An Indifferentiable Hash Function into Elliptic Curves

Jean-Sébastien Coron[1] and Thomas Icart[1,2]

[1]Université du Luxembourg
[2]Sagem Sécurité

Abstract. We provide the first construction of a hash function into an elliptic curve that is indifferentiable from a random oracle. Our construction can be based on any efficient encoding into an elliptic-curve, for example Icart's function or the Shallue-Woestijne-Ulas (SWU) algorithm.

Keywords: elliptic curve, hash algorithm, indifferentiability.

1 Introduction

Hashing into Elliptic-Curves. Many elliptic-curve cryptosystems require to hash into an elliptic curve. For example in the Boneh-Franklin IBE scheme [6], the public-key for identity $id \in \{0,1\}^*$ is a point $Q_{id} = H_1(id)$ on the curve. This is also the case in many other pairing-based cryptosystems including IBE and HIBE schemes [1,17,18], signature and identity-based signature schemes [5,7,8,13,29] and identity-based signcryption schemes [10,21].

Hashing into elliptic curves is also required for some passwords based authentication protocols, for instance the SPEKE (Simple Password Exponential Key Exchange) [20] and the PAK (Password Authenticated Key exchange) [11]. This is also the case for discrete-log based signature schemes such as [14]; when instantiated over an elliptic-curve, a hash function into the curve is required. In all those previous cryptosystems, security is proven when the hash function is seen as a random oracle into the curve. However, it remains to determine which hashing algorithm should be used, and whether it is reasonable to see it as a random oracle.

In [6], Boneh and Franklin use a particular super-singular elliptic curve E for which, in addition to the pairing operation, there exists a one-to-one mapping f from the base field $\mathbb{F}_p$ to E. This enables to hash using $H_1(m) = f(h(m))$ where h is a classical hash function from $\{0,1\}^*$ to $\mathbb{F}_p$. The authors show that their IBE scheme remains secure when h is seen as a random oracle into $\mathbb{F}_p$ (instead of H_1 being seen as a random oracle into E). However, when no pairing operation is required (as in [11,14,20]), it is more efficient to use ordinary elliptic-curves, since super-singular curves require much larger security parameters (due to the MOV attack [23]).

For hashing into an ordinary elliptic curve, the classical approach is inherently probabilistic: one can first compute an integer hash value $x = h(m)$ and then determine whether x is the abscissa of a point on the elliptic curve:

$$y^2 = x^3 + ax + b \mod p;$$

otherwise one can try $x + 1$ and so on. Using this approach the number of operations required to hash a message m depends on m, which can lead to a timing attack (see [9]). To prevent from this attack, one should determine whether $x+i$ is the abscissa of a point, for *all* i between $0 \leq i < k$, and use for example the smallest such i; here k is a security parameter that gives an error probability of roughly 2^{-k}.

The first algorithm to generate elliptic-curve points in *deterministic* polynomial time was published in ANTS 2006 by Shallue and Woestijne [26]. The algorithm has running time $\mathcal{O}(\log^4 p)$

for any p, and $\mathcal{O}(\log^3 p)$ when $p = 3 \mod 4$. The rational maps in [26] were later simplified and generalized to hyper-elliptic curves by Ulas in [28]; we refer to this algorithm as the Shallue-Woestijne-Ulas (SWU) algorithm. Letting $f : \mathbb{F}_p \to E$ be the function defined by SWU, one can then hash in deterministic polynomial time using $H(m) = f(h(m))$ where h is any hash function into $\mathbb{F}_p$.

Another deterministic hash algorithm for ordinary elliptic curves was recently published by Icart in [19]. The algorithm works for $p = 2 \mod 3$ only, with complexity $\mathcal{O}(\log^3 p)$. Given any elliptic-curve E defined over $\mathbb{F}_p$, Icart defines a function f that is a rational function from $\mathbb{F}_p$ into the curve. As previously given any hash function h into $\mathbb{F}_p$, one can use $H(m) = f(h(m))$ to hash into E. As shown in [19], H is one-way if h is one-way.

The Random Oracle Methodology. Many cryptosystems based on elliptic curves have been proven secure in the random oracle model [1,5,6,7,10,8,11,13,17,18,20,21,29]. In the random oracle model (ROM), the hash function is replaced by a publicly accessible random function (the random oracle) [3]. In this model the adversary cannot compute the hash function by himself but instead he must query the random oracle. Obviously, a proof in the random oracle model is not fully satisfactory, because such a proof does not imply that the scheme will remain secure when the random oracle is replaced by a concrete hash function. Numerous papers have shown artificial schemes that are provably secure in the ROM but completely insecure when the RO is instantiated with any function family (see [12]). Despite these separation results, a proof in the ROM is believed to indicate that there are no structural flaws in the design of the system, and that no flaw will suddenly appear when a "well designed" hash function is used instead.

For a cryptosystem that requires a hash function H into an ordinary elliptic curve (such as [11,20]), one possibility could be to use $H(m) = f(h(m))$ where f is either Icart or SWU's function and h is a hash function into $\mathbb{F}_p$. However we know that neither Icart nor SWU's function generate all the points of E; for example Icart's function covers only $\simeq 5/8$ of the points. Therefore the current proofs in the random oracle model for H do not guarantee the security of the resulting scheme when $H(m) = f(h(m))$ is used instead (even if h is assumed to be ideal). In other words, even if a proof in the random oracle for H can indicate that there are no structural flaws in the design of the cryptosystem, using $H(m) = f(h(m))$ could introduce a flaw that would make the resulting cryptosystem completely insecure.[1]

Our Goal. In this paper our goal is to construct a hash function H into elliptic curves with the property that *any cryptosystem* proven secure assuming H is a random oracle should remain secure if our construction is plugged instead (still assuming that the underlying h is a random oracle). For this we use the indifferentiability framework of Maurer *et al.* [22]. As shown in [15], when a construction H is indifferentiable from a random oracle, such construction can then replace a random oracle in any cryptosystem, and the resulting scheme remains secure in the random oracle model for h. Our goal in this paper is to construct a hash function into elliptic curve that is indifferentiable from a random oracle; moreover such H should work for any elliptic curve, not only super-singular ones.

Our Results. We provide the first hash function into elliptic curves satisfying these properties. We first consider general constructions of the form $H(m) = F(h(m))$ and we determine the required properties for F. We introduce the notion of admissible encoding: roughly speaking, an admissible

[1] It is actually easy to construct a cryptosystem provably secure in the random oracle model for H but insecure when $H(m) = f(h(m))$ is used instead, even assuming h is ideal.

encoding F must be efficiently invertible, and the distribution of $F(s)$ must be statistically close to uniform for uniformly distributed s. We show that if F is an admissible encoding, then $H(m) = F(h(m))$ is indifferentiable from a random oracle.

However, we cannot apply this result to Icart or SWU's function directly, since those functions are *not* admissible encodings; namely as mentioned previously their output only covers a fraction of the elliptic curve points. Therefore we introduce a weaker notion called *weak encoding*. Informally, a weak encoding $f : S \to R$ must be efficiently invertible (as for an admissible encoding), but the distribution of $f(s)$ for random s is only required to satisfy a much weaker property. In this paper we show that 1) both SWU and Icart's function satisfy this notion of weak encoding, and 2) we can turn a weak encoding into an admissible encoding when working in a group (*e.g.*, the group of points of an elliptic-curve). This enables to use SWU or Icart's function to build an admissible encoding, and then an indifferentiable hash function.

More precisely, given an elliptic-curve $\mathbb{E}$ defined over $\mathbb{F}_p$ with N points and a generator G, our construction is as follows:

$$H(m) := f(h_1(m)) + h_2(m).G$$

where $h_1 : \{0,1\}^* \to \mathbb{F}_p$ and $h_2 : \{0,1\}^* \to \mathbb{Z}_N$ are two hash functions, and f is SWU or Icart's function (or more generally any weak encoding into $\mathbb{E}$). Our main result in this paper is that $H(m)$ is indifferentiable from a random oracle when h_1 and h_2 are seen as random oracles. Therefore $H(m)$ can be used in any cryptosystem provably secure with random oracle into elliptic curves, and the resulting cryptosystem remains secure in the random oracle model for h_1 and h_2. Intuitively, the term $h_2(m).G$ plays the role of a one-time pad; this ensures that $H(m)$ can behave as a random oracle even though $f(h_1(m))$ does not reach all the points in $\mathbb{E}$. Note that one could not use $H(m) = h_2(m).G$ only since in this case the discrete logarithm of $H(m)$ would be known, which would make most protocols insecure.[2]

Finally, we show how to extend the previous construction to hashing into the subgroup of an elliptic-curve, to hash-functions into strings (instead of $\mathbb{Z}_p$), to primes of the form $p = 2^\ell - \omega$ for small ω, and to fields of characteristic 2. We also describe a slightly more efficient variant of the SWU algorithm when $p = 3 \mod 4$.

1.1 Related Work

An elliptic curve over a field $\mathbb{F}_{p^n}$ where $p > 3$ is defined by a Weierstrass equation:

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

where a and b are elements of $\mathbb{F}_{p^n}$. Throughout this paper, we note $E_{a,b}$ the curve associated to these parameters. It is well known that the set of points forms a group; we denote by $E_{a,b}(\mathbb{F}_{p^n})$ this group and by N its order. We denote $q = p^n$.

Super-singular Curves. A curve $E_{a,b}$ over $\mathbb{F}_p$ is called super-singular when $N = p + 1$. When $p = 2 \mod 3$, the map $x \mapsto x^3$ is a bijection, therefore the curves $Y^2 = X^3 + b$ are super-singular. One can then define the encoding

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

and the hash function $H(m) := f(h(m))$, where h is a classical hash function into $\mathbb{F}_{p^n}$.

