Efficient Indifferentiable Hashing into Ordinary Elliptic Curves

Éric Brier[1] Jean-Sébastien Coron[2] Thomas Icart[2]
David Madore[3] Hugues Randriam[3] Mehdi Tibouchi[2,4]

[1]Ingenico

[2]Université du Luxembourg

[3]TELECOM-ParisTech

[4]École normale supérieure

CRYPTO, 2010-08-16

Outline

Outline

Elliptic curve cryptography

- F finite field of characteristic > 3 (for simplicity's sake).
- Recall that an elliptic curve over F is the set of points $(x, y) \in F^2$ such that:
$$y^2 = x^3 + ax + b$$
(with $a, b \in F$ fixed parameters), together with a point at infinity.
- This set of points forms an abelian group where the Discrete Logarithm Problem and Diffie-Hellman-type problems are believed to be hard (no attack better than the generic ones).
- Interesting for cryptography: for k bits of security, one can use elliptic curve groups of order $\approx 2^{2k}$, keys of length $\approx 2k$. Also come with rich structures such as pairings.

Elliptic curve cryptography

- F finite field of characteristic > 3 (for simplicity's sake).
- Recall that an elliptic curve over F is the set of points $(x, y) \in F^2$ such that:

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

  (with $a, b \in F$ fixed parameters), together with a point at infinity.
- This set of points forms an abelian group where the Discrete Logarithm Problem and Diffie-Hellman-type problems are believed to be hard (no attack better than the generic ones).
- Interesting for cryptography: for k bits of security, one can use elliptic curve groups of order $\approx 2^{2k}$, keys of length $\approx 2k$. Also come with rich structures such as pairings.

Elliptic curve cryptography

- F finite field of characteristic > 3 (for simplicity's sake).
- Recall that an elliptic curve over F is the set of points $(x, y) \in F^2$ such that:

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

  (with $a, b \in F$ fixed parameters), together with a point at infinity.
- This set of points forms an abelian group where the Discrete Logarithm Problem and Diffie-Hellman-type problems are believed to be hard (no attack better than the generic ones).
- Interesting for cryptography: for k bits of security, one can use elliptic curve groups of order $\approx 2^{2k}$, keys of length $\approx 2k$. Also come with rich structures such as pairings.

Elliptic curve cryptography

- F finite field of characteristic > 3 (for simplicity's sake).
- Recall that an elliptic curve over F is the set of points $(x, y) \in F^2$ such that:

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

  (with $a, b \in F$ fixed parameters), together with a point at infinity.
- This set of points forms an abelian group where the Discrete Logarithm Problem and Diffie-Hellman-type problems are believed to be hard (no attack better than the generic ones).
- Interesting for cryptography: for k bits of security, one can use elliptic curve groups of order $\approx 2^{2k}$, keys of length $\approx 2k$. Also come with rich structures such as pairings.

Outline

Hashing to elliptic curves is a problem

- Many cryptographic protocols (schemes for encryption, signature, PAKE, IBE, etc.) involve representing a certain numeric value, often a hash value, as an element of the group $\mathbb{G}$ where the computations occur.
- For $\mathbb{G} = \mathbb{Z}_p^*$, simply take the numeric value itself $\bmod p$.
- However, doesn't generalize when $\mathbb{G}$ is an elliptic curve group; e.g. one cannot put the value in the x-coordinate of a curve point, because only about 1/2 of possible x-values correspond to actual points.
- Elliptic curve-specific protocols have been developed to circumvent this problem (ECDSA for signature, Menezes-Vanstone for encryption, ECMQV for key agreement, etc.), but doing so with all imaginable protocols is unrealistic.

Hashing to elliptic curves is a problem

- Many cryptographic protocols (schemes for encryption, signature, PAKE, IBE, etc.) involve representing a certain numeric value, often a hash value, as an element of the group $\mathbb{G}$ where the computations occur.
- For $\mathbb{G} = \mathbb{Z}_p^*$, simply take the numeric value itself $\bmod p$.
- However, doesn't generalize when $\mathbb{G}$ is an elliptic curve group; e.g. one cannot put the value in the x-coordinate of a curve point, because only about 1/2 of possible x-values correspond to actual points.
- Elliptic curve-specific protocols have been developed to circumvent this problem (ECDSA for signature, Menezes-Vanstone for encryption, ECMQV for key agreement, etc.), but doing so with all imaginable protocols is unrealistic.

Hashing to elliptic curves is a problem

- Many cryptographic protocols (schemes for encryption, signature, PAKE, IBE, etc.) involve representing a certain numeric value, often a hash value, as an element of the group $\mathbb{G}$ where the computations occur.
- For $\mathbb{G} = \mathbb{Z}_p^*$, simply take the numeric value itself $\bmod p$.
- However, doesn't generalize when $\mathbb{G}$ is an elliptic curve group; e.g. one cannot put the value in the x-coordinate of a curve point, because only about 1/2 of possible x-values correspond to actual points.
- Elliptic curve-specific protocols have been developed to circumvent this problem (ECDSA for signature, Menezes-Vanstone for encryption, ECMQV for key agreement, etc.), but doing so with all imaginable protocols is unrealistic.

Hashing to elliptic curves is a problem

- Many cryptographic protocols (schemes for encryption, signature, PAKE, IBE, etc.) involve representing a certain numeric value, often a hash value, as an element of the group $\mathbb{G}$ where the computations occur.
- For $\mathbb{G} = \mathbb{Z}_p^*$, simply take the numeric value itself $\bmod p$.
- However, doesn't generalize when $\mathbb{G}$ is an elliptic curve group; e.g. one cannot put the value in the x-coordinate of a curve point, because only about 1/2 of possible x-values correspond to actual points.
- Elliptic curve-specific protocols have been developed to circumvent this problem (ECDSA for signature, Menezes-Vanstone for encryption, ECMQV for key agreement, etc.), but doing so with all imaginable protocols is unrealistic.

