SwiftEC

Shallue-van de Woestijne Indifferentiable Function to Elliptic Curves

December 6, 2022

Jorge Chavez-Saab[1,2] Francisco Rodríguez-Henríquez[1,2] Mehdi Tibouchi[3]

[1]Cinvestav, Mexico [2]TII, UAE [3]NTT, Japan

Introduction

Our contribution

Cryptography Research Centre

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 $P = nG$ 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 $f : S \to E(\mathbb{F}_q)$.

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

`SwiftEC`

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

`ElligatorSwift`

Admissible encodings

Cryptography Research Centre

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

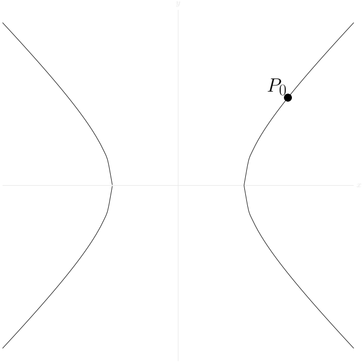
Admissible encoding

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 $f^{-1}(P)$.

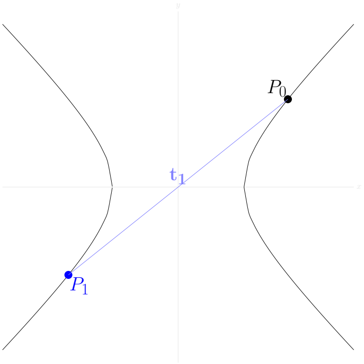
Encoding to a conic

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



Encoding to a conic

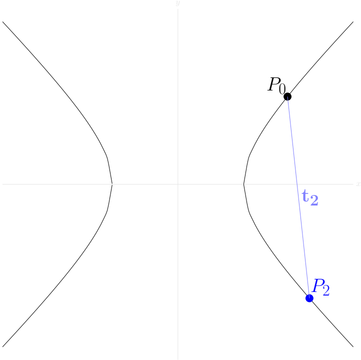
$$C : x^2 - y^2 = 1$$



$$P_1 \leftrightarrow t_1$$

Encoding to a conic

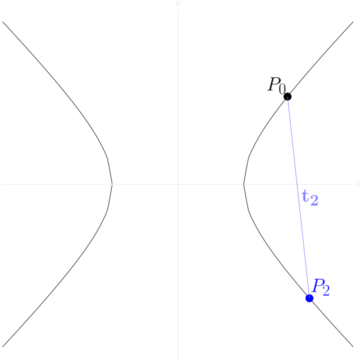
$$C : x^2 - y^2 = 1$$



$$P_2 \leftrightarrow t_2$$

Encoding to a conic

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



This encoding is admissible and one-to-one

Encoding to Elliptic Curves

Encoding to Elliptic Curves

Cryptography Research Centre

Shallue-van de Woestijne Encoding [SW06]

Given $E : y^2 = x^3 + ax + b := g(x)$, and some $u \in \mathbb{F}_q$, we can find a map $\Psi_u : C_u \to V$ where

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

$$V : z^2 = g(x_1)g(x_2)g(x_3)$$

given by

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

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

Encoding to Elliptic Curves

Cryptography Research Centre

Shallue-van de Woestijne Encoding

$\Psi_u : C_u \to V$ where

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

$$V : z^2 = g(x_1)g(x_2)g(x_3)$$

- We know how to encode to C_u (given a fixed point P_u)
- Either one or all of $g(x_i)$ are squares
  - Test quadratic residuocity of each
  - Choose $x = x_1$ when all three are squares (arbitrary)
  - Compute $y = \sqrt{g(x)}$ from scratch

Shallue-van de Woestijne Encoding

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

✓Simple formulas, constant time

✓Main cost is one square-root (for computing y, if needed)

✓Works for almost all elliptic curves and almost all u

✗Still not regular

Squared Encoding

The Squared Encoding [BCIMRT10] construction:

$$F_u(t_1, t_2) = f_u(t_1) + f_u(t_2)$$

is regular.

✓This is an admissible encoding for almost every curve
✗Requires two evaluations of f_u (two square-roots)

SwiftEC

Our construction:

Rather than fixing u, consider

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

Over the full $(\mathbb{F}_q)^2$ domain, this encoding is admissible and requires only one square-root.

Computability of SwiftEC

Cryptography Research Centre

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

Cryptography Research Centre

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

Cryptography Research Centre

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 3 (this work)

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

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

Samplability of SwiftEC

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

Recall

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

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

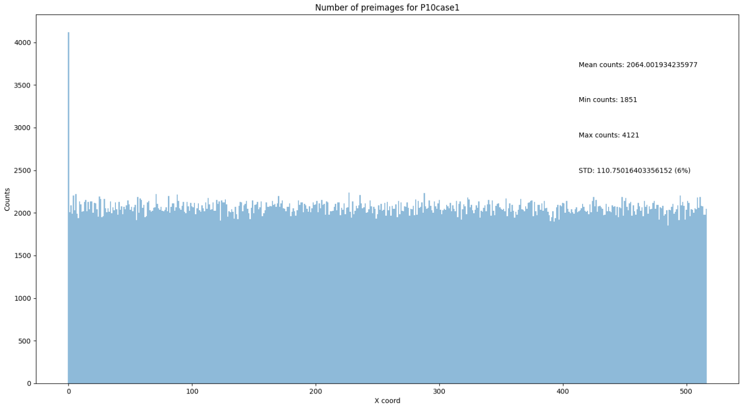
Implementation

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



Conclusion

Conclusions

- SwiftEC is now the most efficient known generic algorithm for constant-time indifferentiable hashing into most ordinary elliptic curves
- ElligatorSwift is the most efficient generic algorithm for point representation of those curves
- Both improved on the previous state-of-the-art with more than double the performance

Future work:

- Efficient C implementation
- Further increase the number of compatible curves

References

[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.

[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.

Thank you!