The Twist-AUgmented Technique for Key Exchange

Olivier Chevassut[1], Pierre-Alain Fouque[2], Pierrick Gaudry[3], and David Pointcheval[2]

[1] Lawrence Berkeley National Lab. – Berkeley, CA, USA – OChevassut@lbl.gov
[2] CNRS-École normale supérieure – Paris, France – {Pierre-Alain.Fouque,David.Pointcheval}@ens.fr
[3] CNRS-LORIA – Nancy, France – Pierrick.Gaudry@loria.fr

Abstract. Key derivation refers to the process by which an agreed upon large random number, often named master secret, is used to derive keys to encrypt and authenticate data. Practitioners and standardization bodies have usually used the random oracle model to get key material from a Diffie-Hellman key exchange. However, formal proofs in the standard model require *randomness extractors* to formally extract the entropy of the random master secret into a seed prior to deriving other keys. Whereas this is a quite simple tool, it is not easy to use in practice –or it is easy to misuse it–.
In addition, in many standards, the acronym PRF (Pseudo-Random Functions) is used for several tasks, and namely the randomness extraction. While randomness extractors and pseudo-random functions are *a priori* distinct tools, we first study whether such an application is correct or not. We thereafter study the case of $\mathbb{Z}_p^\star$ where p is a safe-prime and the case of elliptic curve since in IPSec for example, only these two groups are considered. We present *very efficient* and *provable* randomness extraction techniques for these groups under the DDH assumption. In the special case of elliptic curves, we present a new technique —the so-called *'Twist-AUgmented'* technique— which exploits specific properties of some elliptic curves, and avoids the need of any randomness extractor. We finally compare the efficiency of this method with other solutions.

1 Introduction

Key exchange is an important problem in practice and several schemes have been designed to solve it since the seminal work of Diffie and Hellman [13]. Recently, different works have been published in order to analyze the security of those schemes in various settings (password, public-key, hybrid setting) and security models (random oracle, common reference string, standard model). But for several years, efficiency and security in the standard model have become the main goals to achieve in cryptography. The most widely used network security protocols nowadays are TLS [34], a.k.a SSL, SSH, and the Internet Key Exchange (IKE) protocols [18, 24] from the IPSec standard of the IETF. In all the descriptions, the extraction of the master-key from a common (random) secret element is performed using a PRF, which is often instantiated by HMAC [5] (this is for example the case in IKE). However, it is well-known that such a primitive is not *a priori* well-suited for such a task [15], and the formal analysis requires unusual assumptions.

1.1 The Key Derivation Problem.

Diffie-Hellman (DH) based key exchanges establish a secure communication channel between two parties by securely negotiating a large random element in a given cyclic group, called pre-master secret. Then, this secret is used to derive keys for encrypting and authenticating data. These keys must be bit-strings of some specific length uniformly distributed and used as input parameters to symmetric ciphers (for privacy), message authentication codes (for authentication), and pseudo-random functions (for expansion of a seed into a longer bit-string). However, they cannot be initialized with the *simple* bit-string encoding of the pre-master secret. Even though this secret is indistinguishable from a random element in the cyclic group under some classical computational assumptions, such as the Decisional Diffie-Hellman assumption (DDH), its encoding is not indistinguishable from a random bit-string with a uniform distribution. The entropy of the bit-string encoded secret is indeed high but not high enough to immediately obtain an *almost* uniformly distributed random bit-string: pseudo-entropy generators are not pseudo-random generators even when only considering the property of computational indistinguishability [19].

Most of the cryptographic protocols do not take into account this practical problem since it only appears during the implementation. Cryptographers indeed use "elements in sets" when designing their algorithms while standardization bodies represent and encode these elements. Engineers are left clueless when elements in a given set do not necessarily admit a compact encoding —in bijection with a set of ℓ-bit strings— even for a well-chosen ℓ. Practitioners have no choice but to make educated guesses on which encoding to use and so, may introduce security breaches. This is the case of the Diffie-Hellman version of the SSL protocol [34] where the binary encoding of the random element is used as it. IKE raises this problem too. It explicitly deals with the extraction issue via a mechanism analyzed in [15], and follows the general framework described below.

1.2 Randomness Extraction and Key Derivation

In order to correctly derive several keys from a common (random) secret element —the so-called pre-master key—, two steps are required, with two different tools:

Randomness Extraction – in a first stage, one uses a family of functions $\mathcal{F}$ keyed by *random and public nonces* and applies it to the pre-master secret, to get the master key;

Key Derivation – in the second stage, the output is used as a key to a family of functions $\mathcal{G}$, with known inputs in order to derive further key material to create a secure channel.

This two-phase protocol also appears in the random generator architecture of Barak and Halevi [2]. The aim of the randomness extractor phase is to generate a short seed concentrating the entropy of the source and then in the key derivation, this seed will be used to generate keys. It is important to separate these stages, since different cryptographic primitives are needed. However, in many specifications, $\mathcal{F}$ and $\mathcal{G}$ are asked to be Pseudo-Random Function Families (with the same notation `prf`, such as in IKE [18, 24]).

Before going into more details, let us review informally the main difference between randomness extractors and PRF. A PRF is a family of functions, from a set D on a set R, such that it is computationally hard to distinguish the inputs/outputs of a function taken at random from the set of all functions from D to R and of a function taken at random in the PRF family. It is important to note that the key, or the index of the function taken in the PRF family, must be kept secret, otherwise the distinction becomes easy. A randomness extractor has the property that the output distribution is close to the uniform one, if the input distribution has enough entropy. If the index is known, the randomness extractor is called a *strong* randomness extractor. Hereafter, we only look at strong randomness extractors, where the index is implicitly made public, and we thus simply call them randomness extractors.

As a consequence, one can easily note that the notation `prf` has two different purposes: (1) first stage, `prf` is used as a randomness extractor, with a public and random key and a high-entropy input (but not as a PRF); (2) second stage, `prf` is used as a PRF, to build a PRG. The HMAC function [5], designed and analyzed as a secure MAC, is furthermore the default `prf` in several standards.

In this article, we primarily focus on the randomness extraction phases for DH-based protocol and we show efficient and provable techniques for this task. The key derivation phases can be solved by using techniques coming from the random oracle methodology (see the recently proposed internet draft by Dang and Polk in [12]) or by using a PRP in the counter mode.

