Universidad Politécnica
de Madrid

Ataques Criptográficos a Criptosistemas en Curvas Elípticas

Jorge Alejandro Ayala Talón

Director

Prof. Ángel González Prieto

Universidad Politécnica

de Madrid

Proyecto de Fin de Grado presentado para optar al Grado de
Ingeniería de Software

Departamento de Sistemas Informáticos
Escuela Técnica Superior de Ingeniería de Sistemas Informáticos

Curso 2018/2019

"El conocimiento y la sabiduría me persiguen, pero yo soy más rápido."

Héctor del Mar

ABSTRACT

Ataques Criptográficos a Criptosistemas en Curvas Elípticas

por Jorge Alejandro Ayala Talón

Abstract en Español

El objetivo de este proyecto es comparar el rendimiento y la eficiencia de los algoritmos de ataque conocidos contra criptosistemas en curvas elípticas. Previamente se realizará un estudio del concepto de curvas elípticas para comprender sus propiedades. A continuación, se conducirá un estudio de los ataques criptográficos conocidos, y se discutirá sus fortalezas y debilidades. Estos algoritmos serán implementados y probados para dilucidar qué ataque es mejor utilizar en cada situación.

Palabras clave: *Curvas elípticas, criptografía, ataques criptográficos.*

Abstract in English

The aim of this project is to compare the performance and efficiency of known cryptographic attack algorithms against cryptosystems in elliptic curves. A study of the concept of elliptic curves will be carried out beforehand in order to understand their properties. Next, a study of the known cryptographic attacks, and their strengths and weaknesses will be discussed. These algorithms will be implemented and tested to elucidate which attack is the best option in each situation.

Key words: *Elliptic curves, cryptography, cryptographic attacks.*

Agradecimientos

Gracias a Ángel por su guía incansable a lo largo de todo este proyecto. Sin él este trabajo no sería ni una sombra de lo que es.

Gracias a mi familia y amigos por su paciencia y apoyo conmigo durante este periodo de tiempo. Me habéis ayudado más de lo que os podéis imaginar, muchas gracias.

Gracias a mi abuela por sus pocas enseñanzas que se me grabaron a fuego en la memoria, las he tenido presentes durante todo el proyecto, muchas gracias.

Índice general

Introducción

Esta sociedad está construida sobre la transferencia de información entre los individuos de la misma: los planos de un edificio transmitidos de un arquitecto a un obrero; la contraseña del banco, contraseña de correo electrónico, contraseñas varias, información sensible, etc. Toda esta información viaja a través de varias redes hasta llegar a su destinatario, es decir viaja a través de un medio digital. Este medio es necesariamente compartido con varias personas para poder cumplir su fin, y en el caso de Internet, casi toda la sociedad tiene acceso a él. Todo esto induce la idea de proteger tus datos de ojos no deseados. Aquí nace la **criptografía**.

Aunque es un hecho que en esta época la criptografía es importante, es también un hecho que este concepto tiene mucha historia, remontándose a las guerras de la Antigüedad, donde se quiere evitar revelar la información a otra persona distinta del destinatario. A lo largo de la historia de la humanidad estos métodos han sido refinados, aplicados y vueltos a refinar para aumentar su eficacia y eficiencia. Como resultado de toda esta experiencia y los cambios que introdujo la tecnología, nace la **criptografía moderna**.

Este proyecto se va a centrar en el estudio de los **ataques criptográficos** a los métodos de encriptado moderno asimétrico de información sobre **curvas elípticas** definidas sobre un cuerpo finito. Para ello, es necesario estudiar qué son las curvas elípticas como concepto matemático, así como los cuerpos finitos, y como concepto criptográfico, en tanto que se tienen que entender sus fortalezas y debilidades.

Además, se explicarán algunos ejemplos de cómo se encriptan mensajes en criptografía moderna mediante curvas elípticas con el fin de ilustrar la encriptación que va a ser atacada posteriormente. Para finalizar, se estudiarán los ataques a la encriptación anterior y se llevarán a cabo estos ataques en un entorno informático para, posteriormente, analizar los resultados.

Este análisis constará de la comparación de los ataques sobre curvas elípticas entre sí y su clasificación según los resultados obtenidos. Se realizarán múltiples pruebas y se registrarán los resultados para, posteriormente, elaborar conclusiones en base al trabajo realizado.

En el capítulo 1 se estudiarán las bases de los cuerpos finitos: la **Teoría de Grupos**. En este capítulo se explicará qué son los grupos y su relación con los cuerpos finitos utilizados en criptografía moderna. Se estudiará también las propiedades de los grupos, así como las entidades conocidas como subgrupos, y sus propiedades. Finalmente, se presentarán varios ejemplos de grupos cuyos elementos no son números, si no formas geométricas o permutaciones de una serie de números.

Los **grupos** son una entidad matemática compuesta por un conjunto de elementos y una operación definida entre los elementos de ese conjunto. Este conjunto de elementos puede ser finito o infinito, en el caso de ser finito ese grupo tendrá un orden que indica el número de elementos del grupo. Los **subgrupos** son grupos cuyos conjuntos de elementos son subconjuntos de otro grupo. Una propiedad especial de los subgrupos muy importante también se remarca en este capítulo, esta propiedad es el llamado **Teorema de Lagrange**. Según el Teorema de Lagrange, el orden de los subgrupos de

un grupo son divisores del orden de ese grupo. Este teorema es muy importante para el estudio de los cuerpos finitos, así que se va a estudiar en profundidad. Todas estas ideas serán ampliamente estudiadas y explicadas en el capítulo 1.

En el capítulo 2 se estudia el siguiente paso hacia los cuerpos finitos: la **Teoría de Anillos**. En este capítulo se estudiará el paso de grupos a anillos, así como las nuevas propiedades que tendrán. Además en este capítulo se presentarán, finamente, los **cuerpos** y sus propiedades matemáticas. Asimismo, también se estudiarán las extensiones de cuerpos, así como los cuerpos de escisión de un polinomio. Estas entidades aportarán más propiedades y una mejor comprensión sobre cómo trabajar con anillos y los cuerpos. Al final del capítulo se dará un ejemplo de la construcción de un cuerpo finito. Este capítulo tendrá una estructura similar al capítulo de grupos.

Un **anillo** es un conjunto que consta de dos operaciones definidas en el conjunto de elementos. En este capítulo se clasificarán a los anillos según los elementos que los componen y las propiedades que tienen como el anillo cociente o el anillo de polinomios, ambos importantes para el estudio de anillos. Se explicarán las características de los cuerpos que los diferencian del resto de anillos, así como las características de los subcuerpos. Los anillos también tienen subentidades, estás se llaman **ideales**, y tienen una propiedad muy peculiar: la absorción de elementos. En pocas palabras, todo elemento de un ideal operado con un elemento del anillo resulta en un elemento del ideal. Esta propiedad será importante para el estudio de cuerpos.

De igual forma, se tratará en este capítulo sobre las **extensiones de cuerpos**. Dado un cuerpo K, se dice que el cuerpo E es una extensión si K es un subcuerpo de E. Se estudiarán las propiedades de estas extensiones y su interacción con otras distintas. Los cuerpos de escisión se definen sobre un cuerpo y sobre un polinomio, donde las raíces de ese polinomio crean la situación de **escisión** con la extensión del cuerpo al que pertenecen. Esta escisión será estudiada en detalle en el capítulo 2. Finalmente, gracias a las distintas propiedades de los cuerpos y sus extensiones, se comprueba que existen cuerpos finitos con con orden cualquier potencia de un número primo.

En el capítulo 3 se estudiarán las **curvas elípticas**. Las curvas elípticas son un tipo de curvas algebraicas tal que la ecuación que las define tiene grado 3 y no presenta singularidades. Esta curva está formada por todos los puntos cuyas coordenadas satisfacen la ecuación de la curva. A esta ecuación se la llama **ecuación de Weierstrass**. Teniendo en cuenta esto, se mostrará que es posible formar un grupo finito a partir del conjunto de puntos de la curva. A esta consideración y formalización se la llama la **Ley de Grupo**. Las curvas elípticas tienen determinadas propiedades que facilitarán su estudio y posterior criptoanálisis. Asimismo, se presentan algunos tipos de curvas elípticas basados en estas propiedades.

En el capítulo 4 se lleva a cabo un estudio sobre cómo funciona la criptografía moderna sobre curvas elípticas. Se explicarán algunos de los algoritmos más importantes y más utilizados para encriptar información utilizando curvas elípticas. En concreto, se estudiarán los algoritmos de ElGamal para encriptar y para firmas digitales. Además, se explicará en profundidad cuál es la fortaleza de estos

algoritmos en comparación con otros pertenecientes a la criptografía clásica, como el algoritmo RSA. La gran fortaleza de estos algoritmos reside en el conocido **problema del logaritmo discreto**. A diferencia de RSA, la única forma conocida de atacar a estos algoritmos es atacando al problema del logaritmo discreto; mientras que para atacar a RSA existen varios métodos especializados que resuelven problemas computacionalmente más sencillos que el problema del logaritmo discreto.

En el capítulo 5 se estudiarán los principales algoritmos de ataque contra criptografía en curvas elípticas. Estos ataques son: **Baby Step, Giant Step**; ρ **de Pollard**; **Pohlig-Hellman** y **MOV**. El ataque Baby Step, Giant Step se basa en utilizar las propiedades de las curvas elípticas para encontrar una coincidencia entre dos puntos de dos conjuntos generados realizando saltos en el grupo a dos velocidades (lenta, "Baby Step", y rápida, "Giant Step"). A partir de esta coincidencia, es posible hallar el logaritmo discreto en tiempo constante. El algoritmo de Pollard sigue la misma filosofía, pero recorre el grupo de la curva elíptica siguiendo un camino aleatorio hasta encontrar una coincidencia entre dos puntos. El algoritmo de Pohlig-Hellman utiliza la factorización en factores primos del orden del punto base para hallar el logaritmo discreto en el módulo de cada factor para, después, fusionar los resultados utilizando el Teorema Chino del Resto. El algoritmo MOV utiliza el emparejamiento de Weil para transferir el problema del logaritmo discreto de la curva elíptica a un cuerpo finito. En este capítulo también se discutirá sobre las ventajas y desventajas sobre cada método.

En el capítulo 6 se realizará la implementación software de los métodos de ataque estudiados y se realizarán pruebas sobre varias curvas elípticas de distinto orden para comprobar la eficacia de los métodos. En base a las pruebas realizadas se hará una comparación entre los métodos para constatar cuál es el mejor algoritmo de entre todos los considerados.

Capítulo 1

Teoría de Grupos

El estudio de los grupos se debe a que este concepto matemático es fundamental para la definición de cuerpos finitos. En este capítulo se explicará qué son los grupos, dando una visión general de los mismos, y sus propiedades generales. Asimismo, se estudiarán propiedades más específicas que resultarán útiles más adelante. Este capítulo sentará las bases de los cuerpos finitos. Este capítulo va a estar dedicado a una revisión bibliográfica de [DH07].

1.1. Definición de Grupos

Un grupo es un conjunto G en el cual está definida una operación binaria $\cdot$ que satisface las siguientes propiedades:

(1) La operación binaria $\cdot$ es **cerrada** en el conjunto G. Dados dos conjuntos: G y A, se denomina operación binaria definida en G, a una aplicación definida en el conjunto $G \times G$ con valores en A. Una operación binaria definida en G es cerrada si $A = G$. Esto significa que los conjuntos de entrada y salida son el mismo; es decir, realizar una operación cerrada entre dos elementos de G siempre resultará en un elemento del conjunto G.

(2) La operación binaria $\cdot$ es **asociativa**, esto es, para todo $g_1, g_2, g_3 \in G$, $(g_1 \cdot g_2) \cdot g_3 = g_1 \cdot (g_2 \cdot g_3)$.

(3) Existe un elemento e en G, tal que si cualquier elemento de G se opera con él resulta en ese mismo elemento de G. En otras palabras, existe $e \in G$ tal que, $e \cdot g = g \cdot e = g$ para todo $g \in G$. Este elemento se denomina **elemento neutro**.

(4) Para todo $g \in G$, existe un $g' \in G$, tal que $g \cdot g' = e = g' \cdot g$, siendo e el elemento neutro de G. Este elemento g' se le llama el **elemento inverso** de g y se le denota como g^{-1}.

Si el conjunto G, junto con la operación binaria $\cdot$, satisface las condiciones anteriores se le llamará grupo y se le denotará como $(G, \cdot)$.

Observación 1.1.1. A pesar de la notación, $\cdot$, no tiene por qué ser el producto usual de números.

Teniendo en cuenta las anteriores propiedades, se puede añadir una nueva propiedad y con ella definir un nuevo tipo de grupo. Si la operación $\cdot$ cumple la propiedad **conmutativa** en el conjunto G, es decir, se cumple que $g_1 \cdot g_2 = g_2 \cdot g_1$, para todo $g_2, g_1 \in G$; entonces, el grupo $(G, \cdot)$ se dice que es un grupo **abeliano** o **conmutativo**.

Ejemplo 1.1.2. Tomando como ejemplo el conjunto de los números enteros ($\mathbb{Z}$) y la operación suma, se puede observar que es un grupo **abeliano**, debido a que satisface todas las propiedades anteriores.

Si se suma dos números enteros el resultado siempre será un entero y la suma de números enteros es asociativa. Lo mismo ocurre con la propiedad conmutativa. De igual forma, si se suma cualquier entero por el 0 se quedará igual, por tanto 0 es el elemento neutro del grupo ($\mathbb{Z}$, +); y finalmente, para todo $a \in \mathbb{Z}$, existe un $b \in \mathbb{Z}$, tal que $a + b = 0$, a saber que $b = -a$ es el inverso para la operación del grupo.

Ejemplo 1.1.3. En cambio, si se toma como ejemplo el conjunto $\mathbb{Z}$ y la operación producto, lo que resulta no es un grupo. La operación producto en los enteros es cerrada, la demostración es trivial, del mismo modo que el producto es asociativo y conmutativo. Además, para todo $n \in \mathbb{Z}$, $n \cdot 1 = n$; por tanto, existe un elemento neutro. Sin embargo, este conjunto carece de elemento inverso para la operación de producto. La demostración es sencilla:

$$\text{Si } 2, x \in \mathbb{Z} \Rightarrow 2 \cdot x = 1 \Rightarrow x = \frac{1}{2} \notin \mathbb{Z}.$$

Resolviendo la anterior ecuación se tendría que x es el inverso de 2 en el grupo $(\mathbb{Z}, \cdot)$. Sin embargo, como se puede observar $\frac{1}{2} \notin \mathbb{Z}$, por tanto, no cumple la última propiedad.

Ejemplo 1.1.4. A la luz del ejemplo 1.1.3, se podría pensar que si se elige el conjunto de los racionales ($\mathbb{Q}$) en vez del de los enteros, junto con la operación producto podría resultar en un grupo abeliano, ya que $\frac{1}{2} \in \mathbb{Q}$; pero no es así. Esto es debido a que el 0 no tiene inverso, debido a la sencilla razón de que cualquier número multiplicado por él es 0, que no es el elemento neutro. No obstante, si se le quita el 0 a $\mathbb{Q}$, es decir, se considera: $\mathbb{Q}^* = \mathbb{Q} - \{0\}$, sí que resultará en un grupo abeliano $(\mathbb{Q}^*, \cdot)$.

1.2. Propiedades de grupos

1.2.1. Propiedades cancelativas

Si $(G, \cdot)$ es un grupo, entonces se cumplen:

(1) **Cancelación por la derecha**. Para todo $a, b, c \in G$, entonces si $a \cdot c = b \cdot c$, se deduce $a = b$.

(2) **Cancelación por la izquierda**. Para todo $a, b, c \in G$, entonces si $c \cdot a = c \cdot b$, se deduce $a = b$.

Demostración. Para (1). Dado $c \in G$, por la propiedad (4) de grupos, existe $c' \in G$, tal que $c \cdot c' = e$. Entonces, si $a \cdot c = b \cdot c$, se tiene que $(a \cdot c) \cdot c' = (b \cdot c) \cdot c'$, por la asociatividad, $a \cdot (c \cdot c') = b \cdot (c \cdot c')$. Por la propiedad (3) de grupos, se tiene que $a = a \cdot e = b \cdot e = b$. La parte (2) se demuestra de forma similar, sólo se cambia el lado por el que opera c'. ■

1.2.2. Propiedades de unicidad

(1) El elemento neutro de un grupo $(G, \cdot)$ es único y se denotará como e.

Demostración. Si e y f son elementos neutros del grupo G; al ser e un elemento neutro, se tiene que $e \cdot g = g$ para todo $g \in G$, en particular, $e \cdot f = f$. Pero como $e \cdot f = e$, por ser f también un elemento neutro, se tiene que $e = f$. ■

(2) Todo elemento g de un grupo $(G, \cdot)$ tiene **un solo elemento inverso** y se simbolizará como g^{-1}.

Demostración. Se considera que g' y g'' son inversos de g, con $g', g'' \in G$. Entonces, $g \cdot g' = g \cdot g'' = e$. Por la propiedad cancelativa por la izquierda 1.2.1, se tiene que $g' = g''$. ■

1.2.3. Propiedades del elemento inverso

(1) Sea $(G, \cdot)$ un grupo y $g \in G$; se tiene que $(g^{-1})^{-1} = g$.

Demostración. Sea g' el inverso de g^{-1}, entonces $g' \cdot g^{-1} = e$. Por otro lado, $g \cdot g^{-1} = e$. Utilizando la propiedad cancelativa por la derecha 1.2.1, se tiene que $g' = g$, y por tanto, $(g^{-1})^{-1} = g$. ■

(2) En un grupo $(G, \cdot)$ se tiene que $(a \cdot b)^{-1} = b^{-1} \cdot a^{-1}$ para todo $a, b \in G$.

Demostración. Usando la propiedad asociativa, se tiene que, para todo $a, b \in G$:
$(a \cdot b) \cdot (b^{-1} \cdot a^{-1}) = a \cdot (b^{-1} \cdot b) \cdot a^{-1} = a \cdot e \cdot a^{-1} = e$. Por la propiedad de unicidad del elemento inverso se confirma esta propiedad. La demostración de $(b^{-1} \cdot a^{-1}) \cdot (a \cdot b)$ se prueba del mismo modo. ■

Proposición 1.2.1. *Si $(G, \cdot)$ es un grupo y $a, b \in G$, $(G, \cdot)$ será un grupo abeliano si y sólo si $(a \cdot b)^{-1} = a^{-1} \cdot b^{-1}$.*

Demostración. Si $(G, \cdot)$ es un grupo abeliano quiere decir que en este grupo se cumple la **propiedad conmutativa** y por tanto, $b^{-1} \cdot a^{-1} = a^{-1} \cdot b^{-1}$ y que $a^{-1} \cdot b^{-1} = (a \cdot b)^{-1}$. Por el otro lado, si $(a \cdot b)^{-1} = a^{-1} \cdot b^{-1}$ es cierto, se demuestra:

$$a \cdot b = (a^{-1})^{-1} \cdot (b^{-1})^{-1} = (a^{-1} \cdot b^{-1})^{-1} = (b^{-1})^{-1} \cdot (a^{-1})^{-1} = b \cdot a.$$

Así queda demostrado que $(G, \cdot)$ es un **grupo abeliano**. ■

Si $(G, \cdot)$ es un grupo y G tiene una cantidad finita de elementos, esta cantidad se definirá como el **orden de G** denotado como $|G|$; además, se dice que dicho grupo $(G, \cdot)$ es un **grupo finito de orden $|G|$**, indicando el número de elementos en el grupo. En caso de que G tenga una cantidad infinita de elementos, $(G, \cdot)$ se dirá que es un grupo infinito.

Los grupos finitos se pueden representar totalmente como una tabla en la que cada casilla es el resultado de la operación de los dos elementos correspondientes a esa columna y fila. Así, el resultado de operar los elementos a y b del grupo, se escribiría en la intersección entre la fila a y la columna b. Por ejemplo, para un grupo de 3 elementos se puede observar a continuación en 1.1.

$\cdot$	a	b	c
a	$a \cdot a$	$a \cdot b$	$a \cdot c$
b	$b \cdot a$	$b \cdot b$	$b \cdot c$
c	$c \cdot a$	$c \cdot b$	$c \cdot c$

CUADRO 1.1: Grupos finitos ejemplo 1

Las propiedades de los grupos anteriormente enunciadas se pueden observar gráficamente en una tabla. Las **propiedades cancelativas** son observables fácilmente, al ver que en las filas no hay un elemento repetido ni tampoco en las columnas. Asimismo, en estas tablas tiene que estar representado el elemento neutro, ya que es una parte inherente al grupo. Resulta trivial señalar cuál es el elemento neutro, debido a que tanto su fila como su columna corresponden a los elementos con los que se opera.

Tomando el ejemplo anterior, si el elemento neutro e apareciese en la tabla quedaría del siguiente modo:

$\cdot$	e	b	c
e	e	b	c
b	b	$b \cdot b$	$b \cdot c$
c	c	$c \cdot b$	$c \cdot c$

CUADRO 1.2: Grupos finitos ejemplo 2

Puede observarse también la **propiedad conmutativa** del grupo, si es que la posee. Basta con ver si la tabla resultante es simétrica con respecto a un eje de casillas que representarían a los resultados de operar un elemento consigo mismo. Si se observa esta propiedad eso significa que el grupo es abeliano.

En la tabla 1.3 se puede observar que el grupo sobre el que se opera es un grupo abeliano, debido a que la tabla es simétrica, es decir, $a \cdot b = b \cdot a$. Sobre la creación de esta tabla, se podría pensar que $a \cdot a$ puede ser e ó b, pero como $a \cdot b$ sólo puede ser e, por la propiedad cancelativa, se concluye que $a \cdot a = b$ y $b \cdot b = a$. De hecho, se puede observar que en 1.3 que este grupo es $(\mathbb{Z}_3, +)$, los enteros módulo 3.

$\cdot$	e	a	b
e	e	a	b
a	a	b	e
b	b	e	a

CUADRO 1.3: Grupos finitos ejemplo 3

Otro concepto que será esencial para manejar grupos es el de **homomorfismo**. Estas son funciones entre dos grupos que preservan su estructura algebraica.

Definición 1.2.2 (Homomorfismo de grupos)**.** Sean $(G_1, \cdot)$ y $(G_2, \circ)$ dos grupos y f una aplicación de G_1 en G_2. La aplicación f se dice que es un **homomorfismo de grupos** si, para todo $x, y \in G_1$, $f(x \cdot y) = f(x) \circ f(y)$.

Definición 1.2.3 (Isomorfismo de grupos)**.** Se dirá que f es un **isomorfismo de grupos** si f es biyectiva.

1.3. Subgrupos

Dado un grupo $(G, \cdot)$ y un subconjunto H de G, se dice que H es un subgrupo de $(G, \cdot)$ y se escribe $(H, \cdot) \leq (G, \cdot)$, si H es un grupo con respecto a la operación definida por $\cdot$ en G. Puesto que $(G, \cdot)$ es un grupo, la operación $\cdot$ ya cumple con la condición de asociatividad en todo G; por tanto, para que H sea un subgrupo se tienen que cumplir las siguientes propiedades:

(1) La operación $\cdot$ debe ser cerrada en H.

(2) El elemento neutro de G debe pertenecer también a H.

(3) Para todo $x \in H$, su inverso $x^{-1} \in G$, también pertenece a H.

Como se puede apreciar, son las propiedades de grupos con la excepción de la propiedad de asociatividad que ya ha sido comprobada anteriormente. Desde este punto en adelante, cuando se mencione la operación de los grupos G y H, se utilizará la notación $H \leq G$ para denotar que H es un subgrupo de G.

Ejemplo 1.3.1. $(\mathbb{Z}, +)$ es un subgrupo de $(\mathbb{Q}, +)$, y este a su vez es un subgrupo de los números reales $(\mathbb{R}, +)$.

Ejemplo 1.3.2. Siendo $n\mathbb{Z} = \{nx : x \in \mathbb{Z}\}$, se deduce que el grupo $(n\mathbb{Z}, +)$ es un subgrupo de $(\mathbb{Z}, +)$.

Todo grupo posee dos subgrupos, éstos son: el subgrupo formado por el **elemento neutro** del grupo y el subgrupo formado por **todos los elementos** del grupo. Estos subgrupos son nombrados **subgrupos impropios** del grupo. Al resto de los subgrupos se les denomina **subgrupos propios** del grupo.

Para demostrar que H es un subgrupo de $(G,\cdot)$ tiene que cumplir las propiedades anteriores; sin embargo, es posible demostrarlo sin necesidad de comprobar estas propiedades mediante la siguiente proposición:

Proposición 1.3.3. *Sea $(G,\cdot)$ un grupo y H un subconjunto no vacío de G; H es un subgrupo de $(G,\cdot)$ si y sólo si para todo $x,y\in H$, se da que $x\cdot y^{-1}\in H$.*

Demostración. Se supone que $(H,\cdot)$ es un subgrupo de $(G,\cdot)$. Dados $x,y\in H$, se tiene que $y^{-1}\in H$. En ese caso, $x\cdot y^{-1}\in H$. Se supone, por contra, que se cumple la segunda parte de la proposición; si $x\in H$, $x\cdot x^{-1}=e\in H$; por esto, $e\cdot x^{-1}=x^{-1}\in H$. Por último, si $x,y\in H$, $y^{-1}\in H$, entonces $x\cdot y=x\cdot(y^{-1})^{-1}\in H$. Esto prueba que se cumplen las propiedades anteriores. ■

1.3.1. Construcción y propiedades de subgrupos

A continuación, se presenta una forma de obtener subgrupos partiendo de cualquier subconjunto de éste. Dado un subconjunto S de un grupo $(G,\cdot)$, se denomina **subgrupo generado por S**, y se denota $\langle \boldsymbol{S}\rangle$, al más pequeño de los subgrupos de $(G,\cdot)$ que contienen a S. Según esta definición, se podría deducir que hay dos grandes dificultades: no se sabe si existe tal subgrupo; o si, en caso de existir una manera de generarlo, es fácil de generar ese subgrupo.

La primera dificultad se puede salvar mediante la definición del subgrupo $\langle \boldsymbol{S}\rangle$ como la intersección de todos los subgrupos de $(G,\cdot)$ que contengan a S. Es inmediato verificar que existe tal conjunto. Debido a la proposición 1.3.3 se puede afirmar que esta intersección es un subgrupo.

Lema 1.3.4. *Se tiene que*

$$\langle \boldsymbol{S}\rangle = \bigcap_{H<G,S\subseteq H} H.$$

Demostración. Para abreviar se denotará $\bigcap_{H<G,S\subseteq H}=A$. Se prueba que $\langle \boldsymbol{S}\rangle\subseteq A$, ya que $(A,\cdot)$ es un subgrupo que contiene a S y $\langle \boldsymbol{S}\rangle$ es el subgrupo más pequeño que contiene a S. Sea $(H,\cdot)$ un subgrupo que contiene a S, de la definición de A se deduce que $H\supseteq A$; por tanto, $A\subseteq\langle \boldsymbol{S}\rangle$ y A es el subgrupo más pequeño que contiene a S, es decir, $\langle \boldsymbol{S}\rangle$. ■

La segunda dificultad se responde con la siguiente proposición.

