How to Hash into Elliptic Curves

Thomas Icart

thomas.icart@m4x.org

18/08/2009

Introduction

- Hashing into elliptic curves is needed:
  1. In the IBE scheme of Boneh-Franklin (2001).
  2. In some Password Based protocols over elliptic curves.

Introduction

- Hashing into elliptic curves is needed:
  1. In the IBE scheme of Boneh-Franklin (2001).
  2. In some Password Based protocols over elliptic curves.
- Boneh-Franklin uses a particular super-singular curve on which hashing is easy

Introduction

- Hashing into elliptic curves is needed:
  1. In the IBE scheme of Boneh-Franklin (2001).
  2. In some Password Based protocols over elliptic curves.
- Boneh-Franklin uses a particular super-singular curve on which hashing is easy
- Efficient password based protocols such as the Simple Password Exponential Key Exchange (SPEKE) [Jab 1996] need hash function into ordinary curves.

Introduction

Definition (Notations)

An elliptic curve $E_{a,b}$ is the set of points verifying the equation:

$$X^3 + aX + b = Y^2$$

over a field $\mathbb{F}_p$. The number of points in $E_{a,b}$ is N.

Hashing into Finite Fields

- Hashing into finite field in deterministic polynomial time is easy.

Hashing into Finite Fields

- Hashing into finite field in deterministic polynomial time is easy.

Lemma

- *Let p be a safe prime ($p = 2q + 1$).*
- *Let H be a $|p|$-bit* **one-way** *hash function*

Hashing into Finite Fields

- Hashing into finite field in deterministic polynomial time is easy.

Lemma

- *Let p be a safe prime ($p = 2q + 1$).*
- *Let H be a $|p|$-bit* **one-way** *hash function*
- *Then $H(m)^2 \mod p$ is a* **one-way** *hash function into the prime order subgroup of $\mathbb{F}_p$.*

Hashing into Elliptic Curves

- Hashing into elliptic curves in deterministic polynomial time is much harder.

Hashing into Elliptic Curves

- Hashing into elliptic curves in deterministic polynomial time is much harder.
- It requires a deterministic function from the base field to $E_{a,b}$
- The classical point generation algorithm is not deterministic.

1. Related Works
   - Try and Increment
   - Supersingular
   - SW
   - Wanted

2. Proposal
   - Definition
   - Idea
   - Properties

3. Hashing
   - Preimage
   - Collision

Try and Increment Algorithm

Input: u an integer.
Output: Q, a point of $E_{a,b}(\mathbb{F}_p)$.

1. For $i = 0$ to $k - 1$
   1. Set $x = u + i$
   2. If $x^3 + ax + b$ is a quadratic residue in $\mathbb{F}_p$, then return $Q = (x, (x^3 + ax + b)^{1/2})$
2. end For
3. Return $\perp$

Try and Increment Algorithm

Input: u an integer.
Output: Q, a point of $E_{a,b}(\mathbb{F}_p)$.

1. For $i = 0$ to $k - 1$
    1. Set $x = u + i$
    2. If $x^3 + ax + b$ is a quadratic residue in $\mathbb{F}_p$, then return $Q = (x, (x^3 + ax + b)^{1/2})$
2. end For
3. Return $\perp$

The running time depends on u. This leads to partition attacks [BMN 2001].

Partition Attacks

- When u is related to the password π, different passwords lead to different running times T.

Partition Attacks

- When u is related to the password π, different passwords lead to different running times T.
- Example: $u = H(\pi, PK_C, PK_R)$ in SPEKE.

Partition Attacks

- When u is related to the password π, different passwords lead to different running times T.
- Example: $u = H(\pi, PK_C, PK_R)$ in SPEKE.
- A partition of the password dictionary is possible following the different T.

Possible solutions

Making the Try and Increment algorithm constant time:

Possible solutions

Making the Try and Increment algorithm constant time:
Input: u an integer.
Output: Q, a point of $E_{a,b}(\mathbb{F}_p)$.

1. For $i = 0$ to $k - 1$
   1. Set $x = u + i$
   2. If $x^3 + ax + b$ is a quadratic residue in $\mathbb{F}_p$, then store $Q = (x, (x^3 + ax + b)^{1/2})$
2. end For
3. Return Q

Possible solutions

Making the Try and Increment algorithm constant time:
Input: u an integer.
Output: Q, a point of $E_{a,b}(\mathbb{F}_p)$.