1.3 HMAC as a Randomness Extractor

HMAC, as well as some other constructions, have been recently studied as randomness extractors by Dodis *et al.* in [15]. This is the first formal analysis of practical randomness extractors. They namely prove that variants of these constructions are almost universal hash functions under various assumptions. They basically show how to construct a variable-input length almost universal hash function family from a fixed-input length almost universal hash function family (or even random functions/permutations). Thereafter, a little modification of the Leftover Hash Lemma (LHL) [20] with a *randomly chosen function* from a family of (almost) universal hash functions can be used to extract the entropy of a random source.

Therefore, if the key of the (almost) universal hash function is correctly chosen (not biased by the adversary), the whole construction is correct. But the latter remark is important and not trivial in practice, since this key is not always (cannot always be) authenticated [10]. Finally, although this solution can be proven in the standard model, it is overkill compared with our solutions.

1.4 Randomness Extractors

The notion of a randomness extractor is thus very important from a practical point of view and is often ignored or misused by cryptographers, since solutions are quite theoretical and requirements are strong.

In complexity theory, randomness extraction from a distribution has been extensively studied (see [28] for a survey). For certain random sources, it has been shown that it is impossible to extract even one bit of randomness [26]. One way to solve this last problem is to use a small number of uniformly random bits as a *catalyst* in addition to the bits from the weak random source as in the LHL as said in [23]. However, in some cases, we can eliminate the need for the random catalyst by restricting the class of weak random sources. Trevisan and Vadhan and later Dodis [35, 14] have called such functions *deterministic extractors*. In cryptography, randomness extractors have been studied under different adversaries to construct truly random generators [3], and deterministic extractors have been used to built All-Or-Nothing-Transforms (AONTs) schemes and Exposure-Resilient Functions (ERF) [9, 16].

In the key exchange setting, the problem is to transform the random common secret of small entropy rate into a common secret of entropy rate 1, where the entropy rate is the ratio k/n of a random source of block-length n and of min-entropy k (basically the number of random bits). For example, under the DDH assumption in a 160-bit prime order q subgroup in $\mathbb{Z}_p^\star$, we know that the input random source (in a DH-based key exchange protocol) has 160 bits of min-entropy. So, for a 1024-bit prime p, the entropy rate of the initial source is $160/1024$. Because of the specific structure of the source, deterministic extractors (which exploit the algebraic structure) may be used to derive cryptographic keys. They would avoid problems with probabilistic randomness extractors if the key of a universal hash function can be controlled by the adversary. On the other hand, as we will see, large groups may be required, which would make the overall protocol too inefficient. We will thus introduce a new technique to avoid extractors, which takes advantage of the specific structure of elliptic curves.

1.5 Contribution and Organization

In this paper, we first focus on various techniques to derive a uniformly distributed bit-string from a high-entropy bit-string source. We explain their advantages and drawbacks. Then, we apply Kaliski's technique [22], with quadratic twists of elliptic curves, to avoid them. It is quite well-suited to authenticated key exchange, since it already works on cyclic groups. Therefore, it is more efficient than the Leftover Hash Lemma while retaining the same security attributes (and namely, no additional assumption).

The basic idea is to run twice in parallel, an authenticated Diffie-Hellman protocol on an elliptic curve $\mathbb{E}$ and on the quadratic twist $\tilde{\mathbb{E}}$ of $\mathbb{E}$. This produces two points K and $\tilde{K}$ uniformly distributed on $\mathbb{E}$ and $\tilde{\mathbb{E}}$ respectively. With well-chosen elliptic curves, the random choice of the abscissa of either K or $\tilde{K}$ is an ℓ-bit long random string. Randomness extractors are thus not needed anymore.

This "Twist AUgmented" (TAU) technique is provably secure assuming only the intractability of the decisional Diffie-Hellman problem on elliptic curves.

Even though quadratic twists were previously introduced in the literature [7, 8] in other contexts or with binary curves, we also show here that appropriate prime order curves can be efficiently generated.

2 The Leftover Hash Lemma

In this section, we focus on the most well-known randomness extractor, which makes use of the Leftover Hash Lemma [21, 20]. It provides a probabilistic extractor, which is optimal in general. Whereas in theory, (almost) universal hash functions (AUH) should be used, in practice, one often asks for pseudo-random functions (PRF). Let us see whether the practical way to do it is correct or not, from a theoretical point of view. The definitions are given in the full version [11].

Lemma 1 (LHL [21]). *Let $\mathcal{D}$ be a probabilistic distribution over $\{0,1\}^n$ with min-entropy at least σ. Let e be an integer and $m = \sigma - 2e$. Let $\mathcal{H} = \{h_k\}_k$, with $h_k \in \mathcal{F}_{n,m}$ for any $k \in \{0,1\}^\ell$, be an almost universal hash function family. Let H be a random variable uniformly distributed on $\mathcal{H}$, X denotes a random variable taking value in $\{0,1\}^n$, and H, X are independent. Then, $(H, H(X))$ is $2^{-(e+1)}$-uniform on $\mathcal{H} \times \{0,1\}^m$.*

Impagliazzo and Zuckerman in [21] prove the lemma with an almost universal hash function where $\varepsilon = 1/2^n$. In [15], it is proved for any ε-almost universal hash function family for $\varepsilon \ll 1/2^m$. See also [31] for a proof. Therefore, combined with the analysis of NMAC as an ε-AUH function, this may justify the design of IKE when HMAC is used under a specific assumption on the independence of the two keys in NMAC. We show in the following that the same result holds for some PRFs provided ε be taken into account to estimate the size of the output. However, we begin to prove a slight generalization of the LHL, similar to [15].

Lemma 2 (LHL with ε-AUH). *Let $\mathcal{D}$ be a probabilistic distribution over $\{0,1\}^n$ with min-entropy at least σ. Let e be an integer and $m \leq \alpha - 2e$ where $\alpha = \min(\sigma, \log_2(1/\varepsilon))$. Let $\mathcal{H} = \{h_k\}_k$, with $h_k \in \mathcal{F}_{n,m}$ for any $k \in \{0,1\}^\ell$, be a ε-almost universal hash function family. Let H be a random variable uniformly distributed on $\mathcal{H}$, X denotes a random variable taking value in $\{0,1\}^n$, and H, X are independent. Then, $(H, H(X))$ is 2^{-e}-uniform on $\mathcal{H} \times \{0,1\}^m$.*

Proof. The proof relies on two claims. The first one comes from [31]. It applies to a random variable X distributed according to a distribution $\mathcal{D}$, taking values on the finite set S and of collision probability $\kappa = \kappa(X)$. If X is δ-uniform on S, then $\kappa \geq (1 + 4\delta^2)/|S|$.