Proposición 1.3.5. *Sea $(G,\cdot)$ un grupo y S un subconjunto de G, se tiene que:*

$$\langle S\rangle=\{x_1^{\alpha_1}\cdot x_2^{\alpha_2}\cdot x_3^{\alpha_3}\cdot\ldots\cdot x_n^{\alpha_n}: x_1,x_2,x_3,\ldots,x_n\in S,\ \alpha_1,\alpha_2,\alpha_3,\ldots,\alpha_n\in\mathbb{Z}\}.$$

Demostración. Sea B el subconjunto de G que se encuentra a la derecha de la igualdad que se quiere demostrar; B es un subgrupo de $(G,\cdot)$ debido a que cumple la proposición 1.3.3. Debido a la definición de $\langle \boldsymbol{S}\rangle$ como el subgrupo más pequeño que contiene a S, y B contiene a S, B contiene a su vez a $\langle \boldsymbol{S}\rangle$.

Falta probar la inclusión inversa. Puesto que $\langle \boldsymbol{S} \rangle$ es un subgrupo y contiene a S, si las $x \in S$, entonces el resultado de la expresión de la derecha también pertenece a $\langle \boldsymbol{S} \rangle$ por ser un subgrupo, y por tanto, $\langle \boldsymbol{S} \rangle$ contiene a B. ■

Si el conjunto S está formado por una cantidad finita de elementos, eso significa que esos elementos generan el conjunto mediante operaciones entre sí. De manera similar, se puede definir un subgrupo de $(G, \cdot)$ generado a partir de un elemento x denotado como $\langle \boldsymbol{x} \rangle$.

Definición 1.3.6. Un grupo $(G, \cdot)$ se dice cíclico si existe al menos un elemento $x \in G$ tal que el subgrupo generado por ese elemento sea G, es decir, $\langle \boldsymbol{x} \rangle = G$. En este caso x pasará a denominarse un **generador** de G.

Definición 1.3.7. Dado un grupo $(G, \cdot)$ y un elemento $x \in G$, se define el **orden de $\boldsymbol{x}$** como el número de elementos que posee el subgrupo generado por ese elemento, $\langle \boldsymbol{x} \rangle$, si es finito. Si es infinito, el orden también lo será.

Proposición 1.3.8. *Sea $(G, \cdot)$ un grupo finito y x un elemento de G; existe un entero positivo n tal que $x^n = e$ y $\langle \boldsymbol{x} \rangle = \{x, x^2, x^3, \ldots, x^{n-1}, x^n = e\}$.*

Demostración. Debido a que $(G, \cdot)$ es un grupo finito, el conjunto de las potencias de x debe tener repeticiones, de lo contrario sería infinito; por tanto, existen enteros i, j, tal que $x^i = x^j$. Por tanto, se deduce que hay un n tal que $x^n = e$. En efecto, $e = x^i \cdot x^{-i} = x^j \cdot x^{-i} = x^{j-i}$. Así, $n = j–i$ suponiendo que $j > i$.

A partir de la proposición 1.3.5 se puede definir el subgrupo generado por x como potencias del mismo elemento x. Se considera $x^k = x$, siendo $k > n$, se escribe $k = cn + r$, con $c, r \in \mathbb{N}$; entonces:

$$x^k = x^{cn} \cdot x^r = (x^n)^c \cdot x^r = e^c \cdot x^r = e \cdot x^r = x^r.$$

De este modo se pueden eliminar todos los elementos mayores que n porque son repeticiones, se puede probar de manera similar que los x^{-i}, con $i > 0$, también son repeticiones.

Si $(G, \cdot)$ es un grupo finito y n el menor número entero tal que $x^n = e$, entonces todos los elementos x^i, tal que $i < n$, son distintos. Se puede probar así: se supone que $x^i = x^j$, para algunos $1 \leq i \leq j \leq n$; se deduce que $x^{j-i} = x^j \cdot x^{-i} = x^i \cdot x^{-i} = e$, como $j–i < n$ y n es el menor número positivo que cumple $x^n = e$, se concluye que $j = i$. ■

Corolario 1.3.9. *Si $(G, \cdot)$ es un grupo finito y x un elemento de G, el orden de x coincide con el menor número entero positivo n tal que $x^n = e$. Además, todos los elementos x^i, $1 \leq i \leq n$, son distintos.*

Corolario 1.3.10. *Sea $(G, \cdot)$ un grupo finito y x un elemento de G de orden finito k; si m es un entero positivo tal que $x^m = e$, se tiene que k divide a m.*

Demostración. Si $x^m = e$, se puede escribir $m = ck + r$ siendo $0 \leq r < k$, y por tanto, $e = x^m = x^{ck} \cdot x^r = x^r$, como se comprobó anteriormente. Como $r < k$ y k el menor número entero que satisface $x^k = e$, eso significa que $r = 0$, y $m = ck$; lo que prueba que k divide a m. ■

1.4. Teorema de Lagrange

Desde este punto no se indicará, más que cuando sea necesario, la operación definida en un grupo, y la operación de dos elementos del mismo grupo x e y será simbolizada como xy.

A partir de la observación del Corolario 1.3.10 y de los órdenes de sus elementos se observa una propiedad que resulta ser general.

Proposición 1.4.1. *Si G es un grupo finito y x un elemento de G, el orden de x es un **divisor** del número de elementos de G, es decir, de su orden.*

Demostración. Sea k el orden de x y n el número de elementos de G. Se puede obtener una partición de G en m subconjuntos, tal que m sea el número de elementos de G; estos subconjuntos son definidos como el resultado de operar cada elemento de $\langle \boldsymbol{x} \rangle$ con un elemento de G, g_i. Cada subconjunto es disjunto del resto, debido a que cada subconjunto resulta de la operación de los elementos de $\langle \boldsymbol{x} \rangle$ con un elemento distinto de G, de este modo se obtienen diferentes subconjuntos que poseen el mismo número de elementos, el orden de x. Entonces, $|\langle \boldsymbol{x} \rangle| m = |\boldsymbol{G}| = n$; lo cual prueba el resultado deseado. ■

La demostración anterior está ligada a la noción de particiones de un conjunto; este concepto a su vez está íntimamente ligado con el concepto de relaciones de equivalencia definidas en un conjunto. Esta afirmación sugiere que la demostración podría haberse hecho definiendo una relación de equivalencia en G. Observando la demostración desde el punto de la estructura de $\langle x \rangle$, no parece que la demostración se encuentre ligada a su estructura. Por lo que, la demostración se podría hacer para cualquier subgrupo H de G.

Así, se vislumbra que la afirmación de que "*el número de elementos de un subgrupo H de G divide al número de elementos de G*" puede ser cierta siempre. Esta afirmación se conoce como el **Teorema de Lagrange**. El resto de la sección irá orientado a definir los conceptos involucrados a demostrar este resultado.

Definición 1.4.2. Si G es un grupo, H es un subgrupo de G y x un elemento de G, se define $xH = \{xh : h \in H\}$.

Ejemplo 1.4.3. Si en $(\mathbb{Z}, +)$ se considera el subconjunto $5\mathbb{Z}$, se tiene que

$$2 + 5\mathbb{Z} = \{2 + 5z : z \in \mathbb{Z}\} = 7 + 5\mathbb{Z}.$$

Proposición 1.4.4. *Si G es un grupo finito, H es subconjunto de G y x es un elemento de G, el número de elementos de xH coincide con el número de elementos de H.*

Demostración. La aplicación $f : H \Rightarrow xH$ dada por $f(h) = xh$ es una biyección. Esto es debido a que si $f(h_1) = f(h_2)$ para $h_1, h_2 \in H$, entonces $xh_1 = xh_2$ y debido a la propiedad cancelativa por la izquierda se tiene que $h_1 = h_2$. Además, si $xh \in xH$, basta observar que $f(h) = xh$ para tener que, todo $h \in H$ tiene una imagen en xH dada por $f(h)$, y por tanto, es suprayectiva. ■

Definición 1.4.5. Sea G un grupo y H un subgrupo de G; se dirá que dos elementos $x, y \in G$ están relacionados mediante H, y se escribirá $x \equiv y \mod H$, si $x^{-1}y \in H$.

Proposición 1.4.6. *Sea G un grupo y H un subgrupo de G; la relación definida en la definición 1.4.5 es una relación de equivalencia y la clase de equivalencia de un elemento $x \in G$ en esta relación coincide con xH.*

Demostración. Se va a demostrar que la relación definida por la definición 1.4.5 es, en efecto, una relación de equivalencia, para ello es necesario probar sus tres propiedades: reflexión, simetría y transitividad.

Puesto que H es un subgrupo de G, $x^{-1}x = e \in H$ para todo $x \in G$; por tanto, $x \equiv x \mod H$ y queda probada la propiedad reflexiva. Si $x \equiv y \mod H$ se tiene que $x^{-1}y \in H$; como $H \leq G$, $y^{-1}x \equiv (x^{-1}y)^{-1} \in H$, y por tanto, $y \equiv x \mod H$, con lo que se cumple la propiedad simétrica. Sean ahora x, y, z elementos de G y se supone que, $x \equiv y \mod H$, $y \equiv z \mod H$; se tiene que $x^{-1}y \in H$, e $y^{-1}z \in H$; debido a que $H \leq G$ se tiene que $x^{-1}z \equiv (x^{-1}y)(y^{-1}z) \in H$, así queda probada la propiedad transitiva.

Así queda demostrada la relación de equivalencia definida. A continuación, se va a probar la segunda parte de la proposición.

Sea $x \in G$ y $[x]$ la clase de equivalencia de x, se tiene que:

$$[x] = \{y \in G : x = y \mod H\} = \{y \in G : x^{-1}y \in H\} = \{y \in G : y \in xH\} = \{xh : h \in H\} = xH.$$

■

Teniendo lo anterior demostrado se puede proceder a probar el Teorema de Lagrange.

Teorema 1.4.7 (Teorema de Lagrange)**.** *Si G es un grupo finito y $H \leq G$, el número de elementos de H divide al número de elementos de G.*

Demostración. La relación definida anteriormente se ha demostrado ser de equivalencia; por eso, se tiene que las clases de equivalencia de esta relación establecen una partición de G; puesto que, se ha demostrado que las clases de equivalencia coinciden con xH. Se tiene que G es la unión de todos los subconjuntos x_iH, tal que $x_1H, \ldots, x_mH \in G$ y todas las x_i son diferentes, entonces todos los subconjuntos serían disjuntos, es decir, no comparten ningún elemento. Como todos los subconjuntos tienen el mismo número de elementos, $|\boldsymbol{H}|$, se tiene que:

$$|\boldsymbol{G}| = |x_1H| + |x_2H| + |x_3H| + \ldots + |x_mH| = m|H|.$$

Y así queda demostrado el teorema. ■

Corolario 1.4.8. *Si G es un grupo de orden p, con p primo,* ***G es cíclico****.*

Demostración. Sea $x \in G$ con $x \neq e$, el conjunto $\langle x \rangle$ es un subgrupo de G distinto de $\{e\}$, por el Teorema de Lagrange 1.4.7, $|\langle \boldsymbol{x} \rangle|$ divide a p. Puesto que p es primo, sus únicos divisores son 1 y p; como $\langle x \rangle \neq \{e\}$ que es el único subgrupo con un elemento, $|\langle \boldsymbol{x} \rangle|$ está obligado a ser mayor que 1. Por ello $|\langle \boldsymbol{x} \rangle| = p$, y por tanto, x es un **generador de G**. Como se dijo en la definición 1.3.6 si existe un generador en G, entonces este grupo es **cíclico**. ■

En las condiciones del Teorema de Lagrange 1.4.7, al número entero m, tal que $|\boldsymbol{G}| = m|\boldsymbol{H}|$, se le llama **índice de $\boldsymbol{H}$ en $\boldsymbol{G}$** y se representa como $[G : H]$. A los xH se les llama **clases de equivalencia por la izquierda módulo $\boldsymbol{H}$**, de la misma manera se puede demostrar el Teorema de Lagrange 1.4.7 con clases de equivalencia por la derecha módulo H, y definirlas también de manera similar a las anteriores.

1.5. Ejemplos de grupos

En esta sección se revisarán distintos ejemplos de grupos, desde los convencionales como $(\mathbb{Z}, +)$ hasta las simetrías de un rectángulo y la operación de composición de figuras geométricas: "$\circ$".

1.5.1. Grupos de congruencias

Este tipo de grupos consisten en la definición de un grupo de m elementos. Sea m cualquier número entero positivo. Para la definición de estos grupo se necesita primero la definición de congruencia. Dados dos números a y $b \in \mathbb{Z}$, a se dirá que es **congruente con $\boldsymbol{b}$ módulo $\boldsymbol{m}$**, y se denotará $a \equiv b \mod m$; si $a - b$, es decir, su diferencia es un múltiplo de m, $a - b = km$ tal que $k \in \mathbb{Z}$.

Esta relación de **congruencia** es una relación de equivalencia, a continuación se demostrará. Como se ha mencionado anteriormente se necesita probar que la relación tiene las propiedades reflexiva, transitiva y simétrica para que sea clasificada como **de equivalencia**.

Esta relación se prueba reflexiva así: si $a \equiv a \mod m$, entonces $a - a = km$ para algún $k \in \mathbb{Z}$, a saber, $k = 0$. Por tanto, la propiedad reflexiva la cumple. En cuanto a la propiedad transitiva se demuestra teniendo a, b y $c \in \mathbb{Z}$, tal que $a \equiv b \mod m$ y $b \equiv c \mod m$. Esto significa que $a - b = km$ y $b - c = tm$, siendo $t, k \in \mathbb{Z}$ y como b es congruente con c se tiene que, $a - c = ktm$ y por tanto, $a \equiv c \mod m$. La propiedad simétrica es demostrada teniendo $a, b \in \mathbb{Z}$, tal que $a \equiv b \mod m$ y hay que probar que $b \equiv a \mod m$. Si $a \equiv b \mod m$ se tiene que $a - b = km$, además, si se multiplica por -1 a ambos lados resulta en $b - a = -km \Rightarrow b \equiv a \mod m$, que es lo que se quería demostrar.

Dado que la relación de congruencia se ha demostrado como una relación de equivalencia se puede definir el **conjunto cociente de** $\mathbb{Z}$ sobre esta relación, creando el **conjunto de las clases de congruencias módulo m**, que se denotará como $\mathbb{Z}_m$.

Los elementos del conjunto $\mathbb{Z}_m$ son clases de equivalencia denotadas como $[a]$, donde esta clase de equivalencia agrupa a todos los elementos b, tal que $b \equiv a \mod m$. Dada una clase de equivalenca $[a]$ se puede elegir a un representante de esa clase, b siendo $1 \leq b < m$ y $b \in [a]$. En el caso en el que b sea mayor que m habrá que dividir entre el módulo y coger el resto que pertenecerá a la clase de equivalencia de b. De este modo se puede describir el conjunto $\mathbb{Z}_m$ como el conjunto de todas las clases de equivalencia hasta $m-1$.

$$\mathbb{Z}_m = \boldsymbol{\{[0], [1], [2], [3], \ldots, [m-1]\}}$$

Se puede observar que este conjunto está formado por todos los restos posibles resultantes de dividir entre m. Debido a esta propiedad, las operaciones de multiplicación y suma están definidas y son cerradas en $\mathbb{Z}_m$. Si se usa al conjunto $\mathbb{Z}_m$ junto con la operación de suma se obtiene el grupo abeliano $(\mathbb{Z}_m, +)$, del mismo modo que se demostró en la sección de definición de grupos 1.1.2 con el conjunto de enteros $\mathbb{Z}$. Así queda demostrado que para cualquier número entero positivo m existe un grupo con m elementos.

Observación 1.5.1. Respecto a la operación de multiplicación, $(\mathbb{Z}_m, \cdot)$ no es un grupo; debido a que el 0 carece de inverso, como se explicó en la sección de definición de grupos 1.1.3. Se podría pensar que si se quita el $[0]$ del conjunto, creando $(\mathbb{Z}_m^*, \cdot)$ sería un grupo; sin embargo, esto no es cierto para la mayoría de los casos.

Ejemplo 1.5.2. Por ejemplo, ni [2], ni [3], ni [4] poseen inverso en $(\mathbb{Z}_6^*, \cdot)$. Se puede comprobar tan sólo operando el 2 con todos los números del 1 al 5.

Se puede observar que estos números tienen algo en común con el módulo y es que comparten divisores, es decir, su máximo común divisor es mayor que 1. Por tanto, se puede llegar a la conclusión de que $(\mathbb{Z}_m^*, \cdot)$ es un grupo si y solo si m es un número primo. Se puede llegar a esta conclusión de este modo: Dados $s, r \in \mathbb{Z}$, donde $s, r > 1$.

$$\text{Si } m = s \cdot r, \text{ se tiene que } [s] \cdot [r] = [m] = [0] \text{ en } (\mathbb{Z}_6^*, \cdot).$$

En particular, para 1.5.2 se tiene que $m = 6 = 3 \cdot 2$. Se recuerda que el grupo en el que se opera es el $(\mathbb{Z}_6^*, \cdot) = G$. Dado que $3, 2 \in G$, todos los resultados de la operación $\cdot$ también estarán en G porque la operación es cerrada, por la propiedad (1) de los grupos, 1.1. Pero $3 \cdot 2 = 6 = 0 \mod 6$, y $0 \notin G$. Por tanto, G no es un grupo.

Este ejemplo se formaliza en la siguiente proposición.

Proposición 1.5.3. *Para todo número entero positivo p, $(\mathbb{Z}_p^*, \cdot)$ es un grupo abeliano con $p-1$ elementos si y sólo si p es* ***primo****.*

Demostración. La operación $\cdot$ está definida en el conjunto $\mathbb{Z}_p$, pero no está definida en $\mathbb{Z}_p^*$, así que primero hay que demostrar que es cerrada en este conjunto; para lo cual solo es necesario demostrar que $[a] \cdot [b] = [0]$ es imposible, si $[a], [b] \in \mathbb{Z}_p^*$. En efecto $[a] \cdot [b] = [0]$, significa que $ab = kp$ para algún $k \in \mathbb{Z}$; en ese caso, p divide a ab. En ese caso, como p es primo eso significa que ó a ó b tienen a p como divisor, lo que significa que serían un múltiplo de p, $[a] = [0]$ o $[b] = [0]$, lo que va en contra de que $[a], [b] \in \mathbb{Z}_p^*$.

La clase $[1]$ corresponde al elemento neutro; respecto a las propiedades asociativa y conmutativa la operación $\cdot$ las cumple por heredarlas de $\mathbb{Z}_n$. Para probar la existencia del elemento inverso, es necesario acudir al máximo común divisor y al **Teorema de Bezout**. Este teorema demuestra que siendo $x = \gcd(k, n)$, existen $a, b \in \mathbb{Z}$, tal que $x = ak + bn$.

Recurriendo a la igualdad anterior, se puede llegar a que todo elemento $k \in \mathbb{Z}_p^*$ y $k > 1$ tiene un **elemento inverso** en el grupo. Al ser p un número primo, el máximo común divisor entre k y p es 1, y por Bezout:

$$[1] = [a][k] + [b][p] \Rightarrow [1] = [a][k].$$

Así se llega a que todo elemento de $\mathbb{Z}_p^*$ tiene un elemento inverso que sería $[a]$. La proposición recíproca también es cierta y se prueba utilizando la igualdad del ejemplo 1.5.2. Concretamente, si $(\mathbb{Z}_m^*, \cdot)$ es un grupo abeliano, entonces m es primo. Como se dice en el ejemplo 1.5.2, si se tiene $r, s \in \mathbb{Z}$, mayores que 1, tales que $r \cdot s = m$, entonces se da que $[r] \cdot [s] = [m] = [0]$, pero $[0] \notin \mathbb{Z}_m^*$, por lo que la operación de multiplicación de clases no estaría cerrada en $(\mathbb{Z}_6^*, \cdot)$. Lo que contradice la propiedad (1) de los grupos en 1.1. ■

Otro resultado de gran importancia cuando lidiamos con congruencias es el llamado Teorema Chino del Resto. Este permite resolver sistemas de congruencias lineales, al estilo de los métodos clásicos de álgebra lineal. Su utilización será clave en el algoritmo MOV de la sección 5.2.1.

Teorema 1.5.4 (Teorema Chino del Resto)**.** *Dados dos conjuntos de números* $A = \{a_1, \ldots, a_n\}$ *y* $M = \{m_1, \ldots, m_n\}$*, existe una solución* x *para las siguientes* n *congruencias:*

$$\begin{aligned} x &\equiv a_1 \bmod m_1 \\ x &\equiv a_2 \bmod m_2 \\ &\ldots \\ x &\equiv a_n \bmod m_n \end{aligned}$$

si $a_i \equiv a_j \bmod \gcd(m_i, m_j)$ *y la solución será única módulo* $L = \operatorname{lcm}(m_1, m_2, \ldots, m_n)$.

1.5.2. Grupos de permutaciones

Previamente a dar una definición de los grupos de permutaciones, es necesario definir los grupos de biyecciones para un determinado conjunto.

Dado un conjunto A, se define $B(A)$ como el conjunto de todas las biyecciones de A en sí mismo. Sean f y g dos aplicaciones biyectivas del conjunto A en sí mismo, a su vez, puede comprobarse que $g \circ f$ es también una aplicación biyectiva en A, donde $\circ$ denota la operación de composición de aplicaciones. En relación a demostrar por qué esta aplicación es biyectiva, hay que demostrar que todo elemento tiene una imagen correspondiente en el conjunto objetivo (suprayección), y que elementos distintos tienen imágenes distintas (inyección).

En cuanto a la inyección, si se tiene $x, y \in A$ y se tiene que $g \circ f(x) = g \circ f(y)$, es lo mismo que tener $g(f(x)) = g(f(y))$. Debido a que g es inyectiva, si esos dos términos son iguales significa que $f(x) = f(y)$; como f también es inyectiva, se tiene que $x = y$, lo que demuestra que $g \circ f$ es un aplicación inyectiva. Respecto a la suprayección, para demostrarla sólo se tiene que probar que para un elemento cualquiera $z \in A$, existe un $y \in A$, tal que $g \circ f(y) = z$. Como f es suprayectiva se tiene que existe $x \in A$, tal que $f(x) = y$. Se sustituye en $g \circ f$, y se tiene que $g(f(x)) = z$, lo que demuestra que $g \circ f$ es una aplicación suprayectiva; y por tanto, biyectiva, al ser también inyectiva. Además al ser biyectiva, se tiene que la operación de composición de aplicaciones es cerrada en el conjunto $B(A)$.

Proposición 1.5.5. *Dado un conjunto A, $(B(A), \circ)$ es un grupo.*

Demostración. Como se ha demostrado previamente, la operación de composición es cerrada en $B(A)$. La operación en cuestión es también asociativa:

$$g \circ (f \circ h) = (g \circ f) \circ h \Rightarrow g \circ f(h) = g(f) \circ h \Rightarrow g(f(h)) = g(f(h)).$$

El elemento neutro del grupo es la aplicación identidad en A, que se simboliza como I_A. Por último, con respecto al elemento inverso de $f \in B(A)$, si existe $f^{-1} \in B(A)$ es también una biyección. Si $f^{-1}(x) = f^{-1}(y)$, se tiene que $f(f^{-1}(x)) = (f^{-1}(y))$, y queda que $x = y$, ya que f º f^{-1} = I; por lo tanto, f^{-1} es inyectiva. Si se tiene $y \in A$, $x = f(y)$ por ser f suprayectiva; además, este elemento satisface $f^{-1}(x) = f^{-1}(f(y)) = y$, por lo que f^{-1} también es suprayectiva. Así queda demostrado que $(B(A), \circ)$ es un grupo. ■

Teniendo definido el grupo de biyecciones, la definición del grupo de permutaciones resulta sencilla. El grupo de permutaciones resulta en el momento en el que el conjunto A es el conjunto de los n primeros números naturales, $A = \{1, 2, \ldots, n\}$. De este modo el conjunto $B(A)$ se le conoce como **las permutaciones de n elementos**; este conjunto se denota S_n y el grupo como $(S_n, \circ)$, y se denomina el **grupo simétrico de n elementos**.

Un elemento del conjunto S_n se define completamente en el momento en el que se conocen todas las imágenes de los n primeros números naturales. Si se tiene $\sigma \in S_n$, se puede escribir de este modo:

$$\sigma = \begin{pmatrix} 1 & 2 & \ldots & n \\ \sigma(1) & \sigma(2) & \ldots & \sigma(n) \end{pmatrix}$$

La operación de composición se simbolizará con $\circ$. Unos ejemplos de permutaciones serían:

$$\alpha = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 4 & 2 & 1 \end{pmatrix}$$

$$\beta = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 1 & 3 \end{pmatrix}$$

Como se puede observar estos dos elementos pertenecen al conjunto de permutaciones de S_4. Para añadir otra referencia el elemento neutro del grupo que forma S_4 con $\circ$ sería de esta forma:

$$I = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \end{pmatrix}$$

Para que quede claro cómo se operaría en este grupo se mostrará un ejemplo de composición con los ejemplos anteriores a continuación y se señalarán los puntos claves.

$$\alpha \circ \beta =$$

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 4 & 2 & 1 \end{pmatrix} \circ \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 1 & 3 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 1 & 3 & 2 \end{pmatrix}$$

La operación es una composición, así que habría que dejar observado que primero hay que permutar los valores en base a β y después en base a α. En caso de que sea la operación en el orden inverso, el orden también se invierte.

Para un mejor entendimiento, observando el resultado asociado al número 1. Éste resulta de permutar primero en β; y dado que $\beta(1) = 2$ y el $\alpha(2) = 4$; la imagen del número 1 en $\alpha \circ \beta$ es 4, $\alpha \circ \beta(1) = 4$. Así se completaría el resultado con los valores restantes. La operación en el orden inverso se realiza de este modo: $\beta \circ \alpha =$

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 1 & 3 \end{pmatrix} \circ \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 4 & 2 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 3 & 4 & 2 \end{pmatrix}$$

Como se puede percibir, esta operación no resulta conmutativa; por tanto, el grupo $(S_4, \circ)$ es un **grupo no abeliano**. Esto ocurre para todos los grupos de permutaciones cuyo número n sea $n \geq 3$. Asimismo, también es digno de mención el hecho de que el número de elementos de S_n sea $n!$. Estos grupos son fácilmente generados mediante la composición de dos elementos del grupo, incluso consigo mismos. Sin embargo, conforme se va avanzando en esta generación del grupo se puede observar que si se compone un mismo elemento consigo mismo varias veces se puede obtener el elemento neutro, I. El número de veces puede variar, pero al menos existe un elemento que genera el elemento neutro en n composiciones consigo mismo.

