SwiftEC

Shallue-van de Woestijne Indifferentiable Function to Elliptic Curves

Jorge Chavez-Saab
jorge.saab@tii.ae

Coauthors:
Francisco Rodríguez-Henríquez
Mehdi Tibouchi

July 12, 2023

Our contribution

The most efficient constant-time admissible encoding into a large set of ordinary elliptic curves

- A single-squareroot indifferentiable hash function
- A two-squareroot point representation algorithm

Hashing to Elliptic Curves

Many applications require hashing to a cryptographic group (*e.g.* PAKE schemes, signatures and anything involving Fiat-Shamir transform).

For elliptic curve groups, this is not straightforward.

$$E/\mathbb{F}_q : y^2 = x^3 + ax + b$$

How do we get a random $(x, y) \in E(\mathbb{F}_q)$?

Hashing to Elliptic Curves

Naive constructions:

- Hash to some $x \in \mathbb{F}_q$, and restart until $y = \sqrt{x^3 + ax + b}$ exists.
  Not constant time.
- Hash to some $n \in \mathbb{Z}_N$ and output $P = [n]G$ for some generator $G \in E(\mathbb{F}_q)$.
  Leaks the discrete log.

Encodings

The basic idea: start from a hash h to a set S and compose with an encoding $f: S \to E(\mathbb{F}_q)$.

$$S \xrightarrow{f} E(\mathbb{F}_q)$$

$$S \xleftarrow{f^{-1}} E(\mathbb{F}_q)$$

Encodings

The basic idea: start from a hash h to a set S and compose with an encoding $f: S \to E(\mathbb{F}_q)$.

$$S \xrightarrow{\quad f \quad} E(\mathbb{F}_q)$$

`SwiftEC`

$$S \xleftarrow{\quad f^{-1} \quad} E(\mathbb{F}_q)$$

`ElligatorSwift`

Admissible encodings

What do we need for $f(h(x))$ to be a secure hash function?

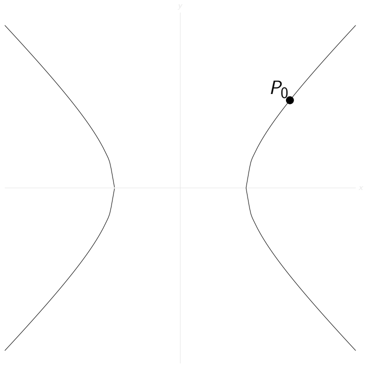
Admissible encoding

The resulting construction is secure if f is admissible [BCIMRT10]:

- **Computable:** $f(x)$ can be evaluated via a deterministic polynomial-time algorithm.
- **Regular:** for $x \in \mathbb{F}_q$ sampled uniformly, the distribution $f(x)$ is statistically indistinguishable from uniform.
- **Samplable:** there exists a PPT algorithm which for any $P \in E(\mathbb{F}_q)$ returns a uniformly random preimage $f^{-1}(P)$.

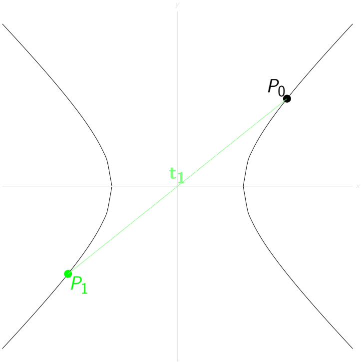
Encoding to a conic

$$C: \; x^2 - y^2 = 1$$



Encoding to a conic

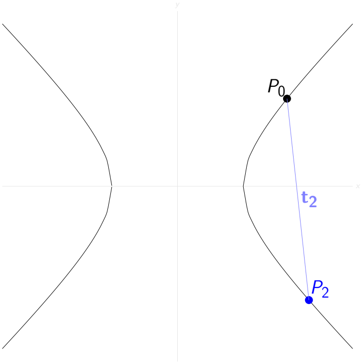
$$C: \; x^2 - y^2 = 1$$



$$P_1 \leftrightarrow t_1$$

Encoding to a conic

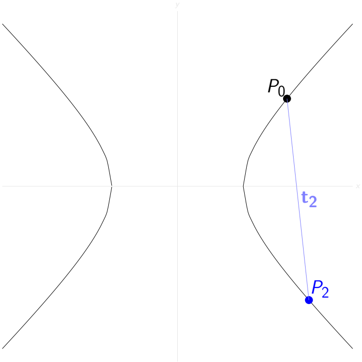
$$C: \; x^2 - y^2 = 1$$



$$P_2 \leftrightarrow t_2$$

Encoding to a conic

$$C : \; x^2 - y^2 = 1$$



This encoding is one-to-one

Encodings to Elliptic Curves

2005
Skałba

2006
SW

2009
Icart

2010
Squared Encoding

2022
Koshelev

2022
SwiftEC

Hashing to elliptic curve has been a problem of great interest for almost two decades.

