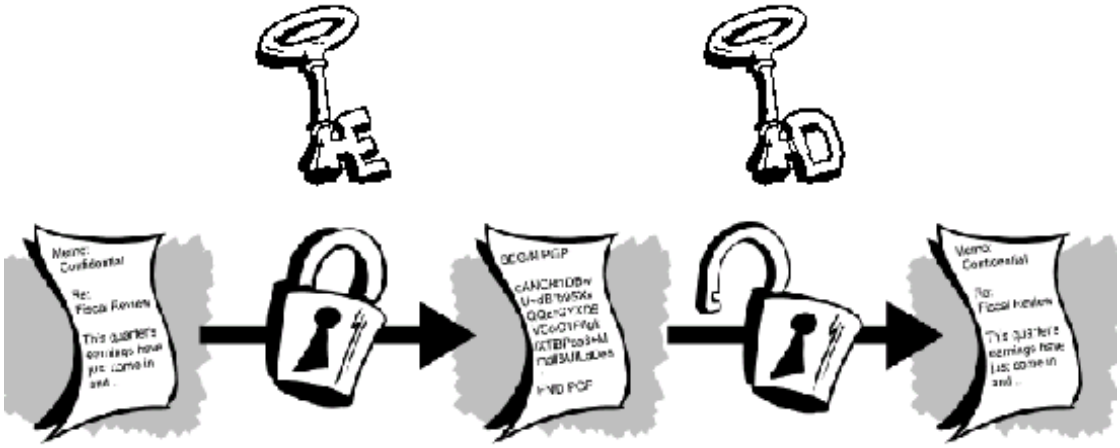
C A P Í T U L O IV

CRIPTOGRAFÍA DE CLAVE PÚBLICA



4.1 Definición

4.2 Algoritmo RSA *(Rivest, Shamir, Adleman)*

4.3 Algoritmo El Gamal

4.4 Criptosistemas basados en Curvas Elípticas

4.5 Criptosistemas basados en Logaritmos Discretos

4.1 DEFINICIÓN

Los algoritmos de llave pública o algoritmos asimétricos, han demostrado su interés para ser empleados en redes de comunicación inseguras. Introducidos por Whitfield Diffie y Martin Hellman en 1976 . Su novedad fundamental con respecto a la criptografía simétrica es que las claves no son únicas, sino que forman pares de ellas, una clave de encriptado y una clave diferente, pero relacionada para desencriptado. Utilizar dos claves tiene consecuencias profundas en las áreas de confidencialidad, distribución de claves y autentificación [LIB 002].

Se basan en general en plantear al atacante problemas matemáticos difíciles de resolver, tales como el problema de la factorización entera, el problema del logaritmo discreto. En la práctica muy pocos algoritmos son realmente útiles, el más popular por su sencillez es RSA, que ha sobrevivido a multitud de ataques, si bien necesita una longitud de clave considerable.

Los algoritmos asimétricos emplean generalmente longitudes de clave mucho mayores que los simétricos. Por ejemplo, mientras que para algoritmos simétricos se considera segura una clave de 128 bits, para algoritmos asimétricos se recomiendan claves de al menos 1024 bits. Además, la complejidad de cálculo que comportan estos últimos los hace considerablemente más lentos que los algoritmos de cifrado por bloques. En la práctica los métodos asimétricos se emplean únicamente para codificar la clave de sesión[17] de cada mensaje.

[17] Una clave simétrica seleccionada aleatoriamente para cada conexión.

En la *figura 4.1* se muestra el proceso de cifrado/descifrado a seguir en un criptosistema asimétrico.

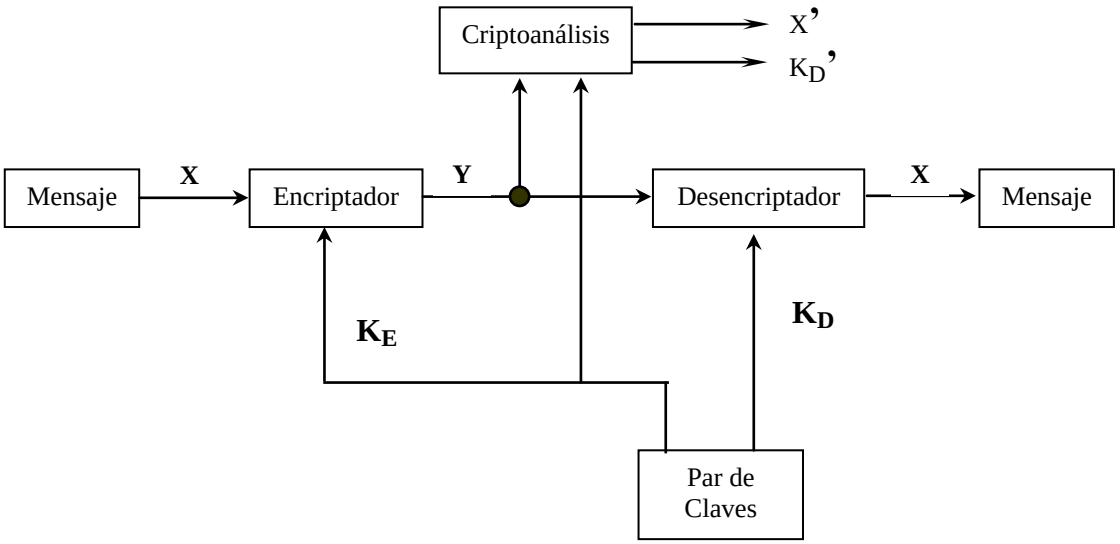


Figura 4.1 Criptosistema de clave pública [LIB 002]

