Deterministic Encoding and Hashing to Odd Hyperelliptic Curves

Pierre-Alain Fouque Mehdi Tibouchi

École normale supérieure

Pairing 2010, 2010-12-14

Outline

Outline

Hashing and encoding to (hyper)elliptic curves

- Many cryptographic protocols (schemes for encryption, signature, PAKE, IBE, etc.) involve representing a certain numeric value as an element of the group $\mathbb{G}$ where the computations occur.
- Two distinct settings.

  Injective encoding: One must be able to retrieve the original numeric value from the group element (e.g. for encryption);

  Hashing: The original value doesn't have to be recovered, but the function should "look like" a random oracle (e.g. for signature).
- For $\mathbb{G} = \mathbb{Z}_p^*$, taking the numeric value itself is a good injective encoding, and reducing a bit-string valued hash function of appropriate size $\bmod p$ provides a good way to hash.
- However, if $\mathbb{G}$ is an elliptic curve group, these techniques have no obvious counterpart; e.g. one cannot put a value in the x-coordinate of a curve point, because only about $1/2$ of possible x-values correspond to actual points. Same problem in higher genus.

The traditional solution for elliptic curves

- For k bits of security:
  1. concatenate the numeric value or hash value with a counter from 0 to $k-1$;
  2. initialize the counter as 0;
  3. if the concatenated value is a valid x-coordinate on the curve, i.e. $x^3 + ax + b$ is a square in the base field, return one of the two corresponding points; otherwise increment the counter and try again.
- Heuristically, the probability of a concatenated value being valid is $1/2$, so k iterations ensure k bits of security.

Problems with this solution

- A natural implementation does not run in constant time: possible timing attacks (especially for PAKE).
- A constant time implementation (always do k steps, compute the Legendre symbol in constant time) is very inefficient, $O(n^4)$.
- Security is difficult to analyze.

Remark: hashing as $H(m) = h(m)G$ where G is a generator of the group is *not* a good idea.

Outline

Supersingular elliptic curves

An optimal solution to both problems was given in Boneh and Franklin's IBE paper for the following supersingular curves:

$$y^2 = x^3 + b$$

over a field with q elements, with $q \equiv 2 \pmod 3$.
Such a curve admits an (almost) bijective deterministic encoding:

$$f : u \mapsto \left((u^2 - b)^{1/3}, u\right)$$

which solves both problems at once.
Convenient for pairing-based protocols at a moderate security level, but higher security (or non-pairing-based settings) call for ordinary curves. Or hyperelliptic curves?

Beyond the supersingular case?

[SW06] First deterministic "point construction algorithm" on most ordinary elliptic curves. One can deduce a deterministic function $F : \mathbb{F}_q \to E(\mathbb{F}_q)$ with large image.

[Ul07] Extension to hyperelliptic curves of the form $y^2 = x^{2g+1} + ax + b$.

[Ic09] Alternate construction for elliptic curves when $q \equiv 2 \pmod 3$.

[FSV10], [FT10] The image size of all such functions is about $c \cdot q$ for some constant $0 < c < 1$ that doesn't depend on the curve. In particular, neither surjective nor injective: neither encoding nor hashing?

Beyond the supersingular case?

[BCIMRT10] For hashing, we really need *indifferentiability from a random oracle*. States some sufficient conditions to achieve it, and gives an efficient construction based on Icart's function.

[KLR10] Generalizes Icart's method to many elliptic and hyperelliptic curves.

[FFSTV] Method to obtain indifferentiable hashing from all known point construction algorithms.

Getting a good grasp on hashing, although even some important elliptic curves (e.g. BN curves) are still missing, and the constructions still have an ad-hoc feel.

Little progress on injective encoding.

Shallue-Woestijne-Ulas

Consider a hyperelliptic curve of the form:

$$y^2 = x^{2g+1} + ax + b$$

with $ab \neq 0$. We sketch the technique by Ulas to construct a function to this curve.