Encodings to Elliptic Curves

2005 Skałba
2006 SW
2009 Icart
2010 Squared Encoding
2022 Koshelev
2022 SwiftEC

Given $E/\mathbb{F}_q : y^2 = x^3 + ax + b := g(x)$ (with $a \neq 0$), Skałba [Ska05] found a rational function $\Psi : \mathbb{F}_q \to V$, where

$$V = \{(x_1, x_2, x_3, z) \in \mathbb{F}_q^4 \mid g(x_1)g(x_2)g(x_3) = z^2\},$$

meaning at least one of the x_i is the x-coordinate of a point in E.

$$f : \mathbb{F}_q \xrightarrow{\Psi} V \xrightarrow{\text{select square}} E(\mathbb{F}_q)$$

Involves degrees up to 26!

Encodings to Elliptic Curves

2005 Skałba
2006 SW
2009 Icart
2010 Squared Encoding
2022 Koshelev
2022 SwiftEC

For some $u \in \mathbb{F}_q$, Shallue-van de Woestijne [SW06] found a one-to-one ivertible function $\Psi_u : \ C_u \to V$, where

$$C_u : X^2 + (3u^2 + 4a)Y^2 = -g(u),$$

given by

$$x_1 = \frac{X}{2Y} - \frac{u}{2} \qquad x_2 = -\frac{X}{2Y} - \frac{u}{2} \qquad x_3 = u + 4Y^2$$

$$z = \frac{g(u+Y^2)}{Y} \cdot \left(u^2 + u\left(\frac{X}{2Y} - \frac{u}{2}\right) + \left(\frac{X}{2Y} - \frac{u}{2}\right)^2 + a \right)$$

Much simpler formulas.

Encodings to Elliptic Curves

2005 Skałba

2006 SW

2009 Icart

2010 Squared Encoding

2022 Koshelev

2022 SwiftEC

We already know how to encode C_u! (given a fixed point P_u)

$$f_u : \mathbb{F} \xrightarrow{\text{conic encoding}} C_u(\mathbb{F}_q) \xrightarrow{\Psi_u} V(\mathbb{F}_q) \xrightarrow{\text{select square}} E(\mathbb{F}_q)$$

✓Family of encodings (one for each u)

✓Main cost is one square-root (for recovering the y-coordinate)

✓Works for almost all elliptic curves and almost all u

✗Not regular: only fills $\approx 1/8$ of the curve

Encodings to Elliptic Curves

2005 Skałba

2006 SW

2009 Icart

2010 Squared Encoding

2022 Koshelev

2022 SwiftEC

Icart's Encoding [Ica09]: $f : t \mapsto (x, y)$ with

$$x = \left(v^2 - b - \frac{t^6}{27}\right)^{1/3} + \frac{t^2}{3} \qquad y = tx + v \qquad v = \frac{4a - t^4}{6t}$$

✓Deterministic, closed-form

✗Requires $q \equiv 2 \mod 3$

✗Expensive cubic root

✗Not regular: only fills $\approx 5/8$ of the curve

Encodings to Elliptic Curves

2005 Skałba
2006 SW
2009 Icart
2010 Squared Encoding
2022 Koshelev
2022 SwiftEC

The "Squared encoding" [BCIMRT10]:

$$F(t_1, t_2) = f(t_1) + f(t_2),$$

where f is Icart's encoding, is regular.

Generalized to most encodings by [FFSTV13].

✓Provides an admissible encoding for any curve
✗Domain is twice as large (bad for point representation)
✗Requires two evaluations of f (two square-roots)

Can one obtain an admissible encoding with a single square-root?

Encodings to Elliptic Curves

2005 Skałba

2006 SW

2009 Icart

2010 Squared Encoding

2022 Koshelev

2022 SwiftEC

Koshelev [Kos22] obtained a single-square-root admissible encoding in some constrained scenarios.