1.5.3. Grupos ligados a configuraciones geométricas planas

Para definir estos grupos, previamente se necesita conocer primero todas las operaciones posibles en el plano. Estas operaciones son llamadas **movimientos** y consisten en los siguientes:

(1) Traslaciones: son las transformaciones que deslizan un vector del plano en base a otro vector fijo. Este vector de deslizamiento se denomina **vector de traslación**.

(2) Rotaciones: son movimientos que hacen corresponder un punto con otro, mediante la rotación alrededor de un punto en el plano y con un determinado ángulo.

(3) Simetrías: son movimientos que transforman a un punto reflejando especularmente respecto a una recta dada, llamada eje de simetría.

Dado un conjunto A, se llamará $S(A)$ a todos los movimientos en el plano que dejen a A inmutable, es decir, todo movimiento M, que cumple $M(A) = A$. Este conjunto se denomina **el conjunto de las simetrías de A**.

Proposición 1.5.6. *El conjunto de las simetrías del conjunto A del plano junto con la operación de composición de movimientos es un grupo, $(S(A), \circ)$.*

Demostración. La composición de simetrías es cerrada en $S(A)$ por definición, ya que cualquier elemento de $S(A)$ deja a A invariante. En efecto, sean $F, G \in S(A)$ se tiene que $F \circ G(A) = F(G(A)) = F(A) = A$, y por tanto, la composición de dos elementos es también un elemento de las simetrías. Como ya se ha demostrado en la proposición 1.5.5, la operación de composición es asociativa. El elemento neutro es el movimiento identidad, I. Si $R(A)$ es una simetría de A, entonces R^{-1} también debido a que $R^{-1}(A) = R^{-1}(R(A)) = A$; por tanto, $R^{-1} \in S(A)$. ■

Si se considera el grupo de las simetrías tomando como figura geométrica en el plano a un triángulo equilátero, se pueden observar varios tipos de movimientos, como la rotación, que se muestra en la ilustración. Hay tres rotaciones que dejan invariante a este triángulo utilizando el centro del triángulo como eje.

Proposición 1.5.7. *Para todo polígono regular de n lados, sea A la rotación de $\dfrac{2\pi}{n}$ radianes hacia la izquierda y B la simetría de un eje que intersecta un vértice del polígono y el centro; las composiciones de estos movimientos, tal que*

$$A \circ B, A^2 \circ B, \ldots, A^{n-1} \circ B$$

son todas sus simetrías.

Demostración. Si se asignan los n primeros números naturales a cada vértice del polígono de n lados, el n-gono, el movimiento A queda representado así:

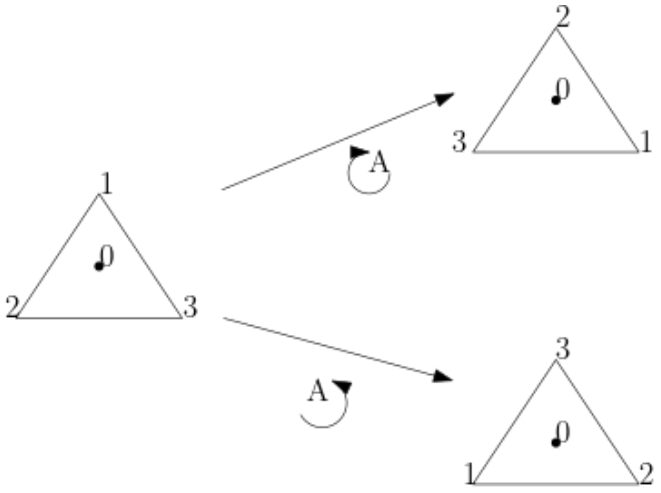

FIGURA 1.1: Simetrías de un triángulo equilátero

$$A(i) = i + 1 \text{ si } 1 \leq i < n$$
$$A(n) = 1 \text{ Caso final.}$$

Debido a que A es una rotación, las ecuaciones anteriores definen al movimiento de la rotación para cualquier polígono regular con n vértices. La simetría B:

$$B(j) = n - j + 2 \text{ si } 1 < j \leq n$$
$$B(1) = 1 \text{ Caso base.}$$

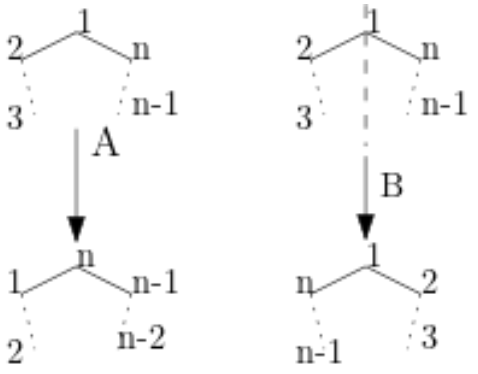

FIGURA 1.2: Simetrías de un polígono regular

Como se puede observar en la imagen, el movimiento A corresponde a la rotación sobre el centro del n-gono y el movimiento B corresponde a la simetría del polígono sobre el eje dibujado. Como breve aclaración de la notación de la figura: la fórmula anterior indica que el número i pasará a simbolizar, en vez del vértice i, el vértice $i + 1$. Sobre la simetría B, se trata simplemente de reflejar los vértices sobre el eje de simetría.

Observación 1.5.8. Si n es impar pasa por un único vértice y si es par pasa por dos vértices.

A continuación se demostrará que $A \circ B$ es una simetría de un eje que cruza el segmento que une los vértices 1 y 2 y el centro. Esta simetría S se define mediante estas ecuaciones:

$$S(j) = n - j + 3 \text{ si } 3 \leq j \leq n$$
$$S(1) = 2 \text{ y } S(2) = 1.$$

Si $3 \leq j \leq n$, entonces se da $A \circ B(j) = A(n - j + 2) = n - j + 2 + 1 = S(j)$; sin embargo, se da que $i < 3$, si por ejemplo, $i = 1$, entonces $A \circ B(1) = A(1) = 2$; si $i = 2$, $A \circ B(2) = A(n-2+2) = A(n) = 1$. Así queda demostrado que las ecuaciones anteriores definen tal simetría.

Si se realiza el movimiento A dos veces, el resultado de $A^2 \circ B$ es una simetría con respecto a un eje que cruza el centro del polígono y el vértice 2. La demostración de esta simetría y del resto del enunciado se prueban de manera similar. Así, estas son los n simetrías del polígono de n lados, el n-gono. ■

Capítulo 2

Teoría de Anillos

Anteriormente se estudiaron los grupos y sus propiedades, así como los distintos tipos de grupos que pueden existir. A continuación se estudiarán lo que se conoce como anillos. Estos anillos serán realmente útiles a la hora de definir qué son los cuerpos finitos y sus distintos usos y propiedades. A su vez, los cuerpos finitos serán la base donde se definirán las curvas elípticas, objeto de estudio. Este capítulo se basa en una revisión bibliográfica de [Nav16].

2.1. Definición de anillos

Los anillos son conjuntos no vacíos con dos operaciones binarias, al contrario que los grupos que solo poseían una, que satisfacen las siguiente condiciones:

Sea R un conjunto no vacío con las operaciones $r + s$ (suma) y rs (multiplicación), siendo $r, s \in R$. R se dirá que es un anillo si:

(1) $(R, +)$ es un grupo abeliano con neutro 0.

(2) Cumple la propiedad distributiva:

$$(r + s)t = rt + st, r(s + t) = rs + rt \text{ para } r, s, t \in R$$

(3) Cumple la propiedad asociativa respecto a la multiplicación:

$$(rs)t = r(st) \text{ para } r, s, t \in R$$

Al cumplir las anteriores condiciones, R se califica como anillo y de ahora en adelante se simbolizará simplemente como el conjunto R, dando a entender implícitamente que las operaciones binarias serán las de la suma y la multiplicación. R se denomina **anillo con identidad** si existe $1 \in R$ tal que, $1r = r1 = r$ para todo elemento del conjunto. En otras palabras, estos anillos poseen un elemento

neutro multiplicativo. De ahora en adelante, todos los anillos mostrados a continuación se supondrán como anillos con identidad.

A modo de entidad paralela a los grupos abelianos en **Teoría de grupos** 1, aquí se tienen los anillos **conmutativos**. Estos anillos son los anillos, R, en los cuales sean $r, s \in R$, se da que

$$rs = sr.$$

En este capítulo se centrará la atención en ellos.

Dentro de los anillos conmutativos se puede encontrar un caso especial que resultará fundamental más adelante en la definición de curvas elípticas, **los cuerpos**.

Definición 2.1.1. Se dice que un anillo conmutativo R no cero (es decir, que no tenga como único elemento el 0), es un **cuerpo** si para todo elemento $r \in R - \{0\}$, existe otro elemento $s \in R - \{0\}$ que cumpla

$$rs = 1.$$

En otras palabras un anillo es un cuerpo si el conjunto del anillo menos el 0, R^*, y la operación multiplicación forman un grupo. R^* ciertamente denota a un conjunto sin el 0, pero en este caso, y de ahora en adelante, denotará a un anillo carente del elemento 0.

Si R es un anillo, se puede definir su **anillo de polinomios**, $R[x]$. Esta entidad resultará muy útil más adelante para la creación de cuerpos finitos sobre los que definir una curva elíptica.

Definición 2.1.2. El **anillo de polinomios** sobre R, $R[x]$, es el conjunto de elementos de la forma:

$$\sum_{n=0}^{m} a_n x^n \text{ con } a_n \in R.$$

Definición 2.1.3. Sea $p(x)$ un polinomio. Si m es el máximo entero tal que $a_m \neq 0$, se dice que p tiene **grado m** y se simboliza $\delta(p) = m$. Todas las a_n se denominan **coeficientes del polinomio** y el **coeficiente director** de p es a_m. Si el coeficiente director es 1, se dice que el polinomio es **mónico**. Los polinomios cuyos coeficientes son cero, salvo para $n = 0$, se denominan constantes. Para expresar mejor todas estas ideas se modela un polinomio de esta forma:

$$p = \sum a_n x^n = a_0 + a_1 x + a_2 x^2 + \ldots + a_m x^m.$$

El polinomio cero es aquel polinomio cuyos coeficientes son todos cero. Para completar la formulación de un anillo de polinomios hace falta definir sus operaciones binarias: suma y multiplicación. Si se tienen dos polinomios, p y q de la forma

$$p = \sum_{n=0}^{m} a_n x^n, \; q = \sum_{n=0}^{l} b_n x^n$$

Se dice que $p = q$, si para todo $n \geq 0$, $a_n = b_n$. Con esto definido se puede pasar a definir las operaciones.

Las operaciones de los polinomios son ampliamente conocidas, pero se da un breve repaso para situar el estudio correctamente. Los polinomios se pueden sumar, y se hace de esta manera:

$$p + q = \sum_{s=0}^{max(n,l)} (a_s + b_s)x^s.$$

De forma similar, se pueden multiplicar:

$$pq = \sum(\sum a_i b_j)x^n, \text{ siendo } i + j = n \text{ y } n \geq 0.$$

Como se puede observar la suma es conmutativa, por tanto, el grupo $(R[x], +)$ es abeliano. Asimismo, es inmediato comprobar que se cumplen la propiedad distributiva y la propiedad asociativa para la multiplicación. Estas operaciones como se puede ver, cambian el grado del polinomio resultado, y lo hace del siguiente modo: Sean $p, q \in R[x]$,

$$\text{para la suma: } \delta(p + q) \leq max(\delta(p), \delta(q));$$
$$\text{para la multiplicación: } \delta(pq) = \delta(p) + \delta(q).$$

En la suma se podría pensar que es simplemente el máximo de los grados de los polinomios, pero en realidad no es así, ya que se podrían sumar coeficientes negativos y anular dicho término. En el caso de la multiplicación sucede algo parecido, la afirmación anterior es cierta siempre que $ab \neq 0$, lo que conecta con el siguiente concepto, el de **dominio de integridad**. Este concepto ayudará a clasificar los anillos según una propiedad muy importante para la definición de cuerpos finitos.

Definición 2.1.4. Se dice que un anillo conmutativo no cero R es un **dominio de integridad** si y solo si cuando para todo $r, s \in R$ se tiene que $rs = 0$ implica que $r = 0$ ó $s = 0$.

Esto viene a ser lo mismo a decir que no existen **divisores de 0** en el anillo, ya que si se tiene que $rs = 0$ y ni r ni s son 0 significa que ambos cumplen que "dividen" a cero.

En el ejemplo anterior, si R es un dominio de integridad, en efecto, la afirmación de la multiplicación se cumple.

Ejemplo 2.1.5. Un ejemplo de dominio de integridad son $\mathbb{Z}$, $\mathbb{Q}$ ó $\mathbb{R}$, es decir, los conjuntos de los números enteros, los racionales y los reales.

Ejemplo 2.1.6. Un ejemplo de no dominio de integridad sería $\mathbb{Z}_6$, en el que $[3] * [2] = [6] = [0] \in \mathbb{Z}_6$.

Siendo R un anillo conmutativo no cero se tiene que sus elementos **inversibles** son aquellos que cumplan la propiedad mencionada anteriormente de tener un elemento en el anillo con el que, si se multiplican, da el elemento identidad del anillo. También se llama **unidades** a estos elementos. Si de

un cuerpo se obtiene su anillo de polinomios las unidades de este anillo serán los polinomios constantes no cero.

Sea R un anillo y S un subconjunto de R que contenga a la identidad. Se dice que S es un **subanillo** de R si, para dos elementos de S, su multiplicación y resta pertenecen a S. Si R es un cuerpo y S un subanillo de R, entonces se dice que S es un **subcuerpo** de R si todo elemento no cero de S es inversible en S.

Los subanillos, al contario de lo que sería normal pensar, no son conceptos análogos a los subgrupos en la Teoría de Grupos 1. En algunos aspectos estos serían los **ideales**. Un ideal I es un subanillo de un anillo que, además, cumple la propiedad de que: para todo $r \in R$ y $s \in I$ se tiene que $rs, sr \in I$.

Lo que quiere decir esta propiedad es que un ideal es un subgrupo que absorbe, es decir, el producto de cualquier elemento por un elemento del ideal resulta en un elemento del ideal. A esta propiedad se la llama **la propiedad de absorción**. Esta propiedad va a ser muy útil a la hora de definir las propiedades de los anillos.

Ejemplo 2.1.7.

- Los ejemplos más triviales de los ideales serían el 0 y el propio anillo R.
- Otro ejemplo trivial es el de los múltiplos de $\mathbb{Z}$, $n\mathbb{Z}$.

2.2. Propiedades de los anillos

En esta sección se estudiarán las propiedades de los anillos y diferente características interesantes que resultarán fundamentales a la hora de construir cuerpos finitos en los que trabajar con curvas elípticas.

Proposición 2.2.1. *Sea R un anillo conmutativo y r un elemento de este. El conjunto:*

$$(r) = Rr = \{sr : s \in R\},$$

es un ideal de R que contiene al elemento r.

Demostración. Es inmediato comprobar que $r = 1r \in (r)$. Para probar que el conjunto generado por r, (r), es un subgrupo aditivo, es suficiente comprobar que $sr - tr = (s - t)r \in (r)$. Por otra parte, si $x, s \in R$, entonces $x(sr) = (xs)r \in (r)$. En otras palabras, (r) tiene la propiedad de absorción. ■

Proposición 2.2.2. *Si I es un ideal de R, entonces el grupos abeliano R/I también es un anillo con la multiplicación $(r + I)(s + I) = rs + I$.*

Demostración. Como se ha afirmado anteriormente, los ideales tienen la propiedad de absorción, por tanto, cuando algo opera con ellos, el resultado pertenece al ideal. Así, al operar la expresión de la proposición queda esta expresión $rs + rI + Is + II$. Tanto rI como Is pertenecen al ideal I. Por tanto, se obtiene el siguiente resultado:

$$rs + rI + Is + II = rs + I.$$

Comprobar que la multiplicación anterior en R/I está bien definida y cumple con las propiedades de un anillo es trivial. La identidad multiplicativa de este anillo es el elemento $1 + I$. ■

Definición 2.2.3. El anillo anterior R/I es un **anillo cociente**. Este tipo de anillo resulta de volver a todo I el elemento neutro del anillo R. En otras palabras, de "eliminar" el conjunto I del anillo R.

Este anillo es similar a otro conjunto que es bien conocido, el **conjunto de restos**. Si se observa con atención su estructura resulta inmediato comprobar que, en efecto, es lo mismo. En efecto, $\mathbb{Z}_n$ no es si no $\mathbb{Z}/n\mathbb{Z}$. Con esto en mente, las propiedades futuras no resultan para nada extrañas.

Proposición 2.2.4. *Sea n un entero positivo, el anillo $\mathbb{Z}/n\mathbb{Z} = \mathbb{Z}_n$ es un cuerpo si y sólo si n es primo.*

Demostración. La demostración es simple. Si n es compuesto, es decir, $n = ed$ con $1 < e, d < n$. Entonces $(e + n\mathbb{Z})(d + n\mathbb{Z}) = ed + n\mathbb{Z} = n + n\mathbb{Z} = 0$. Por tanto, $(d + n\mathbb{Z})$ carece de inverso en $\mathbb{Z}_n$ como se demostró en 1.5.3. Desde el otro lado de la proposición, si n es primo, significa que el máximo común divisor entre n y cualquier $m < n$ es 1. En ese caso existen $x, y \in \mathbb{Z}$ tales que $xn + ym = 1$, por la igualdad de Bezout. Por tanto $(m+nZ)(b+nZ) = 1+nZ$, así se tiene que $m+nZ$ es inversible. ■

Definición 2.2.5. Teniendo a los anillos R y S. Un **homomorfismo de anillos** es un homomorfismo de grupos, 1.2.2, $f : R \to S$ tal que $f(xy) = f(x)f(y)$ y $f(1) = 1$. Si f es biyectiva, se dice que f es un **isomorfismo de anillos**. Si R y S son cuerpos, entonces se dice que f es un **isomorfismo de cuerpos**. Si R es un anillo e I es un ideal de ese anillo, entonces la aplicación $f : R \to R/I$ dada por $f(r) = r + I$ es un homomorfismo de anillos, como ya se ha estudiado.

Lema 2.2.6 (Teorema de Evaluación)**.** *Sea R un anillo conmutativo y a un elemento del anillo. La aplicación $e_a : R[x] \to R$ dada por $e_a(p) = p(a)$ es un homomorfismo de anillos.*

Demostración. Es inmediato comprobar que $e_a(1) = 1$. Si se tiene a polinomios $p(x)$ y $q(x)$, por definición se tiene que $(p + q)(a) = p(a) + q(a)$ y que $pq(a) = p(a)q(a)$. ■

Proposición 2.2.7. *Sea R un anillo y $f, g \in R[x]$ no cero. Se supone que el coeficiente director de g es una unidad de R. Entonces existen dos únicos polinomios $d, r \in R[x]$ tales que*

$$f = dg + r \text{ con } \delta(r) < \delta(g).$$

Demostración. Si f, g no son únicos, significaría que existen f', r' distintos tal que $f = dg+r = d'g+r'$. Entonces $(d - d')g = r' - r$. Se tiene que:

$$\delta(g) \leq \delta((d - d')g) = \delta(r' - r) \leq max(\delta(r), \delta(r')) < \delta(g),$$

lo cual es una contradicción. Por tanto, $d = d'$ y $r = r'$. La comprobación de la existencia de d y r se hace por inducción sobre el grado de f. Si el grado de g es mayor que el de f, $d = 0$ y $r = f$, no habría nada más que probar. Se supone, por tanto, que $\delta(g) \leq \delta(f)$ y se escribe que $m = \delta(g)$ y $n = \delta(f)$. Si se crea un polinomio de grado menor que n con los coeficientes directores de f y g, se multiplica por g y se resta de f:

$$f - a_n b_m^{-1} x^{n-m} g$$

Este polinomio es de grado menor que n. Por inducción, se tiene que existen polinomios $q, r \in R[x]$ tal que $f - a_n b_m^{-1} x^{n-m} g = qg + r$ lo que lleva a que $f = (a_n b_m^{-1} x^{n-m} + q)g + r$. Si se sigue por esta línea se demuestra que estos polinomios existen. ■

Observación 2.2.8. Si R es un anillo conmutativo y $r, s \in R$. Se dice que r divide a s si existe t tal que $s = rt$. Se simboliza $r|s$.

Observación 2.2.9. Se dice que a es una **raíz** de p en R si $p(a) = 0$. En otras palabras, si $e_a(p) = 0$.

Corolario 2.2.10. *Sea $p \in R[x]$, donde R es un anillo conmutativo, y a es un elemento del anillo, entonces a es una raíz si y sólo si $x - a|p(x)$.*

Demostración. Si $p = 0$, el corolario se confirma. Por tanto, se demuestra para $p \neq 0$. Si $p(x) = (x - a)q(x)$ para algún $q(x) \in R[x]$, entonces es cierto que $p(a) = 0$ por el Teorema de Evaluación 2.2.6. Si se utiliza el algoritmo de la división a $p(x)$ en el caso anterior, se tiene que $p(x) = (x-a)d(x)+r$ donde r es constante. Si se aplica de nuevo el Teorema de Evaluación, se tiene que $p(a) = r = 0$. Así queda demostrado. ■

Por el Corolario 2.2.10, si se tiene un polinomio $0 \neq f \in R[x]$, tal que R es un anillo conmutativo y $a \in R$; entonces se puede escribir así:

$$f(x) = (x - a)^m g(x),$$

donde g es un elemento de $R[x]$, $g(a) \neq 0$ y $0 \leq m \leq \delta(f)$. Si $f(a) = 0$, por el Corolario 2.2.10, se tiene que $m > 0$. Si $m > 1$ se dice que a es una **raíz múltiple** de f con **multiplicidad** m. Se observa que m es el mayor entero positivo que cumple que $(x - a)^m|f$.

Proposición 2.2.11. *Sea R un dominio de integridad y f un elemento no constante del correspondiente anillo de polinomios, $R[x]$. Si se tienen n raíces distintas de f en R, entonces $n \leq \delta(f)$.*

Demostración. Se demuestra por inducción sobre n. Si $n = 1$, la demostración es simple, se tiene que $f(x) = (x-a_1)g(x)$, donde g es distinto de 0 y de grado $\delta(f)-1$. Si existe una segunda raíz, a_2 y resulta ser distinta de la primera, por el Teorema de Evaluación 2.2.6, se tiene que $0 = f(a_2) = (a_2 - a_1)g(a_2)$, y por tanto, $g(a_2) = 0$, con lo que se deduce que existen n raíces distintas de g hasta $n-1$. El teorema se seguiría demostrando hasta encontrar las n raíces. ■

La anterior proposición 2.2.11 no se aplica a los no dominios de integridad. Es decir, se aplica a todos los dominios, pero no es verdadero para los anillos que no son dominios de integridad. Por ejemplo, el polinomio $x^2 - 1$ tiene cuatro raíces en $\mathbb{Z}_8$: 1, 3, 5, 7.

Proposición 2.2.12 (Teorema de Ideales Principales)**.** *Sea K un cuerpo e I un ideal no cero de $K[x]$. Entonces existe un único $p \in K[x]$ mónico tal que $I = (p)$.*

Demostración. Sea $0 \neq p \in I$ tal que el grado de p sea el más pequeño posible. La afirmación de que $I = (p)$ incluye dos partes: $I \supseteq (p)$ y $I \subseteq (p)$. Respecto a la primera parte, es evidente que (p) se incluye en I pues es un ideal; se demuestra fácilmente, $r \in (p)$ si y sólo si se tiene que $r = pt \in I$, siendo t un número entero y $p \in I$, por la propiedad de absorción de los ideales se demuestra esta parte.

En cuanto a la segunda parte, sea $f \in I$, por el algoritmo de la división se tiene que $f = qp + r$. En ese caso, $r = qp - f$ y en esta fórmula todos los términos pertenecen a I, por tanto, $r \in I$. Sin embargo, se da esta situación: $\delta(r) < \delta(p)$ por el algoritmo de la división. Pero esto no es posible, ya que p debería ser el elemento no nulo con el menor grado posible, por tanto, $r = 0$. Así queda demostrado el teorema. ■

Este teorema no es cierto en general si K no es un cuerpo.

Observación 2.2.13. En general, si $I \subseteq R$ es un ideal, tal que $I = (a)$, para cierta $a \in R$, se dice que I es principal.

Definición 2.2.14. Un ideal I de un anillo conmutativo R es un **ideal primo** si dados dos elementos cualesquiera $a, b \in R$, tales que $ab \in I$ se tiene que $a \in I$ ó $b \in I$.

Proposición 2.2.15. *Sea R un anillo conmutativo e I un ideal del anillo. I es un ideal primo si y sólo si el anillo cociente R/I es un dominio de integridad.*

Demostración. Sea el anillo E/I un dominio de integridad, se quiere demostrar que I es un ideal primo de R. Se consideran dos elementos $a, b \in R$, tales que $ab \in I$, entonces se tiene que $(a + I)(b + I) = ab + I = I$. Como R/I es un dominio de integridad $a + I = I$ ó $b + I = I$. Lo que implica que $a \in I$ ó $b \in I$, por lo que I es un ideal primo de R.

Para demostrar la proposición recíproca, se supone que I es un ideal primo de R y $(a + I)(b + I) = I$ para dos elementos cualesquiera $a + I, b + I \in R/I$. Como $I = (a + I)(b + I) = ab + I$, se deduce que $ab \in I$ y como I es un ideal primo $a \in I$ ó $b \in I$. Por tanto, $a + I = I$ ó $b + I = I$, demostrando que R/I es un dominio de integridad. ■

Definición 2.2.16. Sea R un anillo conmutativo e I un ideal de ese anillo. Se dice que I es **maximal** del anillo R si y sólo si para todo $I \subseteq R$, tal que $I \nsubseteq I'$, para ningún I'.

Proposición 2.2.17. *Sea R un anillo conmutativo e I un ideal del anillo. Se dice que I es un ideal maximal si y sólo si el anillo R/I es un cuerpo.*

Demostración. Se supone que R/I es un cuerpo y se quiere demostrar que I es maximal. Sea J otro ideal de R con $I \subset J \subset R$. Sea π un homomorfismo de R en R/I, por homomorfismo de anillos, $\pi(J)$ es un ideal de R/I, como R/I es un cuerpo, no tiene otros ideales más que $\{I\}$ y R/I. Por tanto, $\pi(J) = \{I\}$ ó $\pi(J) = R/I$, en el primer caso se tiene que $J = I$ y en el segundo $J = R$, por lo que I es maximal.