The second claim studies the collision probability $\kappa = \kappa(H, H(X))$ where H denotes a random variable with uniform probability on $\mathcal{H}$, X denotes a random variable on the set $\{0,1\}^n$, and H and X are independent. We can easily adapt the proof of [31] to prove that the statistical distance between the distribution of $(H, H(X))$ and the uniform distribution on $\mathcal{H} \times \{0,1\}^m$ is δ, which is at most $(1/2) \cdot \sqrt{2^m \cdot (\kappa + \varepsilon)}$. So it can be upper-bounded by $(1/2) \cdot \sqrt{2^m \cdot (2^{-\sigma} + \varepsilon)}$, since the collision probability κ is less than the guessing probability γ as noted in [11]. If we denote by $\alpha = \min(\sigma, \log_2(1/\varepsilon))$, then we can upper-bound δ by $(1/2) \cdot \sqrt{2^m \cdot 2 \cdot 2^{-\alpha}}$ and so if we want a bias of 2^{-e} we need $m \leq \alpha - 2e$. $\square$

Remark 3. This requires $\varepsilon \ll 1/2^m$ as it is observed in [15], but $\varepsilon \leq 1/2^{m+2e}$ is enough. Anyway, this definitely excludes function families where the key-length is the same as the output-length (as compression functions), unless they are completely balanced, with $\varepsilon = 0$, which is quite a strong assumption.

2.1 Pseudo-Random Functions vs. Almost Universal Hash Functions

We have already discussed the practical meaning of the universal hashing property for compression functions. However, many standards (such as IKE [18, 24]) use the acronym `prf` at several places, for different purposes: randomness extractors and actual PRF. Let us recall here the *crucial* difference between pseudo-random functions and randomness extractors: the former use random *secret* keys, while the latter use random *but known* keys. We thus show below that the *strong* assumption of PRF implies the almost universal hashing property. Therefore, the Leftover Hash Lemma 2 applied with some PRF (namely keyed with uniform random bit-strings and with advantage sufficiently small) provides a good randomness extractor.

Theorem 4. *If a family of functions $\mathcal{F}$ is a $(2, \varepsilon, 2T_f)$-PRF in $\mathcal{F}_{n,m}$, then it is an ε-AUH function family, where T_f denotes the maximal time to evaluate an instance of $\mathcal{F}$ for all $x \in \{0,1\}^n$.*

Proof. We want to show that if the hash function family $\mathcal{F}$ is not $\varepsilon - \mathsf{AUH}$, *i.e.* there exist x, y such that $\Pr_k[f_k(x) = f_k(y)] > 1/2^m + \varepsilon$, then there exists an adversary against the PRF property with advantage at least ε.

Let us consider the following family of distinguishers, $\mathcal{D}_{x,y}$ for each pair (x, y) of elements in $\{0,1\}^n$. The distinguisher $\mathcal{D}_{x,y}$ queries the oracle (either f_k for a random k or a random function) to get $X = f(x)$ and $Y = f(y)$, and simply answers 1 if $X = Y$ and 0 otherwise.

Suppose that $\mathcal{F}$ is not an ε-AUH function family. It means there exists a pair (x, y) for which $\Pr_k[f_k(x) = f_k(y)] > 1/2^m + \varepsilon$. Let us consider the advantage of the corresponding distinguisher $\mathcal{D}_{x,y}$: if f is a truly random function in $\mathcal{F}_{n,m}$, the set of all functions from $\{0,1\}^n$ to $\{0,1\}^m$, then $\Pr[\mathcal{D}_{x,y} = 1] = 1/2^m$; if f is a randomly chosen f_k in $\mathcal{F}$, then $\Pr[\mathcal{D}_{x,y} = 1] > 1/2^m + \varepsilon$. As a consequence, the advantage of $\mathcal{D}_{x,y}$ is not less than ε, which is in contradiction with the above PRF property. $\square$

Therefore, we have the following corollary by combining lemma 2 with the previous theorem.

Corollary 5. *Let $\mathcal{F}$ be a family of functions in $\mathcal{F}_{n,m}$, and T_f denote the maximal time to evaluate an instance of $\mathcal{F}$ on any $x \in \{0,1\}^n$. If $\mathcal{F}$ is a $(2, \varepsilon, 2T_f)$-PRF, when applied on a random source with min-entropy at least σ, then it is a good randomness extractor, of bias bounded by $1/2^e$, as soon as*

$$m \leq \min(\sigma, \log_2(1/\varepsilon)) - 2e.$$

Remark 6. This result is not in contradiction with the example described in [15], since if $\varepsilon = 1/2^m$ with m bits of output, then clearly $\min(\sigma, \log_2(1/\varepsilon)) \leq m$. The above corollary just claims that the bias is less than 1. As a consequence, we cannot extract m bits.

2.2 The Leftover Hash Lemma in Practice

Even if there exist efficient universal hash functions, practitioners and designers usually apply pseudo-random functions, or HMAC, which are clearly less efficient than a simple linear operation. Anyway, a correct application would be valid in both cases (according to the analysis for HMAC [15] — incomplete because of the above problem with compression functions). However, the Leftover Hash Lemma requires the key of the function family to be uniformly distributed, which is not an easy task, since it may be (partly) chosen by a malicious user. This is the case in IKEv1 [18], for compatibility reasons, and thus nothing can be formally proved.

A simple way to guarantee such a uniform distribution is for the users to sign this key (as done in IKEv2). However, such a signature is not always possible, or available, according to the context such as in password-based authenticated key exchange.

Another solution to cope with the randomness extraction error is, as noticed by Shoup [31] and also by Barak *et al.* in [3], to use the same "certified key" or the same hard-coded key in the software. Indeed, they suggest an extension of the LHL which allows the derivation of many random bit-strings with a *unique random* key, and thus a *public and fixed* hash function. However, the quality of the extracted randomness decreases linearly with the number of extractions – due to the hybrid technique. Nevertheless, this is often the unique solution.

3 Deterministic Randomness Extractors

Other alternatives to the LHL are also available, namely when no certification is available, as in the password-based setting, by using deterministic randomness extractors. Several of them exist in the literature and have already been employed by standardization bodies to convert a random element of a group into a random bit-string as in [29].

