Elligator Squared

Uniform Points on Elliptic Curves of Prime Order as Uniform Random Strings

Mehdi Tibouchi

NTT Secure Platform Laboratories
tibouchi.mehdi@lab.ntt.co.jp

Abstract. When represented as a bit string in a standard way, even using point compression, an elliptic curve point is easily distinguished from a random bit string. This property potentially allows an adversary to tell apart network traffic that makes use of elliptic curve cryptography from random traffic, and then intercept, block or otherwise tamper with such traffic.

Recently, Bernstein, Hamburg, Krasnova and Lange proposed a partial solution to this problem in the form of Elligator: an algorithm for representing around half of the points on a large class of elliptic curves as close to uniform random strings. Their proposal has the advantage of being very efficient, but suffers from several limitations:

- Since only a subset of all elliptic curve points can be encoded as a string, their approach only applies to cryptographic protocols transmitting points that are *rerandomizable* in some sense.
- Supported curves all have non-trivial 2-torsion, so that Elligator cannot be used with prime-order curves, ruling out standard ECC parameters and many other cryptographically interesting curves such as BN curves.
- For indistinguishability to hold, transmitted points have to be uniform in the whole set of representable points; in particular, they cannot be taken from a prime order subgroup, which, in conjunction with the non-trivial 2-torsion, rules out protocols that require groups of prime order.

In this paper, we propose an approach to overcome all of these limitations. The general idea is as follows: whereas Bernstein et al. represent an elliptic curve point P as the bit string $\iota^{-1}(P)$, where ι is an *injective encoding* to the curve (which is only known to exist for some curve families, and reaches only half of all possible points), we propose to use a randomly sampled preimage of P under an *admissible encoding* of the form $f^{\otimes 2}\colon (u,v) \mapsto f(u) + f(v)$, where f is essentially any algebraic encoding. Such encodings f exist for all elliptic curves, and the corresponding admissible encodings $f^{\otimes 2}$ are essentially surjective, inducing a close to uniform distribution on the curve.

As a result, our bit string representation is somewhat less compact (about twice as long as Elligator), but it has none of the limitations above, and can be computed quite efficiently when the function f is suitably chosen.

Keywords: Elliptic curve cryptography, Point encoding, Circumvention technology, Anonymity and privacy

1 Introduction

Elliptic curves, whose use in public-key cryptography was first suggested by Koblitz and Miller in the mid-1980s [18,20], offer numerous advantages over more traditional settings like RSA and finite field discrete logarithms, particularly higher efficiency and a much smaller key size that scales gracefully with security requirements. Moreover, they possess a rich geometric structure that enables the construction of additional primitives such as bilinear pairings, which have opened up avenues for novel cryptographic protocols over the past decade, starting with Joux's tripartite key agreement [17] and Boneh and Franklin's construction of an identity-based encryption scheme [5].

On the Internet, adoption of elliptic curve cryptography is growing in general-purpose protocols like TLS, SSH and S/MIME, as well as anonymity and privacy-enhancing tools like Tor (which favors ECDH key exchange in recent versions) and Bitcoin (which is based on ECDSA).

For circumvention applications, however, ECC presents a weakness: points on a given elliptic curve, when represented in a usual way (even in compressed form) are easy to distinguish from random bit strings. For example, the usual compressed bit string representation of an elliptic curve point is essentially the x-coordinate of the point, and only about half of all possible x-coordinates correspond to valid points (the other half being x-coordinates of points of the quadratic twist). This makes it relatively easy for an attacker to distinguish ECC traffic (the transcripts of multiple ECDH key exchanges, say) from random traffic, and then proceed to intercept, block or otherwise tamper with such traffic.

Note that while RSA presents a similar weakness, it is both less severe and easier to mitigate. Namely, an RSA ciphertext or signature with respect to a public modulus N is usually represented as a bit string of length $n = \lceil \log_2 N \rceil$ corresponding to an integer between 1 and $N-1$. This can be distinguished from a random bit string with advantage $\approx (1 - N/2^n)$, which is usually less than $1/2$, and possibly much less for an appropriate choice of N. Moreover, even when N isn't close to 2^n, it is possible to thwart the distinguishing attack by using redundant representations, i.e. transmitting representatives of the classes modulo N chosen in $[0, 2^{n+t})$ (see §3.4).

Countering the distinguishers for elliptic curve points is more difficult. One possible approach is to modify protocols so that transmitted points randomly lie either on the given elliptic curve or on its quadratic twist (and the curve parameters must therefore be chosen to be twist-secure). This is the approach taken by Möller [21], who constructed a CCA-secure KEM and a corresponding hybrid public-key encryption scheme based on elliptic curves, using a binary (to avoid modulus based distinguishers like in RSA) elliptic curve and its twist. Similarly, Young and Yung constructed secure key exchange [26] and encryption [27] without random oracles based on the hardness of DDH in an elliptic curve and its twist.

Möller's approach has already been deployed in circumvention tools, including StegoTorus [24], a camouflage proxy for Tor, and Telex [25], an anticensorship technology that uses a covert channel in TLS handshakes to securely communicate

with friendly proxy servers. However, since protocols and security proofs have to be adapted to work on both a curve and its twist, this approach is not particularly versatile, and it imposes additional security requirements (twist-security) on the choice of curve parameters.

Elligator. A different approach was recently proposed by Bernstein, Hamburg, Krasnova and Lange [4]. Their idea is to leverage an efficiently computable, efficiently invertible algebraic function that maps the integer interval $S = \{0, \ldots, (p-1)/2\}$, p prime, *injectively* to the group $E(\mathbb{F}_p)$ where E is an elliptic curve over $\mathbb{F}_p$ (subject to some conditions on the choice of p and E). Bernstein et al. observe that, since ι is injective, a uniformly random point P in $\iota(S) \subset E(\mathbb{F}_p)$ has a uniformly random preimage $\iota^{-1}(P)$ in S, and use that observation to represent an elliptic curve point P as the bit string representation of the unique integer $\iota^{-1}(P)$ if it exists. If the prime p is close to a power of 2, a uniform point in $\iota(S)$ will have a close to uniform bit string representation.