[2] One can see that $F(s) = s.G$ is not an admissible encoding, since inverting F would require to solve a discrete log.

In the Boneh-Franklin scheme [6], one actually works in a subgroup $\mathbb{G}$ of prime order q of $E_{a,b}(\mathbb{F}_p)$; we let ℓ be the co-factor such that $p+1 = \ell \cdot q$. One requires that q does not divide ℓ (*i.e.* that q^2 does not divide $p+1$). In order to hash into $\mathbb{G}$, one can therefore use the encoding $f_{\mathbb{G}}(u) := \ell.f(u)$ and the hash function into $\mathbb{G}$:

$$H_{\mathbb{G}}(u) := f_{\mathbb{G}}(h(m)) \tag{1}$$

In [6], Boneh and Franklin introduce the following notion of admissible encoding:

Definition 1 (Boneh-Franklin admissible encoding [6]). *A function $f : S \to R$ is an admissible encoding if it satisfies the following properties:*

1. *Computable: f is computable in deterministic polynomial time;*
2. *ℓ-to-1: for any $r \in R$, $|f^{-1}(r)| = \ell$;*
3. *Samplable: there exists a probabilistic polynomial time algorithm that for any $r \in R$ returns a random element in $f^{-1}(r)$.*

The authors of [6] show that if $f : S \to \mathbb{G}$ is an admissible encoding, then the Boneh-Franklin scheme is secure with $H(m) = f(h(m))$, in the random oracle model for $h : \{0,1\}^* \mapsto S$. Since the function $f_{\mathbb{G}}$ is easily seen to be an admissible encoding under the previous definition, this shows that with $H_{\mathbb{G}}$ Boneh-Franklin is provably secure in the random oracle model for h.

In this paper, we introduce a new notion of admissible encoding that is more general than the previous notion. This enables to use Icart or SWU's function that work for any elliptic curve, instead of only super-singular ones. Moreover, we show that the resulting hash function is indifferentiable from a random oracle; therefore, it can be used in any cryptosystem, not only in Boneh-Franklin.

1.2 The Shallue-Woestijne-Ulas Algorithm

The first algorithm to generate elliptic-curve points in *deterministic* polynomial time was published in ANTS 2006 by Shallue and Woestijne [26]. The rational maps were later simplified and generalized to hyper-elliptic curves by Ulas in [28]; we recall those latter rational maps in the following theorem.

Theorem 1 (Ulas [28]). *Let $\mathbb{F}_q$ be a field and let $g(x) := x^3 + ax + b$, where $ab \neq 0$. Let:*

$$X_1(t,u) = u \qquad X_2(t,u) = \frac{-b}{a}\left(1 + \frac{1}{t^4 g(u)^2 + t^2 g(u)}\right)$$
$$X_3(t,u) = t^2 g(u) X_2(t,u) \qquad U(t,u) = t^3 g(u)^2 g(X_2(t,u))$$

Then

$$U(t,u)^2 = g(X_1(t,u)) \cdot g(X_2(t,u)) \cdot g(X_3(t,u)) \tag{2}$$

From equation (2) at least one of $g(X_1(t,u))$, $g(X_2(t,u))$ and $g(X_3(t,u))$ must be a quadratic residue. Therefore, either $X_1(t,u)$, $X_2(t,u)$ or $X_3(t,u)$ is the abscissa of a point on the curve $y^2 = g(x)$. Computing the corresponding y requires to compute a square root; when $q = 3 \mod 4$, this is simply an exponentiation. However for $q = 1 \mod 4$, no deterministic algorithm is known for computing square roots. The Tonelli-Shanks algorithm [24] requires to use a non-quadratic residue, and it is unknown how to generate such non-quadratic residue deterministically. One of the main contribution of [26] is to show a deterministic variant of the Tonelli-Shanks algorithm when an equality of the form $a_0 a_1 a_2 = b^2$ holds, as with (2).

In Section 5.5, we provide a simplified version of the Ulas maps when $q = 3 \mod 4$. This enables to slightly improve the efficiency of the SWU algorithm. In Appendix I we also recall the Shallue-Woestijne algorithm in characteristic 2.

1.3 Icart's Function

We consider the field $\mathbb{F}_{p^n}$ where $p > 3$ and $p^n = 2 \mod 3$. Let E be an elliptic curve over $\mathbb{F}_{p^n}$ with equation:

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

where $a, b \in \mathbb{F}_{p^n}$. In [19], Icart defines the function $f_{a,b} : \mathbb{F}_{p^n} \mapsto E$, with $f_{a,b}(u) = (x, y)$ where:

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

for $u \neq 0$, and $f_{a,b}(0) = \mathcal{O}$, the neutral element of the elliptic curve. It is easy to check that $f_{a,b}(u)$ is indeed a point of E for any $u \in \mathbb{F}_{p^n}$. We recall the following properties of $f_{a,b}$:

Lemma 1 (Icart). *The function $f_{a,b}$ is computable in deterministic polynomial time. For any point $P \in \mathsf{Im}(f_{a,b})$, the set $f_{a,b}^{-1}(P)$ is computable in polynomial time and $|f_{a,b}^{-1}(P)| \leq 4$. Moreover $p^n/4 < |\mathsf{Im}(f_{a,b})| < p^n$.*

We note that Icart's function can also be defined in a field of characteristic 2 (see Appendix F). Given a hash function $h : \{0,1\}^* \mapsto \mathbb{F}_{p^n}$, the following hash function $H : \{0,1\}^* \mapsto E$ is defined in [19]:

$$H(m) = f_{a,b}(h(m))$$

It is shown in [19] that H is one-way if h is one-way. However, it is easy to see that $H(m)$ does *not* behave like a random oracle when the underlying function h is seen as a random oracle; this is because f_{ab} does not reach all the points of E; moreover $f_{a,b}(u)$ is not uniformly distributed in $\mathsf{Im} f_{a,b}$ when u is uniformly distributed in $\mathbb{F}_{p^n}$.

2 Definitions

We recall the notion of indifferentiability introduced by Maurer *et al.* in [22] (see also [15]). We define an *ideal primitive* as an algorithmic entity which receives inputs from one of the parties and delivers its output immediately to the querying party. A *random oracle* [3] into a finite set S is an ideal primitive which provides a random output in S for each new query; identical input queries are given the same answer.

The notion of indifferentiability enables to show that an ideal primitive H (for example, a random oracle into an elliptic-curve E) can be replaced by a construction C that is based on some other ideal primitive h (for example, a random oracle into $\mathbb{F}_p$), and any cryptosystem secure with H remains secure with C and $\mathcal{H}$; see Figure 1 for an illustration of the definition.

Definition 2 (Indifferentiability [22]). *A Turing machine C with oracle access to an ideal primitive h is said to be $(t_D, t_S, q, \varepsilon)$-indifferentiable from an ideal primitive H if there exists a simulator S with oracle access to H and running in time at most t_S, such that for any distinguisher $\mathcal{D}$ running in time at most t_D and making at most q queries, it holds that:*

$$\left|\Pr\left[\mathcal{D}^{C^h,h} = 1\right] - \Pr\left[\mathcal{D}^{H,S^H} = 1\right]\right| < \varepsilon$$

C^h is said to be indifferentiable from H if ε is a negligible function of the security parameter k, for polynomially bounded q, t_D and t_S.

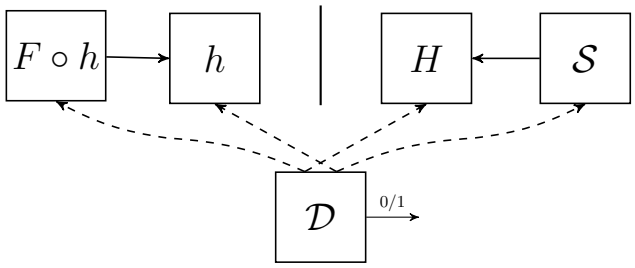


Fig. 1. The indifferentiability notion, illustrated with construction $C^h = F \circ h$ for some function F, and random oracles h and H.

It is shown in [22] that the indifferentiability notion is the "right" notion for substituting one ideal primitive by a construction based on another ideal primitive. That is, if C^h is indifferentiable from an ideal primitive H, then C^h can replace H in any cryptosystem, and the resulting cryptosystem is at least as secure in the h model as in the H model; see [22] or [15] for a proof.

We also recall the definition of statistically indistinguishable distributions.

Definition 3. *Let X and Y be two random variables over a set S. The distribution of X and Y are ε-statistically indistinguishable if:*

$$\sum_{s \in S} \big| \Pr[X = s] - \Pr[Y = s] \big| \leq \epsilon.$$

The two distributions are statistically indistinguishable if ε is a negligible function of the security parameter.

We introduce the definition of α-bounded distribution.

Definition 4. *Let X be a random variable over a finite set R. The distribution of X is α-bounded if the inequality $\Pr[X = r] \leq \alpha/|R|$ holds for any $r \in R$.*

3 Admissible Encoding and Indifferentiability

Our goal is to construct a hash function into elliptic curves that is indifferentiable from a random oracle. First, we introduce our new notion of *admissible encoding*.

Definition 5 (Admissible Encoding). *A function $F : S \mapsto R$ is an ε-admissible encoding if it satisfies the following properties:*

1. *Computable: F is computable in deterministic polynomial time;*
2. *Regular: for s uniformly distributed in S, the distribution of $F(s)$ is ε-statistically indistinguishable from the uniform distribution in R.*
3. *Samplable: there is an efficient randomized algorithm $\mathcal{I}$ such that for any $r \in R$, $\mathcal{I}(r)$ induces a distribution that is ε-statistically indistinguishable from the uniform distribution in $F^{-1}(r)$.*

F is an admissible encoding if ε is a negligible function of the security parameter.

