siguientes aplicaciones (ver Capítulo 2 de [51])

$$f : x \mapsto x + 1, \quad g : x \mapsto -\frac{1}{x}, \quad h : x \mapsto 4x^2 - 3x^7.$$

La aplicación f hará corresponder a cada punto de $i \in \{0, 1 \ldots, 10\}$ el punto $i + 1$ módulo 11, y dejará fijo el punto del infinito ∞. Por otro lado, la aplicación g intercambia los puntos 0 e ∞ y hace corresponder a cada punto distinto de 0 y de ∞ el opuesto de su inverso módulo 11. Por último, h deja fijos el 0 y el ∞ y mueve cada punto restante i al punto $4i^2 - 3i^7$ módulo 11. Así, cada una de estas aplicaciones induce una permutación en el conjunto P; explícitamente:

$$\begin{array}{rcl} f \sim \sigma & = & (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10), \\ g \sim \tau & = & (0, \infty)(1, 10)(2, 5)(3, 7)(4, 8)(6, 9), \\ h \sim \pi & = & (2, 6, 10, 7)(3, 9, 4, 5). \end{array}$$

El grupo de Mathieu M_{12} es el subgrupo de S_P generado por σ, τ y π. Es un grupo simple sharply 5-transitivo en P. Por otra parte, el grupo de Mathieu M_{11} puede definirse como el estabilizador en M_{12} del punto 0 [86]. Los órdenes de estos grupos son $|M_{11}| = 7.920$ y $|M_{12}| = 95.040$. Comprobamos además, que de hecho M_{12} se factoriza como el producto conjuntista

$$M_{12} = K_{12} \cdot M_{11}, \tag{4.1}$$

donde $K_{12} \leq M_{12}$ es el subgrupo de orden 12 generado por:

$$\begin{array}{rcl} \alpha & := & (0, 7)(1, 8)(2, 6)(3, \infty)(4, 10)(5, 9), \\ \beta & := & (0, 6, 4, 8, 9, \infty)(1, 5, 3, 7, 2, 10). \end{array}$$

Podemos expresar α y β en términos de los generadores σ, τ, y π:

$$\alpha = \pi\sigma^2\tau\pi^{-1}\tau\pi\tau^{-1}\sigma^{-2}\pi^{-1}, \; \beta = \pi^{-1}\tau\sigma^2\tau^{-1}\sigma\pi.$$

Por otro lado, K_{12} es abeliano (es isomorfo a un producto de dos grupos cíclicos de órdenes dos y seis). Así, sólo es necesario construir una signatura logarítmica de longitud mínima para M_{11} para tener el resultado también para M_{12} (por ser éste un producto de Zappa-Szép de K_{12} y M_{11}).

Centrémonos por tanto en M_{11}. El estabilizador en M_{11} del punto 1 es de nuevo un grupo de Mathieu (no simple) de orden 720, M_{10} (ver [86]). Se sabe además [99] que M_{10} posee una serie de composición de la forma

$$\{e_{M_{10}}\} < A_6 < M_{10}.$$

Como A_6 tiene una signatura logarítmica de longitud mínima y el cociente de M_{10} por A_6 es isomorfo a un grupo cíclico de orden dos, se sigue que M_{10} tiene una signatura logarítmica de longitud mínima. Además, puede comprobarse que M_{11} es un producto de Zappa-Szép de la forma

$$M_{11} = C_{11} \cdot M_{10},$$

(con C_{11} grupo cíclico de orden 11), sabemos que existen signaturas logarítmicas de longitud mínima para los grupos M_{11} y M_{12}.

II. $\mathbf{M_{22}, M_{23}}$ y $\mathbf{M_{24}}$. El grupo de Mathieu M_{24} puede verse, como ocurría con M_{12}, como un grupo de permutaciones actuando sobre una geometría proyectiva (en este caso, $\mathrm{PG}(2, \mathbb{F}_{23})$). Consideramos de nuevo tres aplicaciones

$$f : x \mapsto x + 1,\ g : x \mapsto -\frac{1}{x} \text{ y } h : x \mapsto -3x^{15} + 4x^4$$

del conjunto de puntos de la geometría en sí mismo. Como antes, representamos los puntos de la geometría por los enteros módulo 23 más el punto del infinito, y las operaciones asociadas a f, g y h se definen consecuentemente. De este modo, f, g y h definen de nuevo (ver [51]) tres permutaciones en los puntos de $\mathrm{PG}(2, \mathbb{F}_{23})$.

Ahora, el grupo de Mathieu M_{24} puede definirse como el grupo generado por las permutaciones inducidas por dichas aplicaciones, es decir: $M_{24} := \langle \sigma, \tau, \pi \rangle$, con,

$$\begin{aligned} f \sim \sigma &= (0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22), \\ g \sim \tau &= (0,\infty)(1,22)(2,11)(3,15)(4,17)(5,9)(6,19)(7,13)(8,20)(10,16) \\ &\quad (12,21)(14,18), \\ h \sim \pi &= (2,16,9,6,8)(3,12,13,18,4)(7,17,10,11,22)(14,19,21,20,15). \end{aligned}$$

El grupo M_{24} puede escribirse como el siguiente producto de Zappa-Szép:

$$M_{24} = S_4 \cdot M_{23},$$

donde el grupo M_{23} es el estabilizador en M_{24} del punto 0 (ver [86]), y el grupo $S_4 := \langle \alpha, \beta \rangle \leq M_{24}$ está generado por las permutaciones

$$\begin{aligned} \alpha &:= (0,2)(1,14)(3,8)(4,\infty)(5,20)(6,17)(7,10)(9,21)(11,18)(12,16) \\ &\quad (13,15)(19,22), \\ \beta &:= (0,1,\infty,12)(2,7,3,9)(4,18,13,22)(5,10,16,19)(6,11,14,21) \\ &\quad (8,20,15,17). \end{aligned}$$

Es fácil ver que dicho grupo es isomorfo al grupo de permutaciones de un conjunto de cuatro elementos, lo que justifica la notación empleada. Ahora, M_{23} puede escribirse como el producto de Zappa-Szép

$$M_{23} = C_{23} \cdot M_{22},$$

donde M_{22} es el estabilizador de los puntos 0 y 1 en M_{24} (ver [86]) y C_{23} es un grupo cíclico de orden 23. De hecho, para construir este producto basta tomar cualquier elemento de orden 23 en M_{23} como generador del factor cíclico.

Para finalizar, considerar los siguientes subgrupos de M_{22} :

$$C_{11} := \langle (2,19,9,7,18,12,\infty,5,11,16,21)(3,17,8,10,13,20,22,14,6,4,15) \rangle$$

y

$$C_2 := \langle (2,10)(3,19)(4,11)(7,13)(8,21)(9,15)(12,14)(16,17) \rangle.$$

Puede comprobarse , que el subconjunto $A := C_2 \cdot C_{11} \subseteq M_{22}$ tiene la siguiente propiedad: para todo punto $x \in \{2, \ldots, 22, \infty\}$ existe un elemento $a \in A$ que cumple $a(2) = x$. Por otro lado, [86] el estabilizador de los puntos 0, 1 y 2 en M_{24} es isomorfo al grupo simple $\mathrm{PSL}_3(4)$ de orden 20.160. Como ya hemos visto que $\mathrm{PSL}_3(4)$ tiene una signatura logarítmica de longitud mínima, podemos construir una signatura logarítmica de longitud mínima para M_{22} a partir de su descomposición (conjuntista) como el producto $M_{22} = \mathrm{PSL}_3(4) \cdot A$. Consecuentemente, también los grupos M_{23} y M_{24} poseeen signaturas logarítmicas de longitud mínima.