Para demostrar el recíproco, se supone que I es un ideal maximal de R, como R es un anillo conmutativo y $I \neq R$, R/I es también un anillo conmutativo. Para demostrar la proposición se busca probar que todo elemento $a + I$, donde $a \notin I$ y el elemento no sea el elemento neutro multiplicativo de R/I, tiene un inverso multiplicativo en R/I. Se considera J como el ideal generado por I y a.

$$J = \langle I, a\rangle = \{i + ra : i \in I, r \in R\}.$$

Es inmediato comprobar que $J \subset I$ y, puesto que $a \notin I$, $I \neq J$, como I es maximal en R, se tiene que $J = R$. Como 1 es un elemento de R, se deduce que existen $i_0 \in I$, $r_0 \in R$, tales que $1 = i_0 + r_0 a$. Por tanto,

$$(r_0 + I)(a + I) = r_0 a + I = 1 - i_0 + I = 1 + I,$$

ya que $i_0 \in I$, esto prueba que $r_0 + I$ es el inverso multiplicativo de $a + I$ y por tanto, R/I es un cuerpo. ■

En los cuerpos de polinomios existe una propiedad de **máximo común divisor**, al igual que en los conjuntos más comunes de la aritmética. Esta propiedad cumple con esas mismas propiedades solo que en un anillo de polinomios. El polinomio d se denominará máximo común divisor de f y g, si es el único polinomio **mónico** tal que

$$(f) + (g) = (d) \text{ y } d|f, d|g$$

Observación 2.2.18. Obsérvese que un tal d existe por el Teorema de los Ideales Principales 2.2.12.

En el capítulo de Teoría de Grupos 1, había interés en estudiar y comprender las propiedades de los subgrupos, al igual que entonces se tiene especial interés en comprender los cuerpos que contienen cierto cuerpo como subcuerpo.

Lema 2.2.19. *Si K es un subcuerpo de E y f, g son elementos no cero de $K[x]$, entonces su máximo común divisor calculado como polinomios de $K[x]$ es el mismo que calculado como polinomios de $E[x]$.*

Demostración. Se supone que existen los polinomios $e \in E[x]$ y $d \in K[x]$, que son el máximo común divisor de f y g calculados respectivamente en $E[x]$ y en $K[x]$. Como d divide a f y a g en $K[x]$, también los dividirá en $E[x]$ y por tanto, d divide a e en $E[x]$. Por el teorema del máximo común divisor y de la división, se tienen que existen los polinomios $a, b \in K[x]$ tales que $af + bg = d$. Como e divide a f y a g, por ser su máximo común divisor, también divide a d. Por tanto, se tiene que $e|d$ y $d|e$; al ser mónicos resulta que $d = e$. ■

De forma análoga al conjunto de números naturales, en los anillos de polinomios existe también el concepto de primalidad, aunque, en este caso, al polinomio se le dice **irreducible**. Este concepto es muy similar al de los primos. Sea R un dominio de integridad. Se dice que $f \in K[x]$ es irreducible si f no se puede expresar como $f = gh$, siendo g y h dos polinomios no constantes cuyos grados son menores que el de f.

El concepto de irreducibilidad no va a resultar inservible, ya que será fundamental para el objetivo de crear cuerpos finitos. La razón se vio previamente en la proposición 2.2.4, donde se demostraba que $\mathbb{Z}_n$ es un cuerpo si n es primo. En el caso de los anillos de polinomios sucede algo idéntico. Este concepto se formaliza en la siguiente proposición.

Proposición 2.2.20. *Sea K un cuerpo y $p \in K[x]$ no constante. Entonces p es irreducible si y sólo si el anillo $K[x]/(p)$ es un cuerpo.*

Demostración. Sea $K[x]/(p)$ un cuerpo. Si p no es irreducible, se puede escribir como $p = fg$, donde $f, g \in K[x]$ tienen grados menores que p. Se tendría, por lo tanto, que $(f + (p))(g + (p)) = (p)$, y por tanto, $(f + (p))$ carece de inverso como se demostró en la proposición 2.2.4. Por tanto, $K[x]/(p)$ no es un cuerpo, lo que lleva a la conclusión de que p debe ser irreducible. Recíprocamente, si p es irreducible, entonces el máximo común divisor de $(f, p) = 1$. Por tanto, existen los polinomios $a, b \in K[x]$ tales que $af + bp = 1$. De este modo,

$$(a + (p))(f + (p)) = 1 + (p),$$

lo que demuestra que $f + (p) \in K[x]/(p)$ tiene inversa, lo que finaliza la proposición. ■

Proposición 2.2.21. *Sea K un cuerpo y G un subgrupo finito de K^*. Entonces G es cíclico.*

Demostración. Si $|G| = n$ y se da que $d|n$. Si existe un elemento $v \in G$ tal que $v^d = 1$, entonces se le llama generador y al grupo en el que se encuentra cíclico por el 1.3.6. ■

2.3. Extensiones de cuerpos

Las extensiones de cuerpos serán necesarios a la hora de la creación de cuerpos finitos en los que definir las curvas elípticas objeto de estudio. Además, las extensiones otorgarán nuevas propiedades a los cuerpos, especialmente criptográficas, que resultarán muy útiles cuando se vayan a generar los cuerpos finitos.

Definición 2.3.1. Sea K un cuerpo. Se dice que el cuerpo E es una **extensión** de K si K es un subcuerpo de E. A partir de este punto se escribirá E/K para indicar que E es una extensión de K.

Sea E/K una extensión. Obsérvese que E es un espacio vectorial sobre K. Se dice que E/K es finita si la dimensión de E como K-espacio vectorial es finita. En ese caso, se denota $|E:K| = dim_K(E)$, siendo $|E:K|$ el **grado** de la extensión E/K.

Así, el grado denota el número de elementos que debe poseer una base de E. La extensión E/K es finita, si y sólo si existe un subconjunto finito $\{a_1, a_2, \ldots, a_n\} \subseteq E$ tal que sea K-libre y todo elemento de E se pueda escribir como una K-combinación lineal de ese subconjunto. El que tal subconjunto sea K-libre indica que ningún elemento de ese subconjunto se puede escribir como una combinación lineal de los otros elemento del subconjunto.

Proposición 2.3.2 (Transitividad de índices)**.** *Sea E/K una extensión y se supone que $K \subseteq L \subseteq E$. Entonces E/K es finita si y sólo si E/L y L/K son finitas. En ese caso, se tiene que*

$$|E:K| = |E:L||L:K|.$$

Demostración. Se supone primero que E/K es finita. Un K-sistema generador de E es evidentemente también un L-sistema generador, y por tanto, se deduce que E/L es finita. Al ser L un K-subespacio vectorial del K-espacio E, se concluye que L/K también es finita. Si se supone que E/L y L/K son finitas, teniendo un subconjunto $\{a_1, a_2, \ldots, a_n\}$ que es una K-base de L y sea $\{b_1, b_2, \ldots, b_m\}$ una L-base de E; es suficiente con probar que el subconjunto $\{a_i b_j\}$ con $1 \leq i \leq n$, $1 \leq j \leq m$ es una K-base de E. Si x es un elemento de E, entonces $x = \sum_m l_j b_j$ para algunos $l_j \in L$, es decir, cualquier elemento de E es resultado de una combinación lineal de la L-base de E. A su vez, se tiene que cada $l_j = \sum_n a_{ij} a_i$ para ciertos $a_{ij} \in K$. Así, se deduce que

$$x = \sum_{n,m} a_{ij} a_i b_j \text{ con } 1 \leq i \leq n, 1 \leq j \leq m.$$

Esto demostraría que el subconjunto anterior $\{a_i b_j\}$, solo si $1 \leq i \leq n, 1 \leq j \leq m$, es un K-sistema generador de E. Si $x = \sum_{n,m} b_{ij} a_i b_j = 0$ para ciertos $b_{ij} \in K$, se tiene que $\sum_m (\sum_n b_{ij} a_i) b_j = 0$ con $1 \leq i \leq n, 1 \leq j \leq m$. Como el subconjunto formado por b_j es una L-base de E, se tiene que para todo $1 \leq j \leq m$

$$\sum_n b_{ij} a_i = 0.$$

Debido a que el subconjunto $\{a_1, a_2, \ldots, a_n\}$ es una K-base de L, ésto solo se puede dar si $b_{ij} = 0$ si $1 \leq i \leq n$ y $1 \leq j \leq m$, ya que este elemento no pertenece a ninguna base. Esto prueba que el K-sistema generador de E formado por el subconjunto $\{a_i b_j\}$ es K-libre, y por tanto, una K-base de E. De esta forma que demostrado el teorema. ■

Si E/K es una extensión y $S \subseteq E$, se denotará como $K(S)$ a la intersección de todos los subcuerpos de E que contengan tanto a K como a S. Evidentemente $K(S)$ es el menor subcuerpo que contiene a K y a S, al ser una intersección de cuerpos. Como en el caso del subgrupo generado (véase 1.3.4).

Si E/K es una extensión y $a \in E$, entonces el cuerpo $K(a)$ contiene a todos los elementos $f(a)$, siendo $f \in K[x]$, y sus inversos. Para entenderlo mejor, se muestra un ejemplo.

Ejemplo 2.3.3. Si se tiene una extensión $\mathbb{R}/\mathbb{Q}$ y $a = \sqrt{5}$, entonces $\mathbb{Q}(a)$ es el cuerpo cuyos elementos son de esta forma $\alpha+\beta\cdot\sqrt{5}$, donde $\alpha, \beta \in \mathbb{Q}$. De este modo se introduce un elemento de una extensión en su subcuerpo. Como tiene que ser cerrado para el producto, pues los números de la forma $\beta\cdot\sqrt{5}$ también deben pertenecer a $\mathbb{Q}(a)$, y como $\alpha \in \mathbb{Q}(a)$, por pertenecer a $\mathbb{Q}$; pues entonces, como los cuerpos son cerrados para la suma, se puede asegurar que $\alpha + \beta \cdot \sqrt{5} \in \mathbb{Q}(a)$. Más aún, los números de esta forma son cerrados para el producto pues $(\alpha_1+\beta_1\cdot\sqrt{5})\cdot(\alpha_2+\beta_2\cdot\sqrt{5}) = \alpha_1\cdot\alpha_2+5\cdot\beta_1\cdot\beta_2+(\alpha_1\cdot\beta_2\cdot+\alpha_2\cdot\beta_1)\cdot\sqrt{5}$ es otro número de $\mathbb{Q}(a)$. Por tanto, es un cuerpo.

Si E/K es una extensión y $a \in E$, se dice que a es **algebraico** sobre K si existe $p(x) \in K[x]$ y $p(x) \neq 0$ tal que $p(a) = 0$.

Todos los elementos $k \in K$ serán algebraicos sobre K, ya que el polinomio $p(x) = x - k \in K[x]$ tiene a k como raíz. Una extensión E/K se dice algebraica si todos los elementos de E son algebraicos sobre K. Si E/K es una extensión y $a \in E$, se dice que a es **trascendente** sobre K si no es algebraico sobre K.

Proposición 2.3.4. *Si E/K es finita, entonces E/K también es algebraica.*

Demostración. Sea $a \in E$, $a \neq 0$ y $|E : K| = n$. Se quiere demostrar que existe un $p(x) \in K[x]$ no cero tal que $p(a) = 0$, es decir, que exista un polinomio en $K[x]$ para el que a es una raíz. Si existen $i, j \in \mathbb{Z}$ tales que $i > j \geq 0$ y se da que $a^i = a^j$, entonces se tiene que a es una raíz del polinomio $x^{i-j} - 1 \in K[x]$. En caso contrario, el conjunto $\{1, a, a^2, \ldots, a^n\}$ posee $n+1$ elementos, y por tanto, es K-ligado. Así, se tiene que existen $k_0, k_1, \ldots, k_n \in K$ no todos cero, tales que $k_0+k_1a+\ldots+k_na^n = 0$. Por tanto, el polinomio no cero $f(x) = k_0 + k_1x + \ldots + k_nx^n \in K[x]$ se anula en a. ■

A continuación se define un concepto que será útil más adelante.

Definición 2.3.5. Sea $f : R \to R'$ un homomorfismo de anillos. Se define el núcleo de f, $\ker(f)$, como el conjunto de todos los elementos $r \in R$ tales que $f(r) = 0 \in R'$.

Proposición 2.3.6 (Elementos Algebraicos)**.** *Sea E/K una extensión y $a \in E$ algebraico sobre K. Entonces*

(a) Existe un único polinomio mónico $p \in K[x]$ irreducible en $K[x]$ tal que $p(a) = 0$.

(b) Si $q \in K[x]$, entonces $q(a) = 0$ si y sólo si $p|q$.

(c) $K(a) = \{f(a) : f \in K[x]\}$.

(d) Si $\delta(p) = n$, entonces $\{1, a, \ldots, a^{n-1}\}$ es una K-base de $K(a)$. Por lo tanto, $|K(a) : K| = \delta(p)$.

Demostración. Primero se va a probar el segundo apartado y a partir de él demostrar el primero. Para probar (b) se define $\phi : K[x] \to E$ así $\phi(f) = f(a)$. Se observa, por tanto, que ϕ es el homomorfismo de Evaluación $E[x] \to E$ aplicado únicamente a $K[x]$. Debido a que a es algebraico, el ideal $I = \ker(\phi)$ no es cero (para la definición de ker véase 2.3.5). Por el Teorema de Ideales Principales 2.2.12 de la sección anterior, se tiene que existe un único polinomio mónico p tal que $I = (p)$. Por ser un ideal se da que, para todo $q \in K[x]$, entonces $q(a) = 0$ si y sólo si $q \in (p)$, lo que quiere decir que p divide a q, lo que prueba (b).

Para probar (a), hay que probar que p es irreducible. Si esto no es así, entonces se puede escribir $p = gh$ donde $g, h \in K[x]$ no constantes. Si $p(a) = 0$, entonces $p(a) = g(a)h(a) = 0$, y por tanto, o $g(a) = 0$ ó $h(a) = 0$, esto se debe a que K es un cuerpo, y por definición, un dominio de integridad. Sin embargo, esto no es posible por el apartado (b), ya que p no divide ni a g ni a h, al tener estos un grado menor que p. Así que p no se puede escribir como producto de otros polinomios, por lo tanto, p es irreducible.

Se puede observar que $\phi(K[x]) = \{f(a) : f \in K[x]\} \subseteq K(a)$. Al ser $K[x]/(p)$ un cuerpo por la proposición 2.2.20, también lo es $\phi(K[x])$ por ser isomorfo y se tiene que $\phi(K[x]) = K(a)$, lo que prueba (c). Para (a) se supone que el grado de p es n. Si el conjunto $\{1, a, a^2, \ldots, a^{n-1}\}$ no fuera K-libre, se podría construir un polinomio no cero g de menor grado que p tal que $g(a) = 0$. Si se diera el caso, entonces p dividiría g, lo cual es imposible porque g tiene un grado menor. Por tanto, se deduce que el conjunto anterior es K-libre. Ahora se tiene que probar que el conjunto anterior es un K-sistema generador de $K(a)$. Al ser p un polinomio mónico de grado n, se tiene que a^n es una K-combinación lineal de $\{1, a, a^2, \ldots, a^n\}$. Por tanto, se tiene que a^m para cualquier $m \in \mathbb{Z}$ también es una K-combinación lineal, con lo que se prueba que el conjunto $\{f(a) : f \in K[x]\}$ es el conjunto K-generado por $\{1, a, a^2, \ldots, a^{n-1}\}$. Así (d) queda demostrado, y se termina la demostración. ■

Al polinomio $p(x)$ de la proposición anterior se le llama el **polinomio mínimo o irreducible** de a sobre K.

Proposición 2.3.7. *Sea E/K una extensión y sean $a_1, \ldots, a_n \in E$ elementos algebraicos sobre K. Se tiene que $K(a_1, \ldots, a_n)/K$ es finita.*

Demostración. Esta proposición se demuestra por inducción sobre n. Si $n = 1$, entonces el teorema queda demostrado por el apartado (d) de la proposición 2.3.6. Por lo que, por inducción de la misma proposición, si $L = K(a_1, \ldots, a_{n-1})$, se puede suponer que L/K es finita. Debido a que a_n es algebraico sobre L, al serlo sobre K, se tiene que $L(a_n)/L$ es finita y la proposición se demuestra por el teorema de transitividad de índices 2.3.2, ya que $L(a_n)/L$, $L/K \subseteq K(a_1, \ldots, a_n)/K$ y $L(a_n)/L$ y L/K son finitas, entonces se tiene que $K(a_1, \ldots, a_n)/K$ es también finita, en concreto

$$|K(a_1, \ldots, a_n) : K| = |L : K||L(a_n) : L|.$$

Por las proposiciones anteriores se comprueba inmediatamente que tanto las sumas como los productos de elementos algebraicos también son algebraicas. ■

Si $\sigma : K_1 \to K_2$ es un isomorfismo de cuerpos, entonces también se tiene que σ se extiende a los anillos de polinomios respectivos de cada cuerpo K. En otras palabras, $\sigma : K_1[x] \to K_2[x]$ es un isomorfismo de anillos. Al ser un isomorfismo se tiene que el polinomio $p \in K_1[x]$ es irreducible si y sólo si $\sigma(p) \in K_2[x]$ es irreducible.

Proposición 2.3.8. *Sean E_1/K_1 y E_2/K_2 extensiones de cuerpos y $\sigma : K_1 \to K_2$ un isomorfismo. Se asume que $p_1 \in K_1[x]$ es irreducible, $p_2 = \sigma(p_1) \in K_2[x]$ y que $a_i \in E_i$ es una raíz de p_i, donde $i = 1, 2$. Entonces σ se extiende a un isomorfismo θ tal que $\theta : K_1(a_1) \to K_2(a_2)$, con $\theta(a_1) = a_2$.*

Demostración. Se supone que p_1, y por tanto, p_2 son mónicos. Por la afirmación anterior, se tiene que al ser p_1 un polinomio irreducible de a_1 sobre K_1, el polinomio p_2 es un polinomio irreducible de a_2 sobre K_2. Además, por el Teorema de Elementos Algebraicos 2.3.6, en particular el apartado (c), se tiene que $K_1(a_1) = \{f(a_1) : f \in K_1[x]\}$ y $K_2(a_2) = \{g(a_2) : g \in K_2[x]\}$. El siguiente paso es definir θ. $\theta : K_1(a_1) \to K_2(a_2)$, esta aplicación se define del siguiente modo. Si $f \in K_1[x]$, entonces

$$\theta(f(a_1)) = \sigma(f)(a_2).$$

La notación anterior indica que la aplicación θ sobre $f(a_1)$ ofrece el mismo resultado que aplicar el homomorfismo evaluación de a_2 sobre $\sigma(f)$; ya que σ transforma los elementos de K_1 en elementos de K_2. Finalmente, se tiene que comprobar que la aplicación está bien definida y es biyectiva. Si se tienen $f, g \in K_1[x]$, entonces se da $f(a_1) = g(a_1)$ si y sólo si $(f - g)(a_1) = 0$ si y sólo si $p_1 \mid f - g$, por el apartado (b) del Teorema de Elementos Algebraicos 2.3.6, si y sólo si $\sigma(p_1) = p_2 \mid \sigma(f) = \sigma(g)$, por el mismo teorema aplicado antes, y ésto sólo ocurre si y sólo si $\sigma(f)(a_2) = \sigma(g)(a_2)$. Así queda demostrado que θ está bien definida y es inyectiva. Es evidente que esta aplicación es suprayectiva. Además, como enuncia la proposición: θ extiende a σ y se tiene que $\theta(a_1) = a_2$. ■

Corolario 2.3.9. *Sean E/K una extensión y $p \in K[x]$ irreducible. Si $a, b \in E$ son raíces de p, entonces existe un isomorfismo de cuerpos $\theta : K(a) \to K(b)$ tal que $\theta(a) = b$ y $\theta(k) = k$ para todo $k \in K$.*

Demostración. Si se toma como base la demostración de la proposición 2.3.8 y se ajusta la demostración, el Corolario quedaría probado. En concreto, los ajustes serían de esta forma: $K_1 = K_2 = K$, $E_1 = E_2 = E$ y $\sigma : K \to K$, la aplicación identidad, $\sigma(k) = k$ para todo $k \in K$.

El recíproco del Corolario también es cierto: teniendo a, b como elementos algebraicos sobre K y existe un isomorfismo θ similar al anterior, entonces a y b son raíces del mismo polinomio irreducible. ■

Si el $f(x) \in K[x]$ es un polinomio de la forma:

$$a_0 + a_1x + a_2x^2 + \ldots + a_nx^n,$$

se define la **derivada formal** de f como el polinomio de la forma:

$$f'(x) = a_1 + 2a_2x + 3a_3x^2 + \ldots + na_nx^{n-1} \in K[x].$$

Proposición 2.3.10. *Sea E un cuerpo, K un subcuerpo de E y $f \in K[x]$ no constante. Se tiene que:*

(a) Sea $a \in E$. En ese caso a es raíz múltiple de f si y sólo si $f(a) = f'(a) = 0$.

(b) Si $\gcd(f, f') = 1$, entonces f no tiene raíces múltiples en E.

(c) Si f es irreducible en $K[x]$ y $f'(x) \neq 0$, entonces todas las raíces de f en E son distintas.

Demostración. Se prueba (a). Por el Corolario 2.2.10, se puede escribir f así: $f(x) = (x-a)^m g(x)$; si esto es así se puede escribir su derivada también de este modo: $f'(x) = m(x-a)^{m-1}g(x)+(x-a)^m g'(x)$. Si a es raíz múltiple, se tiene que $m \geq 2$, de otro modo el término de menor grado a_1 desaparecería y no se cumpliría el apartado.

Se prueba (b). Sea $\gcd(f, f') = 1$. Se sabe que existen dos polinomios $a, b \in K[x]$ tales que se puede escribir que $af + bf' = 1$. Por tanto, f y f' no pueden tener raíces comunes, de otro modo la igualdad no se cumpliría, ya que el resultado sería 0. Luego por el primer apartado se confirma el segundo. A continuación se prueba (c). Sea f un polinomio irreducible con derivada no cero, como tiene grado menor que f, por definición, $\gcd(f, f') = 1$, ya que al ser f un polinomio irreducible no existen polinomios menores cuyo producto sea f. Por lo tanto, se demuestra (c) por el apartado anterior. ■

Sean K un cuerpo y $f \in K[x]$ no constante. Se dice que f se **escinde** en $K[x]$ si existen $a, a_1, a_2, \ldots, a_n \in K$ tales que

$$f = a(x - a_1)\ldots(x - a_n).$$

Como se puede observar todo polinomio $g \in K[x]$ de grado uno se escinde en $K[x]$. $g(x) = ax + b = a(x - (-b/a)) \in K[x]$. Del mismo modo todo polinomio no constante f se escinde en $K[x]$ si todos los polinomios en sus descomposición en productos de polinomios irreducibles son de grado uno.

Si f se escinde en $K[x]$, entonces las raíces de f en cualquier extensión E de K son $a_1, a_2, \ldots, a_n$. Un cuerpo K en el que todo polinomio no constante de $K[x]$ se escinde en $K[x]$ se le llama **algebraicamente cerrado**.

Sean E/K una extensión y $f \in K[x]$ no constante. Se dice que E es un **cuerpo de escisión** de f sobre K si $f = a(x - a_1)(x - a_2)\ldots(x - a_n)$ en $E[x]$ y $E = K(a_1, \ldots, a_n)$.

Proposición 2.3.11. *Sea $p \in K[x]$ irreducible. Entonces existe la extensión E/K tal que p tiene una raíz en E.*

Demostración. Sea $p(x) = a_0 + a_1x + \ldots + a_nx^n \in K[x]$ irreducible. Por la proposición 2.2.20, se tiene que $E = K[x]/(p)$ es un cuerpo, este cuerpo cociente es el resultado de eliminar todos los polinomios múltiplos de p del anillo de polinomios $K[x]$, en otras palabras, $(p) = 0$ en E. Para ilustrar mejor la demostración, se toma el homomorfismo de anillos $\phi : K[x] \to E$ tal que $f \mapsto f + (p)$. Si $a \in K$ es distinto de cero, entonces la clase de a tampoco corresponderá con la de cero. Por lo tanto, vía ϕ se puede ver a K como un subcuerpo de E. Con este isomorfismo el polinomio p se transforma en un polinomio p' cuyos coeficientes corresponden a las clases de los elementos de K. Si se considera el elemento $e = x \in E$. Entonces $p'(e) = p = 0$, y la proposición queda demostrada. La razón de la igualdad anterior reside en que $e \in K[x]/(p) = E$ y que $p = 0$ en E; por tanto, es inmediato comprobar que la clase de $x \in E$ es una raíz de p al hacerlo 0. ■

Proposición 2.3.12 (Existencia de cuerpos de escisión)**.** *Sean K un cuerpo y $f \in K[x]$ no constante. Entonces existe un cuerpo de escisión de f sobre K.*

Demostración. Si f se escinde en $K[x]$ entonces K es un cuerpo de escisión sobre f y la proposición queda demostrada. Por tanto, se puede suponer que hay un factor irreducible f_1 de f que no se escinde en $K[x]$. Por la proposición 2.3.11 se conoce que existe una extensión F/K tal que F contiene una raíz a de f_1. Por tanto, $f(a) = 0$ y se puede escribir $f = (x - a)g$ donde $g \in K(a)[x]$, es el polinomio que queda después de quitarle a f_1 su raíz. Ahora, por inducción, debería existir un cuerpo de escisión para g. Es decir, existe una extensión $E/K(a)$ donde se puede escribir

$$g = b(x - b_1)(x - b_2)\ldots(x - b_m),$$

donde $E = L(b_1, \ldots, b_m)$ y $L = K(a)$. Dando un paso más allá se tiene que

$$E = K(a, b_1, b_2, \ldots, b_m).$$

Como se f se descompuso anteriormente, se vuelve a componer y queda $f = b(x-a)(x-b_1)\ldots(x-b_m)$ y la proposición queda demostrada. ■

Proposición 2.3.13. *Sean $\sigma : K_1 \to K_2$ un isomorfismo de cuerpos, $p_1 \in K_1[x]$ no constante y $\sigma(p_1) = p_2 \in K_2[x]$. Si E_i es el cuerpo de escisión de p_i sobre K_i donde $i = 1, 2$, entonces existe un isomorfismo $\gamma : E_1 \to E_2$ que extiende a σ.*

Demostración. Si p_1 se escinde en $K_1[x]$, entonces las raíces de p_1 se encuentran en K_1; por lo que, $E_1 = K_1$. Del mismo modo, como $\sigma(p_1)$ se escinde en $K_2[x]$, ya que σ es un isomorfismo de anillos, se tiene que $E_2 = K_2$. Para este caso, la proposición queda probada, al existir el isomorfismo σ. Sin embargo, esto sólo se da si $p_1 \in K_1[x]$ se escinde en $K_1[x]$; de otro modo, se tiene que E_1 resulta de añadir las raíces de p_1 a K_1, es decir, $E_1 = K_1(a_1, \ldots, a_n)$; lo mismo ocurre con $E_2 = K_2(b_1, \ldots, b_n)$. Para demostrar esta extensión del isomorfismo σ se utiliza la proposición 2.3.8, la cual afirma que se puede extender σ a θ, siempre que se añada solamente una raíz del polinomio no constante, es decir,

$\theta : K_1(a) \to K_2(b)$. En otras palabras, si p_i sólo constase de una raíz la proposición quedaría probada así; sin embargo, es necesario que se pruebe que esto es cierto para cualquier número de raíces. Por tanto, esta proposición se prueba por inducción sobre $K_i(k_1, \ldots, k_m)$. Al ser esta proposición sólo cierta si se añade una raíz, lo que hay que hacer primero es añadir una raíz a K_i resultando en un isomorfismo θ con $K_i(k_1)$. A continuación, se añade otra raíz de p_i al K_i anterior, generando otro isomorfismo, y así sucesivamente, resultando en un isomorfismo γ general a todas la raíces de p_i tal que

$$\gamma : E_1 = K_1(a_1, \ldots, a_n) \to E_2 = K_2(b_1, \ldots, b_m).$$

Así queda demostrada la proposición. ■

Corolario 2.3.14 (Unicidad de cuerpos de escisión)**.** *Sean K un cuerpo y $f \in K[x]$ no constante. Si E_1 y E_2 son cuerpos de escisión de f sobre K, entonces existe un isomorfismo $\gamma : E_1 \to E_2$ tal que $\gamma(k) = k$, para todo $k \in K$.*

Demostración. Se demuestra del mismo modo que la proposición 2.3.13, sólo se sustituye $\sigma : K_1 \to K_2$ por $\sigma : K \to K$, la aplicación identidad. ■

Definición 2.3.15. Si K es cuerpo, a la intersección de todos los subcuerpos de K se le llama el **cuerpo primo** F de K. En otras palabras, el cuerpo primo de un cierto cuerpo K es el subcuerpo más pequeño de K.

