Ciencias Matemáticas, Vol. 33, No. 1, Pag. 18-27, 2019

Generación de curvas elípticas con buenas propiedades criptográficas sobre campos primos
Generation of elliptic curves with good cryptographic properties over prime fields

Yessica Caridad Castaño Sainz[1]*, Claudia Espinosa Contreras [2], Huber Martínez Rodríguez[3]

Resumen En este trabajo se presentan las funcionalidades del paquete implementado en SAGE, ECDP_Generator, el cual permite generar parámetros de definición de curvas elípticas utilizando los métodos pseudoaleatorio y pseudoaleatorio verificable para las formas de Weierstrass, Edwards y torcida de Edwards y evaluar la seguridad de diferentes juegos de parámetros. Para ello se analizaron los requisitos de seguridad que una curva elíptica debe satisfacer para ser utilizada en sistemas criptográficos teniendo en cuenta los requisitos adoptados por los diferentes estándares y las discusiones sobre la estandarización de nuevas curvas elípticas. Se analizan requisitos técnicos que propician implementaciones eficientes y en el caso de la forma de Edwards (torcida de Edwards) completitud en las operaciones de grupo. Se presentan algoritmos para verificar las condiciones del discriminante y el número de clases que constituyen las restricciones más complejas.

Abstract In this work, the functionalities of the package ECDP_Generator, which is implemented in SAGE are presented. This package allows the generation of secure elliptic curve domain parameters for the Weierstrass, Edwards and Twisted Edwards models, using the pseudo-random and pseudo-random verifiable methods. It also could be used to evaluate the security of different parameter sets to be used in cryptographic applications. The security requirements that an elliptic curve must satisfy to be used in cryptographic systems were analysed taking into account requirements adopted by different standards and recent discussions about the standardization of new elliptic curves. Technical requirements are analyzed allowing efficient implementations and completeness in the group operations in the case of Edwards (twisted Edwards) forms. Algorithms are presented to verify the discriminant and class number conditions which constitute the most complex ones.

Palabras Clave
parámetros de definición de curvas elípticas — requisitos de seguridad — requisitos técnicos

[1] *Instituto de Criptografía, Universidad de la Habana, La Habana, Cuba, yessica.castano@matcom.uh.cu*
[2] *Empresa de Tecnologías para la Defensa (XETID),cespinosa@xetid.cu*
[3] *Departamento de Matemática, Facultad de Ciencias Informáticas, Universidad de Ciego de Ávila, Cuba, huber@unica.cu*
***Autor para Correspondencia**

Introducción

La elección apropiada de los parámetros de dominio de una curva elíptica es fundamental en la seguridad de los criptosistemas de curvas elípticas. La desconfianza en los parámetros utilizados debido a posibles manipulaciones atenta contra la popularidad de estos criptosistemas. Las entidades encargadas de proveer seguridad requieren de opciones para la elección de curvas elípticas seguras o contar con un sistema que las genere.

Tras el aumento de la desconfianza [1] en las curvas previamente estandarizadas [2, 16, 3, 36, 3, 1, 18, 30, 32], varios fueron los grupos de trabajo que publicaron sus puntos de vista sobre las propiedades que deben satisfacer o no y el proceso que debe ser usado para generar las curvas elípticas. En [11, 12, 34, 5, 33, 24] se recogen las posiciones [2] de diferentes grupos de investigación y requisitos de seguridad y rendimiento que una curva debe satisfacer y en [7, 12, 5] nuevas propuestas de curvas.

Las discusiones en el grupo de Investigación IETF Crypto Forum Research Group (CFRG), tras la petición de nuevas curvas por el grupo de trabajo IETF TLS, dieron como resultado que en [31] fueran estandarizadas las curvas Curve25519 [7] y Curve448 [26].

[1] Según [12], la necesidad de cambiar las curvas fue impulsada por los documentos filtrados de la NSA, que sugieren la existencia de una puerta trasera en el generador de bits aleatorios determinista Dual Elliptic Curve [39], aunque ya los criptógrafos sospechaban esto por lo menos desde el 2007 [21]. Estas revelaciones aceleraron desde el 2014 una controversia sobre si las curvas del NIST debían ser sustituidas por curvas con una generación verificable determinista.

[2] Aquí se hace referencia al criterio de los diferentes grupos de investigación en cuanto a la rigidez del proceso de generación el concenso entre eficiencia y seguridad, las requisitos de seguridad, entre otros aspectos que se detallan en cada artículo

Según los análisis que se realizan en [33, 24] la tendencia de los criterios de selección fue aumentar la eficiencia de los cálculos haciendo concesiones que podrían poner en riesgo la seguridad de los sistemas y consideran que se debe seguir tanto para la generación del número primo como para la generación de las curvas un procedimiento pseudoaleatorio verificable.

La seguridad de una curva elíptica para sistemas criptográficos se mide mediante la verificación del cumplimiento de requisitos de seguridad que eliminan posibles curvas débiles. Los requisitos de seguridad contemplan restricciones para evadir ataques al problema del logaritmo discreto, y ataques relacionados con las implementaciones, entre los que se encuentran ataques que son el resultado de utilizar información adicional (consumo de tiempo, radiaciones electromagnéticas, consumo de energía, etc) para obtener información sobre la clave, conocidos como ataques de canal colateral.

La elección de los parámetros de la curva tiene una enorme influencia en el rendimiento del sistema de cifrado resultante, es por ello que en todas las elecciones de curvas se hacen concesiones que permitan ganar en eficiencia sin poner en riesgo la seguridad de los sistemas criptográficos que utilicen las curvas elípticas seleccionadas. Estas concesiones se resumen como requisitos técnicos.

El objetivo de este trabajo es presentar las funcionalidades del paquete ECDP_Generator, el cual posibilita que una entidad proveedora de seguridad pueda generar curvas siempre que sea necesario según los escenarios que se deseen proteger. Además pueden ser añadidas restricciones de seguridad en caso de que se encuentren nuevos grupos de curvas débiles.

1. Modelos de curvas elípticas

Se han propuesto varias representaciones alternativas de curvas elípticas, ver [9].