Dado un mensaje X y la clave pública K_E como entrada, se genera un texto cifrado Y por medio de una función de encriptado E.

$$Y = E_{k_E}(X)$$

El receptor en posesión de la clave privada K_D, es capaz de invertir la transformación y obtener de nuevo el mensaje original por medio de una función de desencriptado D.

$$X = D_{k_D}(Y)$$

Un oponente, teniendo acceso al mensaje cifrado $\boldsymbol{Y}$ y a la clave pública $\boldsymbol{K_E}$ debe intentar recuperar $\boldsymbol{X}$ y/o $\boldsymbol{K_D}$ empleando alguna técnica de criptoanálisis.

4.2 ALGORITMO RSA *(Rivest, Shamir, Adleman)*

Es un criptosistema de clave pública, desarrollado en 1977 por Ron Rivest, Adi Shamir y Leonard Adleman en el MIT [18]. De entre todos los algoritmos asimétricos, quizá RSA sea el más sencillo de comprender e implementar. Desde su nacimiento nadie ha conseguido violar su seguridad, por lo que se le considera como uno de los algoritmos asimétricos más seguros. RSA se basa en la dificultad para factorizar números grandes. Las claves pública y privada se calculan a partir de un número que se obtiene como producto de dos primos grandes. El atacante se enfrentará, si quiere recuperar un texto plano a partir del criptograma y la llave pública, al problema de factorización entera.

La generación de las claves del algoritmo RSA se resume en los siguientes pasos:

1. Seleccionar aleatoriamente dos números primos grandes, por ejemplo mayores que 10^{100} cada uno. Llamémoslos ***p*** y ***q***.
2. Calcular $n = p * q$
3. Calcular $z = (p - 1) * (q - 1)$
4. Seleccionar un número relativamente primo a ***z***, llamémoslo ***d.*** Esto es, que el máximo común divisor de ***z*** y ***d*** debe ser la unidad. Además ***d*** debe ser menor que ***z***, esto es, $1 < d < z$.
5. Calcular ***e*** tal que $e * d \equiv 1 \pmod{z}$.

[18] Massachusetts Institute of Technology (Instituto Tecnológico de Massachusetts)

La clave pública K_E consiste en el par ***{e, n}*** y la clave privada K_D consiste en el par ***{d, n}***. Con estos parámetros calculados, estamos listos para comenzar el proceso de cifrado/descifrado.

Para cifrar el mensaje ***M***, calculamos $C = M^e \pmod n$. Para descifrar ***C***, calculamos $M = C^d \pmod n$.

Un ejemplo sencillo del algoritmo RSA se muestra a continuación.

1. Seleccionamos $p = 11$ y $q = 3$.
2. Calculamos $n = p * q \rightarrow n = 11 * 3 \rightarrow n = 33$
3. Calculamos $z = (p - 1) * (q - 1)$
   $z = (11 - 1) * (3 - 1) \rightarrow z = 10 * 2 \rightarrow z = 20$
4. Seleccionamos ***d*** tal que mcd(z, d)=1 y 1<d<z. Esto es mcd(20, d)=1 y 1<d<20. Podemos seleccionar $d = 3$.
5. Calculamos ***e*** tal que: $e * d \equiv 1 \pmod z \rightarrow 3 * e \equiv 1 \pmod{20}$. Resolviendo la congruencia lineal obtenemos que $e = 7$

La clave pública viene dada por $K_E = \{e, n\} = \{7, 33\}$ y la clave privada por $K_D = \{d, n\} = \{3, 33\}$.

Para cifrar el mensaje ***M = 2***, calculamos:

$C = M^e \pmod n \rightarrow C = 2^7 \pmod{33} \rightarrow C = 128 \pmod{33} \rightarrow C = 29$

Para descifrar ***C***, calculamos:

$M = C^d \pmod n \rightarrow M = 29^3 \pmod{33} \rightarrow M = 24389 \pmod{33} \rightarrow M = 2$

Para cifrar el texto plano "*ataquen",* calculamos:

Mensaje (M)			Texto cifrado (C)	
simbólico	*numérico*	M^7	M^7 *(mod 33)*	*simbólico*
a	01	01	01	a
t	20	1280000000	26	z
a	01	01	01	a
q	17	410338673	8	h
u	21	1801088541	21	u
e	5	78125	14	n
n	14	105413504	20	t

Para descifrar el texto cifrado *"azahunt"*, calculamos:

Texto cifrado (C)			Mensaje (M)	
simbólico	*numérico*	C^3	C^3 *(mod 33)*	*simbólico*
a	01	01	01	a
z	26	17576	20	t
a	01	01	01	a
h	8	512	17	q
u	21	9261	21	u
n	14	2744	5	e
t	20	8000	14	n

Como puede apreciarse, las funciones de cifrado y descifrado son totalmente inversas. La mayoría de las discusiones sobre el criptoanálisis se han centrado en la tarea de factorizar ***n*** en sus dos números primos. Si el criptoanalista pudiera factorizar ***n***, que es conocido públicamente, podría encontrar ***p*** y ***q***, y a partir de estos calcular ***z***. Equipado con el conocimiento de ***z*** y de ***e***, puede encontrar ***d*** usando el algoritmo de Euclides. Afortunadamente esto

representa un problema computacionalmente intratable, siempre que ***p*** y ***q*** (y por lo tanto ***n***) sean lo suficientemente grandes.