3.1 Hash-Diffie-Hellman

The simplest one, and perfectly reasonable in practice, is the use of a cryptographic hash function. In the random oracle model [6], this gives a perfect random bit-string, under the so-called computational Diffie-Hellman assumption. In the standard model, a weaker assumption has been defined, the Hash Diffie-Hellman assumption [1, 17]. But this assumption is, in some sense, the assumption that a hash function is perfectly suited to this goal, while this is not the applications that designers of hash functions have in mind. Everybody may agree on the practical validity of such a construction, but it definitely requires non-standard assumptions, from a theoretical point of view. We would thus prefer to avoid this solution.

3.2 A Simple Deterministic Extractor

Basically, when we want an extractor of the entropy from a random (uniformly distributed) element in a cyclic group $\mathbb{G}$ of order q, a bijection from $\mathbb{G}$ to $\mathbb{Z}_q$ would do the job, since it would transfer the uniform distribution $\mathbb{G}$ into a uniform distribution in $\mathbb{Z}_q$ (an appropriate choice for q thereafter allows the truncation to the $\log q$-rightmost bits to get an almost uniformly distributed bit-string). Let us briefly review such a well-known bijection in the specific case where $\mathbb{G}$ is the group of the quadratic residues modulo p, for a safe prime p, close enough to a power of 2. This result is in the folklore, but some lemmas are useful for the following, we thus briefly review the whole technique.

Theorem 7. *There is an efficient bijection from a subgroup $\mathbb{G}$ of prime order q in $\mathbb{Z}_p^\star$ to $\mathbb{Z}_q$, when $p = 2q + 1$.*

Proof. Let us use a finite field $\mathbb{Z}_p$, with $p = 2q+1$ (a safe prime) and work in the cyclic group of order q: the group $\mathbb{G}$ of the quadratic residues modulo p. Since $p = 3 \bmod 4$, this is a Blum prime, and thus -1 does not lie in $\mathbb{G}$.

We can define the following extractor, for any $y \in \mathbb{G}$: if $y \leq q$, then $f(y) = f_1(y) = y$, else $f(y) = f_2(y) = p - y$. Since -1 is not in $\mathbb{G}$, and $p - y = -y = (-1) \times y \bmod p$, f_1 maps $\mathbb{G}$ to $\mathbb{G}$ (the identity function) and f_2 maps $\mathbb{G}$ to $\mathbb{Z}_p \backslash \mathbb{G}$. Therefore, f is an injective mapping and for $y \in \mathbb{G}$, $f_1(y), f_2(y)$ are in $\mathbb{Z}_q$. A simple counting argument proves that this is a bijection. $\square$

The following lemma analyzes the security when truncation is used in order to get ℓ bits uniformly distributed. The proof of the lemma is done in the full version [11].

Lemma 8. *Let us denote by $\mathcal{U}_q$ the uniform distribution on the space $\mathbb{Z}_q$ and by $\mathcal{U}_{2^\ell}$ the uniform distribution on the space $\{0,1\}^\ell \sim \{0, \ldots, 2^\ell - 1\}$. If $|q| = \ell$ and $|q - 2^\ell| \leq 2^{\ell/2}$, then the statistical distance is bounded by $1/\sqrt{2^\ell}$.*

Therefore, the truncation of f gives a deterministic randomness extractor from $\mathbb{G}$ onto $\mathbb{Z}_q$. However, this requires the use of a safe prime, and thus quite large groups, which make DH-protocols quite inefficient.

4 The "Twist-AUgmented" Technique

In this section, we describe a new mechanism which excludes all the above drawbacks: it does not require any authenticated random value (needed for probabilistic extractors); it is provably secure in the standard model, under classical assumptions; it works in small groups (contrary to the above deterministic example.)

In the early 90's, Kaliski [22] used elliptic curves and their twists for making a random permutation from a random function. This construction can be used to make a uniform distribution in $\mathbb{Z}_{2q}$ from points uniformly distributed on a curve or its quadratic twist, both on the finite field $\mathbb{F}_q$. More recently, quadratic twists have also been used in the context of password-authenticated key exchange [8]. The goal was to make the Bellovin et al.'s encrypted key exchange protocol [4] immune to partition attacks but did not explain how to specify the key-derivation function. It has also been applied to the context of public-key encryption [7].

We can take advantage of elliptic curves and their quadratic twists, as done by Kaliski [22], to come up with a technique that does not require stronger assumptions. This technique, called "Twist-AUgmented" (TAU), uses the fact that a random point on a curve over $\mathbb{F}_p$ has an abscissa uniformly distributed in a set E and that a random point over its twist has an abscissa uniformly distributed in the set $\tilde{E}$ as well, *i.e.* it is the complementary set of E in $\mathbb{F}_p$. Therefore by choosing one of the two abscissae at random, we will get an element almost uniformly distributed in $\mathbb{F}_p$. For well-chosen fields, we thus efficiently get an almost uniformly distributed bit-string, which may be 256 bits long: it is enough to derive two keys (for privacy and for authentication) without any pseudo-random function by simply splitting this bit-string. As a consequence, it avoids the requirement of randomness extractors, and even pseudo-random functions, since we directly get a uniformly distributed *bit-string*, large enough.

4.1 Quadratic Twist of an Elliptic Curve

Let $p > 3$ be a prime number. An elliptic curve is a set of points $\mathbb{E} = \mathbb{E}_{a,b} = \{(x, y) : y^2 = x^3 + ax + b\} \cup \{\infty_{\mathbb{E}}\}$, where a and b are elements of $\mathbb{F}_p$ and $\infty_{\mathbb{E}}$ is a symbol for the point at infinity. It is well known that an elliptic curve $\mathbb{E}$ can be equipped with a group law —the so-called chord and tangent group law— such that the computational and decisional Diffie-Hellman problems are believed to be hard problems in general.

Let c be a quadratic non-residue in $\mathbb{F}_p$, and define the **quadratic twist** of $\mathbb{E}_{a,b}$ to be the curve given by the following equation: $\tilde{\mathbb{E}}_{a,b} = \{(x, y) : cy^2 = x^3 + ax + b\} \cup \{\infty_{\tilde{\mathbb{E}}}\}$.

The change of variables $x' = cx$ and $y' = c^2 y$ transforms the equation of $\tilde{\mathbb{E}}_{a,b}$ into $y'^2 = x'^3 + ac^2x' + bc^3$. This demonstrates that $\tilde{\mathbb{E}}_{a,b}$ is isomorphic to an elliptic curve and can therefore be equipped with a group law. The main interest of the introduction of the quadratic twist here follows directly from the definition: if x is not the abscissa of a point of $\mathbb{E}_{a,b}$, then $x^3 + ax + b$ is not a square in $\mathbb{F}_p$ and therefore $(x^3 + ax + b)/c$ is a square in $\mathbb{F}_p$. Then it is the abscissa of a point of $\tilde{\mathbb{E}}_{a,b}$. The converse is also true.