Sea $E/\mathbb{F}_p$ con $p > 3$ un campo primo grande, hasta la fecha hay esencialmente dos modelos que son los que más se han propuesto para ser utilizados : el modelo de Weierstrass reducido: $E^W_{a,b} : y^2 = x^3 + ax + b$ y el modelo Torcida de Edwards: $E^{Ed}_{a,d} : ax^2 + y^2 = 1 + dx^2y^2$, [8]. El modelo de Montgomery: $E^M_{A,B} : By^2 = x^3 + Ax^2 + x$, [35] también ha sido considerado pero se ha dejado a un lado después de la introducción del modelo Torcida de Edwards. En [8] se demuestra que el conjunto de las curvas de Montgomery sobre k, con k un campo finito de característica diferente de 2, es equivalente al conjunto de las curvas torcidas de Edwards sobre k. Luego esta correspondencia permite utilizar los beneficios de ambas representaciones en dependencia del entorno donde sea utilizada la curva.

En las discusiones sobre nuevas curvas para ser estandarizadas el modelo de curva a utilizar ha sido ampliamente discutido. Las discusiones en [33, 24, 12] evidencian que es importante contar con curvas en forma de Edwards y en forma de Weierstrass ya que cada modelo presenta características que los hacen apropiados dependiendo del escenario donde son utilizados.

2. Requisitos de seguridad

A continuación se resumen los requisitos de seguridad que deben cumplir los parámetros de definición de curvas elípticas para ser utilizados de forma segura en criptosistemas de curvas elípticas. Estos requisitos son el resultado del análisis de los requisitos de seguridad verificados a las curvas estandarizadas y los requisitos que se tuvieron en cuenta en las discusiones para la estandarización de nuevas curvas.

Elección del campo primo base

El número primo no deberá tener formas especiales. Entre los parámetros que definen una curva elíptica para ser utilizada en aplicaciones criptográficas (pdce), se encuentra el campo sobre el cual será considerada la curva. La comunidad criptográfica ha considerado primos pseudoaleatorios y primos de forma especial que aceleran la reducción modular. Según [24] a pesar de la tendencia en la selección de primos en formas especiales que permitan garantizar cálculos más rápidos, esta elección también hace los primos más vulnerables frente a algunos ataques de canal colateral [22, 37, 6, 40, 41, 23]. Considerar primos especiales también requiere de implementaciones específicas para el primo que se esté considerando. Ver los análisis que se hacen de este tema en [24, 34].

Ausencia de subgrupos pequeños

Si el cofactor es mayor que uno se pueden realizar ataques en subgrupos pequeños. Por ejemplo: un atacante puede escoger un punto de menor orden en el protocolo Diffie- Hellman como su clave pública y obtener información sobre la clave privada de otros, ver [19] y sección Twist security en [11]. Evitar que sea efectivo este ataque puede requerir un chequeo adicional o una operación de multiplicación adicional.

Se requiere que el cofactor $h = 1$ para las curvas en forma de Weierstrass.

Las propuestas de curvas de Edwards (torcida de Edwards) requieren utilizar un cofactor mayor que uno. Luego se deberá verificar para estas formas de curvas que tengan cofactor cuatro.

Seguridad frente a ataques por isomorfismos

La curva elíptica E debe ser tal que $\#E(\mathbb{F}_p) \neq p$ o de forma equivalente no debe ser anómala. Una curva elíptica E definida sobre una campo primo $\mathbb{F}_p$ se dice anómala de campo primo cuando $\#E(\mathbb{F}_p) = p$. En este caso el grupo $E(\mathbb{F}_p)$ es cíclico ya que tiene orden primo, por lo tanto $E(\mathbb{F}_p)$ es isomorfo al grupo aditivo $\mathbb{F}^+_p$ de los enteros módulo p y nos encontramos en presencia de una transferencia aditiva. Luego el Problema del Logaritmo Discreto de Curvas Elípticas (PLDCE) puede ser resuelto eficientemente por lo que estas curvas elípticas no deben ser utilizadas en protocolos criptográficos. Mediante el conteo de los puntos en $E(\mathbb{F}_p)$ es fácil determinar si una curva elíptica E sobre un campo primo $\mathbb{F}_p$ es anómala (**comprobando si** $\#E(\mathbb{F}_p) = p$).

La curva elíptica E deberá ser resistente ante los ataques de encrustamiento por emparejamientos de Weil o Tate. Una transferencia multiplicativa se aplica cuando $n = \#\langle P \rangle$

divide a p^k-1, las cotas para k varían de un estándar a otro, ver [4, 15, 17, 32, 11]. **Para evitar transferencias multiplicativas se deberá tener:**$(n-1)/\ell < 100$**, donde** $\ell = \min\{k \mid n$ **divide a** $p^k-1\}$ o de forma equivalente el orden multiplicativo de $p \mod n$.

Restricciones sobre el discriminante y el número de clases

La curva elíptica E deberá tener un número de clases grande. Exigiremos al igual que para el estándar Brainpool que sea > 10000000 y deberá tener $|D_K| > 2^{100}$.

A pesar de que no hay ataques conocidos públicamente al PLDCE que explote un número de clases pequeño, en el estándar de curvas Brainpool [32] se requiere que el número de clases sea mayor que 10000000. Afirman que el artículo [27] puede ser visto como un argumento para la condición del número de clases. En caso de existir una elevación de la curva elíptica E a una curva E' tal que $End(E) = End(E')$ sobre el campo numérico L, el número de clases constituye una cota inferior para el grado de la extensión $L/\mathbb{Q}$. Luego un ataque del tipo Cálculo de Índices no sería viable.

En SafeCurves [11] se requiere que $|D_K|$ sea $> 2^{100}$. Al igual que la restricción de Brainpool no se basa en la existencia de un ataque práctico sino en que si cumplen estas restricciones pueden ser más difíciles de romper. Según SafeCurves si se asume cierta la hipótesis generalizada de Riemann este requisito es más fuerte que la restricción sobre el número de clases que impone Brainpool. No obstante, deberán ser verificadas ambas restricciones.

Seguridad de la curva torcida cuadrática

En las discusiones recientes sobre la estandarización de nuevas curvas se ha añadido como requisito la seguridad de la curva torcida cuadrática, ver sección "Twist security" en [11]. Esta es una de las restricciones más controversiales ya que hay quienes consideran que verificar este requisito puede llevar a ataques en subgrupos de curvas especiales, ver por ejemplo [20, 34].