Otro tipo de ataque a este criptosistema es el conocido ***Ataque de Módulo Común.*** Podrá pensarse que, una vez generados los números primos ***p*** y ***q***, será más rápido generar tantos pares de llaves como queramos, en lugar de tener que emplear dos números primos diferentes en cada caso. Sin embargo, si lo hacemos así, un atacante podría decodificar nuestros mensajes sin necesidad de la llave privada. Sea ***M*** el texto plano, que codificamos empleando dos claves de cifrado diferentes e_1 y e_2. Los criptogramas que obtenemos son los siguientes:

$$C_1 = M^{e1} \pmod{n}$$

$$C_2 = M^{e2} \pmod{n}$$

El atacante conoce pues ***n***, e_1, e_2, c_1 y c_2. Si e_1 y e_2 son primos relativos, el Algoritmo Extendido de Euclides nos permitirá encontrar ***r*** y ***s*** tales que:

$$re_1 + se_2 = 1$$

Ahora podemos hacer el siguiente cálculo:

$$C_1^{\,r}\ C_2^{\,s} = M^{\,r\,.\,e1}\ M^{\,s\,.\,e2} = M^{\,r\,.\,e1\,+\,s\,.\,e2} = M^{\,1} \pmod{n}$$

Recordemos que esto sólo se cumple si e_1 y e_2 son números primos relativos, pero precisamente eso es lo que suele ocurrir en la gran mayoría de los casos. Por lo tanto, se deben generar ***p*** y ***q*** diferentes para cada par de claves.

4.3 ALGORITMO EL GAMAL

El Gamal es un algoritmo de clave pública que basa su seguridad en la dificultad de calcular logaritmos discretos, que está íntimamente relacionado con el de la factorización de números enteros. Es una implementación

concreta de un algoritmo criptográfico más general, Diffie-Hellman (DH). Su patente, por tanto, es la misma que la de DH, que expiró el 29 de abril de 1997, por lo que El Gamal es el primer algoritmo de clave pública "liberado" de la reclamación de patentes en los Estados Unidos.

Para generar un par de llaves, se escoge un número primo ***p*** y dos números aleatorios ***g*** y ***x*** menores que ***p***. Se calcula entonces:

$$y \equiv g^{x} \pmod{p}$$

La llave pública es ***{g; y; p}***, mientras que la llave privada es ***x***.

Un grupo de usuarios puede utilizar la misma ***g*** y la misma ***p*** sin que la seguridad se vea afectada.

La encriptación tiene lugar así: para encriptar el mensaje ***M***, en primer lugar se toma un número aleatorio ***k***, que sea primo relativo de ***p-1*** (es decir, que el máximo común divisor de ***k*** y ***p-1*** sea 1). Entonces, se calculan:

$$a \equiv g^{k} \pmod{p}$$

$$b \equiv M\, y^{k} \pmod{p}$$

En estas condiciones, el texto cifrado ***C*** está constituido por la pareja de valores enteros ***(a, b)***.

La recuperación del mensaje en claro ***M*** a partir del cifrado ***C = (a, b)*** se lleva a cabo mediante el cálculo de la congruencia:

$$M \equiv (b/a^{x}) \bmod p$$

Una de las características más destacables de este procedimiento de cifrado de clave pública es que los cifrados de un mismo mensaje en claro pueden ser diferentes sin mas que computarlos a partir de valores enteros aleatorios ***k*** diferentes. Sin embargo, el sistema de cifrado de *El Gamal* tiene a su vez

algunos inconvenientes, siendo uno de los más serios el hecho de que el cifrado es de longitud doble que el mensaje en claro, lo cual supone un aumento considerable en las necesidades de almacenamiento. Es quizás por esto que su difusión no ha sido igual que la de otros métodos, aunque matemáticamente es igual de seguro que RSA.

Después de todo lo anterior, es fácil darse cuenta de que la ruptura de este sistema de cifrado de clave pública es equivalente a la resolución del problema del logaritmo discreto (mod *p*). En particular, el cálculo de la clave secreta de descifrado ***x*** a partir de la clave pública de cifrado y requiere el cómputo del logaritmo discreto dado por:

$$x \equiv \log_g y \pmod{p}$$

Un ejemplo sencillo del algoritmo *El Gamal* se muestra a continuación.
Sea $\boldsymbol{p} = 17$, $\boldsymbol{g} = 4$ y la clave privada $\boldsymbol{x} = 3$.
Calculamos: $\boldsymbol{y} \equiv g^x \pmod{p} \rightarrow \boldsymbol{y} \equiv 4^3 \pmod{17} \rightarrow \boldsymbol{y} = 13$
En este caso, la clave pública ***{g, y, p}*** viene dada por {4, 13, 17}.