Note 9. In the cryptographic application we have in mind, this is crucial to keep the equation of $\tilde{\mathbb{E}}$ in the non-Weierstrass form. For the internal computations, of course, we apply the above-mentioned transformation so that we can use the classical algorithms, but the result of any computation should be transformed back to the previous representation before usage in cryptographic primitives.

Cardinalities. Hasse-Weil's theorem gives a good bound on the group order of an elliptic curve [33]. Let us write $q = \#\mathbb{E} = p + 1 - t$, then we have $|t| < 2\sqrt{p}$. We could apply the same result to $\tilde{\mathbb{E}}$, but in fact the number of points of a curve and its twist are far from being independent. Starting with the fact that a scalar is either a point on $\mathbb{E}$ or a point on $\tilde{\mathbb{E}}$, it is easy to derive that $\tilde{q} = \#\tilde{\mathbb{E}} = p+1+t$. For maximal security, it is desirable that the group orders are prime numbers. Hence, since p is odd, this implies that t is odd. Then both q and $\tilde{q}$ are odd.

Choice of the Prime Field. We have restricted ourselves to curves defined over prime fields. The notion of a quadratic twist of an elliptic curve also exists for more general finite fields and in particular for fields of characteristic 2. However, they are of less interest in our context where we want to use the property that the abscissae of the points of the groups we are dealing with cover the whole finite field. In characteristic 2, all the non-super-singular curves have a group order that is divisible by (at least) 2. Hence keeping the covering property would imply to work with non-prime order groups. Even if it looks feasible to patch the protocol for that situation, it is certainly less elegant than using a prime-order group with curves over prime fields.

To achieve our goal, we need that the abscissa of a point taken randomly in $\mathbb{E}$ or in $\tilde{\mathbb{E}}$ behaves like a random bit-string of length ℓ. Since all the elements of $\mathbb{F}_p$ are obtainable as abscissae of points of $\mathbb{E}$ and $\tilde{\mathbb{E}}$, we will be able to show that the random abscissa in $\mathbb{E}$ or $\tilde{\mathbb{E}}$ gives a random element in $\mathbb{F}_p$ (see Lemma 10, the proof appears in the full version [11].) To convert this element to a bit-string of length ℓ without any further device and keeping the randomness unbiased, it is necessary to have p very close to 2^ℓ. Hence we propose to use a prime p which can be written $p = 2^\ell - \varepsilon$, where ε is an integer less than $2^{\ell/2}$ (see previous Lemma 8, which proof appears in the full version [11].)

This extra-condition on p is not a practical inconvenience. In fact, the primes that are used in practice are almost always of this form, because they allow a faster arithmetic than more general primes. For instance, the curves proposed by the NIST are defined over a finite field with primes which are often suitable to our case (the prime field, not the curves!).

Finding a Suitable Elliptic Curve and Twist. The basic approach for constructing a curve $\mathbb{E}$ over $\mathbb{F}_p$ such that both q and $\tilde{q}$ are primes is to pick random curves, count their cardinalities with the SEA algorithm, and keep only the good ones. With this strategy, if numbers of points were completely independent and behaved like random numbers in the Hasse-Weil interval, we would expect to have to have to build $O(\log^2 p)$ curves before finding a good one. If $\log p \approx 200$, it means that we have to run the SEA algorithm about 20000 times to construct a good curve, which is prohibitive.

Fortunately, the SEA algorithm [27] is suited for this kind of search, since it computes the order of $\mathbb{E}$ modulo small primes and recombines the group order by Chinese Remaindering. Hence as soon as we know the order of $\mathbb{E}$ modulo a small prime ℓ, we abort the computation if this is zero. Furthermore, the group order of $\tilde{\mathbb{E}}$ modulo ℓ is readily deduced from $\#\mathbb{E} \bmod \ell$, and similar abortion can be played also with the twist. As a consequence, most of the curves are very quickly detected as bad curves, because either the curve or its twist has a non-prime group order.

In fact, the situation is more tricky, since the order of the curve and of its twist are not independent. For instance, imagine that $p \equiv 2 \bmod 3$, then the condition $\#\mathbb{E} \equiv 0 \bmod 3$ is equivalent to $t \equiv 0 \bmod 3$, which in turn is equivalent to $\#\tilde{\mathbb{E}} \equiv 0 \bmod 3$. A rigorous estimation of the running time of the SEA algorithm equipped with the early-abort strategy is out of the scope of this work. We just propose some numerical experiments to justify the claim that the construction of secure pairs of curve and twist is easily feasible on a reasonable computer.

We picked randomly about 30000 200-bit primes, and for each of them we picked a random curve and computed its cardinality and the cardinality of its twist. In the following table, we summarize the percentage of the curves for which both number of points are not divisible by all primes up to P_{max}.

P_{max}	1	2	3	5	7	11	13	17	19
remaining curves	100 %	33 %	12 %	7.2 %	4.9 %	3.9 %	3.3 %	3.0 %	2.7 %

From this data, we see that for 97.3 % of the curves, the SEA algorithm will be stopped at a very early stage, thus spending only a tiny fraction of the running time of the whole computation. With usual reasonable heuristics, it is expected that about 500 full computations are required on average before finding a good pair of curve and twist. A single full SEA computation takes about 20 seconds for this size on a personal computer, hence in about 3 hours, we expect to build good parameters for a key-size of 200 bits. An example curve is given in Appendix A.

If there is a need to construct the curves in a constraint environment, then it is probably a better idea to use the theory of Complex Multiplication. We will not give the details here, since the construction is well described both in the literature and in the standards. For our purpose, it suffices to choose a group order and a twisted group order which are both primes.