Con el objetivo de evitar ataques de curvas no válidas, la noción de seguridad de la curva torcida cuadrática fue introducida en [7] como método para evitar chequeos de pertenecia de parámetros a una curva, en el caso especial en que se trabaja solo con la coordenada x, correspondiente al modelo de Montgomery, $E^M_{A,B}$. En el caso de las curvas en forma de Montgomery la única posibilidad para un ataque de curva no válida es la curva torcida cuadrática. Si se utilizan curvas en forma de Weierstrass, este no es un argumento válido para imponer la seguridad de la curva torcida cuadrática ya que siempre se deben realizar los chequeos triviales para estar seguros de que los puntos transmitidos se encuentran en la curva correcta. No obstante SafeCurves demanda que la curva torcida cuadrática cumpla con los requisitos que garantizan la seguridad del PLDCE para todas las curvas. Afirman que la seguridad de la curva torcida cuadrática es importante en protocolos de curvas elípticas que hacen uso tanto de la curva elíptica original como de su torcida cuadrática, hacen referencia a los protocolos en [28, 29, 13].

Como conclusión se debe contar con curvas que verifiquen la seguridad de la curva torcida cuadrática y curvas que no. Las mismas deberán ser usadas en correspondencia con el escenario, ver el análisis sobre los escenarios que se hace en [34].

Hagamos un análisis de los requisitos que se han venido discutiendo para la curva original, ahora aplicados a la curva torcida cuadrática.
La curva torcida cuadrática $\tilde{E}$ de una curva en forma de Edwards $E^{Ed} : x^2+y^2 = 1+dx^2y^2$ o de una curva en forma de Weierstrass $E : y^2 = x^3+ax+b$ deberá tener $\#E(\mathbb{F}_p) = n'h'$ con $n' > 2^{160}$ un número primo y el mínimo valor para el cofactor h'; para evitar ataques al PLDCE y ataques en subgrupos pequeños. En el caso de las curvas en forma de Weierstrass $h' = 1$ y para las curvas en forma de Edwards $h' = 4$ cuando $p \equiv 3 \mod 4$ y $h' = 8$ cuando $p \equiv 1 \mod 4$ lo cual se sigue del hecho de que las curvas en forma de Edwards siempre tienen un punto de orden cuatro y de que $\#E(\mathbb{F}_p) + \#\tilde{E}(\mathbb{F}_p) = 2p+2$. Se deberá verificar que no sea anómala y la seguridad frente a un ataque por emparejamientos de Weil y Tate. En el caso de la condición del discriminante y el número de clases no se tendrá que verificar nuevamente ya que la curva y la torcida tienen el mismo discriminante de multiplicación compleja.

3. Requisitos técnicos

A continuación se presentan requisitos que pueden ser verificados a los parámetros de definición de curvas elípticas con el objetivo de acelerar las implementaciones sin poner en riesgo la seguridad de los criptosistemas.

- $p \equiv 3 \mod 4$. Este requisito permite una forma de compresión de puntos eficiente: Un método para la transmisión de un punto $P = (x,y)$ de la curva, es transmitir solo la coordenada x y los bits menos significativos de la coordenada y. Usando la ecuación de la curva y el hecho de que si $p \equiv 3 \mod 4$ entonces $(y^2)^{(p+1)/4} = y^{(p-1)/2} \cdot y \mod p$, que es o bien y o $-y$ por el pequeño teorema de Fermat, el valor de y puede ser calculado de forma eficiente.

- La curva en forma de Weierstrass deberá ser isomorfa a una curva con $a \equiv -3 \mod p$. En [14] se demuestra; que es posible obtener el valor deseado $a \equiv -3 \mod p$ para una curva elíptica isogenia y realizar los cálculos en esta curva en lugar de en la original. De esta forma se puede utilizar las ventajas de eficiencia en la aritmética que brinda esta selección partiendo de curvas generadas de forma pseudoaleatoria. Brainpool utiliza estos resultados en el proceso de generación.
  Este requisito es satisfecho por una curva torcida cuadrática E_1 de la curva dada con un cuadrado en $\mathbb{F}_p$. Si $-3 \equiv au^4 \mod p$ es soluble, entonces E y $E_1 : y^2 = x^3+a\cdot u^4\cdot x+u^6\cdot b = x^3-3x+u^6b \mod p$ son $\mathbb{F}_p$-isomorfas a partir del isomorfismo $\phi := (u^2\cdot x, u^3\cdot y)$.

En particular $\#E(\mathbb{F}_p) = \#E_1(\mathbb{F}_p)$ y, aún más importante E y E_1 tienen la misma estructura y por lo tanto ofrecen el mismo nivel de seguridad. Se tiene que proximadamente la mitad de las clases de isomorfismos de las curvas elípticas sobre $\mathbb{F}_p$ con $p \equiv 3 \mod 4$ contiene una curva con $a \equiv -3 \mod p$.

- El coeficiente b en la forma de Weierstrass no deberá ser un cuadrado módulo p. Este requisito se incluye en Brainpool [32] con el objetivo de evitar ambigüedades con la representación del punto al infinito que se hace en algunas implementaciones, además argumentan que este requisito se puede ver como un requisito de seguridad ante el ataque que se presenta en [25].

- Asumir que E es una curva elíptica que es birracionalmente equivalente a alguna curva torcida de Edwards $E_{a,d}$ y hacer las adiciones de puntos en la curva $E_{a,d}$ puede aumentar significativamente la eficiencia. Se ha considerado elegir curvas de Edwards representadas por la ecuación $E_{\pm 1,d} : \pm x^2 + y^2 = 1 + dx^2y^2$ con el fin de ganar en eficiencia al fijar el valor de a para la forma torcida de Edwards en 1 ó -1 cuando $p \equiv 3 \mod 4$ ó $p \equiv 1 \mod 4$ respectivamente. En el requisito técnico 1 se ha asumido que $p \equiv 3 \mod 4$ luego en nuestro caso se estaría utilizando las curvas de Edwards $E^{Ed}_{1,d} : x^2 + y^2 = 1 + dx^2y^2$. La forma torcida de Edwards abarca mayor cantidad de curvas para el caso $p \equiv 1 \mod 4$ al hacer cada curva en forma de Montgomery birracionalmente equivalente sobre $\mathbb{F}_p$ a una curva en forma torcida de Edwards. Cuando $p \equiv 3 \mod 4$ no es necesario ya que las curvas en forma de Edwards cubren todas las curvas en forma de Montgomery. La tabla en la página 14 de [8] resume los costos para la ley del grupo en ambas formas. De la tabla se sigue que la forma torcida de Edwards pierde 1D en cada sistema de coordenadas y para ambas operaciones de grupo.
  Cuando se considera un método pseudoaleatorio o pseudoaleatorio verificable para generar los coeficientes de las curvas, estos generalmente resultan grandes. Este hecho unido a que $p \equiv 3 \mod 4$ justifica la elección de generar parámetros para la forma de Edwards.