Supongamos ahora el mensaje $\boldsymbol{M} = 2$. Elegimos un valor entero $\boldsymbol{k} = 5$, relativamente primo con $\mathbf{p - 1} = 16$. En estas condiciones, el cifrado ***C*** del mensaje ***M*** está constituido por la pareja de valores enteros ***C* = (a, *b*)** dados por las congruencias:

$$\boldsymbol{a} \equiv g^k \pmod{p} \rightarrow \boldsymbol{a} \equiv 4^5 \pmod{17} \rightarrow \boldsymbol{a} = 4$$
$$\boldsymbol{b} \equiv M\, y^k \pmod{p} \rightarrow \boldsymbol{b} \equiv 2 \cdot 13^5 \pmod{17} \rightarrow \boldsymbol{b} = 9$$

Para recuperar el mensaje original ***M*** a partir del cifrado $\boldsymbol{C} = (4, 9)$ calculamos la congruencia lineal:

$$M \equiv (b/a^x) \bmod p$$
$$M \equiv (9/4^3) \bmod 17$$

4^3 M ≡ 9 (mod 17)

M = 2

Si un oponente quisiera obtener ***x*** a partir de la clave pública, debería intentar resolver la congruencia lineal ***x ≡ log $_4$ 13 (mod 17)*** lo que implica un problema intratable cuando se trabaja con números grandes.

4.4 CRIPTOSISTEMAS BASADOS EN CURVAS ELÍPTICAS (CCE)

Otro tipo de criptografía de clave pública es el que usa curvas elípticas definidas en un campo finito. Propuesto en 1985 por Neal Koblitz y Victor Miller [WWW 004]. La diferencia que existe entre este sistema y **RSA** es el problema en el cual basan su seguridad, mientras **RSA** razona de la siguiente manera: se da el número 15 y se reta a encontrar los factores primos, es decir, el 5 y 3. El problema del cual están basados los sistemas que usan curvas elípticas es el problema del logaritmo discreto elíptico (PLDE), en este caso su razonamiento con *números simples* sería algo como: se da el número 15 y el 3 y se reta a encontrar cuantas veces tienes que sumar el mismo 3 para obtener 15. Cabe señalar que los CCE no trabajan sobre *números simples* sino sobre puntos racionales de una curva. En este caso el problema es encontrar cuantas veces hay que sumar un punto racional para obtener otro conocido. Dado los puntos ***P*** y ***Q*** encontrar ***x***, tal que ***xP = Q***.

En lo que sigue nos dedicaremos a explicar lo más importante de los **CCE.**

- ✓ Entenderemos como una *curva elíptica* a una ecuación de dos variables de grado 3, es decir la máxima potencia de las variables debe ser 3. Su forma general es la siguiente:

$$y^2 + axy + by = x^3 + cx^2 + dx + e \qquad \text{(Ecuación 4.1)}$$

Donde las constantes ***a, b, c, d*** y ***e*** pertenecen a cierto conjunto llamado campo ***F,*** que para propósitos de la criptografía, es un campo primo (Z_p) o un campo de característica 2 (F_2^n), o sea conjuntos de ceros y unos de longitud n.

- ✓ A un punto que satisface la ecuación anterior se le llama *punto racional*. Si el campo es finito, entonces el conjunto de puntos (*x, y*) que satisfacen la ecuación es finito y es llamado conjunto de puntos racionales de la curva ***E*** sobre el campo **F**. Al conjunto de puntos racionales lo podemos representar como: E: $O, P_1, P_2, P_3, ..., P_n$

E representa la ecuación y ***O*** es un punto que no tiene coordenadas y hace el papel de cero *(llamado punto al infinito)* ya que en este conjunto los puntos puede sumarse y tiene las mismas propiedades que la suma de los números enteros, es decir lo que se conoce como un grupo abeliano.
Ejemplo: dada la curva elíptica $y^2 = x^3 + 4x + 3$ y el campo $\mathbf{Z}_5$, es decir el conjunto {0,1,2,3,4}, entonces las parejas que satisfacen la ecuación son {(2,2), (2,3)}, por lo tanto la curva elíptica es ***E***: {***O***, (2,2), (2,3)}. En este caso ***E*** tiene 3 puntos racionales.

- ✓ La suma de estos puntos tiene una explicación geométrica muy simple, como se muestra en la *figura 4.2*. Tome dos puntos ***P*** y ***Q*** de la curva, y trace la línea que pasa por ambos puntos. En el caso general esta línea siempre tiene un punto de intersección con la curva. Ahora tome este tercer punto en la intersección de la línea con la curva y trace una línea vertical. El otro punto ***R*** de intersección con la curva de esta línea vertical se define como la suma de ***P*** y ***Q***, es decir, $R = P + Q$.