1. For $i = 0$ to $k - 1$
    1. Set $x = u + i$
    2. If $x^3 + ax + b$ is a quadratic residue in $\mathbb{F}_p$, then store $Q = (x, (x^3 + ax + b)^{1/2})$
2. end For
3. Return Q

The running time is $\mathcal{O}(\log^3 p)$ in general. When using exponentiation for testing quadratic residuosity, running time in $\mathcal{O}(\log^4 p)$.

Supersingular Elliptic Curve

Definition

A curve $E_{0,b}$:

$$X^3 + b = Y^2 \mod p$$

with $p = 2 \mod 3$ has $p + 1$ points and is supersingular.

Supersingular Elliptic Curve

Definition

A curve $E_{0,b}$:

$$X^3 + b = Y^2 \mod p$$

with $p = 2 \mod 3$ has $p+1$ points and is supersingular.

- The function $u \mapsto ((u^2 - b)^{1/3 \mod p-1}, u)$ is a bijection from $\mathbb{F}_p$ to $E_{0,b}$.

Supersingular Elliptic Curve

Definition

A curve $E_{0,b}$:

$$X^3 + b = Y^2 \mod p$$

with $p = 2 \mod 3$ has $p+1$ points and is supersingular.

- The function $u \mapsto ((u^2 - b)^{1/3 \mod p-1}, u)$ is a bijection from $\mathbb{F}_p$ to $E_{0,b}$.
- Because of the MOV attacks, larger p should be used (512 bits instead of 160 bits).

Possible solutions

Previous work:

- Shallue-Woestijne's deterministic algorithm for generating EC points.
- Our algorithm is different, simpler and is an explicit function.

Andrew Shallue and Christiaan van de Woestijne: *Construction of Rational Points on Elliptic Curves over Finite Fields.* ANTS 2006

What do we want?

A function f with the following properties:

- It only requires the elliptic curves parameters,

What do we want?

A function f with the following properties:

- It only requires the elliptic curves parameters,
- f requires a constant number of finite field operations (exponentiations, multiplications, additions)

What do we want?

A function f with the following properties:

- It only requires the elliptic curves parameters,
- f requires a constant number of finite field operations (exponentiations, multiplications, additions)
- f^{-1} can be computed in polynomial time. This ensures that computing the **discrete logarithm of** $f(x)$ **is hard for any** x.

What do we want?

A function f with the following properties:

- It only requires the elliptic curves parameters,
- f requires a constant number of finite field operations (exponentiations, multiplications, additions)
- f^{-1} can be computed in polynomial time. This ensures that computing the **discrete logarithm of** $f(x)$ **is hard for any** x.

The New Function

Fact

- *Over fields such that* $p = 2 \mod 3$, *the map* $x \mapsto x^3$ *is a bijection.*
- *In particular:* $x^{1/3} = x^{(2p-1)/3}$.
- *This operation can be computed in a constant numbers of operations for a constant* p.

The New Function

Definition

$$\begin{array}{rcl} f_{a,b} : \mathbb{F}_p & \mapsto & (\mathbb{F}_p)^2 \cup \{\mathcal{O}\} \\ u & \mapsto & (x, y = ux + v) \end{array}$$

$$\begin{array}{rcl} x & = & \left(v^2 - b - \dfrac{u^6}{27}\right)^{1/3} + \dfrac{u^2}{3} \\ y & = & ux + v \\ v & = & \dfrac{3a - u^4}{6u} \end{array}$$

The idea

Fact

When $p = 2 \mod 3$*, degree* 3 *polynomials* $(x - \alpha)^3 - \beta$ *have a unique root:* $\beta^{1/3} + \alpha$

The idea

Fact

When $p = 2 \mod 3$, *degree* 3 *polynomials* $(x-\alpha)^3 - \beta$ *have a unique root:* $\beta^{1/3} + \alpha$

- Idea: Assume that $y = ux + v$, find $v(u)$ such that:

$$x^3 + ax + b - (ux + v(u))^2 = (x - \alpha(u))^3 - \beta(u)$$

The idea

From the elliptic curve equation and $y = ux + v$:

$$x^3 + ax + b = u^2x^2 + 2uvx + v^2 \quad = \quad (ux + v)^2$$

The idea

From the elliptic curve equation and $y = ux + v$:

$$
\begin{aligned}
x^3 + ax + b = u^2x^2 + 2uvx + v^2 &= (ux + v)^2 \\
x^3 - u^2x^2 + (a - 2uv)x + b - v^2 &= 0
\end{aligned}
$$