  En [10] (donde a=1) y luego en [8], fue probado que si d no es un cuadrado en k, con k un campo de característica impar, y a es un cuadrado en k entonces las fórmulas de las operaciones de grupo para las curvas en formas de Edwards y torcida de Edwards son completas. Luego para la forma torcida de Edwards el coeficiente a deberá ser un cuadrado en $\mathbb{F}_p$ y d no deberá ser un cuadrado en $\mathbb{F}_p$ para garantizar la completitud de la ley del grupo. En el caso de las curvas en forma de Edwards ($a = 1$) y solo se aplica al coeficiente d.

- $\#E(\mathbb{F}_p) < p$ para la forma de Weierstrass. Esta restricción se incluye en Brainpool para evitar sobrecosto en las implementaciones. Ver requisito técnico 5 de [1].
  Como consecuencia del teorema de Hasse, para el caso de Weierstrass, donde se considera $\#\langle P\rangle = \#E(\mathbb{F}_p)$ primo, el cardinal del subgrupo generado por P puede ser mayor que la carácterística p del campo finito $\mathbb{F}_p$, en algunos casos la longitud en bits de $\#\langle P\rangle = \#E(\mathbb{F}_p)$ puede exceder la longitud en bits de p en un bit.

  En el caso de las formas de Edwards y torcida de Edwards como mínimo se deberá tener $\#\langle P\rangle = \#E(\mathbb{F}_p)/4$, de donde se sigue utilizando el teorema de Hasse que $\#\langle P\rangle < p$ y por tanto la longitud en bits de $\#\langle P\rangle$ siempre será menor que la longitud en bits de p. Luego este requisito no es necesario para estas formas.

Deberán ser verificados los requisitos técnicos listados en el cuadro 1:

4. Algoritmos para la generación de parámetros

La aceptación de los parámetros de definición de curvas elípticas depende en gran medida del procedimiento para la generación. Como se mencionó en la introducción este proceso debe ser transparente y permitir ser verificado por quienes usarán las curvas generadas. Un procedimiento pseudoaleatorio verificable que parta de semillas bien justificadas (que pueden ser cifras de consatntes matemáticas comunes como π y e) permite a terceras entidades verificar el origen de las curvas propuestas y descartar posibles manipulaciones.

Por otra parte, un algoritmo pseudoaleatorio evita la elección y justificación de nuevas semillas y de igual forma tiene como salida curvas elípticas que cumplen con los requisitos de seguridad y que pueden ser utilizadas sin temor a ser manipuladas por la propia entidad que las genera.
El paquete ECDP_Generator implementa los algoritmos pseudoaleatorios verificables y pseudoaleatorios propuestos y analizados en [42].

5. Algoritmos para verificar las condiciones del discriminante y del número de clases

El anillo de endomorfismos de $E, End(E)$, es isomorfo a un orden en un campo cuadrático imaginario $K = \mathbb{Q}(\sqrt{-D})$ con $D \in \mathbb{N}$ libre de cuadrados.

El campo de endomorfismos de la curva E es el campo K generado por su endomorfimo de Frobenius π_p. Como π_p es una raíz de la ecuación $x^2 - tx + p$, K es un campo cuadrático imaginario. El discriminante de π_p es el valor $D_{\pi_p} = t^2 - 4p < 0$. Este es el discriminante del orden $\mathcal{O}_{\pi_p} = \mathbb{Z}[\pi_p] \subset K$.

El discriminante de K es el discriminante (fundamental) D_K de su orden maximal $\mathcal{O}_K$. Está estrechamente relacionado con la parte libre de cuadrados de $D_\pi : D_\pi = D_K f^2_{\pi_p}$ para $f_{\pi_p} \in \mathbb{Z}$, llamado conductor del orden $\mathcal{O}_{\pi_p}$, y D_K o $D_K/4$ es un entero libre de cuadrados.

Requisitos	Descripción
1. $p \equiv 3 \mod 4$	
2. El coeficiente a en la ecuación de la curva para la forma de Weierstrass deberá hacer la ecuación $-3 \equiv au^4 \mod p$ soluble.	La curva deberá ser isomorfa a una curva con $a \equiv -3 \mod p$ en el caso de la forma de Weierstrass.
3. El coeficiente b para la forma de Weierstrass no deberá ser un cuadrado módulo p.	Evita ambigüedades con la representación del punto al infinito que se hace en algunas implementaciones. Puede verse como un requisito de seguridad.
4. En la forma de Edwards: El coeficiente d en la ecuación de la curva $E^{Ed} : x^2 + y^2 = 1 + dx^2y^2$ no deberá ser un cuadrado en $\mathbb{F}_p$.	Garantiza completitud de la ley del grupo.
5. En la forma torcida de Edwards: El coeficiente a en $E^{Ed} : ax^2 + y^2 = 1 + dx^2y^2$ deberá ser un cuadrado en $\mathbb{F}_p$ y el coeficiente d no deberá ser un cuadrado en $\mathbb{F}_p$.	Garantiza completitud de la ley del grupo.
6. $\#E(\mathbb{F}_p) < p$ para la forma de Weierstrass.	Evita sobrecosto en las implementaciones.

Cuadro 1. Requisitos Técnicos.

Se puede escribir $End(E) = \mathbb{Z} + c_E\mathcal{O}_K$ donde $\mathbb{Z}$ son los enteros racionales, $\mathcal{O}_K$ es el anillo de enteros de k y c_E denota el conductor $[\mathcal{O}_K : End(E)]$.