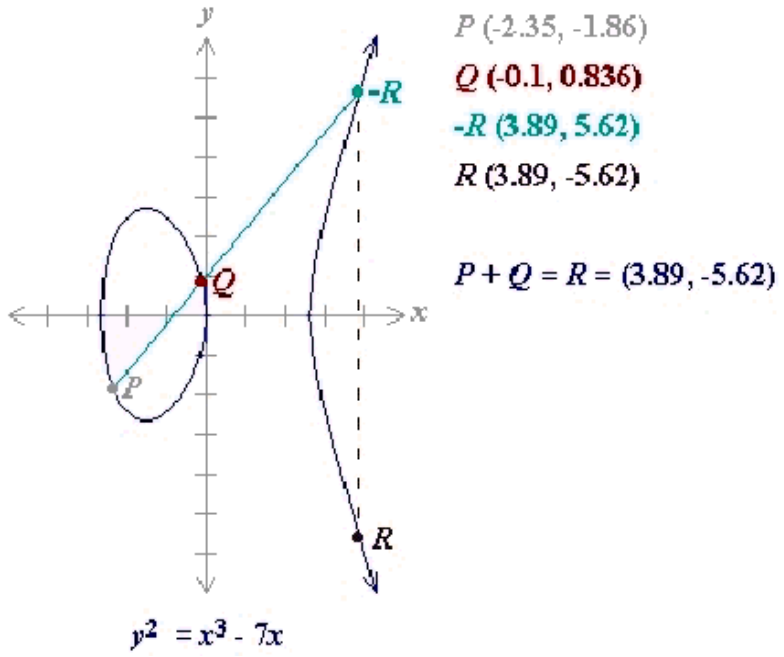


Figura 4.2 Suma de P y Q [WWW 005]

El punto opuesto a $\boldsymbol{R} = (x, y)$ es el punto $\boldsymbol{-R} = (x, -y)$.

Si $P_1 = P_2$, es decir, que queremos obtener R = P + P *(vea la Figura 4.3)*, entonces la línea a ser construida en el primer paso es la tangente a la curva, el cual otra vez tiene otro punto de intersección con la curva.

Una parte compleja es definir el elemento identidad del grupo. La pregunta a responder es: ¿Qué punto de la curva sumado a un punto ***P*** cualquiera resulta en el punto ***P***? Solamente podemos encontrar una respuesta a esta pregunta si un punto extra es agregado a la curva. Este punto extra se llama *punto en el infinito*, y se lo designa con ***O***. El punto ***O*** está en un lugar infinitamente lejos sobre el eje vertical, y es la identidad del grupo de la curva elíptica.

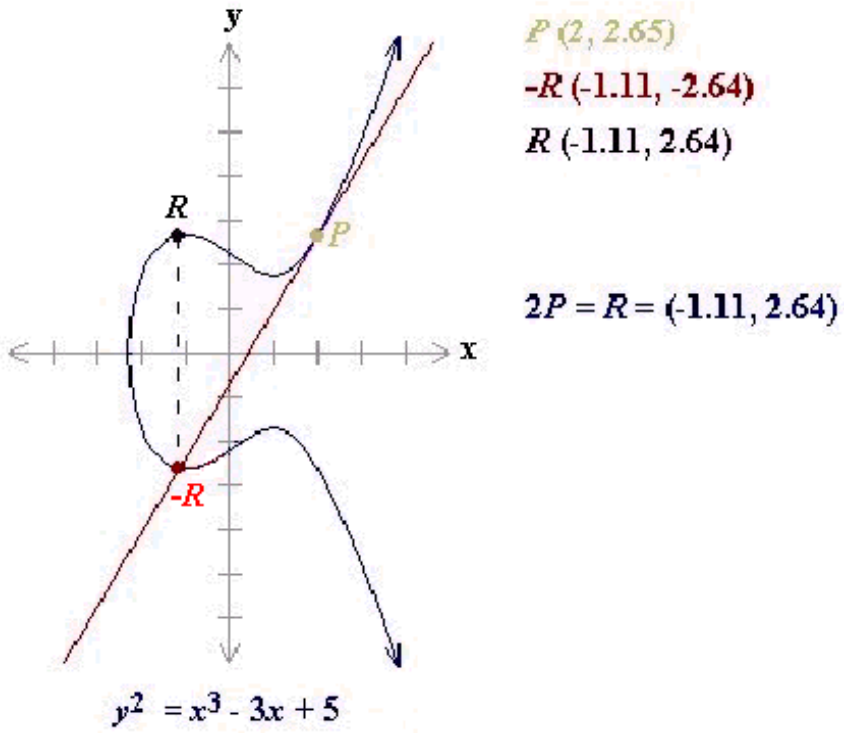


Figura 4.3 Suma del mismo punto P [WWW 005]

Por ejemplo, observe el punto P en la *figura 4.4*. Dada la definición del elemento identidad *(o elemento nulo)*, P + (-P) = ***O***.

- ✓ No se considera práctico utilizar las curvas elípticas en su forma general *(Ec. 4.1)*, ni tampoco utilizarla en el campo de los **R** *(reales)*, debido a los errores de redondeo y truncación de los valores. Por eso, se considerarán las curvas elípticas definidas sobre $\boldsymbol{Z_p}$ donde $\boldsymbol{p}$ es un número primo impar, ya que éstos conjuntos producen las implementaciones más eficientes de la aritmética de curva elíptica [WWW 008].