Our new definition can be seen as a generalization of the definition from [6] recalled in Section 1.1. Namely criteria 2 of our definition gives:

$$\sum_{r \in R} \left| \Pr_s[f(s) = r] - \frac{1}{|R|} \right| = \sum_{r \in R} \left| \frac{|f^{-1}(r)|}{|S|} - \frac{1}{|R|} \right| \leq \varepsilon \tag{3}$$

whereas criteria 2 from Definition 1 requires $|f^{-1}(r)| = |S|/|R|$ for all $r \in R$; this is equivalent to $\varepsilon = 0$ in (3).

The following theorem shows that if $F : S \mapsto R$ is an admissible encoding, then the hash function $H : \{0,1\}^* \to R$ with:

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

is indifferentiable from a random oracle into R when $h : \{0,1\}^* \to S$ is seen as a random oracle. This shows that the construction $H(m) = F(h(m))$ can replace a random oracle into R, and the resulting scheme remains secure in the random oracle model for h.

Theorem 2. *Let $F : S \mapsto R$ be an ε-admissible encoding. The construction $H(m) = F(h(m))$ is $(t_D, t_S, q, \varepsilon')$-indifferentiable from a random oracle, in the random oracle model for $h : \{0,1\}^* \mapsto S$, with $\varepsilon' = 4q\varepsilon$ and $t_S = 2q \cdot t_I$, where t_I is the maximum running time of F's sampling algorithm.*

3.1 Proof of Theorem 2

Let $F : S \mapsto R$ be an ε-admissible encoding with sampling algorithm $\mathcal{I}$. We first prove the following Lemma (see Appendix A).

Lemma 2. *For r uniformly distributed in R, the distribution of $\mathcal{I}(r)$ is 2ε-statistically indistinguishable from the uniform distribution in S.*

Lemma 2 shows that the sampling algorithm $\mathcal{I}(r)$ of an admissible encoding from S to R gives a distribution that is statistically indistinguishable from uniform in S. From this property we can obtain a slightly more general definition of admissible encoding, which we call admissible encoding v2.

Definition 6 (Admissible Encoding v2). *A function $F : S \mapsto R$ is said to be an ε-admissible encoding v2 if it satisfies the following properties:*

1. *Computable: F is computable in deterministic polynomial time;*
2. *Invertible: there exists a probabilistic polynomial time algorithm $\mathcal{I}_F$ such that $\mathcal{I}_F(r) \in F^{-1}(r) \cup \{\perp\}$ for all $r \in R$, and the distribution of $\mathcal{I}_F(r)$ is ε-statistically indistinguishable from the uniform distribution in S when r is uniformly distributed in R.*

F is an admissible encoding v2 if ε is a negligible function of the security parameter.

From Lemma 2 we have that an ε-admissible encoding is a 2ε-admissible encoding v2. However we keep Definition 5 because as a generalization of the original Boneh-Franklin definition it seems easier to understand. We now prove the following lemma; combined with the previous observation this proves Theorem 2.

Lemma 3. *Let $F : S \mapsto R$ be an ε-admissible encoding v2. The construction $H(m) = F(h(m))$ is $(t_D, t_S, q, \varepsilon')$-indifferentiable from a random oracle, in the random oracle model for $h : \{0,1\}^* \mapsto S$, with $\varepsilon' = 2q\varepsilon$ and $t_S = 2q \cdot t_I$, where t_I is the maximum running time of F's sampling algorithm.*

Proof. We first describe our simulator; then we prove the indistinguishability property. As illustrated in Figure 1, the simulator must simulate random oracle h to the distinguisher $\mathcal{D}$, and the simulator has oracle access to random oracle H. It maintains a list L of previously answered queries. Our simulator is based on sampling algorithm $\mathcal{I}$ from F, as an admissible encoding v2.

Simulator $\mathcal{S}$:
Input: $m \in \{0,1\}^*$
Output: $s \in S$

1. If $(m, s) \in L$, then return s
2. Query $H(m) = r$
3. Let $s \leftarrow \mathcal{I}(r)$
4. Append (m, s) to L and return s.

We must show that the systems (C^h, h) and $(H, \mathcal{S}^H)$ are indistinguishable. We consider a distinguisher making at most q queries. Without loss of generality, we can assume that the distinguisher makes all queries to $h(m)$ (or $\mathcal{S}^H$) for which there was a query to $C^h(m)$ (or $H(m)$), and conversely; this gives a total of at most $2q$ queries. We can then describe the full interaction between the distinguisher and the system as a sequence of triples:

$$\mathsf{View} = (m_i, H_i, h_i)_{1 \leq i \leq 2q}$$

In system (C^h, h), we have that the h_i's are uniformly and independently distributed in S, and $H_i = F(h_i)$ for all i.

In system $(H, \mathcal{S}^H)$, by definition of algorithm $\mathcal{I}$ we have that $H_i = F(h_i)$ except if $h_i = \perp$. By definition of an ε-admissible encoding v2, since the distribution of $H_i = H(m_i)$ is uniform in R, the distribution of $h_i = \mathcal{I}(H_i)$ is ε-indistinguishable from the uniform distribution in S. Therefore, the statistical distance between View in system (C^h, h) and View in system $(H, \mathcal{S}^H)$ is at most $2q\varepsilon$. This terminates the proof of Lemma 3. □

4 Weak Encoding

It is easy to see that Icart's function f_{ab} is *not* an admissible encoding into the elliptic-curve E, since as mentioned previously $\mathsf{Im} f_{ab}$ covers only a fraction of the elliptic-curve points. Therefore, we introduce a weaker notion: *weak encoding*.

Definition 7 (Weak Encoding). *A function $f : S \mapsto R$ is said to be an α-weak encoding if it satisfies the following properties:*

1. *Computable: f is computable in deterministic polynomial time.*
2. *α-bounded: for s uniformly distributed in S, the distribution of $f(s)$ is α-bounded in R.*
3. *Samplable: there is an efficient randomized algorithm $\mathcal{I}$ that $\mathcal{I}(r)$ induces the uniform distribution in $f^{-1}(r)$ for any $r \in R$. Additionally $\mathcal{I}(r)$ returns $N_r = |f^{-1}(r)|$ for all $r \in R$.*

f is a weak encoding if α is a polynomial function of the security parameter.

The difference with an admissible encoding is that in criteria 2, the distribution of $f(s)$ is only required to be α-bounded (see Def. 4), instead of being ε-indistinguishable from the uniform distribution. More precisely criteria 2 for a weak encoding requires:

$$\forall r \in R, \quad \Pr_s[f(s) = r] = \frac{|f^{-1}(r)|}{|S|} \leq \frac{\alpha}{|R|} \tag{4}$$

instead of inequality (3). Note that inequality (3) with negligible ε does not necessarily imply inequality (4) with polynomial α; this means that under the previous definitions an admissible encoding is not necessarily a weak encoding. In the next section, we provide a slightly more general definition of weak encoding (called weak encoding v2), for which the implication holds.

From inequality (4) we have that any invertible function with bounded pre-image and bounded $|R|/|S|$ is a weak encoding; in particular, this is the case for Icart's function f_{ab}.

Lemma 4. *Let $f : S \to R$ be a polynomially computable function. Let $B = \max\{|f^{-1}(r)|, r \in R\}$. Assume that there exists a polynomial-time algorithm* Inv *that for any $r \in R$ outputs the set $f^{-1}(r)$. Then f is an α-weak encoding, with $\alpha = B \cdot |R|/|S|$. In particular, Icart's function f_{ab} is an α-weak encoding from $\mathbb{F}_{p^n}$ to $E_{a,b}$ with $\alpha = 4N/p^n$, where N is the order of $E_{a,b}$.*

Proof. We have $\Pr_s[f(s) = r] = |f^{-1}(r)|/|S| \leq B/|S| = \alpha/|R|$ by taking $\alpha := B \cdot |R|/|S|$; therefore, the distribution of $f(s)$ for uniform $s \in S$ is α-bounded in R. Given $\mathsf{Inv}(r) = f^{-1}(r)$, the algorithm $\mathcal{I}(r)$ simply generates a random element in the set $f^{-1}(r)$, and lets $N_r = |f^{-1}(r)|$. The result for Icart's function follows from Lemma 1. □

4.1 From Weak Encoding to Admissible Encoding

We show that when the output set is a group (*e.g.*, the group of points of an elliptic curve), we can turn a weak encoding into an admissible encoding.

Theorem 3 (Weak → Admissible Encoding). *Let $\mathbb{G}$ be a cyclic group of order N and let G be a generator of $\mathbb{G}$. Let $f : S \to \mathbb{G}$ be an α-weak encoding. Then the function $F : S \times \mathbb{Z}_N \to \mathbb{G}$ with $F(s, x) := f(s) + x.G$ is an ε-admissible encoding into $\mathbb{G}$, with $\varepsilon = (1 - 1/\alpha)^t$ for any t polynomial in the security parameter k, and $\varepsilon = 2^{-k}$ for $t = \alpha \cdot k$.*

Using the construction $F(s, x) = f(s) + x.G$ we obtain in Section 5 a hash function into an elliptic curve that is indifferentiable from a random oracle. We note that it is easy to generalize the previous construction to a group with a finite set of generators.