The traditional solution

- For k bits of security:
    1. concatenate the hash value h with a counter c from 0 to $k-1$;
    2. initialize the counter as 0;
    3. if the concatenated value $x = c \| h$ is a valid x-coordinate on the curve (i.e. $x^3 + ax + b$ is a square in F), 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.

The traditional solution

- For k bits of security:
  1. concatenate the hash value h with a counter c from 0 to $k-1$;
  2. initialize the counter as 0;
  3. if the concatenated value $x = c \| h$ is a valid x-coordinate on the curve (i.e. $x^3 + ax + b$ is a square in F), 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.

The traditional solution

- For k bits of security:
  1. concatenate the hash value h with a counter c from 0 to $k-1$;
  2. initialize the counter as 0;
  3. if the concatenated value $x = c \| h$ is a valid x-coordinate on the curve (i.e. $x^3 + ax + b$ is a square in F), 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.

The traditional solution

- For k bits of security:
  1. concatenate the hash value h with a counter c from 0 to $k-1$;
  2. initialize the counter as 0;
  3. if the concatenated value $x = c \| h$ is a valid x-coordinate on the curve (i.e. $x^3 + ax + b$ is a square in F), 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.

The traditional solution

- For k bits of security:
  1. concatenate the hash value h with a counter c from 0 to $k-1$;
  2. initialize the counter as 0;
  3. if the concatenated value $x = c \| h$ is a valid x-coordinate on the curve (i.e. $x^3 + ax + b$ is a square in F), 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

1. A natural implementation does not run in constant time: possible timing attacks (especially for PAKE).
2. A constant time implementation (always do k steps, compute the Legendre symbol in constant time) is very inefficient, $O(n^4)$.
3. Security is difficult to analyze.
   - image is difficult to describe;
   - image size estimate is only heuristic ($\approx q/k$);
   - does not behave at all like a random oracle to the curve: easy distinguisher exists.

Problems with this solution

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

Problems with this solution

1. A natural implementation does not run in constant time: possible timing attacks (especially for PAKE).
2. A constant time implementation (always do k steps, compute the Legendre symbol in constant time) is very inefficient, $O(n^4)$.
3. Security is difficult to analyze.
   - image is difficult to describe;
   - image size estimate is only heuristic ($\approx q/k$);
   - does not behave at all like a random oracle to the curve; easy distinguisher exists.

Problems with this solution

1. A natural implementation does not run in constant time: possible timing attacks (especially for PAKE).
2. A constant time implementation (always do k steps, compute the Legendre symbol in constant time) is very inefficient, $O(n^4)$.
3. Security is difficult to analyze.
   - image is difficult to describe;
   - image size estimate is only heuristic ($\approx q/k$);
   - does not behave at all like a random oracle to the curve; easy distinguisher exists.

Problems with this solution

1. A natural implementation does not run in constant time: possible timing attacks (especially for PAKE).
2. A constant time implementation (always do k steps, compute the Legendre symbol in constant time) is very inefficient, $O(n^4)$.
3. Security is difficult to analyze.
   - image is difficult to describe;
   - image size estimate is only heuristic ($\approx q/k$);
   - does not behave at all like a random oracle to the curve; easy distinguisher exists.

Problems with this solution

1. A natural implementation does not run in constant time: possible timing attacks (especially for PAKE).
2. A constant time implementation (always do k steps, compute the Legendre symbol in constant time) is very inefficient, $O(n^4)$.
3. Security is difficult to analyze.
   - image is difficult to describe;
   - image size estimate is only heuristic ($\approx q/k$);
   - does not behave at all like a random oracle to the curve; easy distinguisher exists.

Outline

The Boneh-Franklin construction

For their elliptic curve-based IBE scheme [BF01], Boneh and Franklin introduced the following hash function construction.

They use *supersingular* elliptic curves, of the form:

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

over $\mathbb{F}_q$ with $q \equiv 2 \pmod 3$. Admit the following deterministic encoding:

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

Solves the problem: efficient, constant-time, quasi-bijective and secure ▶ if h is a good hash function to $\mathbb{F}_q$, $H(m) = f(h(m))$ is well-behaved: has the properties of a RO to the curve if h is modeled as a RO to $\mathbb{F}_q$.The IBE scheme is secure for H in the ROM for h.

Downside: limited to supersingular curves.

The Boneh-Franklin construction

For their elliptic curve-based IBE scheme [BF01], Boneh and Franklin introduced the following hash function construction.

They use *supersingular* elliptic curves, of the form:

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

over $\mathbb{F}_q$ with $q \equiv 2 \pmod 3$. Admit the following deterministic encoding:

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

Solves the problem: efficient, constant-time, quasi-bijective and secure ▶ if h is a good hash function to $\mathbb{F}_q$, $H(m) = f(h(m))$ is well-behaved: has the properties of a RO to the curve if h is modeled as a RO to $\mathbb{F}_q$.The IBE scheme is secure for H in the ROM for h.

Downside: limited to supersingular curves.

The Boneh-Franklin construction

For their elliptic curve-based IBE scheme [BF01], Boneh and Franklin introduced the following hash function construction.

They use *supersingular* elliptic curves, of the form:

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

over $\mathbb{F}_q$ with $q \equiv 2 \pmod 3$. Admit the following deterministic encoding:

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

Solves the problem: efficient, constant-time, quasi-bijective and secure ▶ if h is a good hash function to $\mathbb{F}_q$, $H(m) = f(h(m))$ is well-behaved: has the properties of a RO to the curve if h is modeled as a RO to $\mathbb{F}_q$.The IBE scheme is secure for H in the ROM for h.