El discriminante de $End(E)$ es $c_E^2 \cdot D_K$, donde D_K es el discriminante de $\mathcal{O}_K$ o equivalentemente: $D_K = -D$ si $-D \equiv 1 \mod 4$ y $D_K = -4D$ si $-D \equiv 2 \mod 4$ ó $-D \equiv 3 \mod 4$

Sea $\#E(\mathbb{F}_p) = p + 1 + t$, teniendo en cuenta que $[\mathcal{O}_K : \mathbb{Z}[\pi_p]] = [\mathcal{O}_K : End(E)] \cdot [End(E) : \mathbb{Z}[\pi_p]]$ y $D_\pi = t^2 - 4 \cdot p$, entonces D puede ser calculado como la parte libre de cuadrados de

$$4 \cdot p - t^2 = -c_E^2 \cdot [End(E) : \mathbb{Z}(\pi_p)]^2 \cdot D_K,$$

donde π_p denota el endomorfismo de Frobenius de E. Si $v = \max\{a \,|\, a^2 \text{ divide a } 4 \cdot p - t^2\}$ entonces $D = (4 \cdot p - t^2)/v^2$.

De aquí se puede obtener el valor del discriminante que permite verificar la condición sobre el discriminante impuesta en SafeCurves.

Algoritmo para verificar la condición del discriminante

Entrada: El primo p que define el campo finito $\mathbb{F}_p$ y el cardinal del campo $n = \#E(\mathbb{F}_p)$.

Salida: TRUE: si se cumple la restricción, FALSE: en caso de que falle.

1: Hacer $u = n - p - 1$.
2: Hacer $d \leftarrow$ la parte libre de cuadrados en la factorización de $4p - u^2$.
3: **si** $\text{Mod}(-d, 4) = 1$ **entonces**
4: $\quad D_K = -d$
5: **si no**, **si** Mod(-d,4)=2 or Mod(-d,4)=3 **entonces**
6: $\quad D_K = -4 * d$
7: **fin si**
8: **si** $|D_K| < 2^{100} + 1$ **entonces**
9: $\quad$ **retornar** FALSE y parar.
10: **fin si**
11: **devolver** TRUE

Para verificar la condición del número de clases es necesario comprobar que el número de clases de $\mathcal{O}_K$ es mayor que 10000000.

En la práctica debido al elevado costo computacional del cálculo del número de clases para discriminantes grandes, se usa la biyección con el grupo de clases de formas cuadráticas binarias con discriminante $D_K < 0$ para acotar el número de clases en lugar de calcularlo. Esta idea fue utilizada en el proceso de generación de las curvas de Brainpool. Basándonos en esto, solo tenemos que hallar una forma cuadrática reducida de discriminante fundamental D_K; ya se tendría garantizado que sea primitiva y que sea definida positiva, pues $D_K = b^2 - 4ac$. Esto se hace de manera muy sencilla: se busca un b tal que $q := b^2 - D_K \equiv 0 \mod 4$, luego se buscan a y c tales que $ac = q/4$ y la forma cuadrática $ax^2 + bxy + cy^2$ sea reducida.

Pudiera ocurrir que la primera forma cuadrática que se considere no genere un subgrupo de orden mayor que 10000000, aún en el caso en que ciertamente el valor del número de clases si lo cumpla. El valor de la cantidad de formas a probar puede ser cambiado en dependencia de la complejidad que le agreguen al programa el orden de los discriminantes.

Tener en cuenta que cuando el grupo no es cíclico este método nunca va a retornar el valor del número de clases.

Luego en caso de que con las formas que se pruebe no se obtenga un orden del grupo cíclico generado por ellas > 10000000 no se puede concluir de forma absoluta que $h_K < 10000000$. Lo que se concluye es que se incumple la condición del número de clases por no encontrar una forma cuadrática reducida cuyo orden fuera mayor que 10000000.

Algoritmo para verificar la condición del número de clases

Entrada: El primo p que define el campo finito $\mathbb{F}_p$, el cardinal del campo $n = \#E(\mathbb{F}_p)$, el número de formas cuadráticas reducidas a chequear CotFormas y la cota para el número de clases Cot_Class_Number.

Salida: TRUE: si se cumple la condición, FALSE: en caso de que falle.

1: Hacer $u = n - p - 1$.
2: Hacer $d \leftarrow$ la parte libre de cuadrados en la factorización de $4p - u^2$.
3: **si** Mod$(-d, 4) = 1$ **entonces**
4: $D_K = -d$
5: **si no**, **si** Mod(-d,4)=2 or Mod(-d,4)=3 **entonces**
6: $D_K = -4 * d$
7: **fin si**
8: **para** NumFormas=0 hasta CotFormas **hacer**
9: Encontrar una forma cuadrática $f \neq I$ de discriminante D_K (donde I es el elemento identidad)
10: NumFormas $\leftarrow$ NumFormas+1
11: $cardf \leftarrow 0$
12: $t \leftarrow f$
13: **mientras** $t \neq I$ **hacer**
14: Hacer $t = t \bullet f$.
15: $cardf \leftarrow cardf + 1$
16: **si** $cardf >$ Cot_Class_Number **entonces**
17: **retornar** TRUE y parar.
18: **fin si**
19: **fin mientras**
20: **fin para**
21: **devolver** FALSE

6. Codificación en SAGE

Los algoritmos fueron codificados usando el software matemático de código abierto SAGE dando como resultado el paquete **ECDP_Generation**.
Se debe instalar la versión 7.4 de SAGE en adelante ya que en esta se incorpora la versión 2.8.0 de PARI/GP que incluye el algoritmo de conteo de puntos SEA con posibilidad de utilizar estrategias de aborto temprano (función ellsea.c en PARI/GP) . Se deberán añadir los polinomios modulares del paquete seadata-big.tar para el conteo de los puntos de las curvas de 512 bits. El algoritmo pseudoaleatorio verificable necesita de la inclusión de la función de hash SHA3 ya que SAGE no la tiene incorporada. Se puede descargar la implementación que hicieron de estas funciones Bernstein y Lange de la página `https://bada55.cr.yp.to` o de la dirección `https://zenodo.org/record/34081#.WJOTTc-YphG`.