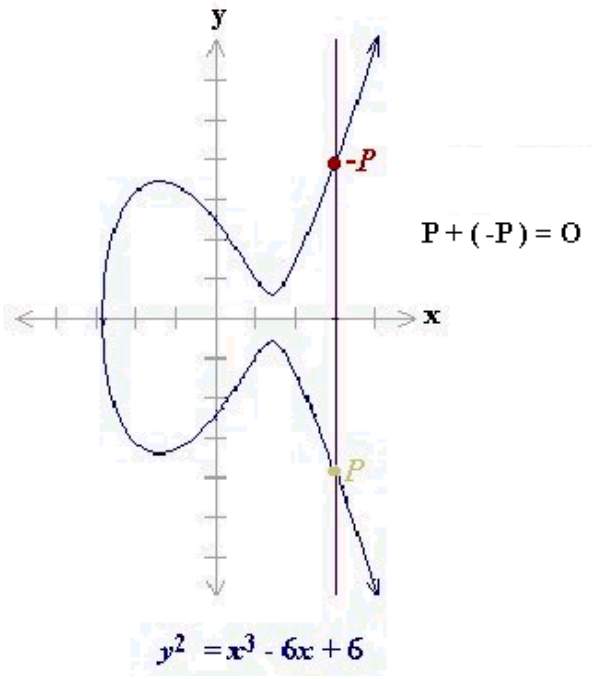


Figura 4.4 *Suma del punto P con su opuesto* [WWW 005]

Una curva elíptica ***E*** sobre Z_p, denotada por $E(Z_p)$ es definida por la ecuación de la forma:

$$y^2 = x^3 + ax + b \pmod p \qquad \text{(Ecuación 4.2)}$$

donde: $a, b \in Z_p$ y $4a^3 + 27b^2 \neq 0 \pmod p$, junto con el punto en el infinito ***O***. El conjunto $E(Z_p)$ consiste de todos los puntos (x, y) tal que $x \in Z_p$; $y \in Z_p$, los cuales satisfacen la ecuación.

Por ejemplo, si consideramos la curva elíptica ***E***: $y^2 = x^3 + x + 5$ definida sobre Z_{17} donde las constantes usadas son a = 1; b = 5.

Note que $4a^3 + 27b^2 = 4*1^3 + 27*5^2 = 4 + 12 = 16 \neq 0 \pmod{17}$, de manera que en realidad es una curva elíptica.

- ✓ La operación de suma de dos puntos de la curva se define así [WWW 011]: Sea $\boldsymbol{P_1} = (x_1, y_1) \in E(Z_p)$ y $\boldsymbol{P2} = (x_2, y_2) \in E(Z_p)$, donde $x_1 \neq x2$. Entonces $P_1 + P_2 = (x_3, y_3)$ es:

$$x_3 \equiv \lambda^2 - x_1 - x_2 \pmod{p}$$
$$y_3 \equiv \lambda(x_1 - x_3) - y_1 \pmod{p}$$
$$\lambda \equiv \frac{y_2 - y_1}{x_2 - x_1} \pmod{p}$$

Sea $\boldsymbol{P_1} = (x_1, y_1) \in E(Z_p)$ con $y_1 \neq 0$. Entonces $P_1 + P_1 = (x_3, y_3)$, donde:

$$x_3 \equiv \lambda^2 - 2x_1 \pmod{p}$$
$$y_3 \equiv \lambda(x_1 - x_3) - y_1 \pmod{p}$$
$$\lambda \equiv \frac{3x_1^2 - a}{2y_1} \pmod{p}$$

Cabe señalar que la constante "***a***" es el coeficiente de la ecuación 4.2

Ejemplo: Considere la curva elíptica $\boldsymbol{E}$: $y^2 = x^3 + x + 1$ definida sobre $\boldsymbol{Z_{23}}$ y los puntos $\boldsymbol{P} = (3, 10)$ y $\boldsymbol{Q} = (9, 7)$ sobre la curva. Entonces $P + Q = (x_3, y_3)$ se calcula así:

$$\lambda = \frac{7-10}{9-3} = \frac{-3}{6} = \frac{-1}{2} \equiv 11 \pmod{23}$$
$$x_3 = 11^2 - 3 - 9 = 6 - 3 - 9 = -6 \equiv 17 \pmod{23}$$
$$y_3 = 11(3-(-6)) - 10 = 11(9) - 10 = 89 \equiv 20 \pmod{23}$$

Por lo tanto, $P + Q = (17, 20)$.

Ejemplo: Sea $\boldsymbol{P} = (3, 10)$. Entonces $2P = P + P = (x_3, y_3)$ se calcula así:

$$\lambda = \frac{3(3^2)+1}{2(10)} = \frac{4+1}{20} = \frac{5}{20} = \frac{1}{4} \equiv 6 \pmod{23}$$
$$x_3 = 6^2 - 2(3) = 13 - 6 = 7 \pmod{23}$$
$$y_3 = 6(3-7) - 10 = -24 - 10 = 22 - 10 = 12 \pmod{23}$$

Por lo tanto, 2P = (7, 12).

- ✓ *El orden de una curva elíptica* ***E*** definida sobre el campo $\boldsymbol{Z_q}$ es el número de puntos sobre la curva elíptica ***E***, incluyendo ***O***. Esto es denotado por $\boldsymbol{\#E(Z_q)}$. En un sistema criptográfico con curvas elípticas es muy importante el número de puntos racionales, ya que este número debe contener como factor a un número primo de al menos 163 bits para considerar que la curva sea segura en criptografía.

- ✓ *El orden de un punto* ***P*** es el entero positivo más pequeño ***n*** tal que ***nP = O*** (el punto en el infinito).