4.2 TAU Distribution

Now, we show that the distribution of the master secret key K, if we take it at random either on the curve $\mathbb{E}$ or $\tilde{\mathbb{E}}$, is uniformly distributed on $\{0,1\}^\ell$, in a statistical way. On the one hand, we prove that it is statistically indistinguishable from the uniform distribution on $\{0, \ldots, p-1\}$ and then that the latter distribution is statistically indistinguishable from the uniform distribution on $\{0,1\}^\ell$ by using lemma 8 by replacing q by p. The proofs of the following lemmas are done in the full version [11]. Let us denote by $\mathcal{D}$ the distribution of K:

$$\begin{aligned}\mathcal{D} &= \{K = [\mathbf{R}_b]_{\mathsf{abs}} \big| b \stackrel{R}{\leftarrow} \{0,1\}, \mathbf{R}_0 \stackrel{R}{\leftarrow} \mathbb{E}, \mathbf{R}_1 \stackrel{R}{\leftarrow} \tilde{\mathbb{E}}\} \\ &= \{K = x_b \big| b \stackrel{R}{\leftarrow} \{0,1\}, x_0 \stackrel{R}{\leftarrow} [\mathbb{E}]_{\mathsf{abs}}, x_1 \stackrel{R}{\leftarrow} [\tilde{\mathbb{E}}]_{\mathsf{abs}}\}.\end{aligned}$$

Lemma 10. *The distribution $\mathcal{D}$ is statistically close to the uniform distribution $\mathcal{U}_p$ in $\mathbb{F}_p \sim \mathbb{Z}_p$:*

$$\delta = \frac{1}{2} \times \sum_{x \in \mathbb{F}_p} \left| \Pr_{K \stackrel{R}{\leftarrow} \mathcal{U}_p}[K = x] - \Pr_{K \stackrel{R}{\leftarrow} \mathcal{D}}[K = x] \right| \leq \frac{1}{\sqrt{2^{\ell-1}}}.$$

Corollary 11. *The statistical distance between the uniform distribution on $\mathcal{U}_\ell$ and the* TAU *technique if $|p - 2^\ell| \leq 2^{\ell/2}$, is upper bounded by $(1+\sqrt{2})/\sqrt{2^\ell}$ according to Lemmas 10 and 8.*

Note 12. However, in an actual scheme, the bit b many not be perfectly uniformly distributed, but biased in a negligible way. Anyway, it will be important to show that such a bias will not impact much the distribution of the key (see the proof of Theorem 13.)

4.3 Working using Abscissae Only

In the basic description, even if only the abscissa of a point is used at the end to derive the key, we worked all along with points on the elliptic curves. In fact, this is not necessary. Let $\mathbf{P}$ be a point on an elliptic curve, then to compute the abscissa of a multiple of $\mathbf{P}$, only the abscissa of $\mathbf{P}$ is required. This is a very classical result, that is used for instance in fast versions of the ECM factoring algorithm [25].

As a consequence, it is possible to improve the TAU protocol as follows (see figure 1): each time there is a point on a curve, we replace it by just its abscissa. In particular, now X_0, X_1, Y_0 and Y_1 are just elements of $\mathbb{F}_p$ which are abscissae of points on the curve or on the twist. We then denote by $x \circ X$ the abscissa of the point $\mathbf{Y}$ which is x times a point $\mathbf{X}$ whose abscissa is X. The space saving is tiny (namely just the one bit that was used to code the ordinate), but this has the advantage to put in light the fact that ordinate's role is irrelevant in the TAU protocol. Furthermore, this improves the time complexity by more than 30%, at least from Bob's view point. Indeed, while in the basic Diffie-Hellman protocol both Alice and Bob have to compute 2 exponentiations, in the TAU version, Alice has to compute 3.5 on average (an additional cost of 75%), and Bob still 2 only (just a negligible additional cost due to the computation with abscissae only.) The use of the 2 coordinates of the points would require an additional square root computation, and thus an exponentiation in the field. Such an operation is much less expensive than the computation of the multiple of a point in the curve, but its cost is not negligible.

Note that not all EC-based protocols can be transformed to work only with abscissae. For instance, El-Gamal signatures involve additions in the elliptic curve, and this cannot be done only with the input of abscissae of the points; only an exponentiation is feasible. TAU can use this improved technique.

4.4 Efficient and Unconditionally Secure Pseudo-random Functions

Roughly, our TAU technique runs twice the basic scheme (but with an actual cost of only 37% more), and provides a long bit-string which is uniformly distributed, under the Elliptic Curve Decisional Diffie-Hellman assumption. Such a long bit-string K allows an efficient and secure key re-generation, to get both a key confirmation k_m and a session/master key sk, without any additional assumption about pseudo-random functions: K can be simply split into k_m and sk, with convenient sizes.

For the same security level, the LHL would require a group of order around q^2, and thus with a complexity exactly twice as much as the basic scheme. With the above improved technique using abscissae, our technique does not double the whole basic scheme, but the complexity is just increased by a factor 1.38. We thus get an average improvement of 30% if we compare to the LHL.

5 The "Twist-AUgmented" Authenticated Diffie-Hellman Protocol

5.1 Description

Using the properties of "Twist-AUgmented" deterministic randomness extractor, we then convert any Diffie-Hellman-like protocol, which provides a random element in a cyclic group, into a protocol which provides a random bit-string, without any additional assumptions. See figure 1 for the description, which implements the above improvement using abscissae only.

5.2 Semantic Security

On Figure 1, we present the TAU-enhancement of a classical authenticated Diffie-Hellman key exchange: basically, some flows are doubled, on each curve. However, Bob randomly chooses the curve which will be used for the Diffie-Hellman computation, and compute correct values on this curve only. For the other part, he plays randomly. This protocol achieves the property of semantic security under the elliptic-curve decisional Diffie-Hellman assumption and does not use ideal-hash functions. In order to prove this claim (the full proof is postponed to the the full version [11]) we consider games that have distances that can be measured easily. We use Shoup's lemma to bound the probability of events in successive