✓Works with some BLS curves (pairings)
✗Requires j-invariant equal to 0 and specific conditions for the base field.

Encodings to Elliptic Curves

2005 Skałba

2006 SW

2009 Icart

2010 Squared Encoding

2022 Koshelev

2022 SwiftEC

`SwiftEC`: a single-square-root admissible function that works with most elliptic curves as long as $q \equiv 1 \mod 3$.

Using the SW family of encodings f_u, we define

$$F(u, t) = f_u(t).$$

Problems that arise

Computability: Is there an efficient algorithm to compute $P_u \in C_u$ on input u?

Regularity: Need a proof that the distribution of $F(u, t)$ is close to uniform

Samplability: Need an "inverse" function to recover a pair (u, t) uniformly at random (`ElligatorSwift`)

Computability of SwiftEC

Encoding to the conic C_u requires knowing a fixed point P_u
Now it must be computed on the go.

Theorem 1 (van Hoeij-Cremona [HC06])

The parametrized projective conic

$$C_u : \; X^2 + h(u)Y^2 + g(u)Z^2 = 0$$

admits a rational point $X(u), Y(u), Z(u)$ iff:

1. $-h$ is a square in $\mathbb{F}_q[u]/(g)$
2. $-g$ is a square in $\mathbb{F}_q[u]/(h)$

Computability of SwiftEC

In our case, $h(u) = 3u^2 + 4a$ and $-g(u) = u^3 + au + b$.

Theorem 2 (this work)

The conditions for Theorem 1 are equivalent to:

1. $q \equiv 1 \mod 3$
2. The discriminant $\Delta_E := -16(4a^3 + 27b^2)$ is a square in $\mathbb{F}_q$
3. At least one of $\nu_\pm := \frac{1}{2}(-b \pm \sqrt{-3\Delta_E}/36)$ is a square

Computability of SwiftEC

- Compatible curves: `P256`, `secp256k1`, as well as all BN and BLS curves as long as $q \equiv 1 \mod 3$.
- Other curves can be rescued by composing with a small isogeny:
  - `Curve25519` has non-square Δ_E, but there is a compatible 2-isogenous curve
  - `P521` has non-square $\nu_\pm$, but there is a compatible 3-isogenous curve
- Curves with $q \not\equiv 1 \mod 3$ cannot be rescued (`P384`, `Ed448-Goldilocks`)

Regularity of SwiftEC

For the distribution to be close to uniform, we want

$$\#F^{-1}(x) \approx \frac{\#\text{Domain}}{\#\text{Codomain}} = \frac{q^2}{\#E(\mathbb{F}_q)/2} \approx 2q$$

for each x.

Theorem 2

The map $F(u,t) = f_u(t)$ is regular in the sense that

$$\frac{1}{2} \sum_{(x,y)\in E(\mathbb{F}_q)} \left| \frac{\#F^{-1}(x)}{q^2} - \frac{1}{\#E(\mathbb{F}_q)/2} \right| < \epsilon$$

for

$$\epsilon = (6 + o(1))q^{-1/2}$$

Sketch of the proof

Recall the SW map

$$x_1 = \frac{X}{2Y} - \frac{u}{2} \qquad x_2 = -\frac{X}{2Y} - \frac{u}{2} \qquad x_3 = u + 4Y^2 \qquad C_u :\ X^2 + h(u)Y^2 = g(u)$$

For fixed $\bar{x}$, define $C_i(\bar{x})$ the set of points (u, X, Y) such that $x_i = \bar{x}$

1. Except for a few $\bar{x}$, $C_i(\bar{x})$ are hyperelliptic curves of genus 2, so $\#C_i(\bar{x}) \approx q$ by the Hasse-Weil bound
2. The number of points where all three x_i are squares is $N(\bar{x}) \approx q/2$ (Perret bound on character sums [Per91])
3. $$\begin{aligned}\#F^{-1}(\bar{x}) &= \#C_1(\bar{x}) + (\#C_2(\bar{x}) - N(\bar{x})) + (\#C_3(\bar{x}) - N(\bar{x})) \\ &\approx q + (q/2) + (q/2) = 2q\end{aligned}$$

Samplability of SwiftEC

We also introduce the ElligatorSwift algorithm which samples a random preimage $(u, t) \in F^{-1}(x)$.