Proposición 2.3.16. *Sea K un cuerpo y F su cuerpo primo. Entonces se da que F es isomorfo a $\mathbb{Q}$ ó a $\mathbb{Z}_p$ para cierto primo p. En el primer caso, $n1 \neq 0$ para todo $0 \neq n \in \mathbb{Z}$. En el segundo caso, $n1 = 0$ si y sólo si p divide a n.*

Demostración. Se considera el homomorfismo de anillos $\alpha : \mathbb{Z} \to F$ definida por $\alpha(n) = n1$ y que $I = \ker(\alpha)$. Si $I = 0$, entonces es sencillo comprobar que la aplicación $\beta : \mathbb{Q} \to F$ definida así $\beta(n/m) = (n1)(m1)^{-1}$ está bien definida, es decir, es inyectiva y es un homomorfismo de anillos también. Esto se debe a que si el $\ker(\alpha) = 0$, entonces no existe ningún n tal que $n1 = 0$, y por tanto, tampoco existe ningún n/m tal que $(n1)(m1)^{-1} = 0$, así se llega a la conclusión de que todo elemento $n/m \in \mathbb{Q}$ distinto tiene una imagen $\beta(n/m)$ distinta, así se concluye que es inyectiva y también un homomorfismo de anillos.

Ahora, se tiene que por el Teorema del Isomorfismo se tiene que $\beta(\mathbb{Q})$ es un cuerpo igual que F. Para este caso, F es un cuerpo isomorfo con $\mathbb{Q}$ y, como se ha explicado anteriormente, $n1 \neq 0$ para todo $0 \neq n \in \mathbb{Z}$. Por otro lado, si se tiene que $I \neq 0$, entonces existe un $0 < p \in \mathbb{Z}$ tal que $(p) = I$. Si $p = uv$, donde $1 \leq u, v \leq p$, entonces $\alpha(p) = \alpha(uv) = \alpha(u)\alpha(v) = 0$, con lo que se tiene que o bien $\alpha(u) = 0$ ó $\alpha(v) = 0$. Por tanto, $u \in I$ ó $v \in I$, es decir, o bien $p|u$ ó $p|v$. Esto prueba que p debe ser primo. Del mismo modo que en la situación anterior, por el Teorema del Isomorfismo se tiene que

$\alpha(\mathbb{Z})$ y $\mathbb{Z}_p = \mathbb{Z}/p\mathbb{Z}$ son anillos isomorfos. Pero, al ser $\mathbb{Z}_p$ un cuerpo por la proposición 2.2.4, se tiene que $\alpha(\mathbb{Z})$ también es un cuerpo. Por tanto, $\alpha(\mathbb{Z}) = \mathbb{Z}_p = F$ y queda demostrado. ■

Definición 2.3.17. Sea K un cuerpo y F su cuerpo primo. Si F es isomorfo a $\mathbb{Q}$, entonces se dice que la **característica** de K es 0. Si F es isomorfo a $\mathbb{Z}_p$, se dice entonces que la característica de K es p. Se puede observar que en un cuerpo K de característica p se tiene que $p1 = 0$.

Proposición 2.3.18. *Si K es un cuerpo finito, entonces $|K| = p^n$ para cierto cualquier p primo y algún $0 < n \in \mathbb{Z}$. El recíproco también es cierto, dado un primo p y algún entero positivo n, existe un único cuerpo finito de p^n elementos, exceptuando los isomorfismos.*

Demostración. Por la proposición 2.3.16, se tiene que el cuerpo finito K tiene un cuerpo primo F tal que es isomorfo a $\mathbb{Z}_p$ para algún primo p. Como K es finito, por necesidad K es un F-espacio vectorial de cierta dimensión finita. Sea $\{a_1, a_2, \ldots, a_n\}$ una F-base de K, se tiene que todo elemento $k \in K$ es producto de una F-combinación lineal de la base anterior, y por tanto, existen p^n elementos, por ser isomorfo con $\mathbb{Z}_p$, en K por combinatoria, donde $n = dim_F(K)$.

Para demostrar el recíproco, se supone que se tiene un p primo y $0 < n \in \mathbb{Z}$. Sean $F = \mathbb{Z}_p$ y E un cuerpo de escisión de $f = x^{p^n} - x \in F[x]$ sobre F. Si V es el conjunto de raíces de f en E. Se observa que $f' = p^n x^{p^n-1} - 1 = -1$ en $F[x] = \mathbb{Z}_p[x]$; por tanto, se tiene que el máximo común divisor de $(f, f') = 1$. Por la proposición 2.3.10 apartado (b), se puede deducir que f tiene p^n raíces distintas en E, es decir, $|V| = p^n$. Si se tienen dos elementos $a, b \in V$, también se tiene que tanto su resta como su división son elementos de V y si $b \neq 0$, $a/b \in V$. Por lo que, se prueba que V es un subcuerpo de E. Como $E = F(V)$, se tiene que $E = V$ y $|E| = p^n$.

Para concluir, si se considera a L como otro cuerpo de p^n elementos y D su cuerpo primo se procederá a demostrar la unicidad del cuerpo de p^n elementos. Como se ha afirmado al principio de la demostración, si se tiene que D es isomorfo a $\mathbb{Z}_q$ para algún primo q, entonces el número de elementos de L es una potencia de q. Así se deduce que D es isomorfo a F. Por la proposición 2.2.21, L^* es un grupo cíclico de $p^n - 1$ elementos.

Ahora bien, como $|L| = p^n$, entonces el grupo $L^* = L - \{0\}$ tiene $p^n - 1$ elementos. Así, si $l \in L^*$, el orden de l divide a $p^n - 1$, luego

$$l^{p^n-1} = l.$$

De este modo, L^* es el cuerpo de escisión de $x^{p^n} - x$ sobre D, y la demostración se continúa con la proposición 2.3.13. Ya que se tiene un isomorfismo de cuerpos entre D y F, entonces por la proposición mencionada se da que existe otro isomorfismo entre E y L. Así queda demostrado que sólo existe un cuerpo de p^n elementos, y de existir otros cuerpos de p^n elementos, entonces éstos serán isomorfos al primero. ■

Para concluir este capítulo, se destaca la importancia de estas últimas proposiciones para la generación de cuerpos finitos en donde definir las curvas elípticas. Gracias a estas proposiciones se demuestra que existen cuerpos finitos de p^n elementos para cualquier primo p y entero positivo n. Este cuerpo finito se pasará a denotarse $\mathbb{F}_{p^n}$.

A continuación, se muestra un ejemplo de cuerpo finito: $\mathbb{F}_9$.

Ejemplo 2.3.19. Por la proposición 2.3.18 se tiene que existe un cuerpo finito de $9 = 3^2$ elementos.

Antes de generar el cuerpo finito, se observa que $\mathbb{F}_{3^2} = \mathbb{Z}_3[x]/(p)$, donde p es un polinomio irreducible de $\mathbb{Z}_3[x]$ con $\delta(p) = 2$. Como se ha demostrado anteriormente en la proposición 2.2.20 $\mathbb{Z}_3[x]/(p)$ es un cuerpo, y por ser una extensión finita de $\mathbb{Z}_3$, es finito. A partir de esta observación el proceso de generación de sigue así:

(1) Encontrar un $p \in \mathbb{Z}_3[x]$ irreducible con $\delta(p) = 2$.

(2) Generar todos los polinomios de $\mathbb{Z}_3[x]$ de grado menor que p.

(3) Generar la tabla de operaciones de los elementos del cuerpo.

Para encontrar el polinomio irreducible p se utiliza el polinomio visto en la proposición 2.3.18, $x^9 - x$. El proceso de encontrar el polinomio irreducible p se basa en elegir al azar coeficientes de $\mathbb{Z}_3$ y comprobar si el polinomio resultante divide a $x^9 - x$, si lo hace es irreducible. En este caso, se utilizará el polinomio $x^2 + 2x + 2$. Obsérvese que también puede probarse directamente que es irreal porque, si se descompusiese, tendría raíces. Pero es una comprobación rutinaria constatar que $x^2 + 2x + 2$ no tiene raíces en $\mathbb{Z}_3$

En el paso (2) se generan los elementos del cuerpo finito. Dado que el polinomio elegido es de grado 2, los elementos del cuerpo serán de la forma

$$ax + b \text{ donde } a, b \in \mathbb{Z}_3.$$

Para concluir, en el paso (3) se operan todos los elementos entre sí dando lugar a la tabla que se muestra a continuación.

$\cdot$	0	1	2	x	$2x$	$x+1$	$x+2$	$2x+1$	$2x+2$
0	0	0	0	0	0	0	0	0	0
1	0	1	2	x	$2x$	$x+1$	$x+2$	$2x+1$	$2x+2$
2	0	2	1	$2x$	x	$2x+2$	$2x+1$	$x+2$	$x+1$
x	0	x	$2x$	$x+1$	$2x+2$	$2x+1$	1	2	$x+2$
$2x$	0	$2x$	x	$2x+2$	$x+1$	$x+2$	2	1	$2x+1$
$x+1$	0	$x+1$	$2x+2$	$2x+1$	$x+2$	2	x	$2x$	1
$x+2$	0	$x+2$	$2x+1$	1	2	x	$2x+2$	$x+1$	$2x$
$2x+1$	0	$2x+1$	$x+2$	2	1	$2x$	$x+1$	$2x+2$	x
$2x+2$	0	$2x+2$	$x+1$	$x+2$	$2x+1$	1	$2x$	x	2

Cuadro 2.1: Operaciones en un cuerpo finito.

Capítulo 3

Curvas elípticas

En este capítulo se definirán las curvas elípticas que servirán como concepto fundamental de las variaciones para algunos algoritmos criptográficos populares, ya sean para encriptar mensajes o firmar un documento. Este capítulo resume algunos conceptos de [Was08]. Cabe destacar que este capítulo sólo va a ofrecer un resumen y se incluirá únicamente la información necesaria para cumplir el objetivo del proyecto. Para más información sobre las curvas elípticas desde el punto de vista criptográfico se aconseja leer [Kob12], [Kob94] y [KM04].

3.1. Definición de Curvas elípticas

Las llamadas curvas elípticas son un tipo especial de curvas algebraicas; así que, para comprender con precisión lo que son es necesario explicar qué son las curvas algebraicas, y a partir de ahí definir las curvas elípticas. A lo largo de toda esta sección se fija un cuerpo base K.

Definición 3.1.1 (Curvas algebraicas planas)**.** Una **curva algebraica plana** X es el conjunto que forman los puntos en un **plano afín** que anulan a determinado polinomio $p(x, y) \in K[x, y]$.

$$X = \{(x, y) : p(x, y) = 0\}$$

Además, se dirá que dicha curva es **lineal** si $\delta(p) = 1$; se dirá **cónica** si $\delta(p) = 2$, y se la llamará **cúbica** si $\delta(p) = 3$.

Más allá del grado 3 existen más curvas, pero para comprender las curvas elípticas las cúbicas son suficientes.

Definición 3.1.2 (Puntos singulares)**.** Sea $x \in X$, se llama a x **punto singular** si $\nabla_x p = 0$, donde $\nabla_x p$ denota el gradiente de p en x. En otras palabras la derivada de la función de la curva en ese punto se anula, causando que no exista una sola tangente, si no infinitas. Un punto **liso** es un punto no singular. Véase la figura 3.1 para observar un dibujo de un punto singular.

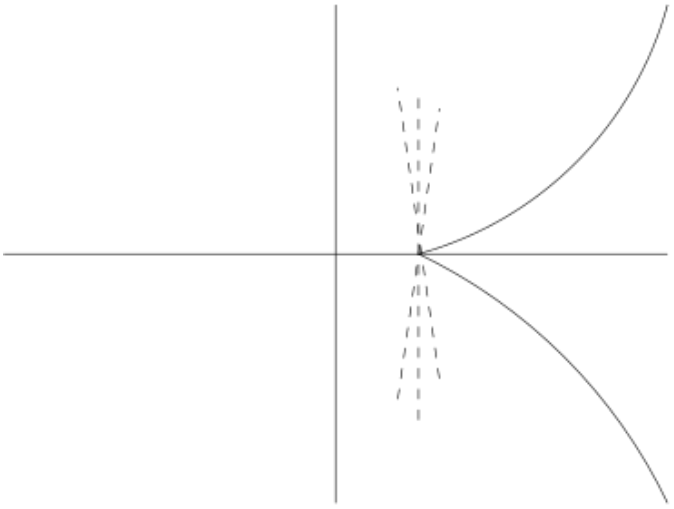

FIGURA 3.1: Curva algebraica plana singular

Definición 3.1.3 (Curva elíptica)**.** Una **curva elíptica** es una curva cúbica lisa, es decir, una curva cúbica cuyos puntos son todos lisos. Además, toda curva elíptica admite su expresión en forma de **ecuación de Weierstrass** (véase la figura 3.2).

La razón de que una curva elíptica no posea ningún punto singular reside en la forma de operar en una curva elíptica que se estudiará más adelante.

3.2. Ecuación de Weierstrass

Toda curva elíptica se puede expresar como el conjunto de puntos de la forma (x, y) que cumplen una ecuación de esta forma:

$$y^2 = x^3 + Ax + B,$$

llamada **ecuación de Weierstrass**, donde $A, B \in K$ son constantes. A efectos criptográficos $K = \mathbb{F}_q$; aunque también pueden usarse otros cuerpos como $\mathbb{C}$ ó $\mathbb{R}$. Sin embargo, por propósitos criptográficos siempre se usarán cuerpos finitos, ya que al encriptar y enviar un mensaje se pretende transmitir la información de forma exacta, algo que no se puede asegurar en un cuerpo infinito.

Esta ecuación de Weierstrass será el polinomio que va generar la curva cúbica junto con sus raíces (no confundir las raíces de este polinomio con que $\mathbb{F}_{p^n}$ se defina como cociente de un anillo de polinomios). Así, si $A, B \in K$, siendo K un cuerpo, se dice que la curva elíptica E se define sobre K, lo que se denota $E(K)$. Además, de esos puntos también se incluye el punto ∞ que resultará de gran ayuda, ya que va a representar el rol de elementos neutro del grupo de la curva elíptica.

Para generar una curva elíptica correcta, no sólo tiene que cumplir los requisitos anteriores, si no también que no haya **raíces múltiples** para la ecuación de Weierstrass.

Es posible comprobar que, de existir raíces múltiples, las funciones de encriptación no pueden funcionar correctamente. En la forma de Weierstrass, es muy fácil comprobar si la cúbica es lisa, pues no tiene singularidades si y sólo si los coeficientes A y B satisfacen $4A^3 + 27B^2 \neq 0$.

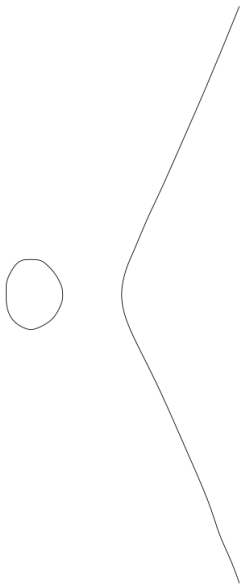

FIGURA 3.2: Curva elíptica de ecuación $y^2 = x^3 - x$

3.3. Ley de Grupo

En esta parte del capítulo se estudiará cómo se opera y se producen otros puntos como resultado de una operación de grupo definido sobre los puntos de una curva elíptica E en un cuerpo K, $E(K)$.

La operación principal que se va a utilizar en curvas elípticas es la **suma de puntos**; sin embargo, la suma de los puntos no consiste en sumar las coordenadas de los puntos, si no en una operación más peculiar. Se empieza el proceso de suma teniendo dos puntos: $P_1, P_2 \in E(K)$. A continuación se dibuja una línea L que intersecte a los dos puntos anteriores y se extiende. Al extenderse, L cortará a la curva elíptica en un tercer punto P_3; una vez que se tiene P_3 se refleja este punto sobre el eje X y se obtiene el resultado final de la suma: P_3'. Como observación, la reflexión del punto anterior consiste solamente en cambiarle el signo a la coordenada y del punto P_3. Toda esta operación se denotará de ahora en adelante:

$$P_1 + P_2 = P_3'.$$

Una vez definida gráficamente la suma de puntos en 3.3, se puede definir de **forma algebraica**. Sean los puntos $P_1 = (x_1, y_1)$ y $P_2 = (x_2, y_2)$. En el caso de que $P_1 \neq P_2$, entonces la pendiente de L es

$$m = \frac{y_2 - y_1}{x_2 - x_1}.$$

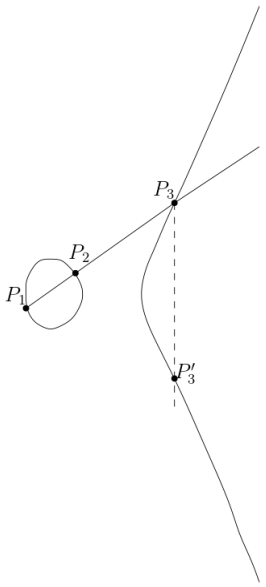

FIGURA 3.3: Ley de Grupo

Suponiendo que $x_1 \neq x_2$; la ecuación de L es $y = m(x - x_1) + y_1$. Para calcular el punto $P_3 = (x, y)$, en el que L corta a E, se sustituye esta ecuación en la ecuación de Weierstrass, resultando en

$$(m(x - x_1) + y_1)^2 = x^3 + Ax + B \Rightarrow 0 = x^3 - m^2x^2 + \ldots + Ax + B.$$

Las raíces de esta ecuación cúbica corresponden a los tres puntos de intersección de L con E. Aunque, de forma general, la resolución de esta ecuación es complicada, en este caso no lo es debido a que ya se conocen dos raíces de esta ecuación, los puntos P_1 y P_2. Para encontrar la tercera raíz, se podría factorizar la ecuación, ya que se conocen dos raíces de la ecuación. Pero al conocer dos raíces existe una manera más simple. En primer lugar si se tiene la ecuación cúbica $x^3 + ax^2 + bx + c$, y sean las tres raíces r, s y t entonces se tiene que

$$x^3 + ax^2 + bx + c = (x - r)(x - s)(x - t).$$

Además, se nota que

$$(x - r)(x - s)(x - t) = x^3 - (r + s + t)x^2 + (rs - rt - st)x - rst.$$

Por tanto, si sigue este razonamiento se tiene que

$$r + s + t = -a,$$

y sabiendo las raíces r, s se puede resolver esta ecuación así: $t = -a - r - s$. Para este caso, se tiene que $P_3 = (x, y)$ cumple

$$x = m^2 - x_1 - x_2,$$
$$y = m(x - x_1) + y_1.$$

Como este punto P_3 tiene que ser reflejado sobre el eje X, las soluciones para $P_3' = (x', y')$ son:

$$x' = m^2 - x_1 - x_2,$$
$$y' = m(x_1 - x) - y_1.$$

En el caso de que las **coordenadas x** de los dos puntos **coincidiesen**, L sería una línea vertical, que corta a E en el punto ∞, cuyo reflejo sobre el eje X sigue siendo ∞. Por tanto, $P_1 + P_2 = \infty$.

Otro caso a estudiar es el caso en el que se suma **un punto consigo mismo**, es decir, $P_1 = P_2 = (x_1, y_1)$. En ese caso, la línea L que corta los dos puntos es la línea tangente a ese punto. Para obtener la ecuación de la pendiente de la recta tangente L se toman derivadas parciales sobre la ecuación de Weierstrass, dando el siguiente resultado:

$$2y\frac{dy}{dx} = 3x^2 + A \Rightarrow m = \frac{3x_1^2 + A}{2y_1}.$$

Si la coordenada $y_1 = 0$, entonces se toma como que el tercer punto es ∞ y que la recta L es una vertical, y por tanto, $P_1 + P_2 = \infty$. Así que se asume que $y_1 \neq 0$, y la ecuación de L resultante es

$$y = m(x - x_1) + y_1,$$

la misma que en el caso anterior. Por tanto, se obtiene la misma ecuación cúbica sobre Weierstrass. Obsérvese que, en este caso sólo se conoce una raíz de la ecuación, aunque esta raíz es doble, ya que L es una recta tangente a $P_1 \in E(K)$. Por lo cual, se sigue el procedimiento como antes,

$$x = m^2 - 2x_1,\ y = m(x_1 - x) - y.$$

Por último, se supone que $P_2 = \infty$. La línea L que intersecta a P_1 y a ∞ es una línea vertical que corta a E en el punto P_1' que es el reflejo de P_1 sobre el eje X. Por la discusión anterior se tiene que

$$P_1 + \infty = P_1$$

para todos los puntos en $E(K)$.

Anteriormente, se explicó que las curvas elípticas tenían que carecer de puntos singulares, 3.1.3, debido a que estos puntos tienen infinitas tangentes como en la figura 3.1. Pues una vez explicada la suma en una curva elíptica se vuelve inmediato comprender por qué eso es así; ya que si se diera una suma de un punto consigo mismo resultaría una operación imposible porque no habría una sola tangente que corte a ese punto.

A partir de lo explicado anteriormente, se puede inferir que la operación de suma de puntos en una curva elíptica E sobre un cuerpo K es cerrada sobre $E(K)$. En otras palabras, si se suman dos puntos que pertenecen a $E(K)$, entonces se tiene que la suma de esos puntos pertenece también a $E(K)$. Sin embargo, esta operación posee otras características especiales.

Las propiedades que satisface la suma de puntos en una curva elíptica E sobre un cuerpo K son las siguientes. Para cualquiera $P_1, P_2, P_3, P \in E(K)$ se tiene

(1) $P_1 + P_2 = P_2 + P_1$.

(2) $P + \infty = P$.

(3) Existe un único $P' \in E(K)$ tal que $P + P' = \infty$. El punto P' se denota $-P$.

(4) $(P_1 + P_2) + P_3 = P_1 + (P_2 + P_3)$.

Por todo lo estudiado en Teoría de grupos 1, se tiene entonces que los puntos de $E(K)$ forman un grupo abeliano junto con el punto ∞ como elementos neutro. Además, dependiendo de la selección de K puede ser un grupo finito o infinito; sin embargo, por propósitos criptográficos siempre se utilizarán grupos finitos.

A partir de la definición de la suma de puntos en una curva elíptica, se puede definir también la multiplicación de un punto $P \in E(K)$ por un entero $k \in \mathbb{Z}$, ya que no es nada más que la suma realizada k veces. Esta operación se denota así kP cuyo significado para $k > 0$ no es otro que indicar $P+P+\ldots+P$ con k sumandos. En el caso de que $k < 0$, entonces solamente indicaría $(-P) + (-P) + \ldots + (-P)$ con $|k|$ operandos.

Observación 3.3.1. Obsérvese que, si k es elevado, entonces computar kP requiere $k - 1$ sumas que es computacionalmente ineficiente. Así que se toma un acercamiento distinto. Este acercamiento consiste en tomar **doblajes sucesivos**. Para ilustrarlo, se considera que $k = 23$. Entonces se puede realizar los siguientes cómputos

$$P + P = 2P,\ 2P + 2P = 4P,\ 4P + 4P = 8P,\ 8P + 8P = 16P,\ 16P + 4P = 20P,\ 20P + 2P = 22P \text{ y}$$
$$22P + P = 23P.$$

Este método permite un cálculo mucho más rápido para k muy grandes; aunque se tiene que tener en cuenta que aún puede haber un problema con el tamaño de las coordenadas de los puntos, pero como se opera en cuerpos finitos $\mathbb{F}_q$ cada vez que se realiza una suma de puntos se reducen las coordenadas módulo q.

Para formalizar este método se crea un algoritmo:

Input:

(1) $E(K)$: curva elíptica definida sobre un cuerpo K.

(2) k: $0 \leq k \in \mathbb{Z}$.

(3) P: un punto perteneciente a la curva elíptica $E(K)$.

Output: La salida $B = kP$.

```
begin
    a ⟵ k;
    B ⟵ ∞;
    C ⟵ P;
    while a ≠ 0 do
        if a % 2 = 0 then
            a ⟵ a/2;
            B ⟵ B;
            C ⟵ 2C;
        end
        else
            a ⟵ a − 1;
            B ⟵ B + C;
            C ⟵ C;
        end
    end
    return B
end
```

Algoritmo 1: Un punto por un entero

Observación 3.3.2. La razón por la que las curvas elípticas son usadas en criptografía, es que dados P y $Q = kP$ puntos de las curvas elípticas resulta muy difícil averiguar cuál es el valor de k. A este problema se le llama el **problema del logaritmo discreto** en curvas elípticas, que será estudiado en el capítulo siguiente, en el que se estudiarán las fortalezas de los algoritmos de encriptación en curvas elípticas.

3.4. Un punto en el infinito

En esta sección se dará una mirada a la razón de la existencia del punto en el infinito. Para dar tal explicación es necesario tomar un **plano proyectivo** sobre el cuerpo K, $\mathbb{P}^2_K$; también llamado espacio proyectivo bidimensional. Para definir un punto en este plano se requieren dar tres coordenadas (x, y, z), tales que $x, y, z \in K$ y al menos una coordenada sea distinta de cero. Ahora bien, es necesario

considerar la relación de **equivalencia** que dice que un punto (x_1, y_1, z_1) es equivalente a un punto (x_2, y_2, z_2) si

$$(x_1, y_1, z_1) = (\lambda x_2, \lambda y_2, \lambda z_2),$$

para algún $\lambda \in K^* = K - \{0\}$. Obsérvese que la clase de equivalencia de un conjunto de tres coordenadas depende de los ratios entre x, y y z. Por eso, de ahora en adelante se notará a una clase de equivalencia de (x, y, z) como $(x : y : z)$.