Let $g(x) = x^{2g+1} + ax + b$. For any $u \in \mathbb{F}_q$, there is a unique x_u such that $g(ux_u) = u^{2g+1} g(x_u)$. This x_u is a rational function of u.
Now take $u = u_0 t^2$ for some fixed quadratic nonresidue u_0. Then exactly one of $g(x_u)$ and $g(ux_u)$ is a square. The image of t is one of the two points on the curve of abscissa x_u.

Considered for European e-passports.

Outline

Odd hyperelliptic curves

In this work, we consider hyperelliptic curves of the following form:

$$y^2 = x^{2g+1} + a_1 x^{2g-1} + \cdots + a_g x$$

(the right-hand side is an odd polynomial), over finite fields $\mathbb{F}_q$ with $q \equiv 3 \pmod 4$.

Many examples in the literature:

- Joux's supersingular curves $y^2 = x^3 + ax$;
- Kawazoe-Takahashi Type II pairing-friendly curves of genus 2;
- the genus 2 curves $y^2 = x^5 + ax^3 + bx$ for which Satoh gave an efficient class group counting algorithm;
- certain Freeman-Satoh pairing-friendly curves of genus 2;
- and more.

Outline

Our new function

Write the curve equation as:

$$H : y^2 = f(x)$$

We define a function $F : \mathbb{F}_q \to H(\mathbb{F}_q)$ as follows. For any $t \in \mathbb{F}_q$, one of $f(t)$ or $f(-t)$ is a square; the x-coordinate of $F(t)$ is $\pm t$ accordingly, and the y-coordinate is chosen such that $F(-t) = -F(t)$.

In short:

$$F(t) = \left(\varepsilon(t) \cdot t \ ; \ \varepsilon(t)\sqrt{\varepsilon(t) \cdot f(t)}\right)$$

where $\varepsilon(t) = \left(\frac{f(t)}{q}\right)$, and $\sqrt{\cdot}$ is the usual square root function in $\mathbb{F}_q$ (raising to the power $(q-1)/4$).

This is well-defined, and almost a bijection $\mathbb{F}_q \to H(\mathbb{F}_q)$. In particular, the curve H has exactly $q+1$ rational points.

Efficient computation

We give an efficient, constant-time algorithm for computing the function F:

1. $\alpha \leftarrow f(t)$
2. $\beta \leftarrow \alpha^r$
3. **return** $(\alpha\beta^2 t, \alpha\beta)$

where $r = (q-3)/4$ or $(q-3)/4 + (q-1)/2$ depending on $q \bmod 8$.

Single exponentiation and a few multiplications in the base field.
Probably the simplest, most efficient encoding function since Boneh-Franklin.

Outline

Encoding and hashing to the Jacobian

- When $g = 1$, H is a supersingular elliptic curve, and just as in the Boneh-Franklin case, the function F provides both injective encoding and hashing to the curve directly.
- When $g > 1$, the set of points on the curve is not a group. The group attached to H is the set of points of its Jacobian J. It is this group that we should seek to encode or hash to.
- Injective encoding with large image:

$$\begin{aligned} F_{\mathrm{inj}} \colon \{g\text{-element subsets of } \mathbb{F}_q\} &\longrightarrow J(\mathbb{F}_q) \\ \{t_1, \ldots, t_g\} &\longmapsto F(t_1) + \cdots + F(t_g) \end{aligned}$$

  Reaches a fraction of about $1/g!$ of all divisors in $J(\mathbb{F}_q)$.
- Using [BCIMRT10] and [FFSTV], we know that the following is a well-behaved hash function to the Jacobian:

$$m \mapsto F(h_1(m)) + \cdots + F(h_{g+1}(m))$$

  when $h_1, \ldots, h_{g+1}$ are seen as independent random oracles into $\mathbb{F}_q$.

Summary

- New, particularly simple bijective function to a nice family of hyperelliptic curves (many pairing-friendly hyperelliptic curves constructed in this form!).
- Efficient to compute in constant-time.
- Gives hashing and injective encoding to the Jacobians of these curves.

- Outlook
  - Injective encodings to more elliptic curves?
  - More systematic, less ad-hoc approach to construction such functions?
  - Applications in actual protocols?

Thank you!