Downside: limited to supersingular curves.

The Boneh-Franklin construction

For their elliptic curve-based IBE scheme [BF01], Boneh and Franklin introduced the following hash function construction.

They use *supersingular* elliptic curves, of the form:

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

over $\mathbb{F}_q$ with $q \equiv 2 \pmod 3$. Admit the following deterministic encoding:

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

Solves the problem: efficient, constant-time, quasi-bijective and secure ▶ if h is a good hash function to $\mathbb{F}_q$, $H(m) = f(h(m))$ is well-behaved: has the properties of a RO to the curve if h is modeled as a RO to $\mathbb{F}_q$.The IBE scheme is secure for H in the ROM for h.

Downside: limited to supersingular curves.

The Boneh-Franklin construction

For their elliptic curve-based IBE scheme [BF01], Boneh and Franklin introduced the following hash function construction.

They use *supersingular* elliptic curves, of the form:

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

over $\mathbb{F}_q$ with $q \equiv 2 \pmod 3$. Admit the following deterministic encoding:

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

Solves the problem: efficient, constant-time, quasi-bijective and secure ▶ if h is a good hash function to $\mathbb{F}_q$, $H(m) = f(h(m))$ is well-behaved: has the properties of a RO to the curve if h is modeled as a RO to $\mathbb{F}_q$. The IBE scheme is secure for H in the ROM for h.

Downside: limited to supersingular curves.

The Boneh-Franklin construction

For their elliptic curve-based IBE scheme [BF01], Boneh and Franklin introduced the following hash function construction.

They use *supersingular* elliptic curves, of the form:

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

over $\mathbb{F}_q$ with $q \equiv 2 \pmod 3$. Admit the following deterministic encoding:

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

Solves the problem: efficient, constant-time, quasi-bijective and secure ▶ if h is a good hash function to $\mathbb{F}_q$, $H(m) = f(h(m))$ is well-behaved: has the properties of a RO to the curve if h is modeled as a RO to $\mathbb{F}_q$.The IBE scheme is secure for H in the ROM for h.

Downside: limited to supersingular curves.

The Boneh-Franklin construction

For their elliptic curve-based IBE scheme [BF01], Boneh and Franklin introduced the following hash function construction.

They use *supersingular* elliptic curves, of the form:

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

over $\mathbb{F}_q$ with $q \equiv 2 \pmod 3$. Admit the following deterministic encoding:

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

Solves the problem: efficient, constant-time, quasi-bijective and secure ▶ if h is a good hash function to $\mathbb{F}_q$, $H(m) = f(h(m))$ is well-behaved: has the properties of a RO to the curve if h is modeled as a RO to $\mathbb{F}_q$.The IBE scheme is secure for H in the ROM for h.

Downside: limited to supersingular curves.

Ordinary curves: Icart

Last year at CRYPTO, Icart presented a construction for ordinary curves when $q \equiv 2 \pmod 3$. Generalization of the supersingular case.

Defined as $f\colon u \mapsto (x, y)$ with

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

Efficient, constant-time, and applies to almost all elliptic curves. *However*, image size is only $\approx 5/8$ of all points. The construction $H(m) = f(h(m))$ is easily distinguished from a RO to the curve even if h is modeled as a RO. ▶ Security?

Many more deterministic encodings to ordinary curves proposed recently, but with the same limitation.

Ordinary curves: Icart

Last year at CRYPTO, Icart presented a construction for ordinary curves when $q \equiv 2 \pmod 3$. Generalization of the supersingular case.

Defined as $f\colon u \mapsto (x, y)$ with

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

Efficient, constant-time, and applies to almost all elliptic curves. *However*, image size is only $\approx 5/8$ of all points. The construction $H(m) = f(h(m))$ is easily distinguished from a RO to the curve even if h is modeled as a RO. ▶ Security?

Many more deterministic encodings to ordinary curves proposed recently, but with the same limitation.

Ordinary curves: Icart

Last year at CRYPTO, Icart presented a construction for ordinary curves when $q \equiv 2 \pmod 3$. Generalization of the supersingular case.

Defined as $f\colon u \mapsto (x, y)$ with

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

Efficient, constant-time, and applies to almost all elliptic curves.

However, image size is only $\approx 5/8$ of all points. The construction $H(m) = f(h(m))$ is easily distinguished from a RO to the curve even if h is modeled as a RO. ▶ Security?

Many more deterministic encodings to ordinary curves proposed recently, but with the same limitation.

Ordinary curves: Icart

Last year at CRYPTO, Icart presented a construction for ordinary curves when $q \equiv 2 \pmod 3$. Generalization of the supersingular case.

Defined as $f\colon u \mapsto (x, y)$ with

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

Efficient, constant-time, and applies to almost all elliptic curves. *However*, image size is only $\approx 5/8$ of all points. The construction $H(m) = f(h(m))$ is easily distinguished from a RO to the curve even if h is modeled as a RO. ▶ Security?

Many more deterministic encodings to ordinary curves proposed recently, but with the same limitation.

Ordinary curves: Icart

Last year at CRYPTO, Icart presented a construction for ordinary curves when $q \equiv 2 \pmod 3$. Generalization of the supersingular case.

Defined as $f\colon u \mapsto (x, y)$ with

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

Efficient, constant-time, and applies to almost all elliptic curves. *However*, image size is only $\approx 5/8$ of all points. The construction $H(m) = f(h(m))$ is easily distinguished from a RO to the curve even if h is modeled as a RO. ▶ Security?

Many more deterministic encodings to ordinary curves proposed recently, but with the same limitation.