En resumen, hemos probado que todos los grupos del Cuadro 4.4 poseen signaturas logarítmicas de longitud mínima.
En particular, esta lista contiene a todos los grupos simples de orden menor o igual que 2^{17} salvo las siguientes excepciones:

Grupo	Orden	Signatura Logarítmica de longitud mínima
C_p (p primo)	p	trivial: $[[g \mid g \in C_p]]$
A_n ($n \geq 5$)	$n!/2$	cadena de estabilizadores
$\mathrm{PSL}_2(q)$ ($q>3$)	$\frac{q\cdot(q^2-1)}{\gcd(2,q-1)}$	producto de subgrupos de Sylow
$\mathrm{PSL}_3(q)$ ($q \not\equiv_3 1$)	$q^3(q^3-1)(q^2-1)$	producto de subgrupos de Sylow
$\mathrm{PSL}_3(4)$	20.160	producto de subgrupos de Sylow (Cuadro 4.1)
$\mathrm{PSU}_4(2)$	25.920	producto de subgrupos de Sylow (Cuadro 4.2)
$\mathrm{PSU}_3(4)$	62.400	producto de subgrupos de Sylow (Cuadro 4.3)
M_{11}	7.920	producto de subgrupos: $C_{11} \cdot A_6 \cdot C_2$
M_{12}	95.040	producto de subgrupos: $(C_2 \times C_6) \cdot M_{11}$
M_{22}	443.520	producto de subgrupos: $\mathrm{PSL}_3(4) \cdot C_2 \cdot C_{11}$
M_{23}	10.200.960	producto de subgrupos: $C_{23} \cdot M_{22}$
M_{24}	244.823.040	producto de subgrupos: $S_4 \cdot M_{23}$

Cuadro 4.4: resumen grupos simples (salvo $\mathrm{PSU}_3(3)$, $\mathrm{Sz}(8)$ y $\mathrm{PSU}_3(5)$).

- $\mathrm{PSU}_3(3)$ (de orden 6048)
- $\mathrm{Sz}(8)$ (de orden 29120)
- $\mathrm{PSU}_3(5)$ (de orden 126000).

Si encontramos signaturas logarítmicas de longitud mínima para esos tres grupos, habremos probado que tal factorización es posible para todo grupo simple de orden menor que el grupo J_1 (el primer grupo esporádico de Janko). De hecho, quedaría demostrado que todo grupo (simple o no) de orden menor que $|J_1| = 175.560$ posee una signatura logarítmica de longitud mínima. Para concluir esta sección, veremos que, de hecho, dicho resultado es cierto encontrando explícitamente signaturas logarítmicas de longitud mínima para los tres grupos anteriores.

III. $\mathrm{PSU}_3(3)$. Representamos el grupo $\mathrm{PSU}_3(3)$ como un subgrupo de S_{28} generado por las permutaciones (ver [99]):

$$\begin{aligned} \alpha &= (2,7,23,26,17,13,6,3)(4,19,11,28,25,24,16,9) \\ &\quad (5,18,10,14,15,8,20,12)(21,27) \quad \text{y} \\ \beta &= (1,2,4,10,8,16,13,22)(3,7,9,17,6,12,21,5) \\ &\quad (11,19,26,14,15,23,24,25)(27,28). \end{aligned}$$

Con ayuda de un sistema de álgebra computacional, vemos que $\mathrm{PSU}_3(3)$ puede factorizarse como un producto (conjuntista) de tres grupos: $\langle \alpha, \beta \rangle = G_1 C_4 C_7$ donde G_1 es el estabilizador del 1, y C_4, C_7 son grupos cíclicos de orden 4 y 7 respectivamente.

En el Cuadro 4.5 especificamos cómo elegir los grupos cíclicos de esta descomposición. El grupo G_1 tiene orden $216 = |\mathrm{PSU}_3(3)|/(4 \cdot 7)$ y es resoluble (por ser de orden $2^3 3^3$), de donde se sigue el resultado deseado: añadiendo a una signatura logarítmica de longitud mínima de G_1, dos secuencias compuestas respectivamente por los elementos de C_4, y C_7, en cualquier orden, obtenemos una signatura logartímica de longitud mínima para $\mathrm{PSU}_3(3)$.

C_4	=	$\langle(1,7,13,21)(2,8,19,4)(3,24,22,18)(5,6,10,14)(9,23)$ $(11,20,28,15)(12,16)(17,27,25,26)\rangle \leq \mathrm{PSU}_3(3)$
C_7	=	$\langle(1,11,8,14,20,27,2)(3,5,12,21,18,4,28)(6,23,24,26,22,9,13)$ $(7,25,16,17,10,15,19)\rangle \leq \mathrm{PSU}_3(3)$.

Cuadro 4.5: factores C_4 y C_7 de la factorización $\mathrm{PSU}_3(3) = G_1C_4C_7$

IV. Sz(**8**). Representamos [99] el grupo de Suzuki, Sz(8), como el subgrupo de S_{65} generado por:

$$\begin{aligned}
\alpha &= (1,2)(3,4)(5,7)(6,9)(8,12)(10,13)(11,15)(14,19)(16,21)(17,23)\\
&\quad (18,25)(20,28)(22,31)(24,33)(26,35)(27,32)(29,37)(30,39)(34,43)\\
&\quad (36,46)(38,48)(41,51)(42,44)(45,55)(47,50)(49,58)(52,60)(53,61)\\
&\quad (54,59)(56,62)(57,63)(64,65)\\
\beta &= (1,3,5,8)(4,6,10,14)(7,11,16,22)(9,12,17,24)(13,18,26,36)\\
&\quad (15,20,29,38)(19,27,31,28)(21,30,40,50)(23,32,41,52)\\
&\quad (25,34,44,54)(33,42,53,43)(35,45,56,63)(37,47,51,46)\\
&\quad (39,49,59,60)(48,57,55,58)(61,64,62,65).
\end{aligned}$$

Haciendo uso de un sistema de álgebra computacional, puede comprobarse que Sz(8) puede factorizarse en un producto $\mathrm{Sz}(8) = G_1C_5C_{13}$ donde G_1 es el estabilizador del 1, y C_5, C_{13} son grupos cíclicos de orden 5 y 13 respectivamente. En el Cuadro 4.6 se especifica cómo construir los grupos cíclicos de esta descomposición. Ahora, el estabilizador G_1 tiene oden $2^6 7 = 448 = |\mathrm{Sz}(8)|/(5 \cdot 13)$ y es por tanto resoluble, de donde se sigue, de modo análogo al caso anterior, que Sz(8) posee signaturas logarítmicas de longitud mínima.