This method, which they call Elligator, has numerous advantages over Möller's twisted curve method: it is easier to adapt to existing protocols using elliptic curves, since there is no need to modify them to also deal with the quadratic twist; it avoids the need to publish a twisted curve counterpart of each public key element, hence allowing a more compact public key; and it doesn't impose additional security requirements like twist-security. But it also has some significant limitations:

- The set $\iota(S)$ of elliptic curve points that can be represented as bit strings using Elligator is of cardinality $\approx p/2$, and hence contains only about half of all points on the curve. As a result, the approach only applies to cryptographic protocols transmitting points that are *rerandomizable* in some sense. For example, Elligator cannot be used in conjunction with a deterministic signature scheme like BLS [6] (short of using e.g. additional padding).
- Not all elliptic curves are known to admit an injective encoding ι as used in the construction of Elligator, and all of those curves have order divisible by a small prime. Bernstein et al. use the injective encoding proposed by Fouque, Joux and Tibouchi [13], which only exists for curves of order divisible by 4 over fields with $p \equiv 3 \pmod 4$, and another new injective encoding which exists for curves of even order. The only other known injective encoding to ordinary curves is due to Farashahi [10] and applies to curves of order divisible by 3. The Elligator construction cannot be used with any other elliptic curve, and in particular does not apply to prime-order curves, which make up essentially all standardized ECC parameters (including NIST [12], SEC 2 [9], Brainpool [19] and ANSSI [1] curves), or to many other cryptographically interesting curves such as Barreto–Naehrig curves [2].
- For indistinguishability to hold, transmitted points have to be uniform in $\iota(S)$; in particular, they cannot be taken from a strict subgroup, which rules out protocols that require groups of prime order, since none of the supported curves has prime order. In particular, many protocols with standard model security cannot be used with Elligator. For example, Bernstein et al. describe

a hybrid encryption scheme constructed from a slightly modified version of the ElGamal key encapsulation mechanism in the whole group of points of their elliptic curve [4, §2.3]. The overall hybrid scheme is secure if the key derivation function is modeled as a random oracle, but the existence of small divisors of the group order breaks the semantic security of the underlying standard model KEM, even though the usual ElGamal KEM is IND-CPA secure in the standard model.

Our contributions. In this paper, we propose a new approach to overcome all of these limitations. The general idea is as follows: whereas Bernstein et al. represent an elliptic curve point P as the bit string $\iota^{-1}(P)$, where ι is an *injective encoding* to the curve (which is only known to exist for some curve families, and reaches only half of all possible points, we propose to use a randomly sampled preimage of P under an *admissible encoding* of the form:

$$f^{\otimes 2}\colon (u, v) \mapsto f(u) + f(v),$$

where f is essentially any algebraic encoding. Such encodings f exist for all elliptic curves, and the corresponding admissible encodings $f^{\otimes 2}$ are essentially surjective, inducing a close to uniform distribution on the curve.

As a result, using our approach, *all* elliptic curve points are representable, and the bit string representation of a random point on the *whole* elliptic curve (rather than just a special subset of it) is statistically indistinguishable from a random bit string. This eliminates the need for repeatedly restarting the protocol until a representable point is found, and for rerandomizability in general (for example, full domain hash-like deterministic signatures such as BLS signatures [6], which we mentioned are not directly usable with Elligator, can be used with our representation algorithm without problem).

In addition, since the kind of encoding functions f we use exist for essentially all elliptic curves, including curves of prime as well as composite order, pairing-friendly curves and so on, our method lifts all the limitations that Elligator sets on curve parameters. In particular, protocols requiring curves of prime order can be used in our setting.

We also recommend specific choices of the function f that are well-suited to various elliptic curve parameters, and propose optimizations of the corresponding algorithms for representing points as bit strings and back. We find that in most setting, our approach is in fact *more efficient* than Elligator for representing generated points as bit strings. It is, however, less compact, since a curve point is represented as two base field elements instead of one.

Organization of the paper. In §2, we introduce notation, definitions and useful results related to discrete probability distributions, regularity and so-called well-distributed encodings to elliptic curves. In §3, we introduce our main construction, and state and establish the theorem on which it is based. Finally, in §4, we present concrete choices of functions f which are well-suited to our

approach, working for large families of curves, and also offer a performance comparison to Elligator.

2 Preliminaries

2.1 Statistical distance and regularity

For $\mathscr{D}$ a probability distribution on a finite set S, we write $\Pr[s \leftarrow \mathscr{D}]$ for the probability assigned to the singleton $\{s\} \subset S$ by $\mathscr{D}$. The uniform distribution on S is denoted by $\mathscr{U}_S$ (or just $\mathscr{U}$ if the context is clear).

Definition 1 (Statistical distance). *Let $\mathscr{D}$ and $\mathscr{D}'$ be two probability distributions on a finite set S. The* statistical distance *between them is defined as the ℓ_1 norm:*[1]

$$\Delta_1(\mathscr{D}, \mathscr{D}') = \sum_{s \in S} \big| \Pr[s \leftarrow \mathscr{D}] - \Pr[s \leftarrow \mathscr{D}'] \big|.$$

We simply denote by $\Delta_1(\mathscr{D})$ the statistical distance between $\mathscr{D}$ and $\mathscr{U}_S$:

$$\Delta_1(\mathscr{D}) = \sum_{s \in S} \Big| \Pr[s \leftarrow \mathscr{D}] - \frac{1}{|S|} \Big|,$$

*and say that $\mathscr{D}$ is ε-*statistically close to uniform *when $\Delta_1(\mathscr{D}) \leq \varepsilon$. When $\Delta_1(\mathscr{D})$ is negligible, we simply say than $\mathscr{D}$ is* statistically close to uniform.[2]

The squared Euclidean imbalance *$\Delta_2^2(\mathscr{D})$ of $\mathscr{D}$ is the square of the ℓ_2 norm between $\mathscr{D}$ and $\mathscr{U}_S$:*

$$\Delta_2^2(\mathscr{D}) = \sum_{s \in S} \Big| \Pr[s \leftarrow \mathscr{D}] - 1/|S| \Big|^2.$$

Definition 2 (Pushforward and pullback). *Let S, T be two finite sets and F any mapping from S to T. For any probability distribution $\mathscr{D}_S$ on S, we can define the* pushforward *$F_*\mathscr{D}_S$ of $\mathscr{D}_S$ by F as the probability distribution on T such that sampling from $F_*\mathscr{D}_S$ is equivalent to sampling a value $s \leftarrow \mathscr{D}_S$ and returning $F(s)$. In other words:*

$$\Pr\left[t \leftarrow F_*\mathscr{D}_S\right] = \Pr\left[s \leftarrow \mathscr{D}_S;\ t = F(s)\right] = \mu_S\big(F^{-1}(t)\big) = \sum_{s \in F^{-1}(t)} \Pr[s \leftarrow \mathscr{D}_S],$$

where μ_S is the probability measure defined by $\mathscr{D}_S$. Similarly, for any probability distribution $\mathscr{D}_T$ on T that assigns a nonzero weight $\mu_T\big(F(S)\big)$ to the image of F, we can define the pullback *$F^*\mathscr{D}_T$ of $\mathscr{D}_T$ by F as the probability distribution on S such that sampling from $F^*\mathscr{D}_T$ is equivalent to sampling a value $t \leftarrow \mathscr{D}_T$,*

[1] An alternate definition frequently found in the literature differs from this one by a constant factor 1/2. That constant factor is irrelevant for our purposes.