4.2 Proof of Theorem 3

We must show that F is an admissible encoding. We first consider criteria 2. We have:

$$F^{-1}(P) = \big\{(s, x) \in S \times \mathbb{Z}_N \mid f(s) + x.G = P\big\} = \big\{\, (s, \log_G(P - f(s))) \mid s \in S\big\} \tag{5}$$

where $\log_G$ denotes the discrete logarithm in base G. This gives $|F^{-1}(P)| = |S|$ for all $P \in \mathbb{G}$. This implies that $F(s, x)$ is uniformly distributed in $\mathbb{G}$ for uniformly distributed $(s, x) \in S \times \mathbb{Z}_N$.

To prove that criteria 3 is satisfied for F, we must construct a sampling algorithm $\mathcal{I}_F(P)$ such that for any $P \in \mathbb{G}$ the distribution of $\mathcal{I}_F(P)$ is ε-statistically indistinguishable from the uniform distribution in $F^{-1}(P)$. Given $P \in \mathbb{G}$, one could first generate a random $x \in \mathbb{Z}_N$, compute $Q = P - x.G$ and then use f's sampling algorithm $\mathcal{I}_f$ to return $s = \mathcal{I}_f(P - x.G)$, which would give $f(s) + x.G = P$ as required. However, we don't necessarily have $P - x.G \in \mathsf{Im} f$, which implies that such s may not exist; moreover, one can see that the distribution of s would not necessarily be uniform in S, because the pre-image sets of f do not necessarily have the same cardinality. In the following we first construct from $\mathcal{I}_f$ a different sampling algorithm $\mathcal{I}'_f$ such that when given as input a uniformly distributed Q, it returns either $\perp$ or a uniformly distributed $s \in S$. Then we simply call $\mathcal{I}'_f$ sufficiently many times with $Q = P - x.G$ for random $x \in \mathbb{Z}_N$ so that $\perp$ appears with negligible probability only.

We consider an α-weak encoding $f : S \mapsto R$ with sampling algorithm $\mathcal{I}_f$. As shown below, the new sampling algorithm $\mathcal{I}'_f$ is constructed from $\mathcal{I}_f$ by artificially aborting with some well chosen probability (dependent on the input); this is to ensure that all the pre-images of f appear with the same probability.

Algorithm $\mathcal{I}'_f(r)$:
Input: $r \in R$
Output: $s \in S$ such that $f(s) = r$ or $s = \perp$

1. Let $N_r = |f^{-1}(r)|$ returned by $\mathcal{I}_f(r)$, and let $\delta_r = |R| \cdot N_r / (\alpha \cdot |S|)$.
2. With probability $1 - \delta_r$ return $\perp$.
3. Return $\mathcal{I}_f(r)$.

Note that since f is an α-weak encoding, we have using (4):

$$\delta_r = \frac{N_r \cdot |R|}{\alpha \cdot |S|} = |f^{-1}(r)| \cdot \frac{|R|}{\alpha \cdot |S|} \leq \frac{\alpha \cdot |S|}{|R|} \cdot \frac{|R|}{\alpha \cdot |S|}$$

which gives $\delta_r \leq 1$ as required. The following lemma shows that $\mathcal{I}'_f$ has the required properties (see Appendix B for the proof).

Lemma 5. *The algorithm $\mathcal{I}'_f$ has the following properties:*

1. $\Pr_r[\mathcal{I}'_f(r) \neq \perp] = 1/\alpha$
2. *For uniformly distributed $r \in R$, the distribution of $\mathcal{I}'_f(r)$ conditioned on $\mathcal{I}'_f(r) \neq \perp$ is uniform in S.*

Actually, we show that the existence of such algorithm $\mathcal{I}'_f$ enables to get a slightly more general definition of a weak encoding that we call a weak encoding v2. We note that under this new definition, an admissible encoding v2 is also a weak encoding v2.

Definition 8 (Weak Encoding v2). *A function $f : S \mapsto R$ is said to be an (α, ε)-weak encoding v2 if it satisfies the following properties:*

1. *Computable: f is computable in deterministic polynomial time.*
2. *Partially invertible: there exists a probabilistic polynomial time algorithm $\mathcal{I}'_f$ such that $\mathcal{I}'_f(r) \in f^{-1}(r) \cup \{\perp\}$ for all $r \in R$, with the following properties:*
   (a) $\Pr_r[\mathcal{I}'_f(r) \neq \perp] \geq 1/\alpha$
   (b) For uniformly distributed $r \in R$, the distribution of $\mathcal{I}'_f(r)$ conditioned on $\mathcal{I}'_f(r) \neq \perp$ is ε-statistically indistinguishable from uniform in S.

f is a weak encoding if α is a polynomial function of the security parameter and ε is a negligible function of the security parameter.

From Lemma 5 we have that an α-weak encoding is also an (α, ε)-weak encoding v2 with $\varepsilon = 0$. However as previously we keep Definition 7 because it seems more natural and easier to understand. We now prove the following lemma; combined with the previous observation, this proves Theorem 3.

Lemma 6. *Let $\mathbb{G}$ be a cyclic group of order N and let G be a generator of $\mathbb{G}$. Let $f : S \to \mathbb{G}$ be an (α, ε)-weak encoding v2. Then the function $F : S \times \mathbb{Z}_N \to \mathbb{G}$ with $F(s, x) := f(s) + x.G$ is an ε'-admissible encoding into $\mathbb{G}$, with $\varepsilon' = (1 - 1/\alpha)^t + \varepsilon$ for any t polynomial in the security parameter k, and $\varepsilon' = 2^{-k} + \varepsilon$ for $t = \alpha \cdot k$.*

Proof. From (5) we have that $|F^{-1}(P)| = |S|$ for all $P \in \mathbb{G}$, which implies that $F(s, x)$ is uniformly distributed in $\mathbb{G}$ for uniformly distributed $(s, x) \in S \times \mathbb{Z}_N$. Our sampling algorithm $\mathcal{I}_F(P)$ is constructed as follows, given algorithm $\mathcal{I}'_f$ from f:

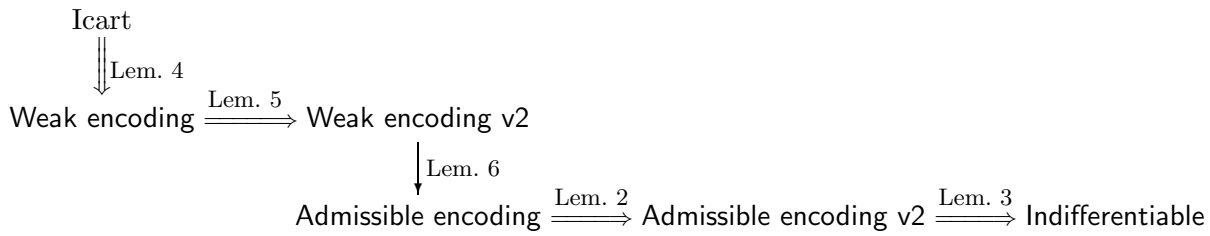
Algorithm $\mathcal{I}_F$:
Input: $P \in \mathbb{G}$.
Output: $(s, x) \in S \times \mathbb{Z}_N$ such that $P = F(s, x) = f(s) + x.G$, or $\perp$

1. For $i = 1$ to t:
   (a) Randomly choose $x \in \mathbb{Z}_N$.
   (b) Let $s \leftarrow \mathcal{I}'_f(P - x.G)$
   (c) If $s \neq \perp$, return (s, x)
2. Return $\perp$.

We must show that for any $P \in \mathbb{G}$, the distribution of (s, x) is statistically close to uniform in $F^{-1}(P)$. From the definition of f as an (α, ε)-weak encoding v2 and the uniform distribution of $P - x.G \in \mathbb{G}$, we have that $s = \perp$ at step i with probability at most $1 - 1/\alpha$. Therefore algorithm $\mathcal{I}_F$ eventually outputs $s = \perp$ with probability at most $(1 - 1/\alpha)^t$. Moreover from the definition of f as an (α, ε)-weak encoding v2, conditioned on $s \neq \perp$ the distribution of s in (s, x) is ε-statistically indistinguishable from uniform in S. From equation (5) this implies that the distribution of (s, x) conditioned on $s \neq \perp$ is also ε-statistically indistinguishable from uniform in $F^{-1}(P)$. Therefore, for any $P \in \mathbb{G}$ the distribution of $\mathcal{I}_F(P)$ is ε'-statistically close to uniform in $F^{-1}(P)$, with $\varepsilon' = (1 - 1/\alpha)^t + \varepsilon$. Taking $t = \alpha \cdot k$, we can take $\varepsilon' = 2^{-k} + \varepsilon$; this terminates the proof of Lemma 6. □

4.3 Summary

We summarize below the relations between the various definitions introduced in this paper. As shown below this enables to construct an indifferentiable hash function from Icart's function; we describe the concrete construction and various extensions in the next section.



5 Our Constructions

Given an elliptic-curve defined $\mathbb{E}$ over $\mathbb{F}_p$ with N points and a generator G, using Theorem 3 we can define our hash function $H : \{0,1\}^* \to \mathbb{E}$ with:

$$H(m) := f(h_1(m)) + h_2(m).G$$

where $h_1 : \{0,1\}^* \to \mathbb{F}_p$ and $h_2 : \{0,1\}^* \to \mathbb{Z}_N$ are two hash functions, and f is any weak encoding into $\mathbb{E}$, such as Icart's function. The following theorem is the main result of this paper.