C_5	=	$\langle(1,38,43,40,64)(2,50,32,63,44)(3,18,10,65,61)$ $(4,7,15,24,12)(5,19,33,54,17)(6,8,36,27,13)$ $(9,31,48,47,30)(11,49,51,20,23)(14,60,16,53,57)$ $(21,39,56,35,37)(22,46,58,45,59)(25,34,62,26,41)$ $(28,55,29,42,52)\rangle \leq \mathrm{Sz}(8)$
C_{13}	=	$\langle(1,46,21,18,47,51,22,31,15,4,39,6,41)$ $(2,56,32,27,5,49,3,20,40,54,50,59,23)$ $(7,42,48,61,37,65,53,36,19,8,17,45,43)$ $(9,52,55,29,38,12,62,25,57,24,11,58,30)$ $(10,63,26,14,44,35,13,34,33,16,64,28,60)\rangle \leq \mathrm{Sz}(8)$

Cuadro 4.6: factores C_5 y C_{13} de la factorización $\mathrm{Sz}(8) = G_1C_5C_{13}$

V. $\mathrm{PSU}_3(5)$**.** Representamos [99] el grupo $\mathrm{PSU}_3(5)$ como el subgrupo de S_{126} generado por

$$\begin{aligned}
\alpha &= (1,2,4,10,25)(3,7,17,41,19)(5,13,33,68,111)(6,14,36,34,70)\\
&\quad (8,20,47,81,48)(9,22,32,61,106)(11,28,37,40,82)(12,30,24,55,65)\\
&\quad (15,38,77,29,64)(16,39,43,85,73)(18,44,75,118,109)\\
&\quad (21,49,93,126,120)(23,53,42,83,52)(26,58,104,122,101)\\
&\quad (27,60,105,110,124)(31,35,72,116,66)(45,87,123,108,96)\\
&\quad (46,89,95,100,59)(50,94,57,76,103)(51,97,114,71,107)\\
&\quad (54,88,74,63,98)(56,102,117,99,121)(62,86,78,90,80)\\
&\quad (67,79,112,92,125)(69,113,84,115,119),\\
\beta &= (2,13,112,108,24,93,45,7)(3,32,53,102,54,65,75,16)\\
&\quad (4,41,52,104,11,68,99,30)(5,81,122,18,56,34,19,63)\\
&\quad (6,17,72,69,101,10,55,31)(8,29,100,40,84,67,37,57)\\
&\quad (9,36,98,124,27,94,117,66)(12,120,42,43,33,15,26,106)\\
&\quad (14,76,64,85,86,96,70,97)(20,73,82,83,125,50,39,113)\\
&\quad (21,74,87,23,59,105,110,61)(22,60,88,123,78,62,90,116)\\
&\quad (28,118,77,79,44,46,115,121)(35,80,109,48,95,49,114,126)\\
&\quad (38,91,47,92,51,107,71,119)(58,103,89,111).
\end{aligned}$$

De nuevo, $\mathrm{PSU}_3(5)$ puede factorizarse como un producto

$$\langle \alpha, \beta \rangle = C_7 C_3^{(1)} C_3^{(2)} C_2 G_1$$

donde G_1 es el estabilizador del 1, C_2 es un grupo cíclico de orden 2, $C_3^{(1)}$ y $C_3^{(2)}$ son grupos cíclicos de orden 3, y C_7 es un grupo cíclico de orden 7 (ver cómo elegirlos en el Cuadro 4.7). Una vez más, el estabilizador G_1 (que ahora es de orden $5^3 2^3 = 1000 = |\mathrm{PSU}_3(5)|/(2 \cdot 3 \cdot 3 \cdot 7)$) es resoluble, de donde se sigue que el grupo $\mathrm{PSU}_3(5)$ posee una signatura logarítmica de longitud mínima.

C_2	=	$\langle(1,60)(3,6)(4,27)(5,21)(7,94)(8,59)(9,12)(10,83)(11,42)$ $(13,30)(14,84)(15,58)(16,47)(17,32)(18,74)(19,115)(20,77)$ $(22,54)(23,78)(24,120)(25,82)(26,43)(28,114)(29,100)(31,76)$ $(33,106)(34,35)(36,45)(37,80)(38,39)(40,105)(41,98)(44,55)$ $(46,67)(48,79)(49,116)(50,61)(52,124)(53,71)(56,91)(57,70)$ $(62,97)(63,119)(64,92)(65,88)(66,111)(68,103)(69,109)$ $(72,96)(73,101)(75,126)(81,122)(86,107)(87,113)(90,110)$ $(93,108)(95,125)(102,121)(104,112)(118,123)\rangle \leq \mathrm{PSU}_3(5)$
$C_3^{(1)}$	=	$\langle(1,81,13)(2,119,26)(3,16,105)(4,121,97)(5,120,109)$ $(6,24,113)(7,10,28)(8,40,74)(9,66,94)(11,39,49)(12,112,62)$ $(14,60,83)(15,126,122)(17,123,56)(18,76,33)(19,69,91)$ $(20,101,45)(21,87,61)(22,78,104)(23,100,65)(25,43,72)$ $(27,67,75)(29,58,36)(30,51,38)(31,118,108)(32,59,89)$ $(34,41,55)(35,82,71)(37,103,110)(42,63,86)(44,114,107)$ $(46,48,116)(47,85,115)(50,90,80)(52,77,70)(53,124,111)$ $(54,98,102)(57,84,117)(64,79,88)(68,93,106)(73,95,99)$ $(92,125,96)\rangle \leq \mathrm{PSU}_3(5)$
$C_3^{(2)}$	=	$\langle(1,8,6)(2,57,56)(3,88,104)(4,79,9)(5,22,97)(7,30,67)$ $(10,58,54)(11,28,27)(12,75,106)(13,19,94)(14,50,51)$ $(15,68,66)(16,34,44)(17,36,73)(18,25,32)(20,93,112)$ $(21,26,70)(23,87,24)(29,64,111)(31,47,116)(33,105,91)$ $(35,98,84)(37,72,121)(38,83,40)(39,59,78)(41,110,71)$ $(42,43,69)(45,108,123)(46,95,102)(48,100,77)(49,115,55)$ $(52,80,63)(53,125,96)(60,82,89)(61,124,76)(62,113,74)$ $(65,126,117)(81,114,118)(85,90,119)(86,120,92)$ $(99,109,101)(103,107,122)\rangle \leq \mathrm{PSU}_3(5)$
C_7	=	$\langle(1,108,125,9,17,95,64)(2,39,63,34,111,25,60)$ $(3,33,68,30,74,47,107)(4,5,32,52,67,57,62)$ $(6,23,78,28,80,46,122)(7,24,50,100,48,104,15)$ $(8,18,85,93,44,51,16)(10,116,113,22,90,126,65)$ $(11,55,72,26,42,124,14)(12,77,121,75,120,84,59)$ $(13,92,56,88,118,94,73)(19,36,45,103,35,21,54)$ $(20,98,110,69,82,29,87)(27,112,109,70,117,97,89)$ $(31,99,96,79,66,81,123)(37,61,91,101,71,102,58)$ $(38,115,105,106,40,114,76)(41,119,53,49,43,86,83)\rangle$