Recall

$$x_1 = \frac{X}{2Y} - \frac{u}{2} \qquad x_2 = -\frac{X}{2Y} - \frac{u}{2} \qquad x_3 = u + 4Y^2$$

1. Pick random $u \in \mathbb{F}_q$ and $i \in \{1, 2, 3\}$
2. Try to invert the map x_i to recover X, Y (restarting if unable)
3. If all $g(x_i)$ are squares and $i \neq 1$, restart
4. Invert the parametrization of C_u to recover t

$$x \xmapsto{\text{random } u,i} (X, Y) \xmapsto{\text{conic encode}} t$$

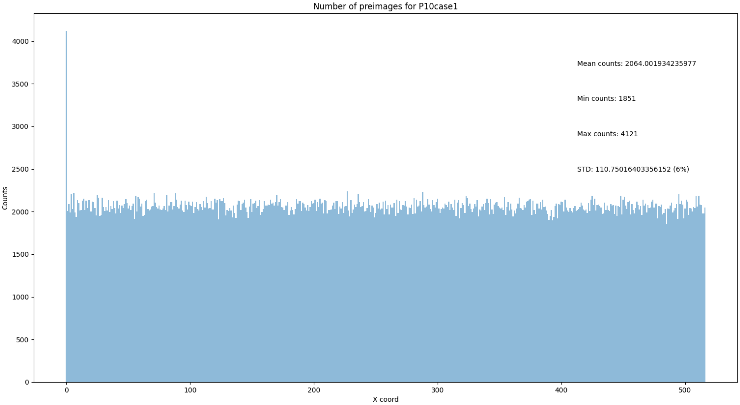
Implementation

We have implemented both `SwiftEC` and `ElligatorSwift` in `Sage`[1].

	Add	Sqr	Mul	Jac	Inv	Sqrt
SwiftEC	25	7	18	2	1	1
X-only proj. SwiftEC	22	9	23	2	0	0

[1] https://github.com/Jchavezsaab/SwiftEC

Preimage Distribution



Benchmarking

Preliminary results from a C implementation for `secp256k1`

	ElligatorSquared	SwiftEC
Encode	49.2 μs	22.1 μs
Decode	14.7 μs	6.9 μs

Conclusions

- SwiftEC is now the fastest known algorithm for hashing into most elliptic curves
- ElligatorSwift is the fastest known algorithm for elliptic curve point representation
- Both improved on the previous state-of-the-art with more than double the performance

Future work:

- Handle the case $q \equiv 2 \mod 3$ and a few other exceptions.

References

[Per91] Marc Perret. "Multiplicative character sums and Kummer coverings". In: *Acta Arith.* 59 (1991), pp. 279–290.

[Ska05] M. Skałba. "Points on elliptic curves over finite fields". eng. In: *Acta Arithmetica* 117.3 (2005), pp. 293–301.

[HC06] Mark van Hoeij and John Cremona. "Solving conics over function fields". In: *Journal de Théorie des Nombres de Bordeaux* 18.3 (2006), pp. 595–606.

[SW06] Andrew Shallue and Christiaan E. van de Woestijne. "Construction of Rational Points on Elliptic Curves over Finite Fields". In: *Algorithmic Number Theory, 7th International Symposium, ANTS-VII*. Ed. by Florian Hess, Sebastian Pauli, and Michael E. Pohst. Vol. 4076. Lecture Notes in Computer Science. Springer, 2006, pp. 510–524.

[Ica09] Thomas Icart. "How to Hash into Elliptic Curves". In: 2009, pp. 303–316.

[BCIMRT10] Eric Brier et al. "Efficient Indifferentiable Hashing into Ordinary Elliptic Curves". In: *Advances in Cryptology – CRYPTO 2010*. Ed. by Tal Rabin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 237–254.

[FFSTV13] Reza Rezaeian Farashahi et al. "Indifferentiable deterministic hashing to elliptic and hyperelliptic curves". In: *Math. Comput.* 82.281 (2013), pp. 491–512.

[Kos22] Dmitrii Koshelev. "Indifferentiable hashing to ordinary elliptic $\mathbb{F}_q$-curves of $j = 0$ with the cost of one exponentiation in $\mathbb{F}_q$". In: *Des. Codes Cryptogr.* 90.3 (2022), pp. 801–812.