Alice		*Bob*
	Common twisted curves $\mathbb{E}_0, \mathbb{E}_1$ over the finite field $\mathbb{F}_p$ of respective prime orders q_0, q_1 $\mathcal{X}_i = [\mathbb{E}_i = \langle \mathbf{P}_i \rangle]_{\mathsf{abs}} = \langle P_i \rangle_{\mathsf{abs}},$ where $P_i = [\mathbf{P}_i]_{\mathsf{abs}},$ for $i = 0, 1$	
Signing Key : sk_A		Signing Key : sk_B
Verification Key : vk_A		Verification Key : vk_B
$\mathsf{accept} \leftarrow \mathsf{false}$		$\mathsf{accept} \leftarrow \mathsf{false}$
$\mathsf{terminate} \leftarrow \mathsf{false}$		$\mathsf{terminate} \leftarrow \mathsf{false}$
$s \stackrel{R}{\leftarrow} \{0,1\}^\star, x_0 \stackrel{R}{\leftarrow} \mathbb{Z}_{q_0}, X_0 = x_0 \circ P_0$		
$x_1 \stackrel{R}{\leftarrow} \mathbb{Z}_{q_1}, X_1 = x_1 \circ P_1$		
$\sigma_A = \mathsf{AUTH.Sign}(\mathsf{sk}_A; (s, X_0, X_1))$	$\xrightarrow[X_0, X_1, \sigma_A]{\textit{Alice}, s}$	Check σ_A
		$\beta \stackrel{R}{\leftarrow} \{0,1\}$
		$y_\beta \stackrel{R}{\leftarrow} \mathbb{Z}_{q_\beta}, Y_\beta = y_\beta \circ P_\beta$
		$Y_{1-\beta} \stackrel{R}{\leftarrow} \mathcal{X}_{1-\beta}$
		$K_\beta = y_\beta \circ X_\beta,$
		$k_m = \mathsf{MacKey}(K_\beta)$
		$\sigma_B = \mathsf{AUTH.Sign}(\mathsf{sk}_B; (s, X_0, X_1, Y_0, Y_1))$
Check σ_B	$\xleftarrow[Y_0, Y_1, \sigma_B, \mu_B]{\textit{Bob}, s}$	$\mu_B = \mathsf{MAC.Sign}(k_m; (\text{“1”}, s, \textit{Bob}))$
$d \stackrel{R}{\leftarrow} \{0,1\}, K = x_d \circ Y_d$		
$k_m = \mathsf{MacKey}(K)$		
Try to check μ_B : in case of failure $d = 1 - d, K = x_d \circ Y_d$ $k_m = \mathsf{MacKey}(K)$		
Check μ_B		
$\mu_A = \mathsf{MAC.Sign}(k_m, (\text{“0”}, s, \textit{Alice}))$		
$\mathsf{accept} \leftarrow \mathsf{true}$	$\xrightarrow[\sigma_A, \mu_A]{s}$	Check μ_A $\mathsf{accept} \leftarrow \mathsf{true}$
$\mathsf{terminate} \leftarrow \mathsf{true}$		$\mathsf{terminate} \leftarrow \mathsf{true}$

$$\mathsf{sk} = \mathsf{SessionKey}(K)$$
$$\mathsf{sid} = s, \textit{Alice}, \textit{Bob}, X_0, X_1, Y_0, Y_1, \sigma_A, \sigma_B, \mu_A, \mu_B$$

where $\mathsf{SessionKey}(K) = \mathsf{PRF}_K(0), \mathsf{MacKey}(K) = \mathsf{PRF}_K(1)$
$[\mathbf{R}]_{\mathsf{abs}}$ is the abscissa of the point $\mathbf{R}$ in $\mathbb{F}_p$
$x \circ P$ is the abscissa of x times a point $\mathbf{P}$ whose abscissa is P
and when a check fails whithout being caught, one stops the execution: $\mathsf{terminate} \leftarrow \mathsf{true}$

Fig. 1. An honest execution of the “Twist-AUgmented” Authenticated Diffie-Hellman protocol.

games [30, 32]. The first game $\mathbf{G}_1$ goes back to the less efficient, but equivalent, protocol using abscissae and ordinates, and the second game $\mathbf{G}_2$ allows us to avoid active attacks, granted signatures, so that in the following games we only have to worry about replay attacks. Proving the claim boils down to coming up with the appropriate games $\mathbf{G}_3$ through $\mathbf{G}_8$, in which we obtain a random master key K uniformly distributed in $\{0, \ldots, 2^\ell - 1\}$. The game $\mathbf{G}_9$, providing random session keys, is then easy to come up with and therefore the proof of the claim easily follows. In the last game $\mathbf{G}_9$, the adversary has indeed clearly no means to get any information about the random bit involved in the Test-query except to flip a coin.

Theorem 13. *For any adversary $\mathcal{A}$ running within time bound t, with less than q_s different sessions*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{ake}}_{\mathsf{TAU}}(\mathcal{A}) \leq {} & 4 \cdot \mathsf{Succ}^{\mathsf{euf-cma}}_{\mathsf{AUTH}}(2t, q_s, q_s) + 10 \cdot \mathsf{Succ}^{\mathsf{euf-cma}}_{\mathsf{MAC}}(2t, 1, 0) \\ & + 2 \cdot \mathsf{Adv}^{\mathsf{ecddh}}_{\mathbf{P},\langle\mathbf{P}\rangle}(t') + 2 \cdot \mathsf{Adv}^{\mathsf{ecddh}}_{\mathbf{Q},\langle\mathbf{Q}\rangle}(t') \\ & + 2q_s \mathsf{Adv}^{\mathsf{prf}}_{\mathcal{F}}(t', 2) + 20 \mathsf{Adv}^{\mathsf{prf}}_{\mathcal{F}}(2t, 1) + \frac{20 + 5q_s}{\sqrt{2^\ell}},\end{aligned}$$

where $t' \leq t + 8 \times q_s T_m$, and T_m is an upper-bound on the time to compute the multiplication of a point by a scalar.

Conclusion

This paper presents a new technique in order to get an appropriate session key with Diffie-Hellman key exchanges. It provides the best efficiency, since it is more than 30% more efficient than using the Leftover Hash Lemma, while it does not require any authenticated randomness.

Acknowledgement

The work described in this paper has been supported in part by the European Commission through the IST Programme under Contract IST-2002-507932 ECRYPT. The first author is supported by the Director, Office of Science, Office of Advanced Scientific Computing Research, Mathematical Information and Computing Sciences Division, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098. This document is report LBNL-54709. Disclaimer available at `http://www-library.lbl.gov/disclaimer`.

References

1. M. Abdalla, M. Bellare, and P. Rogaway. The Oracle Diffie-Hellman Assumptions and an Analysis of DHIES. In *CT – RSA '01*, LNCS 2020, pages 143–158. Springer-Verlag, 2001.