Cuadro 4.7: factores cíclicos de la descomposición $\mathrm{PSU}_3(5) = C_7C_3^{(1)}C_3^{(2)}C_2G_1$

4.2. Un ejemplo de signatura logarítmica salvaje: el peinado de una trenza.

En el análisis de seguridad del esquema MST_1 (ver 2.3.2) resaltábamos la necesidad de encontrar algún criterio que permitiese decidir si la factorización inducida por una signatura logarítmica es difícil de calcular para *la mayoría* de los elementos del grupo. Por los resultados de la Sección (2.3.2), es claro que la definición dada en (2.6) de *signatura logarítmica salvaje* no proporciona un criterio adecuado. Además, en dicho análisis de seguridad vimos que encontrar signaturas logarítmicas salvajes según esa definición no era un problema en absoluto trivial. En esta sección damos un ejemplo de signatura logarítmica salvaje para un grupo infinito, siguiendo las definiciones 3.8 y 3.9 del Capítulo 2. En concreto, dicha signatura logarítmica será análoga a las construidas en la sección anterior, pero con un número infinito de bloques que a su vez serán secuencias infinitas.

Nuestra construcción está basada en la siguiente idea: sea G un grupo infinito para el cual existe una forma canónica F difícil de computar. Si existe una signatura logarítmica α de G tal que ser capaces de computar la factorización inducida por α implique ser capaces de computar F, entonces α será salvaje según la Definición 3.9.

A partir de esta idea construiremos una signatura logarítmica *salvaje* asociada al subgrupo de trenzas puras de un grupo de trenzas cualquiera. Recordar que el grupo de trenzas de n cuerdas, B_n, admite una presentación finita en los llamados generadores de Artin, σ_i, $i = 1, \ldots, n-1$, (ver Definición 3.1). Tal presentación permite establecer un epimorfismo entre B_n y S_n, definido asignando a cada generador σ_i, la transposición $\tau_i = (i, i+1)$. El núcleo de dicho epimorfismo será fundamental en nuestro desarrollo:

Definición 4.4 *Llamamos* grupo de trenzas puras de n cuerdas*, denotado P_n, al subgrupo de B_n que constituyen las trenzas cuya permutación inducida en el conjunto $\{1, \ldots, n\}$ es la identidad.*

Las trenzas puras admiten una forma canónica especial, el *peinado*, muy ligada a su representación geométrica en diagramas.

Definición 4.5 *Decimos que $v \in P_n$ es una* trenza peinada*, si puede representarse por un diagrama en el que las primeras $n-1$ cuerdas son paralelas.*

Definición 4.6 *Dada una trenza pura w, se dice que una palabra $\mathbf{cw}$ en los generadores de Artin es el* peinado de w *si existen $\mathbf{v}_1, \ldots \mathbf{v}_{n-1}$, palabras representando trenzas peinadas $v_1, \ldots v_{n-1}$ tales que $\mathbf{cw} = \mathbf{v}_1 \cdots \mathbf{v}_{n-1}$.*

Notas:

- El peinado de una trenza define una forma canónica en P_n; las trenzas $v_1, \ldots, v_{n-1}$ son únicas y cada una de expresa de manera única como un elemento de cierto grupo libre, como veremos más adelante.

- Suele hablarse indistintamente del peinado de una trenza w o del peinado de una palabra $\mathbf{w}$. Así, llamaremos *peinar* la palabra $\mathbf{w}$ al proceso de construir $\mathbf{cw}$ a partir de $\mathbf{w}$.

La noción de *peinado* juega un papel fundamental en la resolución del problema de la palabra en grupos de trenzas. En concreto, una de las primeras soluciones que se dieron al problema consta de los siguientes pasos:

- *Problema*: Dada una palabra $\mathbf{w}$, en generadores de Artin, decidir si representa o no a la trenza identidad.

- *Algoritmo*:

  Paso I : ¿representa $\mathbf{w}$ a una trenza pura? (si es así, ir al *Paso II*, si no, obviamente $\mathbf{w}$ no representa a la identidad)

  Paso II : *peinar* $\mathbf{w}$,

  Paso III : estudiar el *peinado de* $\mathbf{w}$, $\mathbf{v}_1 \cdots \mathbf{v}_{n-1}$. Sabemos que $\mathbf{w}$ representa a la identidad si y sólo si cada $\mathbf{v}_i$ representa a la identidad en cierto grupo libre (luego el problema de la palabra en grupos de trenzas queda así reducido al problema de la palabra en grupos libres).

Hoy en día existen soluciones mucho más eficientes para el problema de la palabra en grupos de trenzas, puesto que el algoritmo que acabamos de describir es muy lento. A continuación describimos con más detalle cuáles son las dificultades que entraña este proceso.

Peinando una trenza. Pasamos a detallar el *Paso II* del algoritmo esbozado anteriormente; esto es, veamos cómo pueden obtenerse las palabras $\mathbf{v}_1, \ldots, \mathbf{v}_n$ a partir de una representación $\mathbf{w}$ de una trenza pura w.
Sea $w_1 := w$. Denotamos por x_1 a la trenza que resulta al reemplazar la última cuerda de w_1 por una cuerda recta uniendo los correspondientes extremos derechos de las barras del diagrama. Es claro que la trenza y_1 que resulta al multiplicar w_1 por la inversa de x_1 es una trenza peinada.
Tomamos como $\mathbf{v}_1$ cualquier palabra que represente a y_1. Sea ahora $w_2 \in B_{n-1}$ la trenza que forman las $n-1$ primeras cuerdas de x_1. Repetimos el

proceso con w_2 (en lugar de con w_1). De este modo obtenemos una nueva palabra $\mathbf{v}_2$, que representa a una trenza peinada y_2 de B_{n-1}.
Reiterando este proceso, se construyen los demás factores $\mathbf{v}_3, \ldots, \mathbf{v}_{n-1}$. Notar que cada palabra $\mathbf{v}_i$ representa una trenza v_i en P_{n-i+1} con la propiedad de que todas, salvo su última cuerda, son cuerdas paralelas que unen los puntos correspondientes del diagrama.
Así, un algoritmo que realice este proceso sólo tiene que, dada una trenza pura w :

- construir palabras representando a las trenzas

$$w_i \in B_{n-i+1,}, \quad i = 2, \ldots, n-1,$$

- construir palabras representando a las trenzas x_i, que resultan de sustituír la última cuerda de w_i por una cuerda recta,
- construir palabras representando a las trenzas $w_i x_i^{-1}$ que son trenzas en P_{n-i+1}.

Ninguno de los procesos anteriores es especialmente complejo, siempre y cuando hagamos todos los cálculos utilizando los generadores de Artin de B_n. Sin embargo, en el *Paso III* queremos utilizar la solución para el problema de la palabra en ciertos grupos libres, para lo cual es necesario construir palabras $\mathbf{cv}_i$ representando a las trenzas $w_i x_i^{-1}$ en los generadores de dichos grupos. Realizar dicha reescritura conlleva un elevadísimo gasto computacional.