En el caso de que $(x : y : z)$ sea un punto con $z \neq 0$, entonces se tiene que $(x : y : z) = (\frac{x}{z} : \frac{y}{z} : 1)$, que simbolizan los "puntos finitos"de $\mathbb{P}^2_K$. En cambio, si $z = 0$, entonces se tiene que al dividir entre z las demás coordenadas resultan en el resultado ∞; por tanto, estos puntos $(x : y : 0)$ pasarán a denominarse los "puntos en el infinito"de $\mathbb{P}^2_K$.

Volviendo a los **planos afines**, si se tiene un plano afín $\mathbb{A}^2$ sobre un cuerpo K, denotado como $\mathbb{A}^2_K$, se puede definir una inclusión entre este plano y el plano proyectivo $\mathbb{P}^2_K$ definida por

$$(x, y) \in \mathbb{A}^2_K \mapsto (x : y : 1) \in \mathbb{P}^2_K.$$

De este modo, todos los puntos del plano afín se identifican con los puntos finitos del plano proyectivo. Para demostrar que el punto en el infinito es el punto en donde se cortan dos líneas paralelas hace falta recurrir a un segundo concepto: los **polinomios homogéneos**.

Definición 3.4.1. Un polinomio se dice homogéneo de grado n si es una suma de términos de la forma $ax^i y^j z^k$, donde $a \in K$ y $i + j + k = n$.

Ejemplo 3.4.2. Un ejemplo sería

$$F(x, y, z) = x^5 + 7xy^2z^2 + 2z^5,$$

un polinomio homogéneo de grado 5.

Si F es un polinomio homogéneo de grado n, se tiene que $F(\lambda x, \lambda y, \lambda z) = \lambda^n F(x, y, z)$ para todo $\lambda \in K$. Si se relaciona este concepto con el anterior de relaciones de equivalencia, es inmediato pensar que si (x_1, y_1, z_1) (x_2, y_2, z_2), entonces $F(x_1, y_1, z_1) = 0$ si y sólo si $F(x_2, y_2, z_2) = 0$, luego el lugar de anulación de un polinomio homogéneo está bien definido en el plano proyectivo.

Si F es un polinomio cualquiera en vez de ser homogéneo de algún grado, entonces no tiene sentido el hablar de puntos en el plano proyectivo donde el polinomio se anule, porque depende del representante de la clase de equivalencia.

Ejemplo 3.4.3. Por ejemplo, si se toma el polinomio $f(x, y, z) = x^2 + 2y - 3z$, se tiene que, para $(1 : 1 : 1)$, $F(1, 1, 1) = 0$; mientras que, para $(2 : 2 : 2)$, se tiene $f(2, 2, 2) = 2$. Este problema se elude utilizando polinomios homogéneos.

Para transformar un polinomio f definido en un plano afín en homogéneo definido sobre un plano proyectivo simplemente se le añaden las potencias pertinentes de z. Si F es homogéneo, entonces por lo anterior se tiene que

$$F(x, y, z) = z^n f(\tfrac{x}{z}, \tfrac{y}{z}),$$

y por supuesto, $f(x, y) = F(x, y, 1)$.

Con esto en mente, se puede ver por qué dos rectas paralelas se cortan en el infinito. Sean dos rectas paralelas no verticales dadas por las ecuaciones

$$y = mx + b_1 \text{ y } y = mx + b_2,$$

donde $b_1 \neq b_2$. Su forma homogénea es la siguiente:

$$y = mx + b_1 z \text{ y } y = mx + b_2 z$$

Al resolver el sistema de ecuaciones, la solución queda así:

$$z = 0 \text{ y } y = mx.$$

Como ya se ha dicho, al menos una coordenada debe ser distinta de cero; por lo que, se tiene que $x \neq 0$. Así, la clase de equivalencia del punto donde se cortan es $(x : mx : 0) = (1 : m : 0)$.

Ahora hay que estudiar el punto en el infinito de una curva elíptica. Se tiene una curva elíptica E con la ecuación de Weierstrass

$$y^2 = x^3 + Ax + B.$$

Se transforma en homogénea: $y^2 z = x^3 + Axz^2 + Bz^3$. Como ya se ha explicado, los puntos en el infinito tienen la coordenada $z = 0$. Al realizar este cambio se tiene que $x^3 = 0$, resultando en que el punto en el infinito tiene la clase de equivalencia siguiente: $(0 : y : 0)$. Se divide entre y, y resulta que el único punto en infinito es el $(0 : 1 : 0)$. Además, se tiene a su vez que $(0 : 1 : 0) = (0 : -1 : 0)$; por lo que, el infinito se encuentra tanto en la "cima"del eje y como en el "fondo". Por lo que, tiene sentido decir que todas las rectas paralelas se cortan en el infinito.

3.5. Propiedades de las curvas elípticas

En esta sección se estudiarán las distintas propiedades de las curvas elípticas y cómo estas las distinguen en tipos. Los tipos de la curvas elípticas son importantes para la encriptación por diferentes algoritmos de ataque que se estudiarán más adelante; ya que, en función de una curva u otra, puede ser relativamente fácil hallar la clave y romper la encriptación.

En primer lugar se calculará el **orden** de una curva elíptica E sobre un cuerpo finito $\mathbb{F}_q$, donde $q = p^n$ con p un número primo y $n \in \mathbb{Z}$. El orden de una curva elíptica consiste en el número de puntos de la misma, $\#E(\mathbb{F}_q)$. A pesar de que la curva se defina en un cuerpo finito, eso no significa que tenga el mismo número de puntos que valores tenga el cuerpo.

La primera aproximación a dar con un algoritmo para calcular los puntos de $E(\mathbb{F}_q)$ la dio Hasse con el siguiente resultado, en el que limita el número de puntos que puede tener una curva elíptica.

Teorema 3.5.1 (Teorema de Hasse)**.** *Sea E una curva elíptica definida sobre un cuerpo finito $\mathbb{F}_q$. Entonces se tiene que el orden de $E(\mathbb{F}_q)$ cumple*

$$|q + 1 - \#E(\mathbb{F}_q)| \leq 2\sqrt{q}.$$

Hubo más aproximaciones a encontrar una fórmula u algoritmo que calculase la cantidad de puntos de E utilizando el Teorema de Hasse 3.5.1. Sin embargo, eran demasiado costosos y complicados. Más tarde, en 1985, Schoof descubrió y publicó [Sch85] un algoritmo que permitía calcular el número de puntos de una curva elíptica en un cuerpo finito en un tiempo mucho más corto que los algoritmos existentes hasta la fecha. Este algoritmo tiene una complejidad de $O(\log^8 q)$ operaciones de bits y permite cálculos muy rápidos incluso para q grandes. Más adelante, Atkin y Elkies mejoraron el algoritmo de tal forma se puede calcular el orden de una curva para q de varios cientos de dígitos [BSS06].

Definición 3.5.2 (Puntos de torsión)**.** Dados la curva elíptica E definida sobre un cuerpo K y $\overline{K}$ la clausura algebraica de K. Sea n un entero positivo, un punto de n-torsión es todo aquel que

$$E[n] = \{P \in E(\overline{K}) \mid nP = \infty\}$$

En otras palabras, $E[n]$ son los puntos de la curva $E(\overline{K})$ tal que su orden es un divisor de n.

Emparejamiento de Weil. En relación a los puntos de torsión se encuentra el emparejamiento de Weil, que es una técnica sumamente importante para el estudio de curvas elípticas por sus propiedades. Sean E la curva elíptica definida sobre el cuerpo K y $n \in \mathbb{Z}$ tal que la característica de K no divide a n, entonces los puntos de n-torsión 3.5.2, $E[n]$ forman un subgrupo isomorfo a $\mathbb{Z}_n \oplus \mathbb{Z}_n$.

Se considera ahora el grupo de las raíces n-ésimas de la clausura algebraica de K:

$$\mu_n = \{x \in K : x^n = 1\}.$$

Como la característica de K no divide a n, se tiene que la ecuación $x^n = 1$ carece de raíces múltiples, y por tanto, existen n elementos en ese grupo, llamados **raíces n-ésimas de unidad**. Este grupo es cíclico de orden n. Por lo tanto, tiene generadores; todo generador g del grupo μ_n se denomina

raíz primitiva n-ésima de unidad. A modo de comprobación de un generador g, se tiene que $g^k = 1$ si y sólo si n divide a k.

La utilidad de estos grupos reside en el llamado emparejamiento de Weil cuya definición completa y demostración de sus propiedades puede constatarse en 3.5.3.

Teorema 3.5.3 (Emparejamiento de Weil)**.** *Sean E la curva elíptica definida sobre un cuerpo K y $0 \leq n \in \mathbb{Z}$. Suponiendo que la característica de K no divida a n, existe un emparejamiento:*

$$e_n = E[n] \times E[n] \to \mu_n$$

conocido como el ***emparejamiento de Weil****.*

Este emparejamiento es interesante por las propiedades que posee:

(1) e_n es bilineal para cada variable. En otras palabras, para todo $S, S_1, S_2, T, T_1, T_2 \in E[n]$ se tiene que

$$e_n(S_1 + S_2, T) = e_n(S_1, T)e_n(S_2, T)$$
$$\text{y}$$
$$e_n(S, T_1 + T_2) = e_n(S, T_1)e_n(S, T_2).$$

(2) e_n no es degenerada. Es decir, si

$$e_n(S, T) = 1, \text{ para todo } T \in E[n], \text{ entonces } S = \infty.$$

El caso opuesto es también cierto.

(3) $e_n(T, T) = 1$ para todo $T \in E[n]$.

(4) $e_n(T, S) = e_n(S, T)^{-1}$ para todo $S, T \in E[n]$.

(5) $e_n(\sigma S, \sigma T) = \sigma e_n(S, T)$ para todo automorfismo σ de $\overline{K}$ tal que $\sigma(A) = A$ y $\sigma(B) = B$.

(6) $e_n(\alpha(S), \alpha(T)) = e_n(S, T)^{\delta(\alpha)}$ para todo endomorfismo α de E, para más información ver [Eng01], que contiene información interesante sobre las curvas elípticas.

Este emparejamiento resultará muy útil en el ataque MOV (véase en el capítulo 5).

3.6. Algunos Tipos de Curvas Elípticas

Toda curva elíptica tiene una serie de propiedades que satisface como se ha visto en la sección anterior. Debido a estas propiedades, existen ciertas debilidades que ciertos algoritmos de ataque pueden utilizar para romper la criptografía. En sección se verán dos tipos de curvas: curvas supersingulares y curvas anómalas.

Curvas supersingulares. Sea E una curva elíptica definida sobre un cuerpo K de característica p. Se dice que E es una **curva supersingular** si

$$E[p] = \{\infty\},$$

es decir, no tiene puntos de p-torsión.

Usando el algoritmo de Schoof [Sch85] se puede dar una caracterización efectiva de curvas supersingulares como muestra la siguiente definición.

Proposición 3.6.1 (Curva supersingular)**.** *Sea E una curva elíptica definida sobre el cuerpo finito $\mathbb{F}_q$, siendo $q = p^n$, donde $n \in \mathbb{Z}$. Sea $a = q + 1 - \#E(\mathbb{F}_q)$. Entonces E es supersingular si y sólo si $a \equiv 0 \mod p$, lo que se da si y sólo si $\#E(\mathbb{F}_q) \equiv 1 \mod p$.*

La prueba se puede encontrar en [Was08], en el capítulo 4, sección Curvas supersingulares.

Las curvas supersingulares resultan un caso de estudio tan interesante debido a que hay veces en las que realizar operaciones en esta curva es más rápido de lo que se podría esperar. Esto se debe a que los cálculos en la curva elíptica son sustituidos por operaciones sobre el cuerpo $\mathbb{F}_q$, las cuales son rápidas. Sin embargo, son débiles frente al ataque MOV, que se estudiará más adelante.

Por último, decir que existe un método para hallar todas las curvas supersingulares en un cuerpo; pero, este método está fuera del ámbito de estudio de este trabajo. Puede consultarse en [Was08] en el capítulo 4, sección Curvas supersingulares.

Curvas anómalas. Sea E una curva elíptica definida sobre un cuerpo finito $\mathbb{F}_q$, se dice que E es una **curva anómala** si y sólo si

$$\#E(\mathbb{F}_q) = q.$$

A pesar de que este tipo de curvas son resistentes a ataques MOV (puesto que no son supersingulares porque $\#E(\mathbb{F}_q) \equiv 0 \mod p$), la resolución del **problema del logaritmo discreto** en estas curvas es rápido. Hay que observar que aunque E sobre $\mathbb{F}_q$ sea una curva anómala, no significa que E sobre cualquier otro cuerpo finito lo sea, es decir, esta propiedad depende del cuerpo base. Estas curvas son interesantes porque, como las supersingulares, permiten más rapidez para algunas operaciones cuando están definidas sobre extensiones del cuerpo $\mathbb{F}_q$.

Este tipo de curvas se idearon con el fin de evitar que el ataque MOV resultase tan útil contra curvas supersingulares; sin embargo, resultó que existe un ataque distinto al MOV que rompe las criptografías que utilicen este tipo de curvas en un tiempo menor al que lo hace el ataque MOV sobre supersingulares. Así que no resulta recomendable utilizar este tipo de curvas para encriptar. Para más información véase [Sma99], [SA98] y [Sem98].

Capítulo 4

Criptografía de Curvas elípticas

En esta capítulo se presentará un algoritmo de encriptación utilizando curvas elípticas. Se estudiarán sus fortalezas y debilidades; además, se realizará una comparación con los algoritmos de encriptación clásicos.

4.1. Problema del logaritmo discreto

En el capítulo anterior, se mencionó el **problema del logaritmo discreto**. En esta sección se explicará qué es exactamente este concepto. Además, se estudiará cuál es la importancia de este concepto en la criptografía clásica y de curvas elípticas.

El problema del logaritmo discreto **clásico** consiste en lo siguiente:

Problema del logaritmo discreto clásico. Sean p un número primo y $a, b \in \mathbb{Z}_p$ distintos de cero, tal que $a^k = b$. Dados a y b, encontrar k. A ese número se le denomina el logaritmo discreto de b en base a y se denota $k = \log_a(b)$.

Este número k es difícil de averiguar para números p altos, ya que los mejores algoritmos que resuelven este problema son de complejidad subexponencial, y se reducen a buscar todas las posibles soluciones de:

$$\log_a(b).$$

Este problema tiene también otro nombre: el problema de Diffie-Hellman [DH76]. En teoría, el problema es algo distinto pero la dificultad es exactamente la misma: encontrar el exponente exacto que dé el resultado deseado.

La idea de los algoritmos criptográficos modernos es basarse, no en esta versión clásica si no en la siguiente versión de general.

Problema del logaritmo discreto en grupos abelianos finitos. Sea G un grupo abeliano finito. Dados $P, Q \in G$ distintos del elemento neutro y $0 \neq a \in \mathbb{Z}$, tales que $aP = Q$, encontrar a.

Este problema se vuelve más difícil de resolver que el anterior, ya que se está definiendo este problema para un grupo G arbitrario con una operación cualquiera. Esto dificulta el desarrollo de un ataque global contra este problema, a diferencia del logaritmo discreto clásico que sólo usa la multiplicación y el conjunto $\mathbb{Z}_p$.

Observación 4.1.1. El logaritmo discreto en un grupo abeliano finito G sólo está definido mod $|G|$.

4.2. Criptosistemas en curvas elípticas

Este proyecto se va a centrar en atacar los criptosistemas en curvas elípticas, que son grupos abelianos finitos. Para ello se van a estudiar algunos criptosistemas muy usados que están adaptados a curvas elípticas.

La criptografía de curvas elípticas fue propuesta a mediados de la década de 1980 por Neil Koblitz [Kob87] y Viktor Miller [Mil86], y como se ha apuntado anteriormente, su fortaleza reside en el problema del logaritmo discreto en el grupo de los puntos de la curva elíptica en un cuerpo. Esta idea surgió después de la creación de RSA como criptosistema de clave pública, pretendiendo sustituir el cuerpo finito $\mathbb{F}_q$ con la curva elíptica E.

Antes de entrar a estudiar los algoritmos criptográficos en curvas elípticas, es interesante explicar por qué se estudian siquiera las curvas elípticas. Las curvas elípticas presenta una importante ventaja como es el uso de menos bits de la clave para dar un nivel de seguridad mayor. Concretamente, se consigue el mismo nivel de seguridad con 1024 bits en RSA que con 160 bits en la firma digital con curvas elípticas. A pesar de esta ventaja, las desventajas son también notables como que las operaciones en curvas elípticas son costosas; aun utilizando una clave pequeña los documentos cifrados adquieren un peso mucho mayor tras ser encriptados, o el hecho de que diferentes tipos de curvas que pueden ser o no vulnerables según qué ataques. Por todo esto, los criptosistemas que se basan en curvas elípticas son utilizados en su mayoría para firmar documentos; así que a continuación se va a explicar el algoritmo de **ElGamal para la firma digital**, así como **Encriptación ElGamal**.

Observación 4.2.1. Hay varios estándares internacionales que involucran a la criptografía de curvas elípticas como el IEEE P1363 [iee00], así como estándares procedentes de agencias americanas con licencia para publicar estándares como [x9-01] y [x9-05].

4.2.1. ElGamal

Este algoritmo es usado pocas veces, en comparación con su contraparte para las firmas digitales. Sin embargo, es de interés para este trabajo a fin de dar una mejor visión de cómo funciona el algoritmo de ElGamal de firmado en curvas elípticas.

Algoritmo: ElGamal *Prefacio:*

Alicia quiere enviar un mensaje secreto a Berto sin que Eva se entere del contenido del mensaje.

Input:

(1) Berto define una curva elíptica E sobre un cuerpo finito $\mathbb{F}_q$ tal que sea difícil resolver el PLD en él.

(2) Berto escoge un punto $P \in E(\mathbb{F}_q)$ tal que su orden sea lo mayor posible.

(3) Berto elige un entero secreto t y calcula $Q = tP$.

- **Clave pública de Berto**: E, $\mathbb{F}_q$, P y Q.
- **Clave privada de Berto**: t.

Output: Alicia envía el par (M_1, M_2) a Berto.

begin

$M \longleftarrow mensaje$;
$k \longleftarrow random\_integer()$;
$M_1 \longleftarrow kP$;
$M_2 \longleftarrow M + kQ$;

end

Algoritmo 2: ElGamal en curvas elípticas

Procedimiento de desencriptado:

Berto calcula $M = M_2 - tM_1$.

Comprobación:

Por las operaciones descritas, si $\mathcal{E}(M) = (M_1, M_2)$ denota la función de encriptado y $\mathcal{D}(M_1, M_2)$ la de desencriptado, se tiene que

$$\mathcal{D}(\mathcal{E}(M)) = M_2 - tM_1 = M + kQ - tkP = M + ktP - tkP = M.$$

Observación 4.2.2. (1) Si Eva pudiese resolver el PLD, sabiendo la clave pública de Berto, puede averiguar el mensaje resolviendo $Q = tP$, obteniendo t y con ello puede resolver $M = M_2 - tM_1$.

(2) Es importante que Alicia cambie en cada comunicación el número k. De no hacerlo y utilizar el mismo k para M y M', Eva podría darse cuente de eso porque $M_1 = M_1'$. Si el contenido de M ha sido publicado ya, por ejemplo un artículo científico, puede calcular $M' = M - M_2 + M_2'$. Por lo que, a partir de un mensaje conocido y comunicado anteriormente se puede averiguar uno próximo.

(3) La fortaleza de este algoritmo reside en el PLD, mientras que RSA, uno de los algoritmos de encriptación más utilizados, depende en gran medida en la dificultad de descomponer un número en factores primos. De hecho la fortaleza de este algoritmo consiste en que atacar a este algoritmo es equivalente a atacar al PLD; por otro lado, existen varios métodos para atacar RSA sin tener que afrontar el problema de la factorización, en el que reside su fortaleza.

(4) Si $\#E(\mathbb{F}_q)$ es primo, entonces todos los $P \neq \infty$ son generadores del grupo de puntos de la curva elíptica E, luego tienen orden máximo.

4.2.2. Firmas digitales con ElGamal

Las firmas digitales se utilizan para verificar que un mensaje en efecto ha sido enviado por parte de su emisor. Por tanto, se tiene que las firmas se vuelven más importantes que el contenido mismo del mensaje. Por ejemplo, si Ernesto tiene la firma digital de Ana, puede firmar una transferencia bancaria desde la cuenta de Ana hasta la de Ernesto. Para evitar esta situación se adaptaron los algoritmos de clave pública a las firmas digitales. A continuación se muestra el algoritmo de ElGamal para firmas digitales en curvas elípticas.

Algoritmo: firma digital con ElGamal *Prefacio:*
Alicia busca firmar un documento con el fin de que Berto sepa que es ella quien lo ha enviado. Sin embargo, no quiere que Eva o algún espía pueda romper su firma y poder usarla para cualquier documento no deseado. Por lo tanto, se tiene que unir la firma al documento de tal forma que ningún espía pueda apropiársela y Berto pueda confirmar que es Alicia la emisora.

Input:

(1) Alicia establece una curva elíptica E sobre un cuerpo finito $\mathbb{F}_q$, de tal forma que sea difícil romper el problema del logaritmo discreto (PLD).

(2) Alicia escoge un punto $P \in E(\mathbb{F}_q)$, tal que su orden N sea un número primo grande.

(3) Alicia escoge un entero secreto k, y genera $Q = kP$.

(4) Alicia elige una función $f : E(\mathbb{F}_q) \to \mathbb{Z}$ (e.g. $f(x, y) = x$ donde $0 \leq x \leq q$, si q es un número primo) con la menor cantidad de colisiones posible.

- **Clave pública de Alicia**: E, $\mathbb{F}_q$, f, P y Q.
- **Clave privada de Alicia**: k.

Output: Mensaje firmado: (m, R, s).

begin

Integer $m \longleftarrow doc$;
$a \longleftarrow random\_int()$;
while $\gcd(a, N) \neq 1$ **do**
  $a \longleftarrow ranom\_int()$;
end
$R \longleftarrow aP$;
$s \longleftarrow \mathrm{mod}(a^{-1}(m - kf(R)), N)$;
return (m, R, s)

end

Procedimiento verificación:

begin

$U \longleftarrow f(R)Q + sR$;
$V \longleftarrow mP$;
if $U = V$ **then**
  return *True*
end
else
  return *False*
end

end

Algoritmo 3: ElGamal de firmado en curvas elípticas

Comprobación:

Por las operaciones descritas, si $\mathcal{F}(M) = (m, R, s)$ denota la función de firmado y $\mathcal{V}(m, R, s)$ la de validación, se tiene que $\mathcal{V}(m, R, s) = M$ si

$$U = f(R)Q + sR = f(R)kP + saP = f(R)kP + (m - kf(R))P$$
$$= f(R)kP + mP - kf(R)P = mP = V.$$

Observación 4.2.3. (1) Si el entero m es más grande que N, se tiene que considerar o bien cambiar de curva elíptica a una más grande, o usar una función hash.

(2) Alicia no trata de mantener el mensaje m secreto, si fuera así, entonces sería necesaria una función de encriptación, y luego firmar el criptograma.

(3) Una desventaja de este sistema es que el mensaje firmado resulta tres veces aproximadamente más grande que el mensaje original. Una mejora sería la utilización de una función hash criptográfica.

(4) Si Eva buscase romper el sistema y obtener la firma tendría que romper el PLD. Si pudiera calcular los logaritmos discretos, entonces podría coger P y R y calcular a. Como sabe $s, f(R)$ y m, Eva tiene que

$$ks \equiv m - af(R) \bmod N.$$

Normalmente, se tiene que $d = \gcd(f(R), N) \neq 1$, por lo que existen d soluciones para la ecuación anterior. Mientras d sea pequeño Eva puede intentar con todas las posibilidades hasta encontrar a, mediante **ecuaciones diofánticas**.

(5) Si se firma sobre un mensaje al que se le ha aplicado una función hash, se puede utilizar otro algoritmo descrito en [Was08] en el capítulo 6, sección firma con ElGamal, por el que los cálculos de puntos por enteros en la verificación se reducen a dos.

(6) Si se toma N como un primo, entonces cualquier $0 < a < N$ funciona para ejecutar el algoritmo.

Capítulo 5

Ataques a las Curvas elípticas

Este capítulo será dedicado al estudio de los ataques a la criptografía con curvas elípticas mostrada en el capítulo anterior. Se explicarán distintos tipos de ataques y sus ventajas y desventajas, en cuanto a tiempo que tarde en romper el criptosistema, así como en el almacenamiento de datos o capacidad de procesamiento que requiera.

Es importante señalar que tras una búsqueda bibliográfica avanzada se ha llegado a la conclusión de que atacar a la criptografía en curvas elípticas tiene una complejidad similar a atacar al problema del logaritmo discreto, véase [Was08]. Otros ataques criptográficos que resultan efectivos contra curvas elípticas se salen del ámbito de este trabajo pues se basan en atacar la implementación de los criptosistemas y no en atacar el problema del logaritmo discreto. Es interesante observar que esto no siempre se da así, por ejemplo, RSA, [RAS77], depende de la dificultad de la factorización de números grandes en factores primos; sin embargo, atacar a RSA no es lo mismo que atacar al problema de la factorización, existen ataques enfocados al RSA que omiten la resolución de este problema, y en consecuencia, aumentan su eficacia. Se hizo un estudio en profundidad sobre este tema en [Bon98].

En este capítulo se estudiará primero el ataque del algoritmo "*Baby step, Giant step*", seguido de los métodos ρ y λ de Pollard y el método de Pohlig-Hellman. Estos algoritmos atacan el problema del logaritmo discreto en grupos finitos, en especial en curvas elípticas. Finalmente se explicará un método para atacar a curvas especiales, como son las curvas supersingulares, para demostrar que es más fácil romper el problema del logaritmo discreto en esas curvas.

Observación 5.0.1. Para un estudio de la complejidad de estos algoritmos la ISO 9001 realizó un estudio sobre los siguientes ataques resumiendo sus ventajas contra otros algoritmos [SS15].

5.1. Ataques genéricos a grupos abelianos finitos

Como ya se ha mencionado anteriormente, los siguientes ataques funcionan sobre cualquier grupo abeliano finito, y en particular, sobre curvas elípticas. Por este motivo, se cambiará la notación en esta

sección y se trabajará sobre un grupo finito abeliano G arbitrario, escrito con notación aditiva.