The idea

From the elliptic curve equation and $y = ux + v$:

$$\begin{aligned} x^3 + ax + b = u^2x^2 + 2uvx + v^2 &= (ux+v)^2 \\ x^3 - u^2x^2 + (a - 2uv)x + b - v^2 &= 0 \\ \left(x - \frac{u^2}{3}\right)^3 + x\left(a - 2uv - \frac{u^4}{3}\right) &= v^2 - b - \frac{u^6}{27} \end{aligned}$$

The idea

$$\left(x - \frac{u^2}{3}\right)^3 + x\left(a - 2uv - \frac{u^4}{3}\right) = v^2 - b - \frac{u^6}{27}$$

Let

$$v = \frac{3a - u^4}{6u}$$

The idea

$$\left(x - \frac{u^2}{3}\right)^3 + x\left(a - 2uv - \frac{u^4}{3}\right) = v^2 - b - \frac{u^6}{27}$$

Let

$$v = \frac{3a - u^4}{6u}$$

This implies:

$$\left(x - \frac{u^2}{3}\right)^3 = v^2 - b - \frac{u^6}{27}$$

Therefore, we can recover x and $y = ux + v$

Properties

Let $P = (x, y)$ be a point on the curve $E_{a,b}$.

Lemma

The solutions u_s of $f_{a,b}(u_s) = P$ are the solutions of the equation:

$$u^4 - 6u^2x + 6uy - 3a = 0.$$

Properties

Let $P = (x, y)$ be a point on the curve $E_{a,b}$.

Lemma

The solutions u_s of $f_{a,b}(u_s) = P$ are the solutions of the equation:

$$u^4 - 6u^2x + 6uy - 3a = 0.$$

This implies that:

1. $f_{a,b}^{-1}(P)$ is computable in polynomial time,
2. $\left|f_{a,b}^{-1}(P)\right| \leq 4$, for all $P \in E_{a,b}$
3. $\left|\mathrm{Im}\left(f_{a,b}\right)\right| > p/4$

Properties

- $|\mathrm{Im}\,(f_{a,b})| > p/4$

Conjecture

There exists a constant λ such that for any p, a, b

$$\left| |\mathrm{Im}(f_{a,b})| - \frac{5}{8}\,|E_{a,b}(\mathbb{F}_p)| \right| \leq \lambda\sqrt{p}$$

Properties

- $|\mathrm{Im}\,(f_{a,b})| > p/4$

Conjecture

There exists a constant λ such that for any p, a, b

$$\left| |\mathrm{Im}(f_{a,b})| - \frac{5}{8}\,|E_{a,b}(\mathbb{F}_p)| \right| \leq \lambda\sqrt{p}$$

This enables to prove that $(u_1, u_2) \mapsto f_{a,b}(u_1) + f_{a,b}(u_2)$ is a surjective function.

1. Related Works
   - Try and Increment
   - Supersingular
   - SW
   - Wanted

2. Proposal
   - Definition
   - Idea
   - Properties

3. Hashing
   - Preimage
   - Collision

Hashing into Elliptic Curves

We here focus on standard properties for hash functions:

- Resistance against Preimage Attacks
- Resistance against Collision Attacks

Preimage Resistance

Lemma

If h is a one-way hash function then $H(m) = f_{a,b}(h(m))$ is a one-way hash function into elliptic curves.

Preimage Resistance

Lemma

If h is a one-way hash function then $H(m) = f_{a,b}(h(m))$ is a one-way hash function into elliptic curves.

Idea:

1. $f_{a,b}$ is invertible
2. Its preimage size is at most 4

Collision Resistance

Fact

A collision to $H(m) = f_{a,b}(h(m))$ *is either:*

1. *A collision to* h*:* m *and* m' *such that* $h(m) = h(m')$
2. *A collision to* $f_{a,b}$*:* m *and* m' *such that* $h(m) \neq h(m')$ *and* $f_{a,b}(h(m)) = f_{a,b}(h(m'))$

Collision Resistance

Fact

A collision to $H(m) = f_{a,b}(h(m))$ *is either:*

1. *A collision to* h*:* m *and* m' *such that* $h(m) = h(m')$
2. *A collision to* $f_{a,b}$*:* m *and* m' *such that* $h(m) \neq h(m')$ *and* $f_{a,b}(h(m)) = f_{a,b}(h(m'))$

- We did not find a way to prove the collision resistance of $f_{a,b}(h)$ from the collision resistance of h

Collision Resistance

Fact

A collision to $H(m) = f_{a,b}(h(m))$ is either:

1. *A collision to h: m and m' such that $h(m) = h(m')$*
2. *A collision to $f_{a,b}$: m and m' such that $h(m) \neq h(m')$ and $f_{a,b}(h(m)) = f_{a,b}(h(m'))$*

- We did not find a way to prove the collision resistance of $f_{a,b}(h)$ from the collision resistance of h
- We thus propose a 2^{nd} construction.