Ordinary curves: Icart

Last year at CRYPTO, Icart presented a construction for ordinary curves when $q \equiv 2 \pmod 3$. Generalization of the supersingular case.

Defined as $f\colon u \mapsto (x, y)$ with

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

Efficient, constant-time, and applies to almost all elliptic curves. *However*, image size is only $\approx 5/8$ of all points. The construction $H(m) = f(h(m))$ is easily distinguished from a RO to the curve even if h is modeled as a RO. ▶ Security?

Many more deterministic encodings to ordinary curves proposed recently, but with the same limitation.

Outline

Security in the ROM

Is it secure to use $H(m) = f(h(m))$ as a hash function to the curve?

More precisely: if a scheme is proved secure assuming H is a RO, is the security preserved if one instantiates $H(m) = f(h(m))$ with h modeled as a RO?

- For a number of pairing-based schemes: yes (related to random self-reducibility properties of the underlying security assumptions).
- In general: no, security breaks down (counterexample in the paper).
- Difficult to give a simple criterion for the security proof to go through.
- Can we propose constructions that will work all the time instead?

Security in the ROM

Is it secure to use $H(m) = f(h(m))$ as a hash function to the curve?

More precisely: if a scheme is proved secure assuming H is a RO, is the security preserved if one instantiates $H(m) = f(h(m))$ with h modeled as a RO?

- For a number of pairing-based schemes: yes (related to random self-reducibility properties of the underlying security assumptions).
- In general: no, security breaks down (counterexample in the paper).
- Difficult to give a simple criterion for the security proof to go through.
- Can we propose constructions that will work all the time instead?

Security in the ROM

Is it secure to use $H(m) = f(h(m))$ as a hash function to the curve?

More precisely: if a scheme is proved secure assuming H is a RO, is the security preserved if one instantiates $H(m) = f(h(m))$ with h modeled as a RO?

- For a number of pairing-based schemes: yes (related to random self-reducibility properties of the underlying security assumptions).
- In general: no, security breaks down (counterexample in the paper).
- Difficult to give a simple criterion for the security proof to go through.
- Can we propose constructions that will work all the time instead?

Security in the ROM

Is it secure to use $H(m) = f(h(m))$ as a hash function to the curve?

More precisely: if a scheme is proved secure assuming H is a RO, is the security preserved if one instantiates $H(m) = f(h(m))$ with h modeled as a RO?

- For a number of pairing-based schemes: yes (related to random self-reducibility properties of the underlying security assumptions).
- In general: no, security breaks down (counterexample in the paper).
- Difficult to give a simple criterion for the security proof to go through.
- Can we propose constructions that will work all the time instead?

Security in the ROM

Is it secure to use $H(m) = f(h(m))$ as a hash function to the curve?

More precisely: if a scheme is proved secure assuming H is a RO, is the security preserved if one instantiates $H(m) = f(h(m))$ with h modeled as a RO?

- For a number of pairing-based schemes: yes (related to random self-reducibility properties of the underlying security assumptions).
- In general: no, security breaks down (counterexample in the paper).
- Difficult to give a simple criterion for the security proof to go through.
- Can we propose constructions that will work all the time instead?

Indifferentiability

High-level formulation of our problem: find a condition under which an ideal primitive (the RO to the curve) can be replaced by a construction based on another ideal primitive (a RO to $\mathbb{F}_q$) so that all security proof are preserved.

Answer: indifferentiability (Maurer et al., 2004). Roughly speaking, the construction is indifferentiable from the primitive if no PPT adversary can tell them apart with non-negligible probability.

But this is a bit abstract. Easy to test criterion for a hash function construction to work?

Indifferentiability

High-level formulation of our problem: find a condition under which an ideal primitive (the RO to the curve) can be replaced by a construction based on another ideal primitive (a RO to $\mathbb{F}_q$) so that all security proof are preserved.

Answer: indifferentiability (Maurer et al., 2004). Roughly speaking, the construction is indifferentiable from the primitive if no PPT adversary can tell them apart with non-negligible probability.

But this is a bit abstract. Easy to test criterion for a hash function construction to work?

Indifferentiability

High-level formulation of our problem: find a condition under which an ideal primitive (the RO to the curve) can be replaced by a construction based on another ideal primitive (a RO to $\mathbb{F}_q$) so that all security proof are preserved.

Answer: indifferentiability (Maurer et al., 2004). Roughly speaking, the construction is indifferentiable from the primitive if no PPT adversary can tell them apart with non-negligible probability.

But this is a bit abstract. Easy to test criterion for a hash function construction to work?

Admissible encodings

We consider hash function constructions of the form:

$$H(m) = F(h(m))$$

where h is modeled as a RO to a some set S (easy to hash to) and F is a deterministic function $S \to E(\mathbb{F}_q)$.

We prove that H is indifferentiable from a RO to $E(\mathbb{F}_q)$ as soon as the function F is admissible in the following sense:

- **Computable** in deterministic polynomial time;
- **Regular** for s uniformly distributed in S, the distribution of $F(s)$ is statistically indistinguishable from the uniform distribution in $E(\mathbb{F}_q)$;
- **Samplable** there is a PPT algorithm which for any $P \in E(\mathbb{F}_q)$ returns an uniformly distributed element in $F^{-1}(P)$.

Admissible encodings

We consider hash function constructions of the form:

$$H(m) = F(h(m))$$

where h is modeled as a RO to a some set S (easy to hash to) and F is a deterministic function $S \to E(\mathbb{F}_q)$.

We prove that H is indifferentiable from a RO to $E(\mathbb{F}_q)$ as soon as the function F is admissible in the following sense:

Computable in deterministic polynomial time;

Regular for s uniformly distributed in S, the distribution of $F(s)$ is statistically indistinguishable from the uniform distribution in $E(\mathbb{F}_q)$;

Samplable there is a PPT algorithm which for any $\varpi \in E(\mathbb{F}_q)$ returns an uniformly distributed element in $F^{-1}(\varpi)$.