Se tomará que N es el orden de G, aunque es importante señalar que se puede tomar N como el orden del elemento P, ya que el subgrupo $\langle P \rangle$ es el único relevante a efectos del ataque. Obsérvese que en las aplicaciones criptográficas se suele tomar que el orden de G sea primo, y por tanto, los órdenes de sus puntos son divisores de N por el **Teorema de Lagrange** 1.4.7. Como N es un número primo, el orden de sus elementos sólo puede ser 1 o N, así que todos los elementos (distintos de la identidad) de G son generadores. Por tanto, el orden de P y de G son iguales.

5.1.1. Baby Step, Giant Step

Este método fue desarrollado por Daniel Shanks, véase [Sha71]. Este algoritmo sirve para atacar el problema del logaritmo discreto en general. A modo de observación, también se utilizó para calcular el orden de una curva elíptica, pero no se va a estudiar, ya que no entra dentro del alcance del trabajo (véase para más información [Was08] el capítulo 4). El algoritmo es el siguiente:

Algoritmo: Baby step, Giant step. *Prefacio:*
Se tiene que $Q = kP$ tal que $P, Q \in G$ y $k \in \mathbb{Z}$.

Comprobación:

La razón de que este **método** funcione se debe a la elección de la m al principio del algoritmo. Se tiene que $m^2 \geq N$, por lo que se deduce que $0 \leq k < m^2$. Por tanto, se puede escribir de forma única $k = k_0 + mk_1$, lo que significa $k_0 \equiv k \bmod m$, siendo $k_1 = \lfloor (k - k_0)/m \rfloor$; por lo que, $0 \leq k_1 < m$. En el momento en el que se da la coincidencia de elementos en el algoritmo, se tiene que $i = k_0$ y $j = k_1$, por lo que se da

$$Q - k_1 mP = kP - k_1 mP = k_0 P,$$

así se demuestra que hay una coincidencia y que se puede tomar $k = k_0 + k_1 m$.

Observación 5.1.1. (1) Este algoritmo requiere aproximadamente $\sqrt{N}$ pasos para comprobar la lista en busca de coincidencias y $\sqrt{N}$ espacio de almacenamiento, en consecuencia.

(2) Este método se vale de dos pasos importantes: el cálculo de iP, un **Baby step** y el cálculo de $Q - jmP$, un **Giant step**.

(3) Este método puede tener una mejora en espacio y tiempo en el cálculo de iP, calculando y almacenando sólo los puntos donde $0 \leq i \leq m/2$. Luego se comprobaría si $Q - jmP = \pm iP$.

(4) Este método, al contrario que los siguientes, no requiere exactamente el valor de $N = \#G$, sino que con una estimación basta. En el caso de curvas elípticas, la aproximación que se obtiene del **Teorema de Hasse** 3.5.1 es suficiente; por lo que, el rendimiento aumenta.

Input:

(1) Un grupo abeliano finito G.

(2) Elemento $P \in G$.

(3) Elemento $Q \in G$, tal que $Q = kP$, donde $k \in \mathbb{Z}$.

Output: El entero k mod N.

```
begin
    Integer m ⟵ randint(√N, N);
    i ⟵ 0;
    j ⟵ 0;
    salir ⟵ false;
    while i < m do
        list ⟵ iP;
        i ⟵ i + 1;
    end
    while j < m and !salir do
        D ⟵ Q − jmP;
        if D in list then
            r ⟵ list.index(D);
            La función index devuelve el índice en el que se encuentra el elemento en la lista.;
            salir ⟵ True;
        end
        else
            j ⟵ j + 1;
        end
    end
    return mod(r + jm, N);
end
```

Algoritmo 4: Baby Step, Giant Step

5.1.2. ρ de Pollard

El método anterior requiere demasiado espacio de almacenamiento para N grandes, que es el caso más común. En contraste, el método de ρ de Pollard requiere mucho menos espacio de almacenamiento que el "Baby Step, Giant Step" 5.1.1, y el tiempo de ejecución es similar.

Este método consiste en calcular puntos de la curva elíptica de forma pseudoaleatoria hasta que dos de estos puntos coinciden. Según [Pol78], esta coincidencia se da aproximadamente cada

$$0,6267\sqrt{N}$$

elementos. Esta serie de operaciones pueden representarse como el dibujo de una letra griega ρ, por eso el método lleva tal nombre. Obsérvese que siempre se obtendrá una coincidencia de puntos, ya que al ser un grupo finito.

Este algoritmo es algo difícil de implementar, ya que para aprovechar la ventaja de necesitar menos espacio de almacenamiento es indispensable un método o criterio para seleccionar qué puntos almacenar

y cuáles no. De lo contrario, se tiene que se almacenarán el mismo número de elementos en ρ de Pollard y en "Baby Step, Giant Step".

Asimismo, es necesaria una función para generar los puntos del grupo, la principal característica de esta función es que se comporte de manera aparentemente aleatoria. Además, esta función tiene que tener propiedades deterministas, es decir, los elementos que genere tienen que depender de una semilla, y dada la misma semilla, el resultado es el mismo. Por ejemplo,

$$P_{i+1} = f(P_i) = f(P_j) = P_{j+1},$$

si y sólo si, $P_i = P_j$. La clave del algoritmo consiste en que una vez hallada una coincidencia de dos puntos con una diferencia de d elementos entre ellos, los puntos que les seguirán serán todos coincidencias con una diferencia de d elementos.

Observación 5.1.2. Originalmente, Pollard ideó este método como forma de mapear un grupo finito con una gran cantidad de valores [Pol78]. Más tarde, se adaptó a las curvas elípticas dando lugar al algoritmo actual.

Una construcción estándar es hacer $P_{i+1} = f(P_i)$ con P_0 fijado (la semilla). La función de hashing f se construye como sigue:

(1) Fijar un entero s. Una recomendación sería dividir el grupo G en s subconjuntos de tamaño similar, $S_1, \ldots, S_s$.

(2) Elegir aleatoriamente $2s$ números enteros a_i, b_i módulo el orden del grupo.

(3) Definir s elementos M_i, tal que $M_i = a_iP + b_iQ$. En el algoritmo tendrán esta forma $M_i = [a_i, b_i, a_iP + b_iQ]$.

(4) Si $T \in S_i$, entonces se toma $f(T) = T + M_i$. Esta operación se realiza con el último elemento de la lista M_i.

(5) La función f guarda los coeficientes modificados a_i, b_i por la operación anterior en los elementos 0 y 1 de la lista del elemento T.

Teniendo todos estos datos se puede proceder a formular el algoritmo para el método ρ de Pollard.

Algoritmo ρ de Pollard *Prefacio:*
Sea N el orden de G. Se tiene que $Q = kP$ tal que $P, Q \in G$ y $k \in \mathbb{Z}$.

Input:

(1) Grupo abeliano finito G.

(2) Elemento $P \in G$.

(3) Elementos $Q \in G$, tal que $Q = kP$, donde $k \in \mathbb{Z}$.

(4) Entero s (recomendación: un número alrededor de 20).

(5) Función de iteración f descrita previamente.

(6) Lista de elementos M_i.

Output: El entero k mod N.

```
begin
    d ⟵ N;
    while d = N do
        I ⟵ [randint(2, N), randint(2, N), 0];
        I ⟵ [I[0], I[1], I[0]P + I[1]Q];
        list.add(I);
        U ⟵ f(I);
        while U not in list do
            list.add(U);
            U ⟵ f(U);
        end
        V ⟵ list[list.index(U)];
        La función index devuelve el índice en el que se encuentra el elemento en la lista.
         d ⟵ gcd(U[1] − V[1], N);
    end
    nList ⟵ [N, N/d];
    dif ⟵ gcd(U[1] − V[1], nList[1]);
    while dif ≠ 1 do
        nList[1] ⟵ nList[1]/dif;
        dif ⟵ gcd(U[1] − V[1], nList[1]);
    end
    A ⟵ U[1] − V[1];
    B ⟵ V[0] − U[0];
    inv ⟵ inverse_mod(A, nList[1]);
    elem ⟵ inv · B;
    while inv < nList[0] do
        if (elem %nlist[0])P = Q then
            return mod(elem, nList[0]);
        end
        else
            elem ⟵ elem + nList[1];
        end
    end
end
```

Algoritmo 5: ρ de Pollard

Comprobación:

Se podría pensar que este método no calcula realmente la k objetivo debido al cambio de módulo. Sin embargo, es inmediato colegir que, por las ecuaciones planteadas, no puede estar equivocado el resultado, ya que se comprueban los resultados con cada k resultante de los cálculos.

En la última parte, este método, lo que hace es dividir N en d fragmentos y explorar el módulo para encontrar k mod N; no obstante, como se hace en el método, es posible que sea necesario dividir N en más fragmentos. Pero en aplicaciones criptográficas, el número N es comúnmente primo; por lo que, d sólo puede ser 1 ó N. Si $d = N$, entonces hay que seleccionar otro punto inicial y empezar de nuevo, pues los coeficientes son múltiplos de N. En cambio si $d = 1$, se obtiene k mod N.

Observación 5.1.3. (1) Existe otro método que usa estos principios, pero que utiliza un acercamiento algo distinto. Este es el método λ de Pollard. Este método consiste en aplicar el algoritmo anterior en paralelo en varios terminales. Se empieza en cada terminal por un punto distinto, y los puntos que cumplan determinada condición son llamados **distinguidos** y se envían a un ordenador central para su procesamiento. A partir de este punto, todo es similar al anterior algoritmo.

(2) Estos métodos, a diferencia del "Baby Step, Giant Step"5.1.1, son **probabilísticos**, es decir, es probable que acaben en tiempo $\sqrt{N}$. Por contra, "Baby Step, Giant Step"termina en tiempo constante $\sqrt{N}$, pues es **determinista**.

(3) Para seleccionar qué elementos almacenar se utilizará en la implementación software un método extraído de [Was08], según el cual, sólo se almacena una par (P_i, P_{2i}), donde $i \geq 1$; cuyos elementos se pueden calcular así,

$$P_{i+1} = f(P_i),\ P_{2(i+1)} = f(f(P_{2i})).$$

Con esto se reduce significativamente el espacio necesario de este algoritmo.

(6) En la implementación software se realiza una modificación de la función f para adaptarla a curvas elípticas. Se generarán s puntos M de la misma manera que anteriormente, y con estos puntos se realizará la función. El cambio más significativo consiste en la manera de elegir el elemento M_i por el que se va a operar. Se va a seleccionar este elemento según la coordenada x del punto semilla P. Es decir,

$$\text{Si } x_P \equiv i \bmod s, \text{ entonces se toma } f(P) = P + M_i.$$

Los pasos sucesivos sucederán de la misma manera que en el algoritmo presentado.

5.1.3. Método de Pohlig-Hellman

Este método se basa en Teorema Chino del Resto 1.5.4 para calcular el logaritmo discreto. Este método es un ataque al problema del logaritmo discreto en general, pero con una ligera adaptación puede atacar a los criptosistemas basados en curvas elípticas.

La clave de este método es encontrar k para cada uno de los factores del orden del grupo abeliano G, N. A continuación, se utilizará el Teorema Chino del Resto para combinar estos resultados y obtener el entero k objetivo. Por lo tanto, es necesario factorizar N, resultando en una lista de primos elevados a un exponente, de modo que

$$N = \prod_i p_i^{e_i},$$

con p_i primos distintos. La idea clave del método es encontrar k módulo $p_i^{e_i}$ para cada i y luego juntar los resultados usando el Teorema Chino del Resto. Para un factor $p = p_i$ y $e = e_i$. Para generar estos enteros este método computa la expresión de k en la base p de este modo:

$$k = k_0 + k_1 p + k_2 p^2 + \ldots + k_{e-1} p^{e-1},$$

donde $0 \leq k_i < p$. Dadas estas directrices se puede enunciar el algoritmo para este método de ataque a criptosistemas basados en curva elíptica.

Algoritmo: Método de Pohlig-Hellman. *Prefacio:*
Sea N el orden de P, nótese el cambio de notación. Se tiene que $Q = kP$ tal que $P, Q \in G$ y $k \in \mathbb{Z}$.

Input:

(1) Grupo abeliano finito G.

(2) Elemento $P \in G$.

(3) Elemento $Q \in G$, tal que $Q = kP$, donde $k \in \mathbb{Z}$.

Output: El entero k mod N.

begin

$factores \longleftarrow factor(N)$;

Factorización de N en factores primos: $[[p_1, e_1], \ldots, [p_m, e_m]]$;

$kVector \longleftarrow [\,]$;

for $i \longleftarrow 0$ **to** $length(factores)$ **do**

$T \longleftarrow [\,]$;

$kList \longleftarrow [\,]$;

$ind \longleftarrow 0$;

$j \longleftarrow 0$;

$qi \longleftarrow Q$;

while $j < factores[i][0]$ **do**

$T.add(j\frac{N}{factores[i][0]}P)$;

$j \longleftarrow j + 1$;

end

while $ind < factores[i][0]$ **do**

$ind \longleftarrow ind + 1$;

$valor \longleftarrow \frac{N}{factores[i][0]^{ind}}qi$;

$kList.add(T.index(valor))$;

La función index devuelve el índice en el que se encuentra el elemento en la lista. **if** $ind < factores[i][1]$ **then**

$qi \longleftarrow qi - kList.last\_elem() \cdot factores[i][0]^{ind-1} \cdot P$;

end

end

$k \longleftarrow 0$;

for $l \longleftarrow 0$ **to** $factores[i][1]$ **do**

$k \longleftarrow kList[l] \cdot factores[i][0]^{l}$;

$l \longleftarrow l + 1$;

end

$kVector.add(mod(k, factores[i][0]^{factores[i][1]}))$;

end

return $tcr(kVector, factores)$;

La función tcr invoca al Teorema Chino del Resto.

end

Algoritmo 6: Método de Pohlig-Hellman

Comprobación:
Como la parte final es consecuencia del Teorema Chino del Resto, basta demostrar que el algoritmo hace el cálculo correctamente de k mod $p_i^{e_i}$, para todo factor p^e de N. Es decir, hay que probar si realmente ocurre una coincidencia entre T y el valor calculado en el algoritmo 5.1.3. Para ello se observa que

$$\tfrac{N}{p}Q = \tfrac{N}{p}(k_0 + k_1 p + \ldots)P = k_0\tfrac{N}{p}P + (k_1 + k_2 p + \ldots)NP = k_0\tfrac{N}{p}P,$$

porque $NP = \infty$. Por lo que, en efecto se encuentra una coincidencia k_0. A continuación,

$$\begin{gathered} Q_1 = Q - k_0 P = (k_0 + k_1 p + \ldots)P - k_0 P = (k_1 p + k_2 p^2 + \ldots)P; \\ \tfrac{N}{p^2}Q = (k_1 + k_2 p + \ldots)\tfrac{N}{p}P; \\ k_1\tfrac{N}{p}P + (k_2 + k_3 p + \ldots)NP = k_1\tfrac{N}{p}P. \end{gathered}$$

Así que se encuentra k_1, por inducción se demuestra que se pueden encontrar las demás k de forma similar. Por estos cálculos se deduce que el proceso se tiene que detener cuando se llega $e-1$, ya que N/p^{e+1} no es un entero porque p^{e+1} no es un factor de N, por lo que no se puede multiplicar este término por N/p^{e+1}. Sin embargo, en este momento ya se ha calculado k mod p^e, que es lo que se necesita para aplicar el Teorema Chino del Resto; por lo tanto, no es necesario seguir realizando cálculos.

Observación 5.1.4. (1) Este método depende de los factores que compongan a N, ya que si p es un número muy grande será costoso calcular T; por lo que, la eficiencia disminuirá.

(2) En este método se usó, en vez del orden de G, el orden del punto P. Esto se debe a que T genera elementos del subgrupo generado por P, por tanto, si se usa el orden de G se corre el riesgo de que se generen puntos fuera del subgrupo de P y no se hallen las k correspondientes. Esto sucede a la hora de comparar $\frac{N}{p}Q$ y un elemento de T.

(3) Se puede intentar averiguar las k_i sin T, pero se incurriría en atacar al problema del logaritmo discreto en el grupo generado por $\frac{N}{p}P$, con orden p. Si, por algún motivo, tiene una magnitud similar a N, entonces este método es poco efectivo.

(4) Se puede deducir que las aplicaciones criptográficas, usarán grupos abelianos finitos cuyo orden sea el primo más grande posible. Debido a esto, los métodos como Pohlig-Hellman resultarían muy costosos, y por tanto, poco eficaces.

5.1.4. Index Calculus

El ataque "Index Calculus" es uno de los ataques más efectivos contra el problema del logaritmo discreto en el grupo multiplicativo del cuerpo finito $\mathbb{F}_p^*$. Sin embargo, no se aplica a grupos arbitrarios en su forma original.

Sean p, un número primo y g una raíz primitiva módulo p, es decir, generador del grupo finito $\mathbb{F}_p^*$. Sea $k = \log(h)$, para algún $h \not\equiv 0 \bmod p$, el logaritmo discreto de h en base g, tal que $h \equiv g^k$. Se tiene que $g^{\log(h)} \equiv h \bmod p$.

Suponiendo que se tiene h_1 y h_2. Entonces

$$g^{\log(h_1 h_2)} \equiv h_1 h_2 \equiv g^{\log(h_1)+L(h2)} \bmod p,$$

lo que implica que $\log(h_1 h_2) \equiv \log(h_1) + \log(h_2) \bmod p - 1$.

El método "Index Calculus" es un método de calcular valores de la función logaritmo discreto log. La idea detrás de este algoritmo es computar la función log para una cantidad finita de primos pequeños B (llamada la base factorial), y entonces utilizar esta información para calcular $\log(h)$ para h cualquier producto de estos primos. La clave es, pues, que con un numero relativamente pequeño de primos somos capaces de generar una gran cantidad de numeros compuestos.

Algoritmo: Index Calculus. *Prefacio*
Sean p, un número primo y g una raíz primitiva módulo p, se tiene que $h = g^k \bmod p$, donde $k \in \mathbb{Z}$.

Input:

(1) Grupo finito $\mathbb{F}_p$ módulo p.

(2) Generador g del grupo finito módulo p.

(3) Elemento $h \in \mathbb{F}_p^*$, tal que $h = g^k \bmod p$.

(4) Base factorial B, una lista de primos empezando desde el -1 de longitud r.

Output: Entero k tal que $g^k \equiv h \bmod p$.

```
begin
    relations ⟵ [ ];
    k ⟵ 1;
    while length(relations) < r do
        rel ⟵ smooth_factor(g^k, p, B);
        La función smooth_factor(q, p, B) devuelve la factorización del número q mod p utilizando
         elementos de la lista B. Esta función devuelve sólo los exponentes e_i tal que
         [(−1)^e0, 2^e1, ..., p_{r−1}^{e_r}];
        if relations.lineal_independent(rel) then
            rel.add(k);
            relations.add(rel);
        end
        k ⟵ k + 1;
    end
    S ⟵ find_equations_disc_log(relations, p − 1);
    La función find_equations_disc_log(relations, p) utiliza la definición anterior 5.1.4 de
     logaritmo discreto L para hallar las ecuaciones de congruencia de las relaciones en módulo
     p − 1;
    solutions ⟵ solve_equations(S, p − 1);
    Para cada elemento de B se halla el logaritmo discreto en módulo p − 1;
    while True do
        j ⟵ random_int();
        factores ⟵ smooth_factor(g^j, p, B);
        sis_eq ⟵ find_equations_disc_log(factores, p − 1);
        sol ⟵ solve_equations(sis_eq, p − 1);
        return sol[1];
        Se devuelve el segundo elemento de la lista porque el primero es el número del que se
         quiere averiguar el logaritmo discreto.;
    end
end
```

Algoritmo 7: Index Calculus

Comprobación

Este algoritmo utiliza sistemas de ecuaciones de congruencias en base $p-1$ para hallar k utilizando una base de números primos pequeños B. Estos números primos se utilizan para calcular fácilmente logaritmos discretos más pequeños y utilizarlos para calcular el logaritmo discreto sobre números más grandes.

Observación 5.1.5. (1) La elección de la longitud de la base B es importante, pues el tiempo de ejecución del algoritmo dependerá en gran medida de ella. Si el tamaño de B es muy pequeño, el algoritmo tardará en encontrar relaciones para las potencias de g. En cambio, si el tamaño de B es muy grande, las relaciones serán rápidamente calculadas, pero resolver la ecuaciones de congruencias se volverá muy complicado.

(2) Hay varios métodos para producir las relaciones entre las potencias de g y los primos de B. Un método popular utiliza los métodos de criba. Véase [JL02].

(3) El tiempo de ejecución esperado de este algoritmo es aproximadamente una constante por $\exp(\sqrt{2\ln p \ln\ln p})$, véase [MOV18] y [Was97] para más información.

(4) Obsérvese que este algoritmo depende mucho en el hecho de que los número enteros se pueden escribir como productos de primos. Esta es la razón de que no haya una generalización a grupos arbitrarios, un análogo de la factorización en factores primos para grupos arbitrarios no se encuentra disponible.

(5) A modo de ampliación, este método tiene una adaptación a curvas elípticas [MM17]; sin embargo, este método es complicado de implementar y una implementación pobre arruinaría el rendimiento de este algoritmo, que, según [MM17], es mejor que el de ρ de Pollard.

5.2. Ataques por emparejamientos

Aparte de los métodos anteriores que atacan a grupos arbitrarios definidos en una curva elíptica, existen otro tipo de ataques específicos para curvas elípticas. Estos ataques se basan, no en las propiedades del grupo, si no en la capacidad de reducir el problema del logaritmo discreto de ese grupo a otro en el que sea más fácil calcularlo. En otras palabras **transferir** el problema del logaritmo discreto de un grupo a otro.

Este tipo de ataques se denominan **ataques de transferencia**; sin embargo, debido a que utilizan conceptos como los emparejamientos de Weil o de Tate-Lichtenbaum, también se llaman **ataques por emparejamiento**.

En esta sección sólo se estudiará uno de estos ataques que ejemplifica cómo funcionan estos métodos y sus propiedades.

5.2.1. Ataque MOV

Este ataque coge su nombre de sus inventores: Menezes, Okamoto y Vanstone, [MOV93]. El ataque MOV utiliza el **emparejamiento de Weil** estudiado en el capítulo anterior 3.5.3 para **transferir** el problema del logaritmo discreto de $E(\mathbb{F}_q)$ a uno en $\mathbb{F}_{q^m}^*$. El problema del logaritmo discreto en el cuerpo $\mathbb{F}_{q^m}^*$ es mucho más sencillo de resolver que en una curva elíptica, ya que, normalmente, se tiene

que en las aplicaciones criptográficas se utilizan curvas elípticas con muchos puntos, lo que hace que ataques genéricos sean poco efectivos. En cambio, en el cuerpo $\mathbb{F}_{q^m}^*$ se pueden utilizar métodos como el *Index Calculus* para extraer la solución al problema en un tiempo pequeño.

Para las pruebas de este método se utilizarán curvas supersingulares, esto se debe a que para el cálculo de este método es necesario la obtención del conjunto de los **puntos de N-torsión**, $E[N]$. Estos puntos tienen sus coordenadas en el cierre algebraico de $\mathbb{F}_q$, $\overline{\mathbb{F}_q}$. No obstante, como $E[N]$ es una cantidad finita de puntos y $\overline{\mathbb{F}_q} = \bigcup_{n\geq 1} \mathbb{F}_{q^n}$, se tiene que los puntos de $E[N]$ tienen coordenadas en $\mathbb{F}_{q^m}$ para m suficientemente grande. Esta m puede ser arbitrariamente grande, pero en curvas supersingulares 3.6 puede demostrarse que, puede tomarse $m = 2$. La demostración puede verse en [Was08] en el capítulo 5, sección Ataque MOV.

Si el $\gcd(q, N) = 1$ y $S, T \in E[N]$, entonces $e_N(S, T)$ es una raíz de unidad y se puede calcular relativamente rápido. Para más información del emparejamiento de Weil, véase el 3.5.3.

A través de este método las curvas supersingulares se convierten en objetivos muy vulnerables, quedando su seguridad vinculada al PLD clásico. Por lo cual su ventaja de cálculos rápidos se vuelve intrascendente frente a la poca seguridad que ofrecen. Para más propiedades de las curvas supersingulares, véase el 3.6.

Algoritmo: Método MOV. *Prefacio:*
Se tiene que $Q = kP$ tal que $P, Q \in G$ y $k \in \mathbb{Z}$.

Input:

(1) Curva elíptica E definida sobre un cuerpo finito $\mathbb{F}_q$.

(2) Punto $P \in G$.

(3) Punto $Q \in G$, tal que $Q = kP$, donde $k \in \mathbb{Z}$.

(4) Entero $N = \#E(\mathbb{F}_q)$.

(5) Entero $m > 0$ tal que los puntos de N-torsión tengan coordenadas en $\mathbb{F}_{q^m}$.

Output: El entero k mod N.

begin

$F \longleftarrow EllipticCurve(\mathbb{F}_{q^m}, E.weierstrass\_eq())$;

La función weierstrass_eq() devuelve la ecuación de Weierstrass de la curva E;

$kList \longleftarrow [\,]$;

$dList \longleftarrow [\,]$;

while lcm$(dList) \neq N$ **do**

La función lcm *lleva a cabo el mínimo común múltiplo de la lista recibida.*;

$T \longleftarrow F.random\_point()$;

$M \longleftarrow T.order()$;

$d \longleftarrow \gcd(M, N)$;

$T \longleftarrow \frac{M}{d}T$;

$r1 \longleftarrow weil\_pairing(P, T, N)$;

$r2 \longleftarrow weil\_pairing(Q, T, N)$;

La función $weil\_pairing(P, T, N)$ *devuelve el emparejamiento de Weil de* P *y* T *en módulo* N;

$dList.add(r1.order())$;

Importante: El orden de este elemento debe ser el orden multiplicativo.

$kList.add(r2.\log(r1))$;

La función log *calcula el logaritmo discreto de* $r2$ *en base* $r1$ *y devuelve* k mod d;

end

return $tcr(kList, dList)$;

end

Algoritmo 8: Ataque MOV

Comprobación:

La clave del funcionamiento de este algoritmo es el siguiente resultado.

Lema 5.2.1. *Sea $P, Q \in E(\mathbb{F}_q)$ y sea N el orden de P. Existe k tal que $Q = kP$ si y sólo si, $NQ = \infty$ y el emparejamiento de Weil es $e_N(P, Q) = 1$.*

Demostración. Si se tiene que $Q = kP$, entonces $NQ = kNP = \infty$ porque N es el orden de P. Asimismo, $e_N(P, Q) = e_N(P, P + P + P + \ldots + P) = e_N(P, P)^k = 1^k = 1$. Por lo que se demuestra el primer principio.