Se confeccionó el paquete ECDP_Generation con los algoritmos propuestos en [42] y para trabajar con las curvas de Edwards (torcida de Edwards) se utilizaron las clases de Python/SAGE propuestas en [38].
Para ser instalado se debe colocar el paquete en la carpeta upstream, y se instala utilizando una de las siguientes líneas de comando:

```
sage -f ECDP_Generator.spkg

sage -i ECDP_Generator.spkg
```

Se implementaron varias funciones y se ofrecen para la interacción con el paquete las siguientes funcionalidades:

```
Prime_Pseudorandom_Verif(L, seedprime, filename)

Prime_Pseudorandom(L, filename)

ECDP_security_Verification(p, form, parameters,
 security_requirements, filename)

ECDP_technical_Verification(p, form, parameters,
technical_requirements, filename)

ECDP_Point_Verification(p, form, parameters,
P, filename)

ECDP_Generation(p, form, L, filename,
 requirements, seedcurva=0)
```

Función: Prime_Pseudorandom_Verif(L, seedprime, filename):

Esta función implementa el algoritmo pseudoaleatorio verificable para la generación de números primos propuesto en [42], donde:
L: longitud en bits para los diferentes niveles de seguridad.
seedprime: semilla para la generación del número primo en forma hexadecimal.
filename: especifica el archivo donde será guardado el número primo.

Función: Prime_Pseudorandom(L, filename)
Esta función implementa el algoritmo pseudoaleatorio propuesto en [42] para la generación de números primos, donde los parámetros de entrada L y filename se definen como en el algoritmo anterior.

Función ECDP_security_Verification(p, form, parameters, security_requirements, filename)

Esta función puede ser utilizada para chequear los requisitos de seguridad de una curva elíptica dada E en forma de Weierstrass, Edwards o torcida de Edwards. Se debe tener en cuenta que la curva $E^{Ed}_{1,d} : x^2 + y^2 = 1 + dx^2y^2$ en forma de Edwards es la curva $E^{Ed}_{T,a,d} : ax^2 + y^2 = 1 + dx^2y^2$ con parámetro $a = 1$. Los parámetros de entrada se definen de la siguiente forma:

p: primo que define el campo base $\mathbb{F}_p$.

form: especifica la forma de la curva elíptica que puede ser:

```
'Weierstrass', 'Edwards' o 'Twisted_Edwards'
```

parameters: especifica la tupla (a, b) de los parámetros que definen la ecuación reducida de Weierstrass $E^W_{a,b}$ o (a, d) para el caso de la ecuación en forma torcida de Edwards $E^{TE}_{a,d}$. Para el caso de una curva en forma de Edwards la tupla sería (1, d).

security_requirements: vector compuesto por las restricciones de seguridad que serán verificadas a la curva elíptica y que se debe conformar a partir de las entradas:

```
('Cofactor', number)
('Class_Number', CotFormas, CotClassNumber)
```

```
('Weil_Tate', cofactor)
('Discriminant', cota)
('Twist_Secure', twist_security_
requirements)
```

donde:

('Cofactor', number): especifica que se quiere verificar que el cofactor tome el valor number.

('Class_Number', CotFormas, CotClassNumber): especifica que se quiere verificar que el número de clases del anillo de los endomorfismos de la curva elíptica E es mayor que el valor especificado en CotClassNumber. CotFormas, especifica el máximo número de formas cuadráticas a las que se les comprobará el orden, ver sección 5.

('Weil_Tate', cofactor): especifica que se quiere verificar la condición de Weil y Tate. El valor, cofactor, indica el valor h tal que $\#E(\mathbb{F}_p) = n \cdot h$.

('Discriminant', cota): especifica que se quiere verificar que el discriminante con multiplicación compleja D_K sea mayor que el valor especificado en cota.

('Twist_Secure', twist_security_requirements): especifica que se quiere verificar la seguridad de la curva torcida cuadrática, donde:

twist_security_requirements: especifica los requisitos de seguridad que serán verificados a la curva torcida cuadrática conformado a partir de las entradas

```
('Cofactor', number)
('Class_Number', CotFormas, CotClassNumber)
('Weil_Tate', cofactor)
('Discriminant', cota)
```

como se especificaron anteriormente.

filename: especifica el archivo donde se guardan los resultados.

Función ECDP_technical_Verification(p, form, parameters, technical_requirements, filename)

Esta función puede ser utilizada para chequear los requisitos técnicos del cuadro 1 a una curva elíptica dada E en forma de Weierstrass, Edwards o torcida de Edwards. Los parámetros **p**, **form** y **parameters** se definen como anteriormente y:

technical_requirements: vector compuesto por las restricciones técnicas que serán verificadas a la curva elíptica y que se debe conformar a partir de las entradas: 'req1', 'req2', 'req3', 'req4', 'req5', o 'req6', las cuales especifican que se quiere verificar el requisito técnico 1, 2, 3, 4, 5 o 6 en el cuadro 1 respectivamente.

Función ECDP_Point_Verification(p, form, parameters, P, filename)

Esta función permite verificar la pertenencia del punto especificado P, a partir de la tupla (xP, yP), compuesta por las coordenadas x e y del punto, respectivamente. Los parámetros **p** , **form** y **parameters** se definen como se ha hecho para las funciones anteriores y

filename: especifica el archivo donde se imprimirán los resultados de la función.

Función ECDP_Generation(p, form, L, filename, requirements, seedcurva=0)

Esta función implementa los algoritmos propuestos en [42] para las formas de Weierstrass y Edwards (torcida de Edwards). Permite generar curvas elípticas con curva torcida cuadrática segura o de forma equivalente que cumple con los mismos requisitos de seguridad que la curva original. Además permite generar curvas elípticas para las cuales la seguridad de la curva torcida cuadrática no es un requisito. Los parámetros **p** , **L** y **form** se definen como se ha hecho anteriormente y:

filename: especifica el archivo donde serán guardados los pdce y datos de validación.

requirements: Vector que especifica los requisitos que deberán comprobar los pdce generados. Estos requisitos corresponden a los pasos (9, 10 12, 13, 14, 15) y (16, 19, 20, 21, 22) de los algoritmos pseudoaleatorios verificables propuestos para las formas de Weierstrass y Edwards (torcida de Edwards) respectivamente, en [42] y a los pasos (8, 9, 11, 12, 13, 14) y (14, 17, 18, 19, 20) de los algoritmos pseudoaleatorios propuestos para las formas de Weierstrass y Edwards (torcida de Edwards) respectivamente de igual forma en [42]. Se puede conformar a partir de las entradas:

```
('Cofactor', number)
('Class_Number', CotFormas, CotClassNumber)
('Weil_Tate', cofactor)
('Discriminant', cota)
('Twist_Secure', twist_security_requirements)
```

según se especificaron para la función:
Función ECDP_security_Verification(p, form, parameters, security_requirements, filename) y de la entrada

```
'req6'
```

que verifica el requisito técnico 6 del cuadro 1.

seedcurva: especifica una semilla para el procedimiento pseudoaleatorio verificable. Por defecto toma el valor cero para el caso pseudoaleatorio.