Admissible encodings

We consider hash function constructions of the form:

$$H(m) = F(h(m))$$

where h is modeled as a RO to a some set S (easy to hash to) and F is a deterministic function $S \to E(\mathbb{F}_q)$.

We prove that H is indifferentiable from a RO to $E(\mathbb{F}_q)$ as soon as the function F is admissible in the following sense:

Computable in deterministic polynomial time;

Regular for s uniformly distributed in S, the distribution of $F(s)$ is statistically indistinguishable from the uniform distribution in $E(\mathbb{F}_q)$;

Samplable there is a PPT algorithm which for any $\varpi \in E(\mathbb{F}_q)$ returns an uniformly distributed element in $F^{-1}(\varpi)$.

Admissible encodings

We consider hash function constructions of the form:

$$H(m) = F(h(m))$$

where h is modeled as a RO to a some set S (easy to hash to) and F is a deterministic function $S \to E(\mathbb{F}_q)$.

We prove that H is indifferentiable from a RO to $E(\mathbb{F}_q)$ as soon as the function F is admissible in the following sense:

Computable in deterministic polynomial time;

Regular for s uniformly distributed in S, the distribution of $F(s)$ is statistically indistinguishable from the uniform distribution in $E(\mathbb{F}_q)$;

Samplable there is a PPT algorithm which for any $\varpi \in E(\mathbb{F}_q)$ returns an uniformly distributed element in $F^{-1}(\varpi)$.

Remarks

- We can quantify precisely the "loss" in random oracle security when instantiating H in this manner (in terms of the statistical distance between $F(s)$ and uniform, and the running time of the sampling algorithm).
- Icart's function is *not* admissible: computable and samplable, but not regular.
- A construction like $H(m) = h(m) \cdot G$, with G a generator, is *not* admissible: computable and regular but not samplable. Bad idea: leaks the discrete logarithm of the digest!

Remarks

- We can quantify precisely the "loss" in random oracle security when instantiating H in this manner (in terms of the statistical distance between $F(s)$ and uniform, and the running time of the sampling algorithm).
- Icart's function is *not* admissible: computable and samplable, but not regular.
- A construction like $H(m) = h(m) \cdot G$, with G a generator, is *not* admissible: computable and regular but not samplable. Bad idea: leaks the discrete logarithm of the digest!

Remarks

- We can quantify precisely the “loss” in random oracle security when instantiating H in this manner (in terms of the statistical distance between $F(s)$ and uniform, and the running time of the sampling algorithm).
- Icart’s function is *not* admissible: computable and samplable, but not regular.
- A construction like $H(m) = h(m) \cdot G$, with G a generator, is *not* admissible: computable and regular but not samplable. Bad idea: leaks the discrete logarithm of the digest!

Outline

General construction

E ordinary elliptic curve over $\mathbb{F}_q$, G generator of $E(\mathbb{F}_q)$ (assumed cyclic of cardinality N) and $f : \mathbb{F}_q \to E(\mathbb{F}_q)$ deterministic encoding like Icart's function.

Under mild assumptions on f (verified for all deterministic encodings proposed so far), the following is an admissible function $\mathbb{F}_q \times \mathbb{Z}/N\mathbb{Z} \to E(\mathbb{F}_q)$:

$$F(u, v) = f(u) + v \cdot G$$

Thus, $H(m) = f(h_1(m)) + h_2(m) \cdot G$ is indifferentiable from a RO, in the ROM for h_1, h_2.

Works for any deterministic encoding. Extends to the case when $E(\mathbb{F}_q)$ is not cyclic in an obvious way.

Downside: quite inefficient (≈ 10 times slower than Icart's function alone).

General construction

E ordinary elliptic curve over $\mathbb{F}_q$, G generator of $E(\mathbb{F}_q)$ (assumed cyclic of cardinality N) and $f\colon \mathbb{F}_q \to E(\mathbb{F}_q)$ deterministic encoding like Icart's function.

Under mild assumptions on f (verified for all deterministic encodings proposed so far), the following is an admissible function $\mathbb{F}_q \times \mathbb{Z}/N\mathbb{Z} \to E(\mathbb{F}_q)$:

$$F(u,v) = f(u) + v \cdot G$$

Thus, $H(m) = f(h_1(m)) + h_2(m) \cdot G$ is indifferentiable from a RO, in the ROM for h_1, h_2.

Works for any deterministic encoding. Extends to the case when $E(\mathbb{F}_q)$ is not cyclic in an obvious way.

Downside: quite inefficient (≈ 10 times slower than Icart's function alone).

General construction

E ordinary elliptic curve over $\mathbb{F}_q$, G generator of $E(\mathbb{F}_q)$ (assumed cyclic of cardinality N) and $f: \mathbb{F}_q \to E(\mathbb{F}_q)$ deterministic encoding like Icart's function.

Under mild assumptions on f (verified for all deterministic encodings proposed so far), the following is an admissible function $\mathbb{F}_q \times \mathbb{Z}/N\mathbb{Z} \to E(\mathbb{F}_q)$:

$$F(u, v) = f(u) + v \cdot G$$

Thus, $H(m) = f(h_1(m)) + h_2(m) \cdot G$ is indifferentiable from a RO, in the ROM for h_1, h_2.

Works for any deterministic encoding. Extends to the case when $E(\mathbb{F}_q)$ is not cyclic in an obvious way.

Downside: quite inefficient (≈ 10 times slower than Icart's function alone).