Theorem 4. *Let p be a k-bit prime and let $E : y^2 = x^3 + ax + b$ be an elliptic curve over $\mathbb{F}_p$. Let $f_{a,b} : \mathbb{F}_p \mapsto E$ be Icart's function. Let G be a generator of E of order N. The construction $H(m) := f_{a,b}(h_1(m)) + h_2(m).G$ is $(t_D, t_S, q_D, \varepsilon)$-indifferentiable from a random oracle when the hash functions $h_1 : \{0,1\}^* \to \mathbb{F}_p$ and $h_2 : \{0,1\}^* \to \mathbb{Z}_N$ are seen as random oracles, with $\varepsilon = 4 \cdot q_D \cdot 2^{-k}$.*

Proof. From Theorem 3 the function $F(u, x) = f(u) + x.G$ is an ε-admissible encoding from $\mathbb{F}_p \times \mathbb{Z}_N$ into E, with $\varepsilon = 2^{-k}$. From Theorem 2 this implies that $H(m)$ is $(t_D, t_S, q_D, \varepsilon)$-indifferentiable from a random oracle, with $\varepsilon = 4 \cdot q_D \cdot 2^{-k}$. □

This implies that our construction $H(m)$ can be used in any cryptosystem provably secure with random oracles into elliptic curves, and the resulting cryptosystem remains secure in the random oracle model for h_1 and h_2. We note that to prevent timing attacks (as in [9]), our construction H can easily be implemented in constant time, since both Icart's function and scalar multiplication can be implemented in constant time.

5.1 Extension to a Prime Order Subgroup

In many applications only a prime order subgroup of E is used, so we show how to construct a random oracle into a subgroup. Let E be an elliptic-curve over $\mathbb{F}_p$ with N points, and let $\mathbb{G}$ be a subgroup of prime order q and generator G. Let ℓ be the co-factor, *i.e.* $N = \ell \cdot q$. We require that q does not divide ℓ (*i.e.* that q^2 does not divide N). We use the construction $H : \{0,1\}^* \mapsto \mathbb{G}$ with:

$$H(m) := \ell.f_{ab}(h_1(m)) + h_2(m).G$$

with $h_1 : \{0,1\}^* \mapsto \mathbb{F}_p$ and $h_2 : \{0,1\}^* \mapsto \mathbb{Z}_q$.

Lemma 7. *H is $(t_D, t_S, q_D, \varepsilon)$-indifferentiable from a random oracle, in the random oracle model for h_1 and h_2, with $\varepsilon = 4 \cdot q_D \cdot 2^{-k}$.*

First, we show that multiplication by a cofactor is an ε-admissible encoding, with $\varepsilon = 0$. Then we show that the composition of an admissible encoding with a weak encoding is a weak encoding v2. This shows that the function $\ell.f_{ab}(u)$ is a weak encoding v2 into $\mathbb{G}$. Applying Lemma 6 and Theorem 2, this proves Lemma 7. See Appendix C for the full proof.

5.2 Extension to Random Oracles into Strings

The constructions in the previous sections are based on hash functions into $\mathbb{F}_{p^n}$ or $\mathbb{Z}_N$. However in practice a hash function outputs a fixed length string in $\{0,1\}^\ell$. We can modify our construction as follows. We consider an elliptic curve $E_{a,b}(\mathbb{F}_p)$ of prime order N and generator G, with p a $2k$-bit prime. We define the hash function $H : \{0,1\}^* \mapsto E_{a,b}(\mathbb{F}_p)$ with:

$$H(m) := f_{a,b}\big(h_1(m) \mod p\big) + \big(h_2(m) \mod N\big).G$$

where h_1 and h_2 are two hash functions from $\{0,1\}^*$ to $\{0,1\}^{3k}$.

Lemma 8. *The previous hash function H is $(t_D, t_S, q_D, \varepsilon)$-indifferentiable from a random oracle, in the random oracle model for h_1 and h_2, with $\varepsilon = 14 \cdot q_D \cdot 2^{-k}$.*

First, we show that reduction modulo p is an admissible encoding from $\{0,1\}^\ell$ to $\mathbb{Z}_p$ if $2^\ell \gg p$; the same holds modulo N. Secondly, we show that the composition of two admissible encodings remains an admissible encoding v2. This shows that $F(u, v) = f(u \mod p) + (v \mod N).G$ is also an admissible encoding v2 into E. Applying Lemma 3, this proves Lemma 8. See Appendix D for the full proof.

Remark 1. Instead of using two strings of $3k$-bit each, we can use a single string of $5k$-bit only. Namely one can show that the construction:

$$H'(m) := f_{a,b}\big(h(m) \mod p\big) + \big(h(m) \mod N\big).G$$

with $h : \{0,1\}^* \mapsto \{0,1\}^{5k}$, is indifferentiable from a random oracle, in the random oracle model for h.

5.3 Extension to Primes $p = 2^\ell - \omega$

We show a slightly more efficient construction for primes p of the form $p = 2^\ell - \omega$ for small ω, as used for example in the FIPS standardized curves [25]. Let $E_{a,b}(\mathbb{F}_p)$ be an elliptic-curve of prime order N and generator G. Our construction $H : \{0,1\}^* \mapsto E_{a,b}(\mathbb{F}_p)$ is as follows:

$$H(m) := f_{a,b}\big(h_1(m) \mod p\big) + h_2(m).G$$

where h_1 and h_2 are two hash functions from $\{0,1\}^*$ to $\{0,1\}^\ell$. In other words, the output size ℓ of h_1 and h_2 is the same as the bitsize of p, as opposed to the previous section in which we took $\ell = 3k$ for a $2k$-bit prime p.

Lemma 9. *H is $(t_D, t_S, q_D, \varepsilon)$-indifferentiable from a random oracle, in the random oracle model for h_1 and h_2, with $\varepsilon = 8 \cdot q_D \cdot (2^{-\ell/2} + 2^{-\ell} \cdot \omega)$.*

The proof is similar to the proof of Lemma 8, except that since $p \simeq 2^\ell$, reduction modulo p is now an admissible encoding v2 from $\{0,1\}^\ell$ to $\mathbb{Z}_p$; the same holds modulo N since Hasse's bound implies that $N \simeq 2^\ell$ as well. See Appendix E for the full proof.

Remark 2. We only need a single hash function $h : \{0,1\}^* \to \{0,1\}^\ell$ since we can obtain h_1 and h_2 by prepending a bit as input of h.

5.4 Extension to Elliptic Curves over $\mathbb{F}_{2^\ell}$

Let E be an elliptic-curve defined over $\mathbb{F}_{2^\ell}$ with prime order N and generator G. We refer to Appendix F for a description of Icart's function in characteristic 2. When working in $\mathbb{F}_{2^\ell}$ we can use a hash function into $\{0,1\}^\ell$ directly. More precisely, our construction is $H : \{0,1\}^* \mapsto E_{a,b}(\mathbb{F}_{2^\ell})$ is defined as follows:

$$H(m) := f_{a,b}\big(h_1(m)\big) + h_2(m).G$$

Lemma 10. *H is $(t_D, t_S, q_D, \varepsilon)$-indifferentiable from a random oracle, in the oracle model for h_1 and h_2, with $\varepsilon = 8 \cdot q_D \cdot 2^{-\ell/2}$.*

The proof of Lemma 10 is exactly the same as the proof of Lemma 9 with $\omega = 0$.

5.5 Using the Shallue-Woestijne-Ulas (SWU) Algorithm

In this section, we describe a slightly simpler variant of the SWU algorithm over $\mathbb{F}_q$ recalled in Section 1.2, for $q = 3 \mod 4$. Note that this condition is usually satisfied in practice, since it enables to compute square roots efficiently. Our simplified Ulas maps are as follows:

Theorem 5 (Simplified Ulas maps). *Let $\mathbb{F}_q$ be a field and let $g(x) := x^3 + ax + b$, where $ab \neq 0$. Let:*

$$X_2(t) = \frac{-b}{a}\left(1 + \frac{1}{t^4 - t^2}\right), \qquad X_3(t) = -t^2 X_2(t), \qquad U(t) = t^3 g(X_2(t))$$

Then $U(t)^2 = -g\big(X_2(t)\big) \cdot g\big(X_3(t)\big)$

See Appendix G for the proof. When $q = 3 \mod 4$ we have that -1 is a quadratic non-residue in $\mathbb{F}_q$, and therefore either $g(X_2(t))$ or $g(X_3(t))$ must be a quadratic residue. This leads to the following pseudo-code for our simplified SWU algorithm:

Simplified SWU algorithm:
Input: $\mathbb{F}_q$ such that $q = 3 \mod 4$, parameters a, b and input $t \in \mathbb{F}_q$
Output: $(x, y) \in E_{a,b}(\mathbb{F}_q)$

1. $\alpha \leftarrow -t^2$
2. $X_2 \leftarrow \frac{-b}{a}\left(1 + \frac{1}{\alpha^2+\alpha}\right)$
3. $X_3 \leftarrow \alpha \cdot X_2$
4. $h_2 \leftarrow (X_2)^3 + a \cdot X_2 + b$; $h_3 \leftarrow (X_3)^3 + a \cdot X_3 + b$
5. If h_2 is a square, return $(X_2, h_2^{(q+1)/4})$, otherwise return $(X_3, h_3^{(q+1)/4})$

Our simplified SWU algorithm defines a function $f'_{a,b} : \mathbb{F}_q \to E$ which is also a weak encoding into the curve.

Lemma 11. *The function f'_{ab} has pre-image size at most 8 and can be inverted on its image in polynomial time. Then $f'_{a,b}$ is an α-weak encoding with $\alpha = 8N/q$, where N is the elliptic-curve order.*

See Appendix H for the proof. From Lemma 11 one can use SWU's function $f'_{a,b}$ instead of Icart's function $f_{a,b}$ in all previous constructions. In Appendix I, we also recall the Shallue-Woestijne algorithm in characteristic 2, so that Shallue-Woestijne can also be used in the construction of Section 5.4.