Las siguientes composiciones del vector **requirements** verifican los pasos (9, 10 12, 13, 14, 15) y (8, 9, 11, 12, 13, 14) de los métodos pseudoaleatorio verificable y pseudoaleatorio para la forma de Weierstrass propuestos en [42].

```
requirements =[('Cofactor', 1), 'req6',
('Twist_Secure', [('Cofactor', 1)]),
('Weil_Tate', 1),('Twist_Secure',
[('Weil_Tate',  1)]), ('Discriminant',
 2^100), ('Class_Number', 20, 10000000)]

requirements =[('Cofactor', 1), 'req6',
('Weil_Tate', 1), ('Discriminant', 2^100),
('Class_Number', 20, 10000000)]
```

La primera opción genera curvas con curva torcida cuadrática segura y en la segunda opción la seguridad de la curva torcida cuadrática no se considera como requisito.

Para el caso de las curvas en forma de Edwards (torcida de Edwards), las siguientes composiciones del vector **requirements** verifican los pasos (16, 19, 20, 21, 22) y (14, 17, 18, 19, 20) de los métodos pseudoaleatorio verificable y pseudoaleatorio para la forma de Edwards (torcida de Edwards) propuestos en [42].

```
requirements =[('Cofactor', 4),
('Twist_Secure', [('Cofactor', 4)]),
 ('Weil_Tate', 4), ('Twist_Secure',
 [('Weil_Tate', 4)]), ('Discriminant',
  2^100),('Class_Number', 20, 10000000)]

requirements =[('Cofactor', 4),
 ('Weil_Tate', 4), ('Discriminant',
  2^100), ('Class_Number', 20,  10000000)]
```

La primera opción genera curvas con curva torcida cuadrática segura y en la segunda opción la seguridad de la curva torcida cuadrática no se considera como requisito.

7. Conclusiones

El paquete ECDP_Generator permite generar curvas elípticas, con buenas propiedades criptográficas, por los métodos pseudoaleatorio y pseudoaleatorio verificable para las formas de Weierstrass, Edwards y torcida de Edwards. Permite verificar los requisitos de seguridad que se discuten en este artículo como resultado del análisis de los requisitos exigidos a las curvas estandarizadas y a las nuevas propuestas de curvas.

El análisis de los requisitos de seguridad para las curvas candidatas posee un elevado costo computacional. Los requisitos más costosos son el análisis de la inmunidad frente a ataques por emparejamientos de Weil y Tate y la verificación de las condiciones del discriminante y el número de clases. Ambos requisitos requieren de la factorización de números grandes, tarea que se hace bien difícil con las implementaciones de algoritmos de factorización implementados en SAGE y en PARI-GP. Otro aspecto importante es el consumo de memoria por el programa SAGE durante el proceso de generación de las curvas.

La condición de la seguridad de la curva torcida cuadrática complejiza el proceso de generación de curvas con buenas propiedades criptográficas. Según [24] la probabilidad de encontrar una curva elíptica segura y con torcida segura está acotada inferiormente por $\frac{0,5}{\log^2 p}$ y superiormente por $\frac{5}{\log^2 p}$. Esta estimación significa que incluir la seguridad de la curva torcida es costosa ya que todos los chequeos necesarios deberán realizarse en un número de curvas elípticas cuadrático en logaritmo de p, lo cual tiene efecto en el tiempo de corrida de los algoritmos y en la necesidad de recursos de memoria RAM.

Se generaron un conjunto de curvas elípticas que podrán ser utilizadas en aplicaciones criptográficas. La implementación de algoritmos que excluyen curvas débiles mediante la verificación de los requisitos de seguridad, permite generar curvas elípticas con buenas propiedades criptográficas y contar con la certeza de que los parámetros no han sido manipulados por terceros. Este procedimiento permite generar pdce siempre que se considere necesario y añadir o modificar las restricciones de seguridad según el ambiente donde sean utilizados los pdce y las posibles apariciones de nuevos ataques.

Referencias

[1] ECC Brainpool. ECC Brainpool Standard Curves and Curve Generation, 2005.

[2] Accredited Standards Committee X9. American national standard X9.63-2001. Public key cryptography for the financial services industry: key agreement and key transport using elliptic curve cryptography, 2001.

[3] Accredited Standards Committee X9.American national standard X9.62-1999. Public key cryptography for the financial services industry: the elliptic curve digital signature algorithm (ECDSA). Technical report, 1999.

[4] ANSI ANSI. X9. 62: 2005: Public key cryptography for the financial services industry. *The elliptic curve digital signature algorithm (ECDSA)*, 2005.

[5] Diego F. Aranha, Paulo S. L. M. Barreto, C. C. F. Pereira Geovandro, and Jefferson E. Ricardini. A note on high-security general-purpose elliptic curves. *IACR Cryptology ePrint Archive*, 2013:647, 2013.

[6] Naomi Benger, Joop van de Pol, Nigel P. Smart, and Yuval Yarom. .ºoh aah... just a little bit": A small amount of side channel can go a long way. In *Cryptographic Hardware and Embedded Systems - CHES 2014 - 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings*, pages 75–92, 2014.

[7] Daniel J. Bernstein. Curve25519: New diffie-hellman speed records. In *Public Key Cryptography - PKC 2006, 9th International Conference on Theory and Practice of Public-Key Cryptography, New York, NY, USA, April 24-26, 2006, Proceedings*, pages 207–228, 2006.

[8] Daniel J. Bernstein, Peter Birkner, Marc Joye, Tanja Lange, and Christiane Peters. Twisted edwards curves. In *Progress in Cryptology - AFRICACRYPT 2008, First International Conference on Cryptology in Africa, Casablanca, Morocco, June 11-14, 2008. Proceedings* [8], pages 389–405.

[9] Daniel J. Bernstein and Tanja Lange. Explicit-formulas database. `http://hyperelliptic.org/EFD/`. Accessed: 2017-03-16.

[10] Daniel J. Bernstein and Tanja Lange. Faster addition and doubling on elliptic curves. *IACR Cryptology ePrint Archive*, 2007:286, 2007.