Los grupos libres asociados a una cuerda. Denotemos por $A_{i,j}$, para $1 \leq i, j \leq n$, a las trenzas puras constituidas por $n-1$ cuerdas paralelas, donde ninguna cuerda cruza a las demás, salvo la cuerda j-ésima que cruza a la i-ésima por encima (desde la derecha) pasando por encima del resto. En generadores de Artin:

$$A_{i,j} = (\sigma_{j-1}\sigma_{j-2}\cdots\sigma_{i+1}\sigma_i^2\sigma_{i+1}^{-1}\cdots\sigma_{j-2}^{-1}\sigma_{j-1}^{-1}).$$

Puede demostrarse que los elementos $A_{i,j}$ $1 \leq i < j \leq n$, generan el grupo P_n. Además, cada trenza v_i pertenece al grupo libre L_i generado por

$$\{A_{1,i}, A_{2,i}, \ldots, A_{i-1,i}\}.$$

Este grupo es el de las trenzas formadas por $n-1$ cuerdas *inmóviles*, pues en ellas sólo la cuerda i-ésima cruza a las situadas a su izquierda.

Decidir entonces si v_i es o no la identidad es un problema sencillo, siempre y cuando se disponga de una representación de v_i en los generadores $A_{k,i}, k \leq i-1$. Como no existe un algoritmo polinomial que encuentre dicha representación, se considera que usar el método descrito arriba para resolver el problema de la palabra (*peinando* la trenza) es muy ineficiente.

Una signatura logarítmica salvaje. Utilizando los razonamientos del desarrollo anterior, construimos una signatura logarítmica salvaje para el grupo de trenzas puras P_n.
Considerar, para un conjunto finito $\mathcal{X}$ una ordenación cualquiera de los elementos del grupo libre sobre X. Denotemos por $[\mathcal{X}]^*$ a la secuencia que resulta al escribir todos los elementos del grupo libre sobre X según esa ordenación. Supongamos por tanto bien definidas las secuencias:

$$\begin{array}{lcl} \alpha_1 & := & [A_{1,n}, A_{2,n}, \ldots, A_{n-1,n}]^* \\ \alpha_2 & := & [A_{1,n-1}, A_{2,n-1}, \ldots, A_{n-1,n-1}]^* \\ & \cdots & \\ \alpha_{n-2} & := & [A_{1,3}, A_{2,3}]^* \\ \alpha_{n-1} & := & [A_{1,2}]^*, \end{array} .$$

donde para cada $i = 2, \ldots, n$ la secuencia α_{n-i+1} está formada por los infinitos elementos del grupo libre L_i.
Definimos ahora la secuencia $\alpha := [\alpha_1, \cdots, \alpha_{n-1}]$. Es claro que α es una signatura logarítmica de P_n, por definir el peinado una forma canónica: cualquier trenza admite una factorización única como producto de $n-1$ trenzas peinadas, cada una en el grupo libre L_j. Además, α es salvaje, pues un algoritmo polinomial para calcular la factorización que induce sería de hecho un algoritmo que peinara cualquier trenza pura en tiempo polinomial. Caben pocas esperanzas de que los algoritmos conocidos para *peinar trenzas* (ver, por ejemplo, [94]) puedan mejorarse, por lo que es razonable afirmar que α es salvaje según la Definición 3.9.

Aún es pronto para afirmar que esta signatura logarítmica podrá utilizarse para una implementación segura del esquema MST_1. En concreto, para ello sería necesario explorar en profundidad cuál es la dificultad real de romper el esquema, pues ésta será sensiblemente menor a la de resolver el problema teórico. Notar, por ejemplo, que en todo el desarrollo anterior consideramos algoritmos que manejan grupos infinitos, cuando en la práctica habrá solamente un conjunto finito de trenzas involucradas. De cualquier forma, considerando la probada dificultad del problema del *peinado* y las excelentes propiedades computacionales de los grupos de trenzas, es indudable que esta

idea de construcción es un buen punto de partida para el diseño de nuevas herramientas criptográficas.

Conclusiones

A la vista de los trabajos más recientes en seguridad de la información, es incuestionable que la criptografía basada en teoría de grupos jugará un importante papel en el escenario criptográfico de los próximos años. Cabe sin embargo preguntarse qué dirección tomarán las líneas de trabajo actuales. Ciertamente cualquier afirmación que hagamos al respecto puede ser un tanto arriesgada, si bien es seguro que dos puntos clave marcarán el desarrollo de la investigación en este sentido.

Por un lado, las nuevas nociones formales de seguridad; desde la aparición de los esquemas de Cramer y Shoup [29, 30] los investigadores centran sus esfuerzos en conseguir nuevas propuestas de esquemas IND-CCA2. Por otro lado, a la hora de diseñar criptosistemas será fundamental estar al tanto de los últimos avances en computación cuántica. Conviene recordar que la mayoría de los algoritmos cuánticos conocidos en la actualidad pueden ser descritos en términos de teoría de grupos. De hecho, estos algoritmos se formulan como procedimientos de solución de casos particulares del llamado *Problema del Subgrupo Oculto* [16, 80], que incluye, entre otros, el problema de factorización de enteros y el de cálculo de logaritmos discretos.

Así, tomamos como punto de partida la premisa de que los esquemas basados en grupos a considerar en los próximos años estarán fundamentados en problemas no polinomiales (en el modelo clásico y en el cuántico), y tendrán buenas propiedades de seguridad (IND-CCA2). Pero, ¿Es posible construir criptosistemas con estas propiedades a partir de los que hoy conocemos? Al menos podemos ser optimistas en lo que se refiere a la *seguridad cuántica*: hasta la fecha no se conocen algoritmos cuánticos que resuelvan problemas como el de la conjugación o la palabra en grupos infinitos. En realidad, simplemente representar grupos de permutaciones en un ordenador cuántico parece una cuestión extremadamente complicada; hoy por hoy apenas hay algoritmos cuánticos para grupos no abelianos.

En cuanto a alcanzar niveles satisfactorios de seguridad con herramientas basadas en grupos, podemos hacer algunas afirmaciones justificadas por los

resultados de esta Memoria. Es muy probable que a raíz de los llamados *ataques de reacción* que comentamos en el Capítulo 3, la comunidad científica se incline por buscar construcciones seguras de tipo MST_1 en lugar de proponer esquemas similares al de Wagner y Magyarik [97]. Quizá un buen punto de partida sea empezar a buscar signaturas logarítmicas salvajes en grupos infinitos, en la línea de la que construimos en la Sección 4.2.

En cualquier caso, creemos que las signaturas logarítmicas son herramientas criptográficas cuyo potencial merece ser investigado y explotado adecuadamente. Es más, estas secuencias de factorización tienen interés al margen de sus aplicaciones criptográficas, pues son, por ejemplo, una herramienta muy útil a la hora de implementar algoritmos con grupos no abelianos. Desde el punto de vista teórico, sería sin duda interesante completar nuestros resultados del Capítulo 4 encontrando signaturas logarítmicas de longitud mínima para el resto de los grupos simples finitos o demostrando que existen grupos que no disponen de este tipo de factorizaciones.