Para el segundo principio, se supone la segunda parte del lema. Se toma en cuenta que $\gcd(N, q) = 1$. Se elige un punto R tal que $\{P, R\}$ sea la base de $E[N]$. Así se tiene que

$$Q = aP + bR$$

para algunos enteros a, b. Se tiene que $e_N(P, R) = \zeta$, siendo ζ una raíz primitiva N-ésima de la unidad, al ser P y R generadores de $\mathbb{Z}_N \oplus \mathbb{Z}_N$. Por tanto, si $e_N(P, Q) = 1$, entonces se tiene que

$$1 = e_N(P, Q) = e_N(P, aP + bR) = e_N(P, P)^a e_N(P, R)^b = e_N(P, R)^b = \zeta^b.$$

Por tanto, se tiene que $b \equiv 0 \bmod N$, por lo que $bR = \infty$. Así que, $Q = aP$ como se quería demostrar. ■

Las primeras instrucciones del algoritmo indican cómo conseguir un punto de N-torsión. El problema ocurre cuándo se calcula el emparejamiento de Weil de los puntos. El emparejamiento de Weil devuelve una raíz N-ésima de la unidad, pero no es seguro que sea un generador del grupo de raíces, es decir, una raíz primitiva N-ésima de la unidad. Por tanto, se calculan varios logaritmos discretos en diferentes módulos y luego se fusionan con el Teorema Chino del Resto.

Observación 5.2.2. (1) Este ataque es tremendamente efectivo contra curvas elípticas supersingulares, en las que los cálculos son muy rápidos.

(2) El orden que ha de ser añadido a la lista *dList* debe ser el orden multiplicativo del elemento.

(3) Si se quiere atacar a curva elípticas arbitrarias, sería necesario hallar una m tal que $\mathbb{F}_{q^m}^*$ sea el cierre algebraico de $\mathbb{F}_q$, lo cual es bastante costoso; por lo que es poco factible.

(4) Para la ejecución de este método es necesario obtener el orden de un punto de la curva elíptica, lo cual es costoso dependiendo del orden de la curva.

Capítulo 6

Implementación y Pruebas

Este capítulo está dedicado a la implementación software de los algoritmos presentados y estudiados en el capítulo anterior. Asimismo, se llevarán a cabo varias pruebas para comparar estos métodos entre sí teniendo en cuenta el tiempo que necesitan para romper una clave.

Observación 6.0.1. Se recuerda que la implementación de estos algoritmos va a estar orientada a atacar criptosistemas en curvas elípticas.

La implementación de los métodos de ataque se ha llevado a cabo en **SageMath 8.7** [sag], concretamente en el *Notebook* que implementa Sage usando el framework de *Jupyter* integrado en el entorno. SageMath es un entorno que utiliza como lenguaje de programación Python 2.7. El lenguaje Python es de tipado dinámico, multiparadigma, interpretado y multiplataforma, para más información véase [pyd]. El *Notebook* va estar dividido en celdas en las que escribir el código y ejecutarlo. Lo más importante de este entorno es que añade una gran cantidad de librerías que implementan funciones matemáticas; además de proporcionar también librerías de manejo de curvas elípticas, con lo que se hace mucho más fácil la tarea de implementar los ataques.

6.1. Implementación

Los ficheros que contienen los ataques implementados se encuentran en el **Apéndice 3**. Estos ficheros están en formato *Notebook*: ".ipynb". Por este motivo es necesario el entorno SageMath para ejecutar los archivos correctamente. Junto con el **Apéndice 3** hay una archivo guía *Tutorial.pdf* con instrucciones para la correcta ejecución de los ficheros, así como un fichero para probar funciones del *Notebook*: *Tutorial.ipynb*. Además de eso, en el **Apéndice 1** se puede encontrar la documentación del código para entender las funciones que implementa cada fichero.

Hay un fichero de *Notebook* por cada método de ataque, lo que hacen cuatro ficheros:

- *BSGS.ipynb*: implementa el ataque *Baby Step, Giant Step*.

- *RhoPollard.ipynb*: implementa el ataque ρ *de Pollard*.
- *Pohlig-Hellman.ipynb*: implementa el ataque *Pohlig-Hellman*.
- *MOV.ipynb*: implementa el ataque *MOV*.

Además de estos ataques también se ha implementado el algoritmo de firma digital *ElGamalDSA*. Este método se encuentra en el fichero *ElGamalDSA.ipynb*. Por último, el fichero *Pruebas.ipynb* contiene el código utilizado para llevar a cabo las pruebas de los algoritmos.

6.2. Pruebas

Los resultados de las pruebas se encuentran en el **Apéndice 2**. Las pruebas han seguido el siguiente procedimiento:

- Habrá **4 intervalos de orden** diferentes, en función del orden de la curva elíptica utilizada:
  (1) Orden 1: Órdenes entre 5000 y 7000.
  (2) Orden 2: Órdenes entre 7000 y 10000.
  (3) Orden 3: Órdenes entre 10000 y 15000.
  (4) Orden 4: Órdenes entre 15000 y 20000.
- Cada uno de los algoritmos ("Baby Step, Giant Step", ρ de Pollard, método de Pohlig-Hellman) se ejecutará en **10 curvas elípticas distintas** por cada intervalo de orden. Sin embargo, el ataque **MOV** se va a probar con curvas elípticas distintas de los otros algoritmos, usándose así curvas elípticas **supersingulares**.
- Cada algoritmo intentará romper el problema del logaritmo discreto en **20 puntos** por cada curva elíptica dada.
- Todas las curvas tendrán orden primo, por lo que sus puntos, por el **Teorema de Lagrange**, también tendrán el mismo orden. Esta característica no la comparten las curvas supersingulares.
- Los puntos de las curvas con los que se llevarán a cabo las pruebas de los algoritmos son aleatorios.
- Se medirán y registrarán los tiempos que tardan en ejecutarse estos algoritmos. Estos tiempos se medirán en **milisegundos**.
- Utilizando los tiempos, se realizará una media de tiempos de cada algoritmo por cada intervalo de orden. Teniendo estos datos se realizará una gráfica comparativa de la media de los tiempos resultantes.

Toda esta información está recogida en el **Apéndice 2**, además de los detalles de cada curva elíptica utilizada. Para más información acerca de la implementación de las pruebas véase del **Apéndice 1** la sección Pruebas.

La razón por la cual se han elegido específicamente curvas elípticas con órdenes primos es para dar más realismo a las pruebas; ya que las aplicaciones criptográficas utilizan normalemente curvas elípticas cuyos órdenes son números primos de varios cientos de dígitos. Esto se debe a que para romper el problema del logaritmo discreto no hace falta buscar en toda la curva, basta con buscar en el subgrupo del punto de la clave pública. La curva, al tener orden primo, está compuesta por puntos cuyos órdenes de sus respectivos subgrupos son divisores del orden de la curva. Por el Teorema de Lagrange; por tanto, sus órdenes sólo pueden ser 1 ó el orden de la curva, puesto que el único punto de orden 1 es el ∞, sus órdenes sólo pueden ser el mismo que el de la curva.

El equipo utilizado para ejecutar las pruebas de los algoritmos de ataque es un equipo portátil con un procesador Intel Core i5-6200U 2.4 GHz, memoria RAM de 8 GB y una tarjeta gráfica NVIDIA GeForce 940MX.

6.2.1. Resultados

Los resultados de las pruebas se hallan en el **Apéndice 2**, así como los detalles de las mismas. A continuación se realizará un análisis de los resultados obtenidos de las pruebas definidas anteriormente.

En base a los resultados obtenidos de las pruebas de los cuatro algoritmos se ha generado una gráfica con el objetivo de ilustrar mejor cómo se comportan estos algoritmos frente a curvas elípticas de distinto orden y las diferencias en el rendimiento entre los algoritmos.

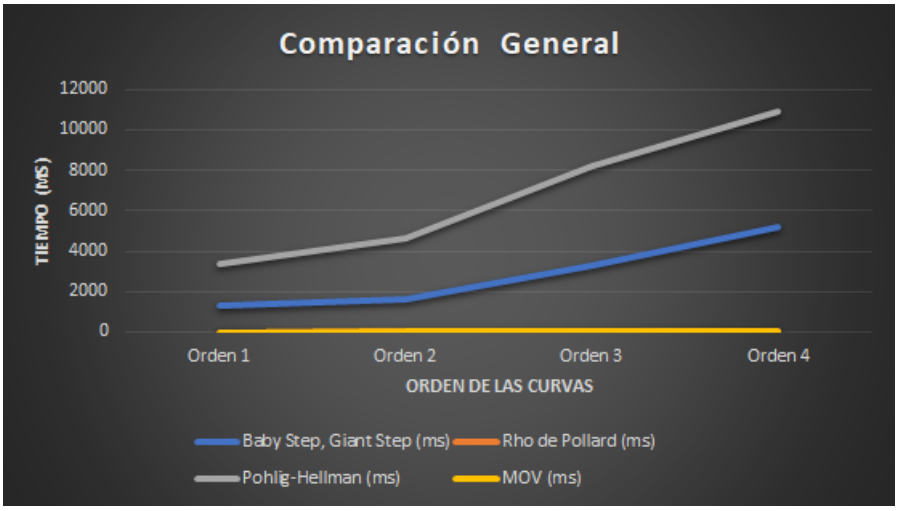

FIGURA 6.1: Comparación del tiempo de ejecución de los cuatro algoritmos

Esta gráfica 6.1 compara los tiempos de ejecución que se han medido para cada algoritmo en las pruebas realizadas. Como se puede observar el tiempo que toma el método de **Pohlig-Hellman** es

superior con mucho al del resto de los algoritmos. Esto se debe a que los órdenes de todas las curvas elípticas usadas en las pruebas son números primos. Se recuerda que la fortaleza de este método reside en romper el PLD para cada uno de los factores del orden de un punto. Por ello cuantos más haya y menores sean, más rápido funciona el algoritmo. Sin embargo, al elegir curvas elípticas de orden primo, la factorización es trivial. Debido a esto se tiene que la factorización de los puntos dificulta la ejecución del método Pohlig-Hellman.

El siguiente algoritmo que más tiempo de computación requiere es el **Baby Step, Giant Step**. Este algoritmo tiene una complejidad exponencial de $\sqrt{N}$, donde N es el orden de la curva. Sin embargo, al ser un algoritmo determinista es comprensible que su tiempo de ejecución sea inferior al del método Pohlig-Hellman que es probabilístico. La razón de que haya una leve diferencia entre los puntos de orden 1 y orden 2 es, probablemente debido a que las curvas de orden 2 utilizadas tienen órdenes cercanos a las de orden 1.

En la gráfica 6.1 se observa que los ataques **MOV** y **ρ de Pollard** tienen tiempos lo suficientemente pequeños en comparación con los ataques anteriores para que no sean apreciados en la gráfica anterior. Así que se ha generado otra gráfica con estos dos métodos de ataque para analizarlos más en profundidad.

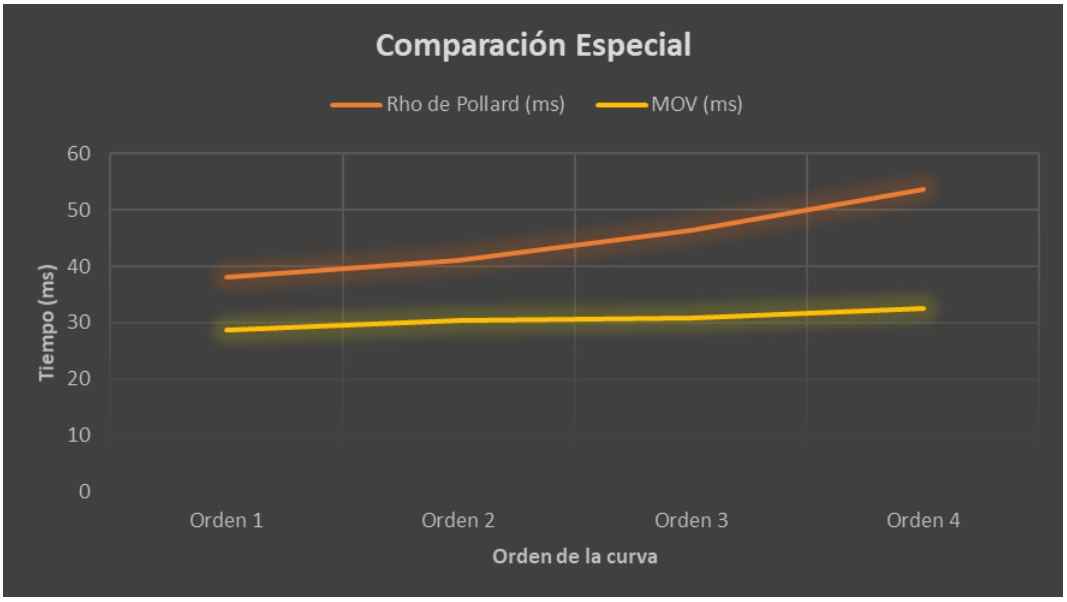

FIGURA 6.2: Comparación exhaustiva de los métodos ρ de Pollard y MOV

Para generar la gráfica 6.2 se han duplicado las pruebas realizadas sobre los algoritmos de MOV y ρ de Pollard, para hacerlas más exhaustivas. De tal forma, estos algoritmos se han probado con 20 curvas elípticas aleatorias y 20 puntos aleatorios en cada curva.

Como se puede apreciar en esta gráfica el algoritmo **MOV** tiene un tiempo menor que el ρ de Pollard. Esto es debido a que este ataque sólo ha sido probado utilizando curvas elípticas supersingulares. La razón de que este algoritmo sea tan rápido reside en que la principal dificultad de este algoritmo es

encontrar el cierre algebraico del cuerpo finito de la curva y trabajar en un cuerpo que puede ser arbitrariamente grande. Al estudiar únicamente el ataque utilizando curvas supersingulares, el cálculo del cierre algebraico se realiza en tiempo constante; ya que para una curva elíptica supersingular E en el cuerpo finito $\mathbb{F}_q$, el cierre algebraico del cuerpo es siempre $\mathbb{F}_{q^2}$. De este modo es fácil hallar el objetivo de la transferencia a otro cuerpo.

Por otro lado, se tiene al algoritmo ρ **de Pollard** que tiene un tiempo superior al del ataque MOV. En contraste con el algoritmo MOV, este ataque ha sido probado utilizando curva elípticas arbitrarias, por lo que es comprensible que su tiempo supere al medido en el ataque MOV. Una razón importante por la cual el algoritmo ρ de Pollard tiene un tiempo de ejecución tan bajo es debido al hecho de que se han usado curvas elípticas de orden primo. Esto causa que el paso final del algoritmo se vuelva mucho más simple, pues es solamente hallar el inverso de un número en un módulo y multiplicarlo por otro.

Los tiempos de ejecución se encuentran registrados en el **Apéndice 2**. De la observación de los tiempos se vuelve clara una cuestión y es que los tiempos aumentan conforme aumenta el orden de la curva. Sin embargo, obsérvese que puede haber unas discordancia con esta afirmación porque para algunas curvas de orden bajo el algoritmo tarda más que con otras de orden superior. Las razones de este comportamiento son varias, ya que los puntos se escogen aleatoriamente puede haber algún retraso o adelanto con el cálculo, puede ser la primera ejecución sobre la nueva curva elíptica ó algún posible fallo con el *Notebook*.

6.3. Conclusiones

En el apartado anterior se han analizado los resultados de las pruebas utilizando las gráficas generadas por los valores resultantes de las pruebas. Como conclusión se puede constatar el algoritmo MOV utilizado contra curvas supersingulares es el que tarda menos de entre todos los ataques. Sin embargo, no se le puede llamar el **mejor algoritmo**, debido a la particularidad de sus objetivos: las curvas supersingulares. Por contra, los algoritmos como el Baby Step, Giant Step; Pohlig-Hellman y ρ de Pollard atacan a cualquier tipo de curva de la misma manera. Por lo que son, al menos, más versátiles que el ataque MOV.

Dejando al algoritmo MOV de lado, respecto a qué algoritmo de entre los restantes es mejor, la respuesta es clara. El algoritmo ρ de Pollard tiene los mejores tiempos de entre los tres, presentando resultados del orden del 50 milisegundos; mientras que los otros presentan tiempos del orden de los 5 y 10 segundos.

Observación 6.3.1. Para más información sobre métodos de ataque sobre curvas elípticas, y en general, seguridad de las mismas, véase [saf]. En esta página web se encuentran varios análisis de distintas curvas elípticas publicadas como seguras por organismos internacionales; así como, explicaciones sobre diversos tipos de ataques efectivos contra este tipo de encriptado.

Trabajo futuro

Otros métodos

El ataque MOV sólo ha sido implementado para atacar a curvas supersingulares debido a la dificultad de encontrar el cuerpo finito donde se encuentran todos los puntos de N-torsión, $E[N]$. Por eso, encontrar una solución a este problema debería ser un objetivo importante para trabajos futuros. Una vez encontrada una solución se podría aplicar este ataque a todo tipo de curvas elípticas.

Sería interesante implementar el método *Index calculus* adaptado para las curvas elípticas basado en polinomios sumatorios. Según [MM17], se ha comprobado que este algoritmo es mejor que el algoritmo ρ de Pollard, en cuanto a velocidad y rendimiento. La implementación del código desarrollado en este proyecto ha sido realizada en SageMath 8.7, así que se podría observar un incremento de rendimiento en estos algoritmos si se desarrolla en otro lenguaje.

En este proyecto se ha implementado y probado el algoritmo ρ de Pollard, pero no el algoritmo λ de Pollard. Este último algoritmo utiliza procesamiento en paralelo entre varias terminales para acelerar el proceso de hallar una coincidencia entre dos puntos que tienen una característica determinada común. En un experimento futuro sería interesante ver las particularidades que podrían surgir de su implementación, dependiendo del lenguaje utilizado y el entorno en el que se da tal implementación. Por ejemplo, este algoritmo requiere de más de un equipo, así que calibrar la calidad necesaria de los equipos para hallar la combinación de sistemas que signifique una mejora de rendimiento significativa con respecto a la ejecución de un equipo en solitario del algoritmo ρ, también es un objetivo importante.

A priori, podría resultar difícil hacer una comparativa entre estos dos algoritmos debido a la diferencia de esfuerzo requerido en cada situación, la diferencia de calidad y número de equipos, la cantidad de recursos necesarios para cada uno ... Aún con todo, realizando un análisis preciso y cuidadoso en condiciones similares para cada uno de los métodos, sería posible dilucidar conclusiones importantes sobre la eficacia de cada uno de estos algoritmos.

Finalmente, en este proyecto sólo se ha estudiado un ataque por emparejamiento: **el ataque MOV**. Sin embargo, existe otro ataque por emparejamiento importante, el **ataque de Frey-Rück**. Este ataque utiliza el **emparejamiento de Tate-Lichtenbaum** para resolver el problema del logaritmo discreto, en vez de utilizar el emparejamiento de Weil. Para más información sobre este emparejamiento véase [FMR98] y [FR91]. Para más información sobre el método en sí, léase el capítulo 5 de [Was08].

Otras curvas elípticas

Los métodos implementados han sido probados utilizando curvas elípticas de orden primo. Esto ha causado que el método de Pohlig-Hellman disminuya su rendimiento debido a que su algoritmo 5.1.3 utiliza la factorización en factores primos del orden de un elemento para hallar el logaritmo discreto

en módulos más pequeños, para luego fusionar los resultados con el Teorema Chino del Resto. Por ello, en el futuro sería interesante realizar pruebas con curvas cuyo orden sea compuesto, para probar la eficacia de este algoritmo.

La mayoría de los métodos presentados en este proyecto sólo se han utilizado sobre curvas elípticas arbitrarias, sin ningún tipo de propiedad especial como con el caso de las anómalas o las supersingulares. No obstante, existen de más tipos como las curvas hiperelípticas; por tanto, se podría realizar un estudio sobre cómo estos ataques se aplicarían a otros tipos de curvas elípticas con diferentes propiedades.

Otro tipo de ataques

Los métodos implementados en este proyecto fueron probados en un ordenador clásico; sin embargo, se podría pensar en utilizar nuevas tecnologías que se están popularizando para acelerar la ejecución de estos algoritmos. Una de estas tecnologías es la **computación cuántica**. Para probar la efectividad de la computación cuántica se ha propuesto el problema del logaritmo discreto para poder comparar la complejidad de este problema en la computación clásica y en la computación cuántica. Con un leve cambio de perspectiva este objetivo se puede transformar en resolver el problema del logaritmo discreto en curvas elípticas, meta que interesaría para futuras investigaciones.

Los métodos de ataque estudiados atacan las propiedades matemáticas de las curvas elípticas definidas en un cuerpo finito. Estos ataques, a su vez, utilizan distintos teoremas y herramientas matemáticas para aprovechar las propiedades ofrecidas por las curvas. Sin embargo, existen ataques que no utilizan estas propiedades matemáticas para atacar al problema del logaritmo discreto. Estos ataques se pueden basar en diferentes datos de los criptogramas para extraer información acerca de cómo fue encriptado. Por ejemplo, se puede utilizar el tiempo que se tarda en cifrar un mensaje para averiguar detalles del algoritmo de encriptación, estos son los llamados **ataques de timing**.

Un tipo especial entre estos ataques está empezando a popularizarse últimamente, los ataques de este tipo se llaman **ataques TEMPEST**. Este tipo de ataques se basan en el análisis de emisiones electromagnéticas no intencionadas que produce el hardware. Este análisis puede revelar información sobre los datos privados de un usuario. De hecho, existe una regulación para evitar este tipo de ataques: ISO 27002, concretamente en el apartado 11.2.1, [iso13]. Por tanto, se podría realizar un estudio con el objetivo de medir cuán efectivo es un ataque TEMPEST frente a la criptografía sobre una curva elíptica.

En [GPP06], se lleva a cabo un estudio sobre la implementación hardware multiprocedural del algoritmo ρ de Pollard. Es un hecho conocido que las operaciones en hardware son mucho más rápidas que en software, y este trabajo prueba con varias arquitecturas hardware que las operaciones se realizan más rápido en hardware que en software. Por tanto, sería interesante analizar hasta qué punto es más rentable intentar resolver el problema del logaritmo discreto por software que por hardware.

Bibliografía

[saf] *Choosing safe curves for elliptic-curve cryptography, https://safecurves.cr.yp.to/.*

[pyd] *https://docs.python.org/2/.*

[sag] *Sagemath, http://www.sagemath.org/.*

[iee00] *Ieee standard specifications for public-key cryptography. IEEE Std 1363-2000*, pp. 1–228, 2000.

[x9-01] *American national standard for financial services: X9.63-2001: public key cryptography for the financial services industry: key agreement and key transport using elliptic curve cryptography.* American Bankers Association, 2001.

[x9-05] *Public key cryptography for the financial services industry - the Elliptic Curve Digital Signature Algorithm (ECDSA): ANSI American national standard for financial services, ANS X9.62-2005.* Accredited Standards Committee X9, Inc., 2005.

[iso13] *Iso/iec 27002:2013, information technology – security techniques – code of practice for information security controls.* Standard, ISOIEC, 2013.

[BSS06] Blake, I. F., Seroussi, G. and Smart, N. P. *Elliptic curves in cryptography*, vol. 265. Cambridge University Press, 2006.

[Bon98] Boneh, D. *Twenty years of attacks on the rsa cryptosystem.* 1998.

[DH76] Diffie, W. and Hellman, M. *New directions in cryptography. IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[DH07] Dorronsoro, J. and Hernandez, E. *Numeros, grupos y anillos.* Addison-Wesley Iberoamericana, 2007.

[Eng01] Enge, A. *Elliptic curves and their applications to cryptography: an introduction.* Kluwer Academic Publishers, 2001.

[FMR98] Frey, G., Müller, M. and Rück, H.-G. *The tate pairing and the discrete logarithm applied to elliptic curve cryptosystems.* Universität Essen. Institut für Experimentelle Mathematik, 1998.

[FR91] Frey, G. and Rück, H.-G. *A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves.* Universität Essen. Institut für Experimentelle Mathematik, 1991.

[GPP06] Güneysu, T., Paar, C. and Pelzl, J. *On the security of elliptic curve cryptosystems against attacks with special-purpose hardware.* 2006.

[JL02] Joux, A. and Lercier, R. *Improvements to the general number field sieve for discrete logarithms in prime fields. a comparison with the gaussian integer method. Mathematics of Computation*, 72(242):953–968, 2002.

[Kob87] Koblitz, N. *Elliptic curve cryptosystems. Mathematics of Computation*, 48(177):203, 1987.

[Kob94] Koblitz, N. *A course in number theory and cryptography.* Springer, 1994.

[KM04] Koblitz, N. and Menezes, A. J. *Algebraic aspects of cryptography.* Springer, 2004.

[Kob12] Koblitz, N. I. *Introduction to elliptic curves and modular forms.* Springer-verlag New York Inc., 2012.

[MM17] McGuire, G. and Mueller, D. *A new index calculus algorithm for the elliptic curve discrete logarithm problem and summation polynomial evaluation.* p. 21, 2017.

[MOV93] Menezes, A. J., Okamoto, T. and Vanstone, S. A. *Reducing elliptic curve logarithms to logarithms in a finite field. IEEE Transactions on Information Theory*, 39(5):1639–1646, 1993.

[MOV18] Menezes, A. J., Oorschot, P. C. V. and Vanstone, S. A. *Handbook of applied cryptography.* 2018.

[Mil86] Miller, V. S. *Use of elliptic curves in cryptography.* In Williams, H. C. (Editor), *Advances in Cryptology — CRYPTO '85 Proceedings*, pp. 417–426. Springer Berlin Heidelberg, Berlin, Heidelberg, 1986.

[Nav16] Navarro, G. *Un curso de álgebra (2a ed.).* Publicaciones de la Universidad de Valencia (PUV, second edn., 2016.

[Pol78] Pollard, J. M. *Monte carlo methods for index computation (mod p). Mathematics of Computation*, 32(143):918, 1978.

[RAS77] Rivest, R., Adleman, L. and Shamir, A. *A method for obtaining digital signatures and public-key cryptosystems.* Laboratory for Computer Science, Massachusetts Inst. of Technology, 1977.

[SA98] Satoh, T. and Araki, K. *Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves. Commentarii Math. Univ. St. Pauli.*, 47:81–92, 1998.

[Sch85] Schoof, R. *Elliptic curves over finite fields and the computation of square roots mod p. Mathematics of Computation*, 44(170):483–494, 1985.

[Sem98] Semaev, I. A. *Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p. Mathematics of Computation of the American Mathematical Society*, 67(221):353–356, 1998.

[Sha71] Shanks, D. *Class number, a theory of factorization, and genera. Proceedings of Symposia in Pure Mathematics 1969 Number Theory Institute*, p. 415–440, 1971.

[SS15] Singh, A. and Singh, R. G. *Various attacks over the elliptic curve-based cryptosystems. International Journal of Engineering and Innovative Technology*, 5:1–3, 2015.

[Sma99] Smart, N. P. *The discrete logarithm problem on elliptic curves of trace one. J. Cryptol.*, 12(3):193–196, 1999.

[Was97] Washington, L. C. *Cyclotomic fields of class number one. Graduate Texts in Mathematics Introduction to Cyclotomic Fields*, p. 205–231, 1997.

[Was08] Washington, L. C. *Elliptic curves: number theory and cryptography.* Chapman and Hall/CRC, second edn., 2008.