- ✓ Sea $\boldsymbol{E(Z_p)}$ una curva elíptica, con ***p*** primo impar, entonces:

$$p+1-2\sqrt{p} \leq \#E(Z_p) \leq p+1+2\sqrt{p}$$

- ✓ Los **CCE** basan su seguridad en el Problema del Logaritmo Discreto Elíptico (**PLDE**), es decir, en el Problema del Logaritmo Discreto **(PL**D) definido en el grupo de puntos racionales de una curva elíptica. Esto quiere decir que dados ***P***, ***Q*** puntos de la curva elíptica hay que encontrar un número entero ***x*** tal que ***xP = Q*** ($xP = P + P + \ldots + P$, x veces). Obsérvese que a diferencia del **PFE** (Problema de Factorización Entera) el **PLDE** no maneja completamente números, lo que hace más complicado su solución. Actualmente el mejor algoritmo para calcular logaritmos discretos es el que se aplica a grupos en general llamado *método de la raíz de Pollard,* que tiene una complejidad $\sqrt{q}$, donde ***q*** es el tamaño del campo, es decir, que en un campo $\boldsymbol{Z_p}$ donde $p \sim 2^{160}$ tendríamos que efectuar $\sqrt{2^{160}} = 2^{80}$ operaciones para poder calcular un logaritmo.

- ✓ La creación de un protocolo con criptografía de curvas elípticas requiere que la elección de la curva sea adecuada, principalmente que sea no-supersingular[19] y que el orden del grupo de puntos racionales tenga un factor primo de al menos 163 bits, si el campo es $\mathbf{Z_p}$, se pide que la curva no sea anómala[20]. Todo esto con el fin de evitar los ataques conocidos.

- ✓ Lo anterior se ve reflejado en las ventajas que ofrecen los **CCE** en comparación con **RSA**, la principal es la longitud de la clave secreta. Se puede mostrar que mientras en **RSA** se tiene que usar una clave de 1024 bits para ofrecer una considerable seguridad, los **CCE** solo usan 163 bits para ofrecer la misma seguridad, así también las claves **RSA** de 2048 bits son equivalentes en seguridad a 210 bits de **CCE**. Esto se debe a que para resolver el **PLDE** el único algoritmo conocido toma tiempo de ejecución totalmente exponencial, mientras que el algoritmo que resuelve **PFE** incluso también el **PLD** en $\mathbf{Z_p}$, toman un tiempo subexponencial.

- ✓ Lo anterior permite que los **CCE** sean idóneos para ser implementados en donde el poder de cómputo y el espacio del circuito sea reducido, donde sea requerida una alta velocidad de procesamiento o grandes volúmenes de transacciones, donde el espacio de almacenamiento, la memoria o el ancho de banda sea limitado. Lo que permite su uso en smart cards, teléfonos celulares, Fax, PCs, etc. [WWW 004]

[19] Son curvas elípticas que son inmunes al MOV (de Menezes, Okamoto, Vanstone) el cual permite calcular logaritmos discretos.

[20] Una curva anómala es una curva elíptica que tiene tantos puntos racionales como elementos tiene el campo finito.

- ✓ Los **CCE** son el mejor candidato para reemplazar a las aplicaciones que tienen implementado **RSA**, estas definen también esquemas de firma digital, Intercambio de claves simétricas y otros.

COMPARACIÓN ENTRE RSA Y ECC

RSA	ECC
✓ Basa su seguridad en la dificultad para factorizar números grandes. ✓ Utiliza longitudes de claves grandes (512, 1024, 2048 bits) ✓ Es sencillo de comprender e implementar. ✓ Se considera como uno de los algoritmos asimétricos más seguros.	✓ Basa su seguridad en el problema del logaritmo discreto elíptico (PLDE) ✓ Utiliza longitudes de claves cortas (163, 210 bits) ✓ Es complejo de comprender e implementar. ✓ Son el mejor candidato para reemplazar a las aplicaciones que tienen implementado RSA

HACIENDO CRIPTOGRAFÍA CON ECC

Parámetros del Dominio de Curvas Elípticas.- Los parámetros del dominio de las curvas elípticas son los valores básicos necesarios para definir el campo finito a usar, los valores ***a*** y ***b*** que definen la curva, etc. Debe notarse que estos parámetros deben ser compartidos por las partes que

quieren comunicarse, de manera que en general se trata de utilizar siempre los mismos parámetros recomendados por las organizaciones productoras de estándares.

Los parámetros que definen el dominio de las curvas elípticas sobre Z_p es la séxtupla: $T = (p; a; b; G; n; h)$

Consistiendo de un número entero primo ***p*** que especifica el campo finito Z_p, dos elementos ***a***, ***b*** є Z_p especificando una curva elíptica $E(Z_p)$ definida en la ecuación (Ec. 4.2), un punto base $\boldsymbol{G} = (x_G, y_G)$ є $E(Z_p)$, un número primo ***n*** el cual es el orden de G, con $n > 2^{160}$ y $n > 4\sqrt{q}$; además $n \neq q$ y ***n*** no divide a $(q^k - 1)$ para cada $1 \leq k \leq 20$. Por último, un número entero ***h*** que es el cofactor $h = \#E(Z_p)/n$.