En lo que respecta a los esquemas basados en grupos de trenzas, las construcciones actuales necesitarán revisarse si los criptoanálisis que detallamos en el Capítulo 3 se demuestran efectivos. En particular, es probable que a raíz del criptoanálisis de Lee y Park [65] las herramientas criptográficas basadas en el problema de la conjugación con grupos de trenzas pierdan interés práctico. Será por tanto necesario encontrar propuestas más seguras, bien a partir de otro tipo de grupos o bien utilizando otros problemas más complejos en grupos de trenzas. Muchos expertos investigan diferentes alternativas apoyándose en la sólida base teórica construida alrededor de la teoría de nudos. Nuestra idea de utilizar *cadenas de Markov* para el cifrado no es aún una propuesta concreta con garantías de seguridad, pero esperamos que sea un primer paso en esa dirección.

Por último, es también interesante investigar posibles modificaciones del esquema MST_2 generalizado con objeto de diseñar una construcción segura según la noción IND-CCA2. Quizá un esquema mixto entre el MST_2 generalizado y el criptosistema de Cramer y Shoup para grupos abelianos [29] sirva de marco para construir criptosistemas de clave pública IND-CCA2 a partir de grupos no abelianos.

La criptografía es una ciencia extremadamente dinámica y es por tanto arriesgado hacer predicciones acerca de su desarrollo a largo plazo. Las propuestas de nuevas herramientas criptográficas están continuamente expuestas al análisis crítico de la comunidad científica, y precisamente esta interacción entre diseñadores y criptoanalistas garantiza el avance hacia esquemas cada

vez más seguros. Nuestro trabajo se enmarca fundamentalmente dentro del criptoanálisis, por lo que la mayoría de los resultados expuestos tienen garantizada su vigencia en el futuro. Por otro lado, las ideas de diseño introducidas en esta Memoria son propuestas analizadas desde el punto de vista teórico. Estudios posteriores con un enfoque más práctico determinarán si es posible construir a partir de ellas herramientas eficientes y seguras.

Bibliografía

[1] S.S. Abhyankar. Galois Theory on the Line in Nonzero Characteristic. *Bulletin of the American Mathematical Society*, 27(1):68–133, 1992.

[2] M. Agrawal, N. Kayal, and N. Saxena. PRIMES is in P. *Preprint*, 2002.

[3] W. Alexi, B. Chor, O. Goldreich, and C. P Schnorr. RSA and Rabin functions: certain parts are as hard as the whole. *IAM Journal on Computing*, 17:194–209, 1988.

[4] I. Anshel, M. Anshel, B. Fisher, and D. Goldfeld. New key agreement protocols in braid group cryptography. In *CT RSA 2001*, Lecture Notes in Computer Science, pages 13–27, Berlin, Heildelberg, 2001. Springer-Verlag.

[5] I. Anshel, M. Anshel, and D. Goldfeld. A Method and Apparatus for Cryptographically Secure Algebraic Key Stablishment Protocols. *International Patent*, WO 99/44324, 1999.

[6] I. Anshel, M. Anshel, and D. Goldfeld. An algebraic method for public-key cryptography. *Mathematical Research Letters*, 6:1–5, 1999.

[7] I. Anshel, M. Anshel, and D. Goldfeld. Ko-Lee-Cheon-Han-Kang-Park Public Key Cryptosystem is a Special Case of the Anshel-Anshel-Goldfeld Public Key Cryptosystem. *Preprint*, 2000.

[8] M. Anshel. Constructing Public Key Cryptosystems via Combinatorial Group Theory. *Disponible en* `http://cryptome.org/pkc-cgt.htm`, 1999.

[9] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations Among Notions of Security for Public-Key Encryption Schemes. In *Advances in Cryptology — CRYPTO 1998*, volume 1462 of *Lecture Notes in Computer Science*, pages 26–46. Springer, 1998.

[10] M. Bellare and P. Rogaway. Entity Authentication and Key Distribution. In D. Stinson, editor, *Advances in Cryptology — CRYPTO 1993*, volume 773 of *Lecture Notes in Computer Science*. Springer, 1994.

[11] M. Ben-Or, B. Chor, and A. Shamir. On the cryptographic security of single RSA bits. In *Proceedings of the 15th ACM STOC*, pages 421–430, 1983.

[12] C. Bennet and G. Brassard. The dawn of a new era for quantum cryptography: the experimental prototype is working. *SICACT News*, 20(4):78 ff., 1989.

[13] Th. Beth. The State of Public-Key Cryptography - Not Only After the Invention of Quantum Computers . *Information Security Technical Report*, 4(4):47–52, 1999.

[14] Th. Beth. El estado de la Criptografía de Clave Pública a la vista de las posibilidades de la Computación Cuántica. *Actas de la VI Reunión Española sobre Criptología y Seguridad de la Información, VI RECSI*, pages 39–50, 2001.

[15] Th. Beth, S. González, M.I. González Vasco, C. Martínez, and R. Steinwandt. Cryptographic Shelter for the Information Society: Modeling and Fighting Novel Attacks on Cryptographic Primitives. *Por aparecer en el libro Techno-Legal Aspects of Information Society and New Economy, an Overview*, 2003.

[16] Th. Beth and M. Rötteler. Polynomial-Time Solution to the Hidden Subgroup Problem for a Class of non-abelian Groups. *LANL Preprint, disponible en* `http://uni-ausburg.de/abs/quant-ph/9812070`, 1998.

[17] J.C. Birget, S. Magliveras, and W. Wei. Trapdoors from subgoup chains and recombinant bilateral transversal. *Actas de la VII Reunión Española sobre Criptología y Seguridad de la Información, VII RECSI*, pages 31–49, 2002.

[18] J. Birman. *Braids, Links and Mapping Class groups*. Annals of Mathematics studies, 68, 1975.

[19] J. Birman, K.H. Ko, and S.J. Lee. A new solution to the word and conjugacy problems in the braid groups. *Advances in Mathematics*, 139:322–353, 1998.

[20] N. Blackburn and B. Huppert. *Finite Groups III*. Die Grundlehren de Mathematischen Wissenschaften. Springer-Verlag, Berlin- New York, 1982.

[21] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM Journal on Computing*, 13:850–864, 1986.

[22] J.M. Bohli, M.I. González Vasco, C. Martínez, and R. Steinwandt. Weak Keys in MST_1. *Preprint, disponible en* `http://eprint.iacr.org/2002/070/`, 2002.

[23] D. Boneh and I.E. Shparlinski. On the unpredictability of bits of the elliptic curve Diffie–Hellman scheme. In *Advances in Cryptology - CRYPTO 2001 Proceedings*, volume 2139 of *Lecture Notes on Computer Science*, pages 201–212, Berlin, 2001. Springer.

[24] D. Boneh and R. Venkatesan. Hardness of computing the most significant bits of secret keys in Diffie-Hellman and related schemes. In *Advances in Cryptology — CRYPTO ' 96*, volume 921 of *Lecture Notes in Computer Science*, pages 129–142. Springer, 1996.

[25] W. Bosma, J. Cannon, and C. Playoust. The Magma Algebra System I: The User Language. *Journal of Symbolic Computation*, 24:235–265, 1997.