[11] Daniel J. Bernstein and Tanja Lange. Safecurves: choosing safe curves for elliptic-curve cryptography. http://safecurves.cr.yp.to, 19 January 2014. accedido 23 de Diciembre de 2016.

[12] Joppe W. Bos, Craig Costello, Patrick Longa, and Michael Naehrig. Selecting elliptic curves for cryptography: an efficiency and security analysis. *J. Cryptographic Engineering*, 6(4):259–286, 2016.

[13] Colin Boyd, Paul Montague, and Khanh Nguyen. Elliptic curve based password authenticated key exchange protocols. In *Information Security and Privacy*, pages 487–501. Springer, 2001.

[14] 'Eric Brier and Marc Joye. Fast Point Multiplication on Elliptic Curves Through Isogenies. In *M. Fossorier, T. Høholdt, and A. Poli, Eds., Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, vol. 2643 of Lecture Notes in Computer Science, pp. 43-50, Springer-Verlag*, 2003.

[15] Certicom Research. SEC 1: Elliptic curve cryptography, version 1.0, 2000.

[16] Certicom Research. SEC 2: Recommended elliptic curve domain parameters, version 1.0, 2000.

[17] Certicom Research. SEC 1: Elliptic curve cryptography, version 2.0, 2009.

[18] Certicom Research. Standards for Efficient Cryptography 2 (SEC 2): Recommended Elliptic Curve Domain Parameters. Technical report, Certicom Corp, 2010.

[19] Chae Hoon Lim and Pil Joong Lee. A key recovery attack on discrete log based schemes using a prime order subgroup. In Burton S. Kaliski Jr., editor, *Advances in cryptology crypto 97, 17th annual international cryptology conference, santa barbara, california, usa.*, volume 1294 of *Lecture Notes in Computer Science*, pages 379–392. Springer Berlin, Heidelberg, 1997.

[20] Craig Costello, Patrick Longa, and Michael Naehrig. A brief discussion on selecting new elliptic curves. Technical report, Microsoft Research, 2014.

[21] D. Shumow and N. Ferguson. On the possibility of a back door in the NIST SP800-90 dual ec prng. http://rump2007.cr.yp.to/15-shumow.pdf, 2007.

[22] William Dupuy and Sébastien Kunz-Jacques. Resistance of randomized projective coordinates against power analysis. In *Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings*, pages 1–14, 2005.

[23] Benoit Feix, Mylène Roussellet, and Alexandre Venelli. *Side-Channel Analysis on Blinded Regular Scalar Multiplications*, pages 3–20. Springer International Publishing, Cham, 2014.

[24] Jean-Pierre Flori, Jérôme Plût, Jean-René Reinhard, and Martin Ekerå. Diversity and transparency for ECC. *IACR Cryptology ePrint Archive*, 2015:659, 2015.

[25] Louis Goubin. A refined power-analysis attack on elliptic curve cryptosystems. In *International Workshop on Public Key Cryptography*, pages 199–211. Springer, 2003.

[26] Mike Hamburg. Ed448-goldilocks, a new elliptic curve. *IACR Cryptology ePrint Archive*, 2015:625, 2015.

[27] Ming-Deh A. Huang and Wayne Raskind. Signature calculus and discrete logarithm problems. In *Algorithmic Number Theory, 7th International Symposium, ANTS-VII, Berlin, Germany, July 23-28, 2006, Proceedings*, pages 558–572, 2006.

[28] Burton S. Kaliski Jr. A pseudo-random bit generator based on elliptic logarithms. In *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, volume 263 of *Lecture Notes in Computer Science*, pages 84–103. Springer, 1986.

[29] Burton Stephen Kaliski. *Elliptic curves and cryptography: A pseudorandom bit generator and other tools*. PhD thesis, Massachusetts Institute of Technology, 1988.

[30] Cameron F. Kerry, Acting Secretary, and Charles Romine Director. Fips pub 186-4 federal information processing standards publication digital signature standard (dss), 2013.

[31] A. Langley, M. Hamburg, and S. Turner. Elliptic Curves for Security. RFC 7748, RFC Editor, January 2016.

[32] Manfred Lochter and Johannes Merkle. Elliptic Curve Cryptography (CCE) Brainpool Standard Curves and Curve Generation. RFC 5639 (Informational), 2010.

[33] Manfred Lochter, Johannes Merkle, and Jöne-Marc Schmidt. Requirements for Elliptic Curves for High-Assurance Applications. *Torsten Schütze*, 2015.

[34] Manfred Lochter, Johannes Merkle, Jörn-Marc Schmidt, and Torsten Schütze. Requirements for standard elliptic curves. *IACR Cryptology ePrint Archive*, 2014:832, 2014.

[35] P. L. Montgomery. Speeding the Pollard and elliptic curve methods of factorization. *Mathematics of computation*, 48(177):243–264, 1987.

[36] National Institute of Standards and Technology. FIPS 186-2. Digital Signature Standard. Technical report, NIST, 2000.

[37] Yasuyuki Sakai and Kouichi Sakurai. *Simple Power Analysis on Fast Modular Reduction with NIST Recommended Elliptic Curves*, pages 169–180. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[38] Stale Zerener Haugnæss. On the Generation of Strong Elliptic Curves: For Cryptographic Applications. Master's thesis, University of Oslo, 2015.

[39] The New York Times. Government announces steps to restore confidence on encryption standards. https://bits.blogs.nytimes.com/2013/09/10/government-announces-steps-to-restore-confidence-on-encryption-standards/?_r=1, 10 September 2013.

[40] Joop van de Pol, Nigel P. Smart, and Yuval Yarom. Just a little bit more. In *Topics in Cryptology - CT-RSA 2015, The Cryptographer's Track at the RSA Conference 2015, San Francisco, CA, USA, April 20-24, 2015. Proceedings*, pages 3–21, 2015.

[41] Yasuyuki Sakai and Kouichi Sakurai. Simple power analysis on fast modular reduction with generalized Mersenne prime for elliptic curve cryptosystems. *IEICE Transactions*, 89-A(1):231-237, 2006.

[42] Yessica Caridad Castaño Sainz. Generación de curvas elípticas con buenas propiedades criptográficas sobre campos finitos primos. Master's thesis, Universidad de la Habana, January 2018.