References

1. J. Baek and Y. Zheng. Identity-based threshold decryption. In Bao et al. [2], pages 262–276.
2. F. Bao, R. H. Deng, and J. Zhou, editors. *Public Key Cryptography - PKC 2004, 7th International Workshop on Theory and Practice in Public Key Cryptography, Singapore, March 1-4, 2004*, volume 2947 of *Lecture Notes in Computer Science*. Springer, 2004.
3. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.
4. E. Biham, editor. *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, volume 2656 of *Lecture Notes in Computer Science*. Springer, 2003.
5. A. Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. In Desmedt [16], pages 31–46.
6. D. Boneh and M. K. Franklin. Identity-based encryption from the weil pairing. In J. Kilian, editor, *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.
7. D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In Biham [4], pages 416–432.
8. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the weil pairing. In C. Boyd, editor, *ASIACRYPT*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532. Springer, 2001.
9. C. Boyd, P. Montague, and K. Q. Nguyen. Elliptic curve based password authenticated key exchange protocols. In V. Varadharajan and Y. Mu, editors, *ACISP*, volume 2119 of *Lecture Notes in Computer Science*, pages 487–501. Springer, 2001.
10. X. Boyen. Multipurpose identity-based signcryption (a swiss army knife for identity-based cryptography). In D. Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 383–399. Springer, 2003.
11. V. Boyko, P. D. MacKenzie, and S. Patel. Provably secure password-authenticated key exchange using diffie-hellman. In *EUROCRYPT*, pages 156–171, 2000.
12. R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, 2004.
13. J. C. Cha and J. H. Cheon. An identity-based signature from gap diffie-hellman groups. In Desmedt [16], pages 18–30.

14. B. Chevallier-Mames. An efficient cdh-based signature scheme with a tight security reduction. In V. Shoup, editor, *CRYPTO*, volume 3621 of *Lecture Notes in Computer Science*, pages 511–526. Springer, 2005.
15. J. S. Coron, Y. Dodis, C. Malinaud, and P. Puniya. Merkle-damgård revisited: How to construct a hash function. In *CRYPTO*, 2005.
16. Y. Desmedt, editor. *Public Key Cryptography - PKC 2003, 6th International Workshop on Theory and Practice in Public Key Cryptography, Miami, FL, USA, January 6-8, 2003, Proceedings*, volume 2567 of *Lecture Notes in Computer Science*. Springer, 2002.
17. C. Gentry and A. Silverberg. Hierarchical id-based cryptography. In Zheng [30], pages 548–566.
18. J. Horwitz and B. Lynn. Toward hierarchical identity-based encryption. In L. R. Knudsen, editor, *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 466–481. Springer, 2002.
19. T. Icart. How to hash into an elliptic-curve. In *CRYPTO 2009*. Publicly available on `http://eprint.iacr.org/`.
20. D. P. Jablon. Strong password-only authenticated key exchange. *SIGCOMM Comput. Commun. Rev.*, 26(5):5–26, 1996.
21. B. Libert and J.-J. Quisquater. Efficient signcryption with key privacy from gap diffie-hellman groups. In Bao et al. [2], pages 187–200.
22. U. M. Maurer, R. Renner, and C. Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In M. Naor, editor, *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 21–39. Springer, 2004.
23. A. Menezes, T. Okamoto, and S. A. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Transactions on Information Theory*, 39(5):1639–1646, 1993.
24. I. Niven, H. S. Zuckerman, and H. L. Montgomery. *An Introduction to the Theory of Numbers (5th edition ed.)*. Wiley. ISBN 0-471-62546-9.
25. N. I. of Standards and Technology. *FIPS PUB 186-3: Digital Signature Standard (DSS)*. june 2009.
26. A. Shallue and C. van de Woestijne. Construction of rational points on elliptic curves over finite fields. In F. Hess, S. Pauli, and M. E. Pohst, editors, *ANTS*, volume 4076 of *Lecture Notes in Computer Science*, pages 510–524. Springer, 2006.
27. V. Shoup. A new polynomial factorization algorithm and its implementation. *J. Symb. Comput.*, 20(4):363–397, 1995.
28. M. Ulas. Rational points on certain hyperelliptic curves over finite fields, 2007.
29. F. Zhang and K. Kim. Id-based blind signature and ring signature from pairings. In Zheng [30], pages 533–547.
30. Y. Zheng, editor. *Advances in Cryptology - ASIACRYPT 2002, 8th International Conference on the Theory and Application of Cryptology and Information Security, Queenstown, New Zealand, December 1-5, 2002, Proceedings*, volume 2501 of *Lecture Notes in Computer Science*. Springer, 2002.

A Proof of Lemma 2

We let $\omega \in \Omega$ be the random used by $\mathcal{I}$. We must show that $\delta \leq 2\varepsilon$, where:

$$\delta := \sum_{s \in S} \left| \Pr_{\omega,r}[\mathcal{I}(r) = s] - \frac{1}{|S|} \right|$$

Given $s \in S$, we have $\mathcal{I}(r) = s$ only if $r = f(s)$. Therefore, $\Pr_{\omega,r}[\mathcal{I}(r) = s] = \Pr_{\omega}[\mathcal{I}(r) = s]/|R|$ with $r = f(s)$. This gives:

$$\delta = \sum_{r \in R} \sum_{s \in F^{-1}(r)} \frac{1}{|R|} \left| \Pr_{\omega}[\mathcal{I}(r) = s] - \frac{|R|}{|S|} \right| \tag{6}$$

Since F is an ε-admissible encoding, by definition we have $\delta_1 \leq \varepsilon$, where:

$$\delta_1 := \sum_{r \in R} \left| \Pr_{s}[f(s) = r] - \frac{1}{|R|} \right| = \sum_{r \in R} \left| \frac{\left|F^{-1}(r)\right|}{|S|} - \frac{1}{|R|} \right| = \sum_{r \in R} \sum_{s \in F^{-1}(r)} \frac{1}{|R|} \left| \frac{|R|}{|S|} - \frac{1}{|F^{-1}(r)|} \right| \tag{7}$$

Since for all $r \in R$ the distribution of $\mathcal{I}(r)$ is ε-statistically indistinguishable from the uniform distribution in $F^{-1}(r)$, we have for all $r \in R$:

$$\sum_{s \in F^{-1}(r)} \left| \Pr_{\omega}[\mathcal{I}(r) = s] - \frac{1}{|F^{-1}(r)|} \right| < \varepsilon$$

which gives by summation over $r \in R$:

$$\delta_2 := \sum_{r \in R} \sum_{s \in F^{-1}(r)} \frac{1}{|R|} \left| \Pr_{\omega}[\mathcal{I}(r) = s] - \frac{1}{|F^{-1}(r)|} \right| < \varepsilon \tag{8}$$

From (6), (7) and (8) we have $\delta \leq \delta_1 + \delta_2$, which gives $\delta \leq \varepsilon + \varepsilon = 2\varepsilon$.

B Proof of Lemma 5

Given $r \in R$, we have that $\mathcal{I}'(r) \neq \perp$ with probability δ_r, where:

$$\delta_r = \frac{|R| \cdot |f^{-1}(r)|}{\alpha \cdot |S|}$$

This gives:

$$\Pr_r[\mathcal{I}'(r) \neq \perp] = \sum_{r \in R} \frac{1}{|R|} \cdot \delta_r = \frac{1}{\alpha \cdot |S|} \sum_{r \in R} |f^{-1}(r)| = \frac{1}{\alpha}$$

By definition we have that $\mathcal{I}(r)$ is uniformly distributed in the set $f^{-1}(r)$; this gives for any $r \in R$ and any $s \in f^{-1}(r)$:

$$\Pr[\mathcal{I}'(r) = s] = \delta_r \cdot \frac{1}{|f^{-1}(r)|} = \frac{|R|}{\alpha \cdot |S|}$$

Since $\mathcal{I}'(r) = s$ only if $r = f(s)$, this gives for any $s \in S$:

$$\Pr_r[\mathcal{I}'(r) = s] = \frac{1}{\alpha \cdot |S|}$$

and finally:

$$\Pr_r[\mathcal{I}'(r) = s | \mathcal{I}'(r) \neq \perp] = \frac{\Pr_r[\mathcal{I}'(r) = s]}{\Pr_r[\mathcal{I}'(r) \neq \perp]} = \frac{1}{|S|}$$

which shows that the distribution of $\mathcal{I}'(r)$ conditioned on $\mathcal{I}'(r) \neq \perp$ is uniform in S. This terminates the proof of Lemma 5.

C Proof of Lemma 7

We start by showing that multiplication by a co-factor is an admissible encoding.

Lemma 12. *The map $M_\ell : E \mapsto \mathbb{G}$ with $M_\ell(P) = \ell.P$ is an ε-admissible encoding, with $\varepsilon = 0$.*

Proof. The proof is the same as the proof of Lemma 5.1 in [6]. Since $|M_\ell^{-1}(Q)| = \ell$ for all $Q \in \mathbb{G}$, the distribution of $M_\ell(P)$ is uniform in $\mathbb{G}$ for uniform $P \in E$. Let G_E be a generator of E. Given $Q \in \mathbb{G}$, the sampling algorithm generates a random $u \in \mathbb{Z}_\ell$ and returns $P = (1/\ell) \cdot Q + u \cdot q \cdot G_E$. Here $1/\ell$ is computed in $\mathbb{Z}_q^*$. We have that $\ell.P = Q$ as required and P is uniformly distributed in $M_\ell^{-1}(Q)$. □

Secondly, we show that the composition of an admissible encoding with a weak encoding is a weak encoding v2.