[2] For this to be well-defined, we of course need a family of random variables on increasingly large sets S. Usual abuses of language apply.

returning a uniformly random preimage $s \in F^{-1}(t)$ if one exists, and restarting otherwise. In other words:

$$\Pr\left[s \leftarrow F^*\mathscr{D}_T\right] = \frac{1}{\mu_T\big(F(S)\big)} \cdot \frac{\Pr[t \leftarrow \mathscr{D}_T]}{\#F^{-1}(t)} \quad \textit{where} \quad t = F(s).$$

Definition 3 (Regularity). *Let S, T be two finite sets and F any mapping from S to T. We say that F is* ε-regular *(resp.* ε-antiregular*) when $F_*\mathscr{U}_S$ (resp. $F^*\mathscr{U}_T$) is ε-close to the uniform distribution. We may omit ε if it is negligible.*

Lemma 1. *Let S, T be two finite sets and F an ε-regular mapping from S to T. Then F satisfies:*

$$1 - \frac{\#F(S)}{\#T} \leq \varepsilon,$$

and is also a 2ε-antiregular mapping.

Proof. This result is similar to [7, Lemma 3]. Since F is ε-regular, we have:

$$\Delta_1(F_*\mathscr{U}_S) = \sum_{t \in T} \Big| \Pr[t \leftarrow F_*\mathscr{U}_S] - \frac{1}{\#T} \Big| = \sum_{t \in T} \Big| \frac{\#F^{-1}(t)}{\#S} - \frac{1}{\#T} \Big| \leq \varepsilon.$$

On the other hand, that sum is larger than the same sum restricted to $T \setminus F(S)$, which is:

$$\sum_{t \notin F(S)} \Big| \frac{\#F^{-1}(t)}{\#S} - \frac{1}{\#T} \Big| = \#\big(T \setminus F(S)\big) \cdot \Big| 0 - \frac{1}{\#T} \Big| = 1 - \frac{\#F(S)}{\#T}.$$

Hence the first assertion that $1 - \#F(S)/\#T \leq \varepsilon$. Turning to the second assertion, we compute $\Delta_1(F^*\mathscr{U}_T)$:

$$\begin{aligned}
\Delta_1(F^*\mathscr{U}_T) &= \sum_{s \in S} \Big| \Pr[s \leftarrow F^*\mathscr{U}_T] - \frac{1}{\#S} \Big| \\
&= \sum_{s \in S} \Big| \frac{\#T}{\#F(S)} \cdot \frac{Pr[F(s) \leftarrow \mathscr{U}_T]}{\#F^{-1}\big(F(s)\big)} - \frac{1}{\#S} \Big| \\
&= \sum_{s \in S} \Big| \frac{1}{\#F(S) \cdot \#F^{-1}\big(F(s)\big)} - \frac{1}{\#S} \Big| \\
&= \sum_{t \in F(S)} \#F^{-1}(t) \cdot \Big| \frac{1}{\#F(S) \cdot \#F^{-1}(t)} - \frac{1}{\#S} \Big| \\
&\leq \sum_{t \in F(S)} \Big| \frac{1}{\#F(S)} - \frac{1}{\#T} \Big| + \Big| \frac{1}{\#T} - \frac{\#F^{-1}(t)}{\#S} \Big| \\
&\leq \Big| 1 - \frac{\#F(S)}{\#T} \Big| + \Delta_1(F_*\mathscr{U}_S) \leq 2\varepsilon
\end{aligned}$$

as required. □

2.2 Well-distributed encodings

Let E be an elliptic curve over a finite field $\mathbb{F}_q$, and $f\colon \mathbb{F}_q \to E(\mathbb{F}_q)$ any function. Farashahi et al., in [11], show that regularity properties of the tensor square $f^{\otimes 2}$ defined by:

$$\begin{aligned} f^{\otimes 2}\colon \mathbb{F}_q^2 &\to E(\mathbb{F}_q) \\ (u, v) &\mapsto f(u) + f(v) \end{aligned}$$

can be derived formally from the behavior of f with respect to characters of the group $E(\mathbb{F}_q)$. More precisely, they call the function f a *well-distributed encoding* when it satisfies good bounds with respect to character sums of the form $\sum_{u\in\mathbb{F}_q} \chi(f(u))$, for nontrivial characters χ of $E(\mathbb{F}_q)$.

Definition 4. *A function $f\colon \mathbb{F}_q \to E(\mathbb{F}_q)$ is said to be a* B-well-distributed encoding *for a certain constant $B > 0$ if for any nontrivial character χ of $E(\mathbb{F}_q)$, the following holds:*

$$\Big| \sum_{u\in\mathbb{F}_q} \chi(f(u)) \Big| \leq B\sqrt{q}.$$

Farashahi et al. then show that if f is a well-distributed encoding, then $f^{\otimes 2}$ is regular. They also provide a bound on the Euclidean imbalance of $(f^{\otimes 2})_*\mathscr{U}$.

Lemma 2 ([11, Theorem 3 & Corollary 4]). *Let $f\colon \mathbb{F}_q \to E(\mathbb{F}_q)$ be a B-well-distributed encoding, and $\mathscr{D} = (f^{\otimes 2})_*\mathscr{U}_{\mathbb{F}_q^2}$ the distribution on $E(\mathbb{F}_q)$ induced by $f^{\otimes 2}$. Then, we have:*

$$\Delta_1(\mathscr{D}) \leq \frac{B^2}{q}\sqrt{\#E(\mathbb{F}_q)} \quad \text{and} \quad \Delta_2^2(\mathscr{D}) \leq \frac{B^4}{q^2}.$$

Note that since $\#E(\mathbb{F}_q) = q + O(q^{1/2})$ by the Hasse–Weil bound, this implies $\Delta_1(\mathscr{D}) = O(q^{-1/2})$, so the distribution induced by $f^{\otimes 2}$ on $E(\mathbb{F}_q)$ is indeed statistically close to uniform.

We also mention a special case of the general geometric result that Farashahi et al. use to show that concrete maps are well-distributed encodings.

Lemma 3 ([11, Theorem 7]). *Let $h\colon C \to E$ a morphism over $\mathbb{F}_q$ from a curve C of genus g to the elliptic curve E. Assume that h does not factor through a nontrivial unramified morphism $Z \to E$. Then, for all nontrivial characters χ of $E(\mathbb{F}_q)$, we have:*

$$\Big| \sum_{P\in\mathbb{F}_q} \chi\big(h(P)\big) \Big| \leq (2g-2)\sqrt{q}.$$

3 Our construction

3.1 Elligator Squared

As explained in the introduction, our new approach to representing $\mathbb{F}_q$-points on an elliptic curve E as bit strings is to fix a suitable point encoding function $f\colon \mathbb{F}_q \to E(\mathbb{F}_q)$, and to use the tensor square function:

$$\begin{aligned} f^{\otimes 2}\colon \mathbb{F}_q^2 &\to E(\mathbb{F}_q) \\ (u,v) &\mapsto f(u) + f(v). \end{aligned}$$

A point $P \in E(\mathbb{F}_q)$ is then represented as (a bit string representation of) a uniformly random preimage $(u,v) \in (f^{\otimes 2})^{-1}(P) \subset \mathbb{F}_q^2$, and a pair (u,v) is converted back to a point by applying $f^{\otimes 2}$.