Collision Resistance

- **Heuristically**, for sufficiently small value of u, $f_{a,b}(u)$ is collision free.

Collision Resistance

- **Heuristically**, for sufficiently small value of u, $f_{a,b}(u)$ is collision free.
- We use pair-wise independent functions to get a **probabilistic** result (i.e. a non-heuristic one). [CW 1981]

Collision Resistance

- **Heuristically**, for sufficiently small value of u, $f_{a,b}(u)$ is collision free.
- We use pair-wise independent functions to get a **probabilistic** result (i.e. a non-heuristic one). [CW 1981]

Definition (Pair-wise Independent Function)

A family of functions $g : \mathbb{F}_p \mapsto \mathbb{F}_p$ is pair-wise independent if given any couple (x_1, x_2) with $x_1 \neq x_2$ and any couple (u_1, u_2), $\Pr_g [g(x_1) = u_1 \wedge g(x_2) = u_2]$ is negligible.

- The affine functions $x \mapsto c.x + d$ for $(c, d) \in (\mathbb{F}_p \times \mathbb{F}_p)$ are pair-wise independent functions

- The affine functions $x \mapsto c.x + d$ for $(c, d) \in (\mathbb{F}_p \times \mathbb{F}_p)$ are pair-wise independent functions
- For **sufficiently small value** of x, $f_{a,b}(c.x + d)$ is collision free with a very high probability.

- The affine functions $x \mapsto c.x + d$ for $(c, d) \in (\mathbb{F}_p \times \mathbb{F}_p)$ are pair-wise independent functions
- For **sufficiently small value** of x, $f_{a,b}(c.x + d)$ is collision free with a very high probability.

Lemma

For a random choice of c, d, the function $m \mapsto f_{a,b}(c.h(m) + d)$ is collision resistant with a high probability for a good choice of size parameter assuming that h is collision resistant.

- The affine functions $x \mapsto c.x + d$ for $(c, d) \in (\mathbb{F}_p \times \mathbb{F}_p)$ are pair-wise independent functions
- For **sufficiently small value** of x, $f_{a,b}(c.x + d)$ is collision free with a very high probability.

Lemma

For a random choice of c, d, the function $m \mapsto f_{a,b}(c.h(m) + d)$ is collision resistant with a high probability for a good choice of size parameter assuming that h is collision resistant.

- If $h(m)$ is a 160-bit hash function, $f_{a,b}(c.h(m) + d)$ is collision resistant if p is a 400-bit integer.

Conclusion

- $f_{a,b}$ enables to deterministically generate points into elliptic curves.

Conclusion

- $f_{a,b}$ enables to deterministically generate points into elliptic curves.
- $f_{a,b}$ exists in characteristic 2.

Conclusion

- $f_{a,b}$ enables to deterministically generate points into elliptic curves.
- $f_{a,b}$ exists in characteristic 2.
- When the cofactor $r \neq 1$, $r.f_{a,b}$ can be used to hash into the subgroup of the curves.

Conclusion

- $f_{a,b}$ enables to deterministically generate points into elliptic curves.
- $f_{a,b}$ exists in characteristic 2.
- When the cofactor $r \neq 1$, $r.f_{a,b}$ can be used to hash into the subgroup of the curves.
- $f_{a,b}$ is based on cube root extraction: over RSA rings, generating a point into elliptic curves only requires a cube root oracle.

Conclusion

- $f_{a,b}$ enables to deterministically generate points into elliptic curves.
- $f_{a,b}$ exists in characteristic 2.
- When the cofactor $r \neq 1$, $r.f_{a,b}$ can be used to hash into the subgroup of the curves.
- $f_{a,b}$ is based on cube root extraction: over RSA rings, generating a point into elliptic curves only requires a cube root oracle.
- $f_{a,b}$ can be used on any curve model (Edwards Curve, etc) whenever the model is birationally equivalent to the Weierstrass model.

Thank You

Thank You

Questions?