Lemma 13. *Let $f : S \mapsto R$ be an α-weak encoding and $G : R \mapsto T$ be an ε-admissible encoding. Then $G \circ f$ is an $(\alpha/(1 - 2\varepsilon\alpha), 4\alpha\varepsilon)$-weak encoding v2 from S to T.*

Proof. We have that G is a 2ε-admissible encoding v2. Therefore given t uniformly distributed in T, the random variable $r = \mathcal{I}_G(t)$ is 2ε-statistically indistinguishable from the uniform distribution in R. Moreover we have that f is an $(\alpha, 0)$-weak encoding v2. Let $\mathcal{I}'_f$ be the corresponding sampling algorithm. For r uniformly distributed in R we have that $\Pr[\mathcal{I}'_f \neq \perp] \geq 1/\alpha$ and $s = \mathcal{I}'_f(r)$ is uniformly distributed in S conditioned on $s \neq \perp$.

We define the sampling algorithm for $G \circ f$ as $\mathcal{I}'_f \circ \mathcal{I}_G$. Therefore, we must study the distribution of $\mathcal{I}'_f(r)$ when r is only 2ε-statistically indistinguishable from the uniform distribution in R, instead of being uniform. We prove the following simple lemma.

Lemma 14. *Let X and Y be two random variables over a set Ω. Let δ be the statistical distance between X and Y. Let $A \subset \Omega$ and let $p_x = \Pr[X \in A]$ and $p_y = \Pr[Y \in A]$. Then $|p_y - p_x| \leq \delta$; moreover if $p_x > \delta$ then:*

$$\sum_{r \in \Omega} \left| \Pr[X = r | X \in A] - \Pr[Y = r | Y \in A] \right| \leq \frac{2\delta}{p_x}$$

Proof. Since the statistical distance between X and Y is δ, we have $|p_y - p_x| \leq \delta$; from $p_x > \delta$ this implies $p_y > 0$. We let

$$\delta_2 = \sum_{r \in \Omega} \left| \Pr[X = r | X \in A] - \Pr[Y = r | Y \in A] \right|$$

We have:

$$\begin{aligned}
\delta_2 &= \sum_{r \in A} \left| \frac{\Pr[X = r]}{p_x} - \frac{\Pr[Y = r]}{p_y} \right| \\
&\leq \sum_{r \in A} \left| \frac{\Pr[X = r]}{p_x} - \frac{\Pr[Y = r]}{p_x} \right| + \sum_{r \in A} \left| \frac{\Pr[Y = r]}{p_x} - \frac{\Pr[Y = r]}{p_y} \right| \\
&\leq \frac{\delta}{p_x} + p_y \cdot \left| \frac{1}{p_x} - \frac{1}{p_y} \right| = \frac{1}{p_x} \cdot (\delta + |p_y - p_x|) \leq \frac{2\delta}{p_x}
\end{aligned}$$

□

We apply Lemma 14 with $X = \mathcal{I}'_f(r)$ for r uniformly distributed in R and $Y = \mathcal{I}'_f(r)$ with $r = \mathcal{I}_G(t)$ for uniformly distributed $t \in T$. Moreover we take $\Omega = S \cup \{\perp\}$ and $A = S$. Since the statistical distance between X and Y is at most 2ε, we obtain:

$$\Pr[Y \neq \perp] \geq \Pr[X \neq \perp] - 2\varepsilon \geq \frac{1}{\alpha} - 2\varepsilon = \frac{1 - 2\varepsilon\alpha}{\alpha}$$

Using the notations of Lemma 14 we have $p_x \geq 1/\alpha$. Moreover by definition the distribution of X conditioned on $X \neq \perp$ is uniform in S. Therefore the statistical distance of $Y = \mathcal{I}'_f(r)$ conditioned on $Y \neq \perp$ from the uniform distribution in S is at most $2 \cdot (2\varepsilon)/p_x \leq 4\alpha\varepsilon$. This terminates the proof of Lemma 13. □

From Lemma 12 and Lemma 13 we obtain that the function $\ell.f_{ab}(u)$ is an $(\alpha, 0)$-weak encoding v2 into $\mathbb{G}$. Applying Lemma 6 and Theorem 2, this proves Lemma 7.

D Proof of Lemma 8

Let p be an integer. First we show that reduction modulo p is an admissible encoding from $\{0,1\}^\ell$ to $\mathbb{Z}_p$ if $2^\ell \gg p$.

Lemma 15 ($\{0,1\}^\ell \mapsto \mathbb{Z}_p$)**.** *Let p be an integer and k be a security parameter. Let $\ell = k + \lceil \log_2 p \rceil + 1$. The function $\text{MOD}_p : [0, 2^\ell - 1] \mapsto \mathbb{Z}_p$ with $\text{MOD}_p(x) = x \mod p$ is a 2^{-k}-admissible encoding v2.*

Proof. Let $\mu \in \mathbb{Z}$ such that $2^\ell - p < \mu \cdot p \leq 2^\ell$. We consider the sequence:

$$\{0,1\}^\ell \xrightarrow{F} [0, \mu \cdot p[\xrightarrow{G} \mathbb{Z}_p$$

where $F(x) = x \mod (\mu \cdot p)$ and $G(y) = y \mod p$. We show that both F and G are admissible encodings, and that the composition of two admissible encodings v2 remains an admissible encoding v2. For F we actually prove a slightly more general result:

Lemma 16. *Let $F : S \mapsto (S \cup \Delta_2) \setminus \Delta_3$ be a polynomially computable function such that $F(x) = x$ for all $x \in S \setminus \Delta_1$. Assume that set membership for $S \setminus \Delta_1$ can be decided in polynomial time. Then F is an ε-admissible encoding v2, with $\varepsilon = (|\Delta_1| + |\Delta_2| + |\Delta_3|)/|S|$.*

Proof. Given $x \in S \cup \Delta_2$, the sampling algorithm $\mathcal{I}_F(x)$ returns x for $x \in S \setminus \Delta_1$ and $\perp$ otherwise. Therefore for uniform $x \in (S \cup \Delta_2) \setminus \Delta_3$ the distribution of $\mathcal{I}_F(x)$ is ε-indistinguishable from the uniform distribution in S, with $\varepsilon = (|\Delta_1| + |\Delta_2| + |\Delta_3|)/|S|$. □

Applying Lemma 16 with $S = \{0,1\}^\ell$, $\Delta_1 = [\mu \cdot p, 2^\ell - 1[$, $\Delta_2 = \emptyset$ and $\Delta_3 = \Delta_1$, we obtain that F is an ε-admissible encoding v2, with $\varepsilon = 2p/2^\ell \leq 2^{-k}$. Moreover, it is easy to see that G is an admissible encoding according to Definition 1; therefore it is an ε-admissible encoding v2 with $\varepsilon = 0$. The following Lemma shows that the composition of two admissible encodings v2 remains an admissible encoding v2. This terminates the proof of Lemma 15. □

Lemma 17. *Let $F : S \mapsto R$ be an ε_1-admissible encoding v2 and $G : R \mapsto T$ be an ε_2-admissible encoding v2. Then $G \circ F$ is an $(\varepsilon_1 + \varepsilon_2)$-admissible encoding v2 from S to T.*

Proof. Firstly, $G \circ F$ is computable in polynomial time. Secondly, given t uniformly distributed in T, the random variable $r = \mathcal{I}_G(t)$ is ε_2-statistically indistinguishable from the uniform distribution in R. Then $s = \mathcal{I}_F(r)$ is $(\varepsilon_1 + \varepsilon_2)$-statistically indistinguishable from the uniform distribution in S. □

We now proceed with the proof of Lemma 8. With p a $2k$-bit prime and $\ell = 3k$, from Lemma 15 we obtain that reduction mod p is a 2^{-k+1}-admissible encoding from $\{0,1\}^\ell$ to $\mathbb{Z}_p$. Similarly, since $N \leq 2p$, we have that reduction modulo N is also a 2^{-k+2}-admissible encoding from $\{0,1\}^\ell$ to $\mathbb{Z}_N$. Using Lemma 17, this shows that $F : \{0,1\}^\ell \times \{0,1\}^\ell \to E$ with:

$$F(u,v) = f(u \mod p) + (v \mod N).G$$

is an ε-admissible encoding v2 with $\varepsilon = 2^{-k} + 2^{-k+1} + 2^{-k+2} = 7 \cdot 2^{-k}$. Applying Lemma 3, this proves Lemma 8.

E Proof of Lemma 9

We first prove that the function $F : \{0,1\}^\ell \times \{0,1\}^\ell \mapsto \mathbb{Z}_p \times \mathbb{Z}_N$ with:

$$F(u,v) = \big(u \mod p, v \mod N\big)$$

is an ε-admissible encoding v2 with $\varepsilon = \omega \cdot 2^{-\ell+2} + 2^{-\ell/2+2}$.

We first consider the function $G_1 : \{0,1\}^\ell \mapsto \mathbb{Z}_p$ with $G_1(u) = u \mod p$. Since $p = 2^\ell - \omega$, applying Lemma 16 with $S = \{0,1\}^\ell$, $\Delta_1 = [p, 2^\ell[$, $\Delta_2 = \emptyset$ and $\Delta_3 = \Delta_1$, we obtain that G_1 is an ε_1-admissible encoding v2 with $\varepsilon_1 = 2\omega \cdot 2^{-\ell}$.