Leaving aside the question of how elements of $\mathbb{F}_q^2$ are represented as bit string for now (we discuss it in §3.4), we now describe the type of function f we will consider, formally define our construction, and state the corresponding main results. In what follows, we fix a finite field $\mathbb{F}_q$ and an elliptic curve E over $\mathbb{F}_q$. When stating asymptotic results, we implicitly assume as usual that q, E, and functions depending on them fit in infinite families indexed by a security parameter λ.

Definition 5. *We call a function $f\colon \mathbb{F}_q \to E(\mathbb{F}_q)$ a* (d,B)-well-bounded encoding*, for positive constants d, B, when f is B-well-distributed and all points in $E(\mathbb{F}_q)$ have at most d preimages under f. We may occasionally omit the constant B or both d and B as appropriate.*

Our main result pertaining to well-bounded encodings says that, on the one hand, if we sample a uniformly random preimage under $f^{\otimes 2}$ of a uniformly random point P on the curve, we get a pair $(u,v) \in \mathbb{F}_q^2$ which is statistically close to uniform; and on the other hand, that sampling uniformly random preimages under $f^{\otimes 2}$ can be done efficiently for all points $P \in E(\mathbb{F}_q)$ except possibly a negligible fraction of them.

Theorem 1. *Let $f\colon \mathbb{F}_q \to E(\mathbb{F}_q)$ be a (d,B)-well-bounded encoding. Then, the distribution on $\mathbb{F}_q^2$ obtained by picking a uniformly random point P in $E(\mathbb{F}_q)$, and then a uniformly random preimage $(u,v) \in \mathbb{F}_q^2$ of P under $f^{\otimes 2}$ if one exists is ε-statistically close to uniform for $\varepsilon = 2B^2\sqrt{\#E(\mathbb{F}_q)}/q = O(q^{-1/2})$. Moreover, there exists a probabilistic algorithm which, on input of any point $P \in E(\mathbb{F}_q)$, returns a uniformly random preimage of P under $f^{\otimes 2}$ if it exists, and whose average running time $T(P)$ on input P satisfies:*

$$T(P) \leq T_{f^{-1}} + \big(1 + \varepsilon_T(P)\big) \cdot d \cdot (T_f + T_\ominus + T_{\#f^{-1}})$$

where T_f, $T_\ominus$, $T_{\#f^{-1}}$ and $T_{f^{-1}}$ are the respective running times of the algorithms computing f, a subtraction in $E(\mathbb{F}_q)$, the number of preimages of a point under

f, and all the preimages of a point under f, and the coefficient $\varepsilon_T(P)$ is bounded, for all P except possibly a fraction of $\leq q^{-1/2}$ of them, as:

$$\varepsilon_T(P) \leq \frac{2B^2+2}{q^{1/4}-2B^2} = O(q^{-1/4}). \tag{1}$$

In other words, for all $P \in E(\mathbb{F}_q)$ except possibly a negligible fraction of them, the time it takes to sample a uniformly random preimage of P under $f^{\otimes 2}$ is one evaluation of f^{-1} and about d evaluations of f, of point subtractions on $E(\mathbb{F}_q)$ and of the function that counts preimages under f.

Proof. The first assertion says that $f^{\otimes 2}$ is ε-antiregular, which is a direct consequence of Lemma 1 and Lemma 2. We describe the preimage sampling algorithm in §3.3 below. The assertion on the running time is an immediate consequence of Lemmas 4 and 5 from that subsection.

Definition 6. *For a given well-bounded encoding $f\colon \mathbb{F}_q \to E(\mathbb{F}_q)$, the* Elligator Squared construction *for f is the pair formed by a randomized algorithm $E(\mathbb{F}_q) \to \mathbb{F}_q^2$ as in Theorem 1, called the* Elligator Squared representation algorithm*, which samples uniform preimages under $f^{\otimes 2}$, and the deterministic algorithm, called the* Elligator Squared recombination algorithm*, which computes the function $f^{\otimes 2}$.*

3.2 Example: ECDH using Elligator Squared

As an example of how this construction can be used in practice, we describe a standard elliptic curve Diffie–Hellman key exchange protected with Elligator Squared. Let P be a generator of $E(\mathbb{F}_q)$ (which we assume is a cyclic group of order N), $f\colon \mathbb{F}_q \to E(\mathbb{F}_q)$ a well-bounded encoding, and $\mathrm{KDF}\colon E(\mathbb{F}_q) \to \{0,1\}^\lambda$ a key derivation function. To derive a common secret, Alice and Bob proceed as follows.

1. Alice and Bob generate short term secrets (the values computed by Alice, resp. Bob, are indicated with indices A, resp. B, below):
   (a) Pick a uniformly random $r \xleftarrow{\$} \{0, \ldots, N-1\}$.
   (b) Compute the point $R = rP$.
   (c) Sample a random preimage $(u, v) \xleftarrow{\$} (f^{\otimes 2})^{-1}(R)$ under $f^{\otimes 2}$ using the Elligator Squared representation algorithm.
2. Alice sends (u_A, v_A) to Bob; Bob sends (u_B, v_B) to Alice.
3. Alice uses the Elligator Squared recombination algorithm to compute $R_B = f^{\otimes 2}(u_B, v_B)$. Similarly, Bob computes $R_A = f^{\otimes 2}(u_A, v_A)$.
4. Alice computes the shared secret as $k_{AB} = \mathrm{KDF}(r_A R_B)$, and similarly, Bob computes it as $k_{AB} = \mathrm{KDF}(r_B R_A)$.

The transmitted values (u_A, v_A) and (u_B, v_B) are elements of $\mathbb{F}_q^2$ that are statistically close to uniform, as shown by Theorem 1, so a transcript of this protocol cannot be distinguished from random messages.[3]

[3] With the caveat that an actual implementation transmits bit strings rather than field elements, but this is addressed in §3.4.

Moreover, in contrast with the same protocol implemented with Bernstein et al.'s Elligator [4, §2.3], our approach doesn't require any kind of rejection sampling during the computation of the pairs (u, v), and therefore only one elliptic curve scalar multiplication is needed to generate the short term secrets, compared to an average of two, and possibly more, with Elligator. Indeed, Theorem 1 ensures that with overwhelming probability on the choice of r, the representation algorithm samples a random preimage of $R = rP$ efficiently.