2. B. Barak and S. Halevi. An architecture for robust pseudo-random generation and applications to `/dev/random`. In *Proc. of ACM CCS*, ACM, 2005.
3. B. Barak, R. Shaltiel and E. Tromer. True Random Number Generators Secure in a Changing Environment. In *CHES '03*, pages 166–180. LNCS 2779, 2003.
4. S. M. Bellovin and M. Merritt. Encrypted Key Exchange: Password-Based Protocols Secure against Dictionary Attacks. In *Proc. of the Symposium on Security and Privacy*, pages 72–84. IEEE, 1992.
5. M. Bellare, R. Canetti and H. Krawczyk. Keying Hash Functions for Message Authentication. In *Crypto '96*, LNCS 1109, pages 1–15. Springer-Verlag, 1996.
6. M. Bellare and P. Rogaway. Random Oracles Are Practical: a Paradigm for Designing Efficient Protocols. In *Proc. of ACM CCS*, pages 62–73. ACM Press, 1993.
7. B. Möller. A Public-Key Encryption Scheme with Pseudo-Random Ciphertexts. In *ESORICS '04*, LNCS 3193, pages 335–351. Springer-Verlag, Berlin, 2004.
8. C. Boyd, P. Montague, and K. Nguyen. Elliptic Curve Based Password Authenticated Key Exchange Protocols. In *ACISP '01*, LNCS 2119, pages 487–501. Springer-Verlag, 2001.
9. R. Canetti, Y. Dodis, S. Halevi, E. Kushilevitz and A. Sahai. Exposure-Resilient Functions and All-Or-Nothing Transforms. In *Eurocrypt '00*, LNCS 1807, pages 453–469. Springer-Verlag, 2000.
10. O. Chevassut, P. A. Fouque, P. Gaudry, and D. Pointcheval. Key Derivation and Randomness Extraction. ePrint Report 2005/061. Available at `http://eprint.iacr.org/`.
11. O. Chevassut, P. A. Fouque, P. Gaudry, and D. Pointcheval. The Twist-Augmented Technique for Key Exchange. Full version available at `http://www.di.ens.fr/users/pointche/pub.php`.
12. Q. Dang and T. Polk. Hash-Based Key Derivation. draft-dang-nistkdf-00.txt. Available at `http://www.ietf.org/internet-drafts/`.
13. W. Diffie and M. E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, IT–22(6):644–654, November 1976.
14. Y. Dodis. Exposure-Resilient Cryptography. *PhD Thesis*, MIT, August 2000.
15. Y. Dodis, R. Gennaro, J. Håstad, H. Krawczyk, and T. Rabin. Randomness Extraction and Key Derivation Using the CBC, Cascade and HMAC Modes. In *Crypto '04*, LNCS, pages 494–510. Springer-Verlag, 2004.
16. Y. Dodis, A. Sahai, A. Smith. On perfect and adaptive security in exposure-resilient cryptography. In *Eurocrypt '01*, LNCS 2405, pages 301–324. Springer-Verlag, 2001.
17. R. Gennaro, H. Krawczyk, and T. Rabin. Secure Hashed Diffie-Hellman over Non-DDH Groups. In *Eurocrypt '04*, LNCS 3027, pages 361–381. Springer-Verlag, 2004.
18. D. Harkins and D. Carrel. The Internet Key Exchange (IKE). RFC 2409, 1998.
19. J. Håstad, R. Impagliazzo, L. Levin, and M. Luby. A Pseudorandom Generator from any One-Way Function. *SIAM Journal of Computing*, 28(4):1364–1396, 1999.
20. I. Impagliazzo, L. Levin, and M. Luby. Pseudo-Random Generation from One-Way Functions. In *Proc. of the 21st STOC*, pages 12–24. ACM Press, New York, 1989.
21. I. Impagliazzo and D. Zuckerman. How to Recycle Random Bits. In *Proc. of the 30th Annual IEEE FOCS*, pages 248–253, 1989.
22. B. Kaliski. One-Way Permutations on Elliptic Curves. *Journal of Cryptology*, 3(3):187–199, 1991.
23. J. Kamp and D. Zuckerman. Deterministic Extractors for Bit-Fixing Sources and Exposure-Resilient Cryptography. In *Proc. of the 44th Annual IEEE Symposium on Foundations of Computer Science*, 2003.

24. C. Kaufman. The Internet Key Exchange (IKEv2) Protocol. INTERNET-DRAFT draft-ietf-ipsec-ikev2-17.txt, September 23, 2004. Available at `http://www.ietf.org/internet-drafts/draft-ietf-ipsec-ikev2-17.txt`
25. P. L. Montgomery. *An FFT Extension of the Elliptic Curve Method of Factorization.* PhD thesis, University of California – Los Angeles, 1992.
26. M. Santha and U. V. Vazirani. Generating quasi-random sequences from semi-random sources. In *J. of Computer and System Sciences*, 63:612–626, 1986.
27. R. Schoof. Counting Points on Elliptic Curves over Finite Fields. In *J. Théor. Nombres Bordeaux*, 7:219–254, 1995.
28. R. Shaltiel. Recent developments in Extractors. In *Bulletin of the European Association for Theoretical Computer Science*, Volume 77, June 2002, pages 67–95. Available at `http://www.wisdom.weizmann.ac.il/~ronens/papers/survey.ps`, 2002.
29. V. Shoup. A Proposal for an ISO Standard for Public-Key Encryption, december 2001. ISO/IEC JTC 1/SC27.
30. V. Shoup. OAEP Reconsidered. In *Crypto '01*, LNCS 2139, pages 239–259. Springer-Verlag, Berlin, 2001.
31. V. Shoup. A Computational Introduction to Number Theory Algebra. In *Cambridge University Press*, 2005. Freely available at `http://www.shoup.net/ntb/`.
32. V. Shoup. Sequences of Games: A Tool for Taming Complexity in Security Proofs. Available at `http://www.shoup.net/papers/`, 2004.
33. J. H. Silverman. *The Arithmetic of Elliptic Curves*, volume 106 of *Graduate Texts in Mathematics*. Springer-Verlag, 1986.
34. T. Dierks and C. Allen. The TLS Protocol Version 1.0. RFC 2246, January 1999. OpenSSL. version 0.9.7e
35. L. Trevisan and S. Vadhan. Extracting Randomness from Samplable Distributions. In *Proc. of the 41st Annual IEEE FOCS*, 2000.

A An Example 200-bit Pair of Curve and Twist

We give a pair of curve and twist suitable for implementing the TAU protocol. This curve was produced using the method sketched in Section 4.1. We choose a curve with $a = -3$, to allow the use of the fast projective group law.

Let $\ell = 200$, and let $p = 2^\ell - 978579$. Let b in $\mathbb{F}_p$ be given by

$$b = 386119362724722930774569388602676779780560253666503462427823.$$

The trace of the curve $\mathbb{E}$ of equation $y^2 = x^3 - 3x + b$, is

$$t_{\mathbb{E}} = -1864972684066157296039917581949.$$

Hence, the group orders of $\mathbb{E}$ and of its twist $\tilde{\mathbb{E}}$ are $p + 1 \pm t_{\mathbb{E}}$, which are both prime numbers.