Now we consider the function $G_2 : \{0,1\}^\ell \mapsto \mathbb{Z}_N$. From Hasse's bound, we have:

$$|N - (p+1)| < 2\sqrt{p} < 2^{\ell/2+1}$$

which gives using $p = 2^\ell - \omega$:

$$\left|N - 2^\ell\right| < 2^{\ell/2+1} + \omega$$

We distinguish two cases. If $N < 2^\ell$, we apply Lemma 16 as done previously with p. If $N \geq 2^\ell$, we apply Lemma 16 with $S = \{0,1\}^\ell$, $\Delta_1 = \emptyset$, $\Delta_2 = [2^\ell, N[$ and $\Delta_3 = \emptyset$. In both cases, we obtain that G_2 is an ε_2-admissible encoding v2, with $\varepsilon_2 = 2\omega \cdot 2^{-\ell} + 2^{-\ell/2+2}$.

Therefore F is an ε-admissible encoding v2 with $\varepsilon = \varepsilon_1 + \varepsilon_2 = \omega \cdot 2^{-\ell+2} + 2^{-\ell/2+2}$. Applying Lemma 3 this proves Lemma 9.

F Icart's function in Characteristic 2

In characteristic 2 we consider an elliptic-curve defined by the following equation:

$$E_{ab} : Y^2 + XY = X^3 + aX^2 + b$$

where a and b are elements of $\mathbb{F}_{2^n}$. When n is odd, we have $2^n - 1 \neq 0 \mod 3$, which implies that the map $x \mapsto x^3$ is a bijection over $\mathbb{F}_{2^n}$. Let

$$\begin{aligned} f_{a,b} : \mathbb{F}_{2^n} &\mapsto (\mathbb{F}_{2^n})^2 \\ u &\mapsto (x, y) \end{aligned}$$

where

$$\begin{aligned} x &= (v^4 + v^3 + b)^{1/3} + v \\ y &= ux + v^2 \\ v &= a + u + u^2 \end{aligned}$$

One can check that for any $u \in \mathbb{F}_{2^n}$, $f_{a,b}(u)$ is indeed a point of E_{ab}; we refer to [19] for more details. Icart's function in characteristic 2 satisfies the same property as in characteristic $p > 2$:

Lemma 18 (Icart [19]). *The function $f_{a,b}$ is computable in deterministic polynomial time. For any point $P \in \mathsf{Im}(f_{a,b})$, the set $f_{a,b}^{-1}(P)$ is computable in polynomial time and $|f_{a,b}^{-1}(P)| \leq 4$.*

G Proof of Theorem 5

We explain how the new maps from Theorem 5 are derived. We start from the original Ulas maps recalled in Section 1.2. Letting $g(x) = x^3 + ax + b$ and:

$$\begin{aligned} X_1(t,u) &= u & X_2(t,u) &= \frac{-b}{a}\left(1 + \frac{1}{t^4 g(u)^2 + t^2 g(u)}\right) \\ X_3(t,u) &= t^2 g(u) X_2(t,u) & U(t,u) &= t^3 g(u)^2 g(X_2(t,u)) \end{aligned}$$

it is shown in [28] that the following equality holds:

$$U(t,u)^2 = g(X_1(t,u)) \cdot g(X_2(t,u)) \cdot g(X_3(t,u)) \tag{9}$$

We note that knowing the value of u is not required to compute X_2, X_3 and U. Indeed, X_2, X_3 and U only depend on $g(u)$. For this reason, u does not have to be explicitly computed and we can take $g(u) = -1$, which is a quadratic non residue since $q = 3 \mod 4$. Even if such u does not necessarily exist in $\mathbb{F}_q$, it exists in $\mathbb{F}_{q^3}$ and with such a u, the Ulas' formulas are still correct. We can then rewrite the Ulas' maps as maps in one variable t with $g(u) = -1$; this gives the maps of Theorem 5.

H Proof of Lemma 11

To compute the pre-images of a point $P = (X_P, Y_P)$, the equations $X_2(t) = X_P$ and $X_3(t) = X_P$ must be solved. Since $X_2(t)$ and $X_3(t)$ are rational functions over the finite field $\mathbb{F}_q$, efficient algorithms such as the Berlekamp algorithm [27] can be used to compute the roots of the corresponding polynomial equations. The Berlekamp algorithm has complexity $\mathcal{O}(d^2 \log^3 q)$, where d is the degree of the equation. Since $\deg X_2(t) = 4$ and $\deg X_3(t) = 4$, we have that each equation has at most 4 solutions; therefore a point has at most 8 pre-images which can be efficiently computed. This proves Lemma 11.

We note that computing the roots of the equations is not sufficient to compute a pre-image. Once these roots are computed, it is necessary to check whether the roots correspond effectively to P. Firstly because these roots could correspond to $-P$; namely the SWU algorithm computes a point from abscissa only. Secondly, a root t_0 of $X_3(t) = X_P$ is not necessarily a pre-image of P since it could be that $g(X_2(t_0))$ is also a quadratic residue; in this case, the SWU algorithm would return a point $P' = (X_2(t_0), Y)$, different from P. Nevertheless, these two required verifications do not change the upper-bound on the pre-image size.

I Shallue-Woestijne in Characteristic 2

In this section, we recall the Shallue-Woestijne algorithm in characteristic 2 (see [26]). An elliptic curve over a field $\mathbb{F}_{2^n}$ is a set of points $(x, y) \in (\mathbb{F}_{2^n})^2$ verifying the equation:

$$E_{ab} : Y^2 + X \cdot Y = X^3 + a \cdot X^2 + b$$

where $a, b \in \mathbb{F}_{2^n}$. Let g be the rational function

$$g : x \mapsto x^{-2} \cdot \left(x^3 + a \cdot x^2 + b\right)$$

Letting $Z = Y/X$, the equation for E_{ab} can be rewritten as:

$$Z^2 + Z = g(X) \tag{10}$$

Theorem 6 (Shallue-Woestijne [26]). *Let $g(x) = x^{-2} \cdot \left(x^3 + a \cdot x^2 + b\right)$ where $a, b \in \mathbb{F}_{2^n}$. Let*

$$X_1(t,w) = \frac{t \cdot c}{1 + t + t^2} \qquad X_2(t,w) = t \cdot X_1(t,w) + c \qquad X_3(t,w) = \frac{X_1(t,w) \cdot X_2(t,w)}{X_1(t,w) + X_2(t,w)}$$

where $c = a + \omega + \omega^2$. Then $g(X_1(t,w)) + g(X_2(t,w)) + g(X_3(t,w)) \in \mathsf{Im}\, h$ where h is the map $h : z \mapsto z^2 + z$.

From Theorem 6, we have that at least one of the $g(X_i(t,w))$ must be in $\mathsf{Im}\, h$, which leads to a point in E_{ab}. Namely, we have that $\mathsf{Im}\, h = \{z \in \mathbb{F}_{2^n} \mid \mathrm{Tr}\,(z) = 0\}$, where Tr is the trace operator $\mathrm{Tr} : \mathbb{F}_{2^n} \mapsto \mathbb{F}_2$ with:

$$\mathrm{Tr}\,(z) = \sum_{i=0}^{n-1} z^{2^i}$$

From Theorem 6 we have $\sum_i \mathrm{Tr}\,(g(X_i)) = 0$ and therefore at least one of the X_i must satisfy $\mathrm{Tr}\,(g(X_i)) = 0$.

Given such a X_i it remains to compute a solution of the equation $Z^2 + Z = g(X_i)$. This is a linear equation in $\mathbb{F}_{2^n}$, so finding Z amounts to solving a linear system. When n is odd, the solutions can be found more efficiently: the solutions of $Z^2 + Z = \beta$ are given by $\mathrm{HTr}(\beta)$ and $\mathrm{HTr}(\beta) + 1$, where HTr is the half trace, defined as:

$$\mathrm{HTr} : z \mapsto \sum_{i=0}^{(n-1)/2} z^{2^{2i}}$$

We note that in practice a prime extension degree n is generally used, which means that n is usually odd.

We obtain the following pseudo-code for the Shallue-Woestijne algorithm in characteristic 2, when the extension degree n is odd. Note that the algorithm takes as input two values $t, w \in \mathbb{F}_{2^n}$; in practice, one can take $w = 0$.

Shallue-Woestijne algorithm in characteristic 2:
Input: parameters $a, b \in \mathbb{F}_{2^n}$ and input $t, w \in \mathbb{F}_{2^n}$
Output : $(x, y) \in E_{a,b}$

1. $c \leftarrow a + w + w^2$
2. $X_1 \leftarrow t \cdot c/(1 + t + t^2)$
3. $X_2 \leftarrow t \cdot X_1 + c$
4. $X_3 \leftarrow X_1 \cdot X_2/(X_1 + X_2)$
5. For $i = 1$ to 3:
   (a) $h_i \leftarrow ({X_i}^3 + a \cdot {X_i}^2 + b)/{X_i}^2$
   (b) If $\mathrm{Tr}\,(h_i) = 0$, return $(X_i,\ \mathrm{HTr}(h_i) \cdot X_i)$

I.1 Analysis

As for the Ulas maps, two parameters are needed to generate a point. A general way to analyze the algorithm is to fix one of the parameter and to let the other vary.

Lemma 19 (fixed t). *The Shallue-Woestijne algorithm with fixed t such that $t^2 + t \neq 1$ has pre-image size at most 6. The encoding can be inverted on its image in polynomial time.*

Lemma 20 (fixed w). *The Shallue-Woestijne algorithm with fixed w such that $w^2 + w \neq a$ has pre-image size at most 6. The encoding can be inverted on its image in polynomial time.*

Proof. We first give the expression of the rational function $X_3(t, w)$:

$$X_3(t, w) = \frac{c \cdot (t^2 + t)}{1 + t + t^2}$$

Since we have for all i, $\deg_t(X_i(t, w)) = 2$ and $\deg_w(X_i(t, w)) = 2$, it is easy to see that each point has at most 6 preimages in both cases. □