3.3 The sampling algorithm

Let $f\colon \mathbb{F}_q \to E(\mathbb{F}_q)$ be a (d, B)-well-bounded encoding. We now turn to the sampling algorithm for preimages of $f^{\otimes 2}$ whose existence was asserted as Theorem 1. It is described as Algorithm 1. This algorithm generalizes the sampling algorithm proposed, but not thoroughly analyzed, by Brier et al. [7, Algorithm 1] for the tensor square of Icart's encoding [16].

Algorithm 1 Preimage sampling algorithm for $f^{\otimes 2}$.

1: **function** SAMPLEPREIMAGE(P)
2: **repeat**
3: $u \xleftarrow{\$} \mathbb{F}_q$
4: $Q \leftarrow P - f(u)$
5: $t \leftarrow \#f^{-1}(Q)$
6: $j \xleftarrow{\$} \{1, \ldots, d\}$
7: **until** $j \leq t$
8: $\{v_1, \ldots, v_t\} \leftarrow f^{-1}(Q)$
9: **return** (u, v_j)
10: **end function**

Lemma 4. *On all inputs $P \in E(\mathbb{F}_q)$ in the image of $f^{\otimes 2}$, Algorithm 1 terminates almost surely, and returns a uniformly random preimage of P under $f^{\otimes 2}$, after an average of $N(P)$ iterations of the main loop (Steps 2–7), where:*

$$N(P) = d \cdot \frac{q}{\#(f^{\otimes 2})^{-1}(P)}.$$

On inputs P that have no preimage under $f^{\otimes 2}$, Algorithm 1 does not terminate.

Proof. The probability to exit the main loop after Step 7 for a given random choice of $u \in \mathbb{F}_q$ is t/d, where $t = \#f^{-1}\big(P - f(u)\big)$ (note that since f is d-well bounded, we know that t is always less or equal to d). As a result, taking all possible choices of u into account, the overall probability $\varpi(P)$ to exit the main

loop for a given input P is:

$$\varpi(P) = \frac{1}{q}\sum_{u\in\mathbb{F}_q}\frac{\#f^{-1}\big(P-f(u)\big)}{d} = \frac{1}{d\cdot q}\sum_{u\in\mathbb{F}_q}\sum_{v\in\mathbb{F}_q}\big[f(v)=P-f(u)\big]$$
$$= \frac{1}{d\cdot q}\sum_{(u,v)\in\mathbb{F}_q^2}\big[f^{\otimes 2}(u,v)=P\big] = \frac{1}{d\cdot q}\#(f^{\otimes 2})^{-1}(P),$$

where $[\cdot]$ is the usual Iverson bracket notation: for a statement U, $[U]=1$ if U is true and 0 otherwise. As a result, we see that Algorithm 1 does not terminate when $\#(f^{\otimes 2})^{-1}(P)=0$, and terminates almost surely otherwise, after an average of $N(P)=1/\varpi(P)=d\cdot q/\#(f^{\otimes 2})^{-1}(P)$ iterations of the main loop as required. Moreover, all outputs are clearly preimages of P under $f^{\otimes 2}$, so all it remains to prove is that each preimage is output with equal probability.

Fix a preimage (u_0,v_0) of P in $\mathbb{F}_q^2$. The probability that Algorithm 1 outputs (u_0,v_0) on input P conditionally to the first coordinate being u_0 is clearly $1/t_0$ where $t_0=\#f^{-1}\big(P-f(u_0)\big)$. Furthermore, the rejection sampling in the main loop ensures that any given first coordinate u is chosen with probability proportional to $t=\#f^{-1}\big(P-f(u)\big)$. As a result, we obtain, using the previous computation, that the probability of Algorithm 1 returning (u_0,v_0) on input P is exactly:

$$\frac{1}{t_0}\cdot\frac{t_0}{\sum_{u\in\mathbb{F}_q}\#f^{-1}\big(P-f(u)\big)} = \frac{1}{d\cdot q\cdot\varpi(P)} = \frac{1}{\#(f^{\otimes 2})^{-1}(P)}$$

as required. □

Lemma 5. *With the same notation as in Lemma 4, write, for all $P\in E(\mathbb{F}_q)$, $\varepsilon_T(P)=N(P)/d-1=q/\#(f^{\otimes 2})^{-1}(P)-1$. Then, for all $P\in E(\mathbb{F}_q)$ except possibly a fraction of $\leq q^{-1/2}$ of them, we have:*

$$\varepsilon_T(P)\leq\frac{2B^2+2}{q^{1/4}-2B^2}=O(q^{-1/4}).$$

(This is the same bound as (1) above).

Proof. Define $\delta=B^2q^{5/4}/\sqrt{\#E(\mathbb{F}_q)}$ (in particular, $\delta\sim B^2q^{3/4}$), and let α be the fraction of all points in $E(\mathbb{F}_q)$ such that:

$$\left|\#(f^{\otimes 2})^{-1}(P)-\frac{q^2}{\#E(\mathbb{F}_q)}\right|>\delta.$$

Now, according to Lemma 2, we have:

$$\Delta_2^2\big((f^{\otimes 2})_*\mathscr{U}_{\mathbb{F}_q^2}\big)=\sum_{P\in E(\mathbb{F}_q)}\left|\frac{\#(f^{\otimes 2})^{-1}(P)}{q^2}-\frac{1}{\#E(\mathbb{F}_q)}\right|^2\leq\frac{B^4}{q^2}.$$

On the other hand, by definition of α:

$$\Delta_2^2\big((f^{\otimes 2})_* \mathscr{U}_{\mathbb{F}_q^2}\big) = \frac{1}{q^4} \sum_{P \in E(\mathbb{F}_q)} \left| \#(f^{\otimes 2})^{-1}(P) - \frac{q^2}{\#E(\mathbb{F}_q)} \right|^2 \geq \frac{1}{q^4} \cdot \alpha \#E(\mathbb{F}_q) \cdot \delta^2.$$

Putting both inequalities together, we get:

$$\alpha \leq \frac{B^4 q^2}{\#E(\mathbb{F}_q) \cdot \delta^2} = q^{-1/2}.$$

Hence, for all $P \in E(\mathbb{F}_q)$ except a fraction $\alpha \leq q^{-1/2}$, the number $\#(f^{\otimes 2})^{-1}(P)$ of preimages of P under $f^{\otimes 2}$ is within δ of $q^2/\#E(\mathbb{F}_q)$. For all such P, we get:

$$\varepsilon_T(P) = \frac{q}{\#(f^{\otimes 2})^{-1}(P)} - 1 \leq \frac{q}{\frac{q^2}{\#E(\mathbb{F}_q)} - \delta} - 1 = \frac{(q+\delta)\#E(\mathbb{F}_q) - q^2}{q^2 - \delta \#E(\mathbb{F}_q)}.$$