El proceso para generar la séxtupla $T = (p; a; b; G; n; h)$ es el siguiente:

1. Elegir el nivel de seguridad aproximado para los parámetros de dominio de curva elíptica. Este debe ser un entero ***t*** є {56; 64; 80; 96; 112; 128; 192; 256} de manera que calcular logaritmos sobre la curva generada tome aproximadamente $\mathbf{2^t}$ operaciones.
2. Seleccionar un número primo ***p*** tal que $\log_2 p = 2t$ si $t \neq 256$ y tal que $\log_2 p = 521$ si $t = 256$ para determinar el campo finito Z_p.

Seleccionar los elementos ***a***, ***b*** є Z_p para determinar la curva $E(Z_p)$ definida por la ecuación (Ec. 4.2), un punto base $\boldsymbol{G} = (x_G, y_G)$ sobre $E(Z_p)$, un número primo ***n*** el cual es el orden de ***G***, y un cofactor entero $\boldsymbol{h} = \#E(Z_p)/n$, sujeto a las siguientes restricciones:

(i) $4a^3 + 27b^2 \neq 0 \pmod p$.

(ii) $\#E(Z_p) \neq p$.

(iii) $p^B \neq 1 \pmod n$ para todo $1 \leq B < 20$.

(iv) $h \leq 4$.

La condición (ii) se pide para evitar las curvas anómalas, la (iii) para evitar las curvas supersingulares, y la (iv) para que $\#E(Z_p)$ sea un número "cerca" de un primo grande.

Dado un punto ***G***, es fácil calcular ***n*** y ***h*** cuando se conoce $\#E(Z_p)$. Encontrar $\#E(Z_p)$ involucra el cálculo del algoritmo de Schoof, o en métodos basados en Multiplicación Compleja, o en métodos basados en el Teorema de Weil.

Pares de Claves para Curvas Elípticas.- Dado los parámetros de dominio de curva elíptica T = (p; a; b; G; n; h), un par de claves de curva elíptica formado por ***(d, Q)*** asociado con ***T***. Consiste de una clave secreta ***d*** la cual es un entero en el intervalo [1, n - 1], y una clave pública ***Q*** = (x_Q, y_Q) el cual es el punto Q = dG.

Para generar un par de claves, una entidad procede como sigue, dados los parámetros válidos de dominio T = (p; a; b; G; n; h):

1. Seleccionar aleatoriamente un número entero ***d*** en el intervalo [1, n - 1].
2. Se calcula el punto Q = dG.
3. La llave pública es el punto ***Q***
4. La llave privada es el entero ***d***

Ejemplo: Si consideramos la curva elíptica sobre el campo finito F_{23} con a = 2 y b = 8, la ecuación de la curva elíptica es $y^2 = x^3 + 2x + 8$. Los 15 puntos que satisfacen esta ecuación son:

(0,10) (0,13) (3,8) (3,15) (6,11) (6,12) (10,4) (10,19)
(11,2) (11,21) (12,9) (12,14) (13,0) (15,3) (15,20)

Si elegimos un punto base $\boldsymbol{G} = (x_G, y_G) = (3,8) \in E(Z_p)$, y n=5.
Ahora $h = \#E(Z_p)/n = 15/5 = 3$.
Los parámetros que definen el dominio de la curva elíptica sobre Z_{23} es la séxtupla: T = (p; a; b; G; n; h) = (23; 2; 8; (3,8); 5; 3)

4.5 CRIPTOSISTEMAS BASADOS EN LOGARITMOS DISCRETOS

Curiosamente, la inmensa mayoría de los algoritmos asimétricos que se usan en la actualidad para proporcionar seguridad en la información se apoyan en el llamado *problema de los Logaritmos Discretos (PLD)*, que se cree es computacionalmente tan complejo como el problema de la factorización de números primos grandes.

Criptosistemas como: El Gamal *(sección 4.3)*, Diffie-Hellman, e incluso la moderna Criptografía de Curva Elíptica *(sección 4.4)* depositan su seguridad en el problema del logaritmo discreto. Todos ellos descansan en la supuesta imposibilidad de resolverlos de forma algorítmicamente eficiente.

El problema del *Logaritmo Discreto* se define sobre aritmética modular, y consiste en averiguar cuántas veces hay que multiplicar un número consigo mismo para que nos dé otro concreto. Por ejemplo: $\log_4 12 \pmod{13}$ es decir, el logaritmo en base ***4*** de ***12*** módulo ***13*** es el número de veces que hay que multiplicar el ***4*** por sí mismo para que nos dé ***12*** módulo ***13***.
El resultado es ***3***, ya que $4 * 4 * 4 = 64 \equiv 12 \pmod{13}$.

Una función típica es la exponenciación modular dada por la ecuación: $y \equiv g^x \ (mod\ p)$, con ***g, x*** valores enteros y siendo ***p*** un número primo grande de 200 dígitos ó más. El cálculo de la función ***y*** es posible, ya que tiene una complejidad **O(**log p**)**, pero el cálculo de la función inversa $x \equiv log_g y \ (mod\ p)$ para números grandes tiene tanta complejidad que es totalmente inviable resolverla. Naturalmente, este ***x*** es el *logaritmo discreto* de ***y***.

En la actualidad existen algunos métodos para calcular *Logaritmos Discretos*. Hasta la fecha el método más adecuado para calcular logaritmos discretos es el *Método del Índice*.