[26] J.C. Cha, K.H. Ko, S.J. Lee, and J.W. Hanand J.H. Cheon. An Efficient Implementation of Braid Groups. In *Advances in Cryptology. Proceedings of Asiacrypt 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 144–156, Santa Barbara, California, USA, 2000. Springer.

[27] J.H. Cheon and B. Jun. Diffie-Hellman Conjugacy Problem on Braids. *Cryptology ePrint Archive: Report 2003/019*, 2003. Disponible electrónicamente en `http://eprint.iacr.org/2003/019/`.

[28] B. Chor and O. Goldreich. *RSA/Rabin least significant bits are* $\frac{1}{2} + \frac{1}{poly(\log n)}$ *secure*. In *Advances in Cryptology –CRYPTO '84*, volume 196 of *Lecture Notes in Computer Science*, pages 303–313, 1984.

[29] R. Cramer and V. Shoup. Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption. *Cryptology ePrint Archive: Report 2001/085*, 2001. Disponible electrónicamente en `http://eprint.iacr.org/2001/085/`.

[30] R. Cramer and V. Shoup. Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption. In Lars Knudsen, editor, *Advances in Cryptology — EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 45–64. Springer, 2002.

[31] C.A. Cusack. Group Factorizations in Cryptography. *Phd. Thesis. University of Nebraska*, 2000.

[32] W. Diffie and M.E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.

[33] J.P. Duval. Factoring words over an ordered alphabet. *J. Algorithms*, 4:363–381, 1983.

[34] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithm . *IEEE Trans. Info. Theory*, 31:469–472, 1985.

[35] R. Fischlin and C. P. Schnorr. Stronger security proofs for RSA and Rabin bitsStronger security proofs for RSA and Rabin bits. *Journal of Cryptology*, 13(2):221–244, 2000.

[36] N. Franco and J. González-Meneses. Conjugacy problem for braid groups and Garside groups. *Journal of Algebra, por aparecer*, 2003.

[37] M. Garzon and Y. Zalcstein. The Complexity of Grigorchuk groups with application to cryptography. *Theoretical Computer Science*, 88:83–98, 1991.

[38] R. Gennaro and D. Micciancio. Cryptanalysis of a Pseudorandom Generator Based on Braid Groups. In Lars Knudsen, editor, *Advances in Cryptology — EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 1–13. Springer, 2002.

[39] O. Goldreich and L. A. Levin. A hard core predicate for any one way function. In *Proceedings of the 21st ACM STOC*, pages 25–22, 1989.

[40] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28:270–299, 1984.

[41] M.I. González Vasco, C. Martínez, and R. Steinwandt. Towards a Uniform Description of Several Group Based Cryptographic Primitives. *Designs, Codes and Cryptography, por aparecer.*

[42] M.I. González Vasco, C. Martínez, and R. Steinwandt. Un Marco Común para Varios Esquemas de Clave Pública Basados en Grupos. *Actas de la VII Reunión Española sobre Criptología y Seguridad de la Información, VII RECSI*, pages 353–364, 2002.

[43] M.I. González Vasco and M.Ñäslund. A survey of hard core functions. In *Proc. Workshop on Cryptography and Computational Number Theory Singapure, 1999*, Birkäuser, pages 227–255, 2000.

[44] M.I. González Vasco, M.Ñaslund, and I.E. Shparlinski. The hidden number problem inextension fields and its applications. In *Proc. the 3rd Latin American Theoretical Informatics Conference, Cancun, 2002*, Lecture Notes in Computer Science, pages 105–117, Berlin, 2001. Springer.

[45] M.I. González Vasco, M. Rötteler, and R. Steinwandt. On Minimal Length Factorizations of Finite Groups. *Experimental Mathematics, por aparecer*, 2003.

[46] M.I. González Vasco and I.E. Shparlinski. On the security of Diffie Hellman bits . In *Proc. Workshop on Cryptography and Computational Number Theory Singapure, 1999*, Birkäuser, pages 256–268, 2001.

[47] M.I. González Vasco and I.E. Shparlinski. Security of the most significant bits of the Shamir message passing scheme. *Mathematics of Computation*, 71:333–342, 2002.

[48] M.I. González Vasco and R. Steinwandt. Clouds over a Public Key Cryptosystem based on Lyndon words. *Information Processing Letters*, 80:239–242, 2001.

[49] M.I. González Vasco and R. Steinwandt. Obstacles in Two Public-Key Cryptosystems Based on Group Factorizations. In *Cryptology. Editors: K. Nemoga, O. Grošek*, Tatra Mountains Math. Publications, pages 23–37, 2002.

[50] M.I. González Vasco and R. Steinwandt. Reaction Attacks on Public Key Cryptosystems Based on the Word Problem. *Cryptology ePrint Archive, Report 2002/139*, 80, 2002.

[51] D. Gorenstein. *Finite Simple Groups*. University series in mathematics. Plenum Press, New York, 1982.

[52] D. Gorenstein, R. Lyons, and R. Solomon. *The Classification of the Finite Simple Groups*, volume 40 of *Mathematical Surveys and Monogrpahs*. AMS, Providence, RI, 1998.

[53] J. Gruska. *Quantum Computing*. MacGraw-Hill, 1999.

[54] S.G. Hahn, E. Lee, and J.H. Park. Complexity of the Generalized Conjugacy Problem. *Discrete Applied Mathematics, por aparecer*, 2003.

[55] C. Hall, I. Goldberg, and B. Schneider. Reaction Attacks Against Several Public-Key Cryptosystems. In Vijay Varadharajan and Yi Mu, editors, *Information and Communication Security, Second International Conference, ICICS'99*, volume 1726 of *Lecture Notes in Computer Science*, pages 2–12. Springer, 1999.

[56] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo random number generators from any one way function. *SIAM Journal on Computing*, 28:1364–1396, 1999.

[57] J. Håstad and M.Ñaslund. The security of individual rsa and discrete log bits. *Journal of the ACM, por aparecer*, 2002.

[58] D. Hofheinz and R. Steinwandt. 187–198. In *Public Key Cryptography, 6th International Workshop on Practice and Theory in Public Key Cryptosystems, PKC 2003 Proceedings*, volume 2567 of *Lecture Notes on Computer Science*. Springer, 2003.

[59] D.F. Holt and P. Rowley. On products of Sylow subgroups in finite groups. *Arch. Math. (Basel)*, 60(2):105–107, 1993.

[60] T. Horváth. *Das TST Kryptosystem*. PhD. Thesis, Essen, 1998.

[61] J. Katz and M. Yung. Threshold cryptosystems based on factoring. `citeseer.nj.nec.com/457523.html`, 2001.

[62] A.Y. Kitaev, A.H. Shen, and M.N. Vyalyi. *Classical and Quantum Computation*, volume 47 of *Graduate Studies in Mathematics*. AMS Press, 2002.

[63] K.H. Ko, S.J. Lee, J.H. Cheon, J.W. Han, J. Kang, and C. Park. New Public-Key Cryptosystem using Braid Groups. In *Advances in Cryptology. Proceedings of CRYPTO 2000*, Lecture Notes in Computer Science, pages 166–183, Santa Barbara, California, USA, 2000. Springer.