The Hasse–Weil bound gives $\#E(\mathbb{F}_q) \leq q + 2\sqrt{q} + 1 = (\sqrt{q}+1)^2$, and hence $\delta \#E(\mathbb{F}_q) = B^2 q^{5/4} \#E(\mathbb{F}_q) \leq 2B^2 q^{7/4}$. As a result, again for all P except a fraction $\leq q^{-1/2}$:

$$\begin{aligned} \varepsilon_T(P) &\leq \frac{q^2 + 2q^{3/2} + q + 2B^2 q^{7/4} - q^2}{q^2 - 2B^2 q^{7/4}} \\ &\leq \frac{2B^2}{q^{1/4}} \cdot \frac{1 + \frac{1}{B^2} q^{-1/4} + \frac{1}{2B^2} q^{-3/4}}{1 - 2B^2 q^{-1/4}} \leq \frac{2B^2 + 2}{q^{1/4} - 2B^2} \end{aligned}$$

as required. □

With these lemmas, the proof of Theorem 1 is now complete. We also note that we can deduce the following result of independent interest as an easy corollary. This result is hinted to in [11], but not formally stated, let alone proven, although it is quite important if the results of that paper are to be applied to hash function constructions.

Corollary 1. *Let $f\colon \mathbb{F}_q \to E(\mathbb{F}_q)$ be a (d, B)-well-bounded encoding such that both f and f^{-1} are computable in polynomial time. Then $f^{\otimes 2}$ is $2q^{-1/2}$-*samplable *in the sense of [7, Definition 2], i.e. there exists a randomized algorithm $\mathscr{I}$ taking points $P \in E(\mathbb{F}_q)$ as inputs, running in polynomial time on all inputs, and such that $\mathscr{I}(P)$ is an element of $(f^{\otimes 2})^{-1}(P) \cup \{\perp\}$ whose distribution is $2q^{-1/2}$-statistically close to the uniform distribution on $(f^{\otimes 2})^{-1}(P)$. In particular, if $\mathfrak{h}$ is a random oracle with values in $\mathbb{F}_q^2$, $(f^{\otimes 2}) \circ \mathfrak{h}$ is indifferentiable from a random oracle with values in $E(\mathbb{F}_q)$.*

Proof. The only subtle point is that Algorithm 1 samples exactly uniform preimages under $(f^{\otimes 2})$, but may run in superpolynomial time, or even fail to terminate, on a negligibly small fraction of possible inputs. We can convert it to an algorithm that terminates in polynomial time on all inputs but induces a sampling that is only statistically close to uniform using early termination: for example,

modify Algorithm 1 to return $\perp$ if more than $\log q / \log(d/(1-d))$ iterations of the main loop are executed. Then, by Lemma 5, we obtain the algorithm returns a uniform preimage with probability $\geq 1 - q^{-1/2}$ and $\perp$ otherwise on all inputs except possibly a fraction $\leq q^{-1/2}$ of them, which gives the stated samplability result. The indifferentiability of the corresponding hash function construction in then a consequence of [7, Theorem 1], since f is also regular and efficiently computable. □

3.4 Bit-string representation

The Elligator Squared construction represents uniform elliptic curve points as close to uniform elements (u, v) of $\mathbb{F}_q^2$, but in practice, one wants to transmit bit strings rather than field elements. Can we obtain close to uniform bit strings instead?

Let us say for simplicity's sake that $q = p$ is a large prime (the prime power setting can be treated similarly). Then, the simplest way to represent an element in $\mathbb{F}_p$ is as the basic n-bit representation of the corresponding integer in $\{0, \ldots, p-1\}$, where $n = \lceil \log_2 p \rceil$. Then, it is easy to see that the statistical distance between a uniform element of $\mathbb{F}_p$ in that representation and a uniform bit string of the same length is given by $2 \cdot (1 - p/2^n)$.

If p is very close to 2^n, which is often the case for standardized curve parameters (including most NIST and SEC 2 curves [12,9], as well as Edwards curves such as Curve25519 and Curve1174 [3,4]) as such special primes offer efficient modular reduction, then we can simply transmit the basic n-bit representations of u and v directly, since they are close to uniform bit strings.

In some cases, however (like Brainpool curves [19], most families of pairing-friendly curves, etc.), p is not close to 2^n. Then, one possible approach to get close to uniform bit strings is to use a redundant representation as a bit string of length $n+t$ for some suitable t, i.e. represent $u \in \mathbb{F}_p$ as the basic $(n+t)$-bit representation of a randomly chosen integer of the form $u + kp$ with $k \in \left\{0, \ldots, \left\lfloor \frac{2^{n+t}-u}{p} \right\rfloor\right\}$. For a uniform $u \in \mathbb{F}_p$, the statistical distance to uniform of the corresponding distribution on $(n+t)$-bit strings is given by:

$$\sum_{u \in \mathbb{F}_p} \left| \frac{\left\lfloor \frac{2^{n+t}-u}{p} \right\rfloor + 1}{2^{n+t}} - \frac{1}{p} \right| \leq \frac{p}{2^{n+t}} \leq 2^{-t}.$$

Therefore, taking $t \approx n/2$ is sufficient. In fact, we can represent the whole pair $(u, v) \in \mathbb{F}_p^2$ as a close to uniform bit string of length $\approx 2n + n/2$ by first packing u and v as an integer in $\{0, \ldots, p^2 - 1\}$ and then using the same technique.

4 Application to specific curve families

One drawback of the Elligator Squared construction when applied to general well-bounded encodings f is that the representation algorithm involves the

computation of f^{-1}, which usually amounts to finding the roots of a possibly complicated polynomial over $\mathbb{F}_q$.

For example, Icart's encoding [16], defined for an elliptic curve $E\colon y^2 = x^3 + ax + b$ over a field $\mathbb{F}_q$ with $q \equiv 2 \pmod 2$ and $ab \neq 0$, is a $(4, 14)$-well-bounded encoding by [11, Theorem 8], so we can use it with Elligator Squared. In particular, many curves of prime order are of that form and are thus supported by our construction. But computing the preimages of a point (x, y), or even counting those preimages, involves solving quartic equation $u^4 - 6xu^2 + 6yu - 3a = 0$ over $\mathbb{F}_q$, which would probably be done using a rather costly algorithm such as Berlekamp or Cantor–Zassenhaus.

However, in many cases, we can choose a well-bounded encoding f such that f^{-1} is much easier to compute (it might take a couple of base field exponentiations, say), and counting the number of preimages of a point is even faster. We present several large classes of curves that admit such a convenient well-bounded encoding below. The curves considered here will be defined over a field $\mathbb{F}_q$ with $q \equiv 3 \pmod 4$. In such a field $\mathbb{F}_q$, we denote by $\chi_q(\cdot) : \mathbb{F}_q \to \{-1, 0, 1\}$ the nontrivial quadratic character (which is the Legendre symbol when q is prime), and by $\sqrt{\cdot}$ the standard square root, defined by $\sqrt{u} = u^{(q+1)/4}$ when $\chi_q(u) \neq -1$.