General construction

E ordinary elliptic curve over $\mathbb{F}_q$, G generator of $E(\mathbb{F}_q)$ (assumed cyclic of cardinality N) and $f : \mathbb{F}_q \to E(\mathbb{F}_q)$ deterministic encoding like Icart's function.

Under mild assumptions on f (verified for all deterministic encodings proposed so far), the following is an admissible function $\mathbb{F}_q \times \mathbb{Z}/N\mathbb{Z} \to E(\mathbb{F}_q)$:

$$F(u, v) = f(u) + v \cdot G$$

Thus, $H(m) = f(h_1(m)) + h_2(m) \cdot G$ is indifferentiable from a RO, in the ROM for h_1, h_2.

Works for any deterministic encoding. Extends to the case when $E(\mathbb{F}_q)$ is not cyclic in an obvious way.

Downside: quite inefficient (≈ 10 times slower than Icart's function alone).

Proof sketch

The function F is:

Computable Clearly.

Regular With v uniformly distributed in $\mathbb{Z}/N\mathbb{Z}$ it is clear that $f(u) + v \cdot G$ is uniformly distributed in $E(\mathbb{F}_q)$, regardless of the behavior of f.

Samplable To sample $F^{-1}(\varpi)$, pick a random $v \in \mathbb{Z}/N\mathbb{Z}$ and solve the algebraic equation $f(u) = \varpi - v \cdot G$ for u. For Icart, there are at most 4 solutions, easy to enumerate. Return (u, v) for one of those solutions u at random, or try again if there are none.

Proof sketch

The function F is:

Computable Clearly.

Regular With v uniformly distributed in $\mathbb{Z}/N\mathbb{Z}$ it is clear that $f(u) + v \cdot G$ is uniformly distributed in $E(\mathbb{F}_q)$, regardless of the behavior of f.

Samplable To sample $F^{-1}(\varpi)$, pick a random $v \in \mathbb{Z}/N\mathbb{Z}$ and solve the algebraic equation $f(u) = \varpi - v \cdot G$ for u. For Icart, there are at most 4 solutions, easy to enumerate. Return (u, v) for one of those solutions u at random, or try again if there are none.

Proof sketch

The function F is:

Computable Clearly.

Regular With v uniformly distributed in $\mathbb{Z}/N\mathbb{Z}$ it is clear that $f(u) + v \cdot G$ is uniformly distributed in $E(\mathbb{F}_q)$, regardless of the behavior of f.

Samplable To sample $F^{-1}(\varpi)$, pick a random $v \in \mathbb{Z}/N\mathbb{Z}$ and solve the algebraic equation $f(u) = \varpi - v \cdot G$ for u. For Icart, there are at most 4 solutions, easy to enumerate. Return (u, v) for one of those solutions u at random, or try again if there are none.

Outline

Efficient construction

A much more efficient construction of an admissible encoding is as follows:

$$F(u,v) = f(u) + f(v)$$

where f is Icart's function.

Thus, $H(m) = f(h_1(m)) + f(h_2(m))$ is indifferentiable from a RO, in the ROM for h_1, h_2.

Only requires two evaluations of Icart's function, so quite efficient. No restriction on the curve.

Downside: the proof is significantly more difficult, and only applies to Icart's function, not other deterministic encodings.

More precisely, computability and samplability are proved like before. The hard part is regularity: showing that the cardinality of $F^{-1}(\varpi)$ is almost constant along the curve.

Efficient construction

A much more efficient construction of an admissible encoding is as follows:

$$F(u, v) = f(u) + f(v)$$

where f is Icart's function.

Thus, $H(m) = f(h_1(m)) + f(h_2(m))$ is indifferentiable from a RO, in the ROM for h_1, h_2.

Only requires two evaluations of Icart's function, so quite efficient. No restriction on the curve.

Downside: the proof is significantly more difficult, and only applies to Icart's function, not other deterministic encodings.

More precisely, computability and samplability are proved like before. The hard part is regularity: showing that the cardinality of $F^{-1}(\varpi)$ is almost constant along the curve.

Efficient construction

A much more efficient construction of an admissible encoding is as follows:

$$F(u, v) = f(u) + f(v)$$

where f is Icart's function.

Thus, $H(m) = f(h_1(m)) + f(h_2(m))$ is indifferentiable from a RO, in the ROM for h_1, h_2.

Only requires two evaluations of Icart's function, so quite efficient. No restriction on the curve.

Downside: the proof is significantly more difficult, and only applies to Icart's function, not other deterministic encodings.

More precisely, computability and samplability are proved like before. The hard part is regularity: showing that the cardinality of $F^{-1}(\varpi)$ is almost constant along the curve.

Efficient construction

A much more efficient construction of an admissible encoding is as follows:

$$F(u, v) = f(u) + f(v)$$

where f is Icart's function.

Thus, $H(m) = f(h_1(m)) + f(h_2(m))$ is indifferentiable from a RO, in the ROM for h_1, h_2.

Only requires two evaluations of Icart's function, so quite efficient. No restriction on the curve.

Downside: the proof is significantly more difficult, and only applies to Icart's function, not other deterministic encodings.

More precisely, computability and samplability are proved like before. The hard part is regularity: showing that the cardinality of $F^{-1}(\varpi)$ is almost constant along the curve.

Efficient construction

A much more efficient construction of an admissible encoding is as follows:

$$F(u,v) = f(u) + f(v)$$

where f is Icart's function.

Thus, $H(m) = f(h_1(m)) + f(h_2(m))$ is indifferentiable from a RO, in the ROM for h_1, h_2.

Only requires two evaluations of Icart's function, so quite efficient. No restriction on the curve.

Downside: the proof is significantly more difficult, and only applies to Icart's function, not other deterministic encodings.