[64] E. Lee, S.J. Lee, and S.G. Hahn. Pseudorandomness from Braid Groups. In Joe Kilian, editor, *Advances in Cryptology — CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 486–502. Springer, 2001.

[65] E. Lee and J.H. Park. Cryptanalysis of the Public Key Encryption based on Braid Groups. *Advances in Cryptology - EUROCRYPT 2003, por aparecer*, 2003.

[66] H.K. Lee, H.S. Lee, and Y.R. Lee. An Authenticated Group Key Agreement Protocol on Braid Groups. *Cryptology ePrint Archive: Report 2003/018*, 2003. Disponible electrónicamente en `http://eprint.iacr.org/2003/018/`.

[67] S.J. Lee and E. Lee. Potential Weaknesses of the Commutator Key Agreement Protocol Based On Braid Groups. In Lars Knudsen, editor, *Advances in Cryptology — EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 14–28. Springer, 2002.

[68] M.W. Liebeck, C.E. Praeger, and J. Saxl. *The maximal factorizations of the finite simple groups and their automorphism groups*, volume 86 of *Memoirs of the AMS*. AMS, Providence, RI, 1990.

[69] S.S. Magliveras. A cryptosystem from logarithmic signatures of finite groups. In *Proceedings of the 29'th Midwest Symposium on Circuits and Systems*, pages 972–975. Elsevier Publishing Company, 1986.

[70] S.S. Magliveras. Secret- and Public-key Cryptosystems from Group Factorizations. In *Cryptology. Editors: K. Nemoga, O. Grošek*, Tatra Mountains Math, Publication, pages 11–22, 2002.

[71] S.S. Magliveras and N.D. Memon. Linear complexity profile analysis of the PGM cryptosystem. *Congresus Numerantium, Utilitas Mathematica*, 72:51–60, 1989.

[72] S.S. Magliveras and N.D. Memon. Properties of cryptosystem PGM. In *Advances in Cryptology. Proceedings of CRYPTO 1989*, Lecture Notes on Computer Science, pages 447–460, Berlin, 1989. Springer-Verlag.

[73] S.S. Magliveras and N.D. Memon. Complexity tests for cryptosystem PGM. *Congresus Numerantium, Utilitas Mathematica*, 79:61–68, 1990.

[74] S.S. Magliveras and N.D. Memon. Algebraic properties of cryptosystem PGM. *Journal of Cryptology*, 5:167–183, 1992.

[75] S.S. Magliveras and P. Pedersen. Software implementation of the PGM encryption system. *Center for Communication and Information Science, University of Nebraska-Lincon*, CCIS report, en preparación, 1991.

[76] S.S. Magliveras, D.R. Stinson, and T. Trung. New approaches to designing public key cryptosystems using one-way functions and trap-doors in finite groups. *Journal of Cryptology*, 15:285–297, 2002.

[77] L. Mathew and R. Siromoney. A Public Key Cryptosystem Based on Lyndon Words. *Information Processing Letters*, 35:33–36, 1990.

[78] A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[79] P.W. Michor. Knit Products of Graded Lie Algebras and Groups. In *Proceedings of the Winter School in Geormetry and Physiscs, Srni 1988*, volume 22 of *II*, pages 171–175. Suppl. Rendiconti Circolo Matematico di Palermo, 1989.

[80] M. Mosca and A. Ekert. The Hidden Subgroup Problem and Eigenvalue Estimation on a Quantum Computer. *Proceedings of the 1st NASA International Conference on Quantum Computing and Quantum Comunication*, 1509, 1999.

[81] K. Murasugi and B.I. Kurpita. *A Study of Braids*, volume 484 of *Mathematics and its Aplications*. Kluwer, 1999.

[82] S.H. Paeng, K.C. Ha, J.H. Kim, S. Chee, and C. Park. New Public Key Cryptosystem Using Finite Non Abelian Groups. In J. Kilian, editor, *Advances in cryptology — CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 470–485. Springer, 2001.

[83] S.H. Paeng, D. Kwon, K.C. Ha, and J.H. Kim. Improved public key cryptosystem using finite non abelian groups. *Cryptology ePrint Archive: Report 2001/066*, 2001. Disponible electrónicamente en `http://eprint.iacr. org/2001/066/`.

[84] M. Rabin. Digital signatures and public key functions as intractable as factorization. *MIT Laboratory for computer science*, Technical report MIT/LCS/TR-212, 1979.

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

[86] J.J. Rotman. *An Introduction to the Theory of Groups*, volume 148 of *Graduate Texts in Mathematics*. Springer, 1999.

[87] A. Salomaa and S. Yu. On a Public Key Cryptosystem based on iterated morphisms and substitutions. *Theoretical Computer Science*, 48:283–296, 1986.

[88] P. Shor. Polynomial time algorithms for prime factorization and discrete logarithms on quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.

[89] P.W. Shor. Algorithms for Quantum Computation. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, , 1994. IEEE Computer Society Press.

[90] H. Sibert, P. Dehornoy, and M. Girault. Entity authentication schemes using braid word reduction, 2002. Disponible electrónicamente en `http://eprint.iacr.org/2003/019/`.

[91] R. Steinwandt and D. Hofheinz. Cryptoanalysis of a Public Key Cryptosystem Based on Grigorchuk Groups. *Preprint*, 2002.

[92] K.G. Subramanian and R. Siromomey. A DOL-TOL Public Key Cryptosystem. *Information Processing Letters*, 26:95–97, 1987.

[93] The GAP Team. GAP—Groups, Algorithms, and Programming. Lehrstuhl D für Mathematik, RWTH Aachen, Germany and School of Mathematical and Computational Sciences, Univ. St. Andrews, Scotland, 1997.

[94] R.S.D. Thomas. An algorithm for combing braids. In *Proceedings of the Second Louisina Conference on Combinatorics, Graph Theory and Computing*, pages 517–532. Lousiana State University, Baton Rouge, La., 1971.

[95] C. Tobias. Security Analysis of the MOR Cryptosystem. In *Public Key Cryptography, 6th International Workshop on Practice and Theory in Public Key Cryptosystems, PKC 2003 Proceedings*, volume 2567 of *Lecture Notes on Computer Science*, page 175 ff. Springer, 2003.

[96] Y. Tsiounis and M. Yung. On the Security of ElGamal based Encryption. In *PKC 98*, volume 1431 of *Lecture Notes on Computer Science*, page 117 ff., , 1998. Springer.

[97] N.R. Wagner and M.R. Magyarik. A Public Key Cryptosystem Based on the Word Problem. In G. R. Blakley and David Chaum, editor, *Advances in Cryptology: Proceedings of CRYPTO 84*, volume 196 of *Lecture Notes in Computer Science*, pages 19–36. Springer, 1985.

[98] S. Wiesner. Conjugate Coding. *SIGACT News*, 15:78–88, 1983.

[99] R. Wilson, P. Walsh, J. Tripp, I. Suleiman, S. Rogers, R. Paker, S.Norton, S. Linton, and J. Bray. *Atlas of Finite Group Representations*. `http://www.mat.bham.ac.uk/atlas/html`, 1996.