4.1 Ordinary curves with $q \equiv 3 \pmod 4$

Let $E\colon y^2 = x^3 + ax + b$ be an elliptic curve over $\mathbb{F}_q$, $q \equiv 3 \pmod 4$, with $ab \neq 0$, and let g be the polynomial $X^3 + aX + b \in \mathbb{F}_q[X]$. Based on earlier constructions by Shallue and van de Woestijne [22] and Ulas [23], Brier et al. [7] define the simplified SWU encoding to $E(\mathbb{F}_q)$ as follows (we follow the slightly modified presentation from [14,11]).

Definition 7. *Define rational functions $X_0, X_1 \in \mathbb{F}_q(u)$ as:*

$$X_0(u) = -\frac{b}{a}\Big(1 + \frac{1}{u^4 - u^2}\Big) \quad \textit{and} \quad X_1(u) = -u^2 X_0(u).$$

The simplified SWU encoding *to $E(\mathbb{F}_q)$ is the following mapping, which is well-defined (where we denote by O the point at infinity on E).*

$f\colon \mathbb{F}_q \to E(\mathbb{F}_q)$

$$u \mapsto \begin{cases} O & \text{if } u \in \{-1, 0, 1\}; \\ \Big(X_0(u), \sqrt{g\big(X_0(u)\big)}\Big) & \text{if } u \notin \{-1, 0, 1\} \text{ and } g\big(X_0(u)\big) \text{ is a square;} \\ \Big(X_1(u), -\sqrt{g\big(X_1(u)\big)}\Big) & \text{otherwise.} \end{cases}$$

It is shown in [11, §5.3] that f is a $(52 + O(q^{-1/2}))$-well-distributed encoding, and that for all $u \in \mathbb{F}_q \setminus \{-1, 0, 1\}$:

$$x = X_0(u) \iff u^4 - u^2 + \frac{1}{\omega} = 0$$
$$x = X_1(u) \iff u^4 - \omega u^2 + \omega = 0$$

where $\omega = \frac{a}{b}x + 1$. Since these are equations of degree 4 in u, it follows that any point $P = (x, y) \in E(\mathbb{F}_q)$ has at most 4 preimages under f (which must come from X_0 if $\chi_q(y) \geq 0$ and from X_1 otherwise). Therefore, f is a 4-well-bounded encoding. Moreover, the equations are biquadratic: therefore, f^{-1} can be computed with at most two square root computations on any input. And we can often compute the number of preimages under f with only quadratic character evaluations.

Indeed, to compute the number of preimages of (x, y) under f where, without loss of generality, $\chi_q(y) \geq 0$, we have to count the number $N = \#f^{-1}(x, y)$ of roots of the biquadratic equation $u^4 - u^2 + 1/\omega = 0$, where $\omega = \frac{a}{b}x + 1$. Let $\Delta = 1 - 4/\omega$ be the discriminant of the corresponding quadratic equation $v^2 - v + 1/\omega = 0$. Clearly, if $\chi_q(\Delta) = -1$, we have $N = 0$, and if $\Delta = 0$, the equation becomes $u^2 = v = 1/2$, hence $N = 0$ or 2 depending on whether $1/2$ is a square in $\mathbb{F}_q$. Finally, suppose $\chi_q(\Delta) = 1$. Then, the equation $v^2 - v + 1/\omega = 0$ has two simple roots whose product is $1/\omega$. Therefore, if $\chi_q(1/\omega) = -1$, exactly one of those roots is a square, and we get its two square roots as solutions for u, hence $N = 2$. If, however, $\chi_q(1/\omega) = 1$, we compute one of the roots, say $v_0 = (1 + \sqrt{\Delta})/2$, and we get $N = 0$ or 4 depending on whether $\chi_q(v_0) = \pm 1$.

Thus, as we can see, we can compute N with at most one exponentiation, and no exponentiation at all (only quadratic character evaluations) most of the time. This makes the Elligator Square construction quite efficient: the representation algorithm has an average total cost of 6.5 field exponentiations, while the recombination algorithm costs 2 field exponentiations (ignoring faster operations like field arithmetic and quadratic character evaluations).

4.2 Elligator 1 curves

Consider now an Elligator 1 curve E over $\mathbb{F}_q$ in the sense of [4, §3]. It is associated with a map $\phi\colon \mathbb{F}_q \to E(\mathbb{F}_q)$ such that each point in $E(\mathbb{F}_q)$ has either 0 or 2 preimages under ϕ (except one special point, which has a single preimage). Bernstein et al. show that computing and inverting ϕ both cost about one exponentiation in the base field, while counting the number of preimages of a given point can be done with only a quadratic character evaluation and a few multiplications.

Moreover, one can prove that ϕ is well-distributed. This is because ϕ can be expressed in terms of a degree 2 covering $h\colon H \to E$ of E by a certain elliptic curve H of genus 2, as described by Fouque et al. in [13]. As a result, character sums of the form $\sum_{u \in \mathbb{F}_q} \chi(\phi(u))$ can be rewritten up to a constant as $\sum_{P \in H(\mathbb{F}_q)} \chi(h(P))$. Moreover, the covering $h\colon H \to E$ is of prime degree, so does not factor nontrivially, and it cannot be unramified since H is not elliptic. Therefore, Lemma 3 ensures that:

$$\left| \sum_{P \in H(\mathbb{F}_q)} \chi\big(h(P)\big) \right| \leq (2g - 2)\sqrt{q} = 2\sqrt{q}$$

for all nontrivial characters χ of $E(\mathbb{F}_q)$. Therefore, we get that ϕ is $(2 + O(q^{-1/2}))$-well-distributed, and hence also $(2, 2 + O(q^{-1/2}))$-well-bounded.

This allows us to apply the Square Elligator construction to ϕ. It is even more efficient that for the simplified SWU encoding: the representation algorithm has an average total cost of $2 \times 1 + 1 = 3$ field exponentiations, while the recombination algorithm costs 2 field exponentiations (ignoring faster operations again).

4.3 BN curves

In [15], Fouque and Tibouchi have analyzed the Shallue–van de Woestijne encoding [22] in the particular case of Barreto–Naehrig curves [2], and found that it was a $(62 + O(q^{-1/2}))$-well-distributed. Moreover, preimages under this encoding are of three types, and the analysis in [15] makes it clear that each curve point can have at most one preimage of type 1, one preimage of type 2 and 2 preimages of type 3. As a result, the Shallue–van de Woestijne encoding f to any BN curve is a 4-well-bounded encoding.

Moreover, since the equations satisfied by preimages are quadratic for type 1 and 2 and biquadratic for type 3, f^{-1} can be computed with at most 4 square root computations, and the number of preimages of a given point can again be estimated with at most one square root computations and none at all most of the time. Therefore, even for BN curves, the Elligator Square construction is quite efficient.