More precisely, computability and samplability are proved like before. The hard part is regularity: showing that the cardinality of $F^{-1}(\varpi)$ is almost constant along the curve.

Proof idea

We want to show that the number of solutions $(u, v) \in (\mathbb{F}_q)^2$ to the equation $f(u) + f(v) = \varpi$ is constant up to negligible deviations when ϖ varies along the curve (possibly with a few exceptions).

Key idea: the set of solutions (u, v) forms a curve in the plane. The Hasse-Weil bound ensures that such a curve always has $q + O(\sqrt{q})$ points. QED.

Technical difficulties:

- Icart's function f is not a morphism, only an algebraic correspondance. The correct geometric pictures involves a curve C with morphisms $h: C \to E$ and $p: C \to \mathbb{P}^1$ such that $f = h \circ p^{-1}$.
- Show that $s: C \times C \to E$ is geometrically "nice", except at a few exceptional points (to be found and dealt with).
- Show that the preimage of "nice" points is indeed an irreducible curve on $C \times C$. Compute its genus (it's 49).
- Justify that we can ignore what happens at infinity (intersection theory on $C \times C$), and push everything down to $(\mathbb{F}_q)^2$.

Proof idea

We want to show that the number of solutions $(u,v) \in (\mathbb{F}_q)^2$ to the equation $f(u)+f(v)=\varpi$ is constant up to negligible deviations when ϖ varies along the curve (possibly with a few exceptions).

Key idea: the set of solutions (u,v) forms a curve in the plane. The Hasse-Weil bound ensures that such a curve always has $q+O(\sqrt{q})$ points. QED.

Technical difficulties:

- Icart's function f is not a morphism, only an algebraic correspondance. The correct geometric pictures involves a curve C with morphisms $h\colon C \to E$ and $p\colon C \to \mathbb{P}^1$ such that $f = h \circ p^{-1}$.
- Show that $s\colon C \times C \to E$ is geometrically "nice", except at a few exceptional points (to be found and dealt with).
- Show that the preimage of "nice" points is indeed an irreducible curve on $C \times C$. Compute its genus (it's 49).
- Justify that we can ignore what happens at infinity (intersection theory on $C \times C$), and push everything down to $(\mathbb{F}_q)^2$.

Proof idea

We want to show that the number of solutions $(u, v) \in (\mathbb{F}_q)^2$ to the equation $f(u) + f(v) = \varpi$ is constant up to negligible deviations when ϖ varies along the curve (possibly with a few exceptions).

Key idea: the set of solutions (u, v) forms a curve in the plane. The Hasse-Weil bound ensures that such a curve always has $q + O(\sqrt{q})$ points. QED.

Technical difficulties:

- Icart's function f is not a morphism, only an algebraic correspondance. The correct geometric pictures involves a curve C with morphisms $h\colon C \to E$ and $p\colon C \to \mathbb{P}^1$ such that $f = h \circ p^{-1}$.
- Show that $s\colon C \times C \to E$ is geometrically "nice", except at a few exceptional points (to be found and dealt with).
- Show that the preimage of "nice" points is indeed an irreducible curve on $C \times C$. Compute its genus (it's 49).
- Justify that we can ignore what happens at infinity (intersection theory on $C \times C$), and push everything down to $(\mathbb{F}_q)^2$.

Proof idea

We want to show that the number of solutions $(u, v) \in (\mathbb{F}_q)^2$ to the equation $f(u) + f(v) = \varpi$ is constant up to negligible deviations when ϖ varies along the curve (possibly with a few exceptions).

Key idea: the set of solutions (u, v) forms a curve in the plane. The Hasse-Weil bound ensures that such a curve always has $q + O(\sqrt{q})$ points. QED.

Technical difficulties:

- Icart's function f is not a morphism, only an algebraic correspondance. The correct geometric pictures involves a curve C with morphisms $h: C \to E$ and $p: C \to \mathbb{P}^1$ such that $f = h \circ p^{-1}$.
- Show that $s: C \times C \to E$ is geometrically "nice", except at a few exceptional points (to be found and dealt with).
- Show that the preimage of "nice" points is indeed an irreducible curve on $C \times C$. Compute its genus (it's 49).
- Justify that we can ignore what happens at infinity (intersection theory on $C \times C$), and push everything down to $(\mathbb{F}_q)^2$.

Proof idea

We want to show that the number of solutions $(u, v) \in (\mathbb{F}_q)^2$ to the equation $f(u) + f(v) = \varpi$ is constant up to negligible deviations when ϖ varies along the curve (possibly with a few exceptions).

Key idea: the set of solutions (u, v) forms a curve in the plane. The Hasse-Weil bound ensures that such a curve always has $q + O(\sqrt{q})$ points. QED.

Technical difficulties:

- Icart's function f is not a morphism, only an algebraic correspondance. The correct geometric pictures involves a curve C with morphisms $h\colon C \to E$ and $p\colon C \to \mathbb{P}^1$ such that $f = h \circ p^{-1}$.
- Show that $s\colon C \times C \to E$ is geometrically "nice", except at a few exceptional points (to be found and dealt with).
- Show that the preimage of "nice" points is indeed an irreducible curve on $C \times C$. Compute its genus (it's 49).
- Justify that we can ignore what happens at infinity (intersection theory on $C \times C$), and push everything down to $(\mathbb{F}_q)^2$.

Proof idea

We want to show that the number of solutions $(u, v) \in (\mathbb{F}_q)^2$ to the equation $f(u) + f(v) = \varpi$ is constant up to negligible deviations when ϖ varies along the curve (possibly with a few exceptions).

Key idea: the set of solutions (u, v) forms a curve in the plane. The Hasse-Weil bound ensures that such a curve always has $q + O(\sqrt{q})$ points. QED.