4.4 Performance comparison with Elligator

Consider again a protocol such as the ECDH key exchange described in §3.2. The ephemeral key generation involves a single elliptic curve scalar multiplication, as well as one evaluation of the Elligator Squared representation algorithm, which costs an average of 6.5 base fields exponentiations with a general elliptic curve as in §4.1, or 3 base fields exponentiations with an Elligator 1 curve as in §4.2. In contrast, the corresponding algorithm implemented using Elligator [4, §2.4] costs an average of two scalar multiplications, plus one base field exponentiation for computing the representation. This is likely to make this phase of the protocol significantly *faster* with Elligator Squared compared to Elligator (certainly so at least when comparing implementations on the same curve). This is on top of the other advantages of Elligator Squared, including much more freedom in terms of supported curve parameters (prime order curves, BN curves, etc.), support for non-rerandomizable protocols and encoding of all curve points.

On the other hand, the transmitted data with Elligator Squared is twice as large, and the recombination algorithm about twice as slow (although for both Elligator and Elligator Squared this recombination time is usually dwarfed by a subsequent scalar multiplication on the curve).

References

1. ANSSI. Publication d'un paramétrage de courbe elliptique visant des applications de passeport électronique et de l'administration électronique

française. http://www.ssi.gouv.fr/fr/anssi/publications/publications-scientifiques/autres-publications/publication-d-un-parametrage-de-courbe-elliptique-visant-des-applications-de.html, Nov. 2011.
2. P. S. L. M. Barreto and M. Naehrig. Pairing-friendly elliptic curves of prime order. In B. Preneel and S. E. Tavares, editors, *Selected Areas in Cryptography*, volume 3897 of *Lecture Notes in Computer Science*, pages 319–331. Springer, 2005.
3. D. J. Bernstein. Curve25519: New Diffie-Hellman speed records. In M. Yung, Y. Dodis, A. Kiayias, and T. Malkin, editors, *Public Key Cryptography*, volume 3958 of *Lecture Notes in Computer Science*, pages 207–228. Springer, 2006.
4. D. J. Bernstein, M. Hamburg, A. Krasnova, and T. Lange. Elligator: Elliptic-curve points indistinguishable from uniform random strings. In V. Gligor and M. Yung, editors, *ACM CCS*, 2013.
5. 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.
6. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. *J. Cryptology*, 17(4):297–319, 2004.
7. E. Brier, J.-S. Coron, T. Icart, D. Madore, H. Randriam, and M. Tibouchi. Efficient indifferentiable hashing into ordinary elliptic curves. Cryptology ePrint Archive, Report 2009/340, 2009. http://eprint.iacr.org/. Full version of [8].
8. E. Brier, J.-S. Coron, T. Icart, D. Madore, H. Randriam, and M. Tibouchi. Efficient indifferentiable hashing into ordinary elliptic curves. In T. Rabin, editor, *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 237–254. Springer, 2010.
9. Certicom Research. SEC 2: Recommended elliptic curve domain parameters, Version 2.0, Jan. 2010.
10. R. R. Farashahi. Hashing into Hessian curves. In A. Nitaj and D. Pointcheval, editors, *AFRICACRYPT*, volume 6737 of *Lecture Notes in Computer Science*, pages 278–289. Springer, 2011.
11. R. R. Farashahi, P.-A. Fouque, I. Shparlinski, M. Tibouchi, and J. F. Voloch. Indifferentiable deterministic hashing to elliptic and hyperelliptic curves. *Math. Comp.*, 82(281), 2013.
12. FIPS PUB 186-3. *Digital Signature Standard (DSS)*. NIST, USA, 2009.
13. P.-A. Fouque, A. Joux, and M. Tibouchi. Injective encodings to elliptic curves. In C. Boyd and L. Simpson, editors, *ACISP*, volume 7959 of *Lecture Notes in Computer Science*, pages 203–218. Springer, 2013.
14. P.-A. Fouque and M. Tibouchi. Estimating the size of the image of deterministic hash functions to elliptic curves. In M. Abdalla and P. S. L. M. Barreto, editors, *LATINCRYPT*, volume 6212 of *Lecture Notes in Computer Science*, pages 81–91. Springer, 2010.
15. P.-A. Fouque and M. Tibouchi. Indifferentiable hashing to barreto-naehrig curves. In A. Hevia and G. Neven, editors, *LATINCRYPT*, volume 7533 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2012.
16. T. Icart. How to hash into elliptic curves. In S. Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 303–316. Springer, 2009.
17. A. Joux. A one round protocol for tripartite Diffie-Hellman. In W. Bosma, editor, *ANTS*, volume 1838 of *Lecture Notes in Computer Science*, pages 385–394. Springer, 2000.
18. N. Koblitz. Elliptic curve cryptosystems. *Math. Comp.*, 48:203–209, 1987.
19. M. Lochter and J. Merkle. Elliptic curve cryptography (ECC) Brainpool standard curves and curve generation. RFC 5639 (Informational), Mar. 2010.

20. V. S. Miller. Use of elliptic curves in cryptography. In H. C. Williams, editor, *CRYPTO*, volume 218 of *Lecture Notes in Computer Science*, pages 417–426. Springer, 1985.
21. B. Möller. A public-key encryption scheme with pseudo-random ciphertexts. In P. Samarati, P. Y. A. Ryan, D. Gollmann, and R. Molva, editors, *ESORICS*, volume 3193 of *Lecture Notes in Computer Science*, pages 335–351. Springer, 2004.
22. 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.
23. M. Ulas. Rational points on certain hyperelliptic curves over finite fields. *Bull. Pol. Acad. Sci. Math.*, 55(2):97–104, 2007.
24. Z. Weinberg, J. Wang, V. Yegneswaran, L. Briesemeister, S. Cheung, F. Wang, and D. Boneh. StegoTorus: a camouflage proxy for the Tor anonymity system. In T. Yu, G. Danezis, and V. D. Gligor, editors, *ACM CCS*, pages 109–120. ACM, 2012.
25. E. Wustrow, S. Wolchok, I. Goldberg, and J. A. Halderman. Telex: Anticensorship in the network infrastructure. In *USENIX Security Symposium*. USENIX Association, 2011.
26. A. L. Young and M. Yung. Space-efficient kleptography without random oracles. In T. Furon, F. Cayre, G. J. Doërr, and P. Bas, editors, *Information Hiding*, volume 4567 of *Lecture Notes in Computer Science*, pages 112–129. Springer, 2007.
27. A. L. Young and M. Yung. Kleptography from standard assumptions and applications. In J. A. Garay and R. D. Prisco, editors, *SCN*, volume 6280 of *Lecture Notes in Computer Science*, pages 271–290. Springer, 2010.