Technical difficulties:

- Icart's function f is not a morphism, only an algebraic correspondance. The correct geometric pictures involves a curve C with morphisms $h: C \to E$ and $p: C \to \mathbb{P}^1$ such that $f = h \circ p^{-1}$.
- Show that $s: C \times C \to E$ is geometrically "nice", except at a few exceptional points (to be found and dealt with).
- Show that the preimage of "nice" points is indeed an irreducible curve on $C \times C$. Compute its genus (it's 49).
- Justify that we can ignore what happens at infinity (intersection theory on $C \times C$), and push everything down to $(\mathbb{F}_q)^2$.

Outline

Additional contributions

Extensions of our construction:

- hashing to a subgroup;
- extension to even characteristic;
- using a bit-string-valued hash function as the basis;
- formal results on the composition of admissible encodings, etc.

New encodings to ordinary curves:

- simpler, more efficient variant of the Shallue-Woestijne-Ulas encoding;
- deterministic encodings to curves of characteristic 3.

Additional contributions

Extensions of our construction:

- hashing to a subgroup;
- extension to even characteristic;
- using a bit-string-valued hash function as the basis;
- formal results on the composition of admissible encodings, etc.

New encodings to ordinary curves:

- simpler, more efficient variant of the Shallue-Woestijne-Ulas encoding;
- deterministic encodings to curves of characteristic 3.

Additional contributions

Extensions of our construction:

- hashing to a subgroup;
- extension to even characteristic;
- using a bit-string-valued hash function as the basis;
- formal results on the composition of admissible encodings, etc.

New encodings to ordinary curves:

- simpler, more efficient variant of the Shallue-Woestijne-Ulas encoding;
- deterministic encodings to curves of characteristic 3.

Additional contributions

Extensions of our construction:

- hashing to a subgroup;
- extension to even characteristic;
- using a bit-string-valued hash function as the basis;
- formal results on the composition of admissible encodings, etc.

New encodings to ordinary curves:

- simpler, more efficient variant of the Shallue-Woestijne-Ulas encoding;
- deterministic encodings to curves of characteristic 3.

Additional contributions

Extensions of our construction:

- hashing to a subgroup;
- extension to even characteristic;
- using a bit-string-valued hash function as the basis;
- formal results on the composition of admissible encodings, etc.

New encodings to ordinary curves:

- simpler, more efficient variant of the Shallue-Woestijne-Ulas encoding;
- deterministic encodings to curves of characteristic 3.

Additional contributions

Extensions of our construction:

- hashing to a subgroup;
- extension to even characteristic;
- using a bit-string-valued hash function as the basis;
- formal results on the composition of admissible encodings, etc.

New encodings to ordinary curves:

- simpler, more efficient variant of the Shallue-Woestijne-Ulas encoding;
- deterministic encodings to curves of characteristic 3.

Summary and Outlook

- Consider the instantiations of random oracles in elliptic curve-based cryptosystems;
- Suggest a framework for constructing well-behaved hash functions to ordinary elliptic curves;
- Propose two such constructions, one more general, the other more efficient.

Further problems:

- Extend the efficient construction to any deterministic encoding to elliptic and hyperelliptic curves (done!)
- Construct *injective encodings* to ordinary curves?
- Understand how the possibility of encoding scalars as curve points affects elliptic curve-based protocols?

Summary and Outlook

- Consider the instantiations of random oracles in elliptic curve-based cryptosystems;
- Suggest a framework for constructing well-behaved hash functions to ordinary elliptic curves;
- Propose two such constructions, one more general, the other more efficient.

Further problems:

- Extend the efficient construction to any deterministic encoding to elliptic and hyperelliptic curves (done!)
- Construct *injective encodings* to ordinary curves?
- Understand how the possibility of encoding scalars as curve points affects elliptic curve-based protocols?

Summary and Outlook

- Consider the instantiations of random oracles in elliptic curve-based cryptosystems;
- Suggest a framework for constructing well-behaved hash functions to ordinary elliptic curves;
- Propose two such constructions, one more general, the other more efficient.

Further problems:

- Extend the efficient construction to any deterministic encoding to elliptic and hyperelliptic curves (done!)
- Construct *injective encodings* to ordinary curves?
- Understand how the possibility of encoding scalars as curve points affects elliptic curve-based protocols?

Summary and Outlook

- Consider the instantiations of random oracles in elliptic curve-based cryptosystems;
- Suggest a framework for constructing well-behaved hash functions to ordinary elliptic curves;
- Propose two such constructions, one more general, the other more efficient.

Further problems:

- Extend the efficient construction to any deterministic encoding to elliptic and hyperelliptic curves (done!)
- Construct *injective encodings* to ordinary curves?
- Understand how the possibility of encoding scalars as curve points affects elliptic curve-based protocols?

Summary and Outlook

- Consider the instantiations of random oracles in elliptic curve-based cryptosystems;
- Suggest a framework for constructing well-behaved hash functions to ordinary elliptic curves;
- Propose two such constructions, one more general, the other more efficient.

Further problems:

- Extend the efficient construction to any deterministic encoding to elliptic and hyperelliptic curves (done!)
- Construct *injective encodings* to ordinary curves?
- Understand how the possibility of encoding scalars as curve points affects elliptic curve-based protocols?

Summary and Outlook

- Consider the instantiations of random oracles in elliptic curve-based cryptosystems;
- Suggest a framework for constructing well-behaved hash functions to ordinary elliptic curves;
- Propose two such constructions, one more general, the other more efficient.

Further problems:

- Extend the efficient construction to any deterministic encoding to elliptic and hyperelliptic curves (done!)
- Construct *injective encodings* to ordinary curves?
- Understand how the possibility of encoding scalars as curve points affects elliptic curve-based protocols?

Thank you!