ON IMPLEMENTING PAIRING-BASED PROTOCOLS WITH ELLIPTIC CURVES OF EMBEDDING DEGREE ONE

SANJIT CHATTERJEE, ALFRED MENEZES, AND FRANCISCO RODRÍGUEZ-HENRÍQUEZ

Abstract. We observe that the conventional classification of pairings into Types 1, 2, 3 and 4 is not applicable to pairings from elliptic curves with embedding degree one. We define three kinds of pairings from these elliptic curves, and discuss some subtleties with using them to implement pairing-based protocols.

1. Introduction

Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be groups of prime order n. A cryptographic pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a map that is bilinear, non-degenerate and efficiently computable. Since 2000, when Boneh and Franklin proposed their identity-based encryption scheme [11], cryptographic pairings have been used extensively to design a wide variety of cryptographic protocols.

Cryptographic pairings are constructed from elliptic curves of small embedding degree. More precisely, let E be an elliptic curve defined over the finite field $\mathbb{F}_q$. Let n be a prime divisor of $\#E(\mathbb{F}_q)$ with $\gcd(n, q) = 1$, and let k be the smallest positive integer such that $n \mid q^k - 1$; the number k is called the embedding degree of E (with respect to n). Then $\mathbb{G}_1$ is an order-n subgroup of $E(\mathbb{F}_q)$, $\mathbb{G}_2$ is an order-n subgroup of $E(\mathbb{F}_{q^k})$, $\mathbb{G}_T$ is the order-n subgroup of $\mathbb{F}^*_{q^k}$, and the map e is derived from the classical Weil and Tate pairings. Among the elliptic curves that have been used to implement pairings are supersingular curves of embedding degree 4 over finite fields of characteristic two (see [7]), supersingular curves of embedding degree 6 over finite fields of characteristic three (see [7]), supersingular and ordinary curves of embedding degree 2 over finite fields of prime order (see [44]), and Barreto-Naehrig (BN) ordinary curves of embedding degree 12 over finite fields of prime order [9].

A necessary condition for the security of pairing-based protocols is that the discrete logarithm problem (DLP) in $\mathbb{F}_{q^k}$ is intractable. Until recently, the assumption was that the fastest algorithm for computing logarithms in small-characteristic finite fields $\mathbb{F}_q$ was Coppersmith's algorithm [19] with running time $L_q[\frac{1}{3}, (\frac{32}{9})^{1/3}] \approx L_q[\frac{1}{3}, 1.526]$, and the fastest algorithm for computing logarithms in large-characteristic finite fields $\mathbb{F}_q$ was the Number Field Sieve (NFS) [25, 41] with running time $L_q[\frac{1}{3}, (\frac{64}{9})^{1/3}] \approx L_q[\frac{1}{3}, 1.923]$. Here, $L_Q[\alpha, c]$ with $0 < \alpha < 1$ and $c > 0$ denotes the expression

$$O\left(\exp\left((c + o(1))(\log Q)^{\alpha}(\log\log Q)^{1-\alpha}\right) \right)$$

Date: April 22, 2016.

that is subexponential in $\log Q$. However, recent work has shown that these assumptions were overly optimistic. Most dramatically, in 2013 a quasi-polynomial time algorithm (with running time $L_q[\epsilon, c]$ for any $\epsilon > 0$) was devised for computing discrete logarithms in small-characteristic finite fields $\mathbb{F}_q$ [6], effectively breaking all pairings derived from supersingular elliptic curves over small-characteristic finite fields (see [1, 26]). In 2015, Kim and Barbulescu [35] devised a variant of the NFS that computes logarithms in $\mathbb{F}_q = \mathbb{F}_{p^k}$ in time $L_q[\frac{1}{3}, (\frac{48}{9})^{1/3}] \approx L_q[\frac{1}{3}, 1.759]$ when p is a medium-sized prime (more precisely, $p = L_q[\alpha, c]$ with $\frac{1}{3} < \alpha < \frac{2}{3}$). Kim and Barbulescu state that their algorithm can be applied to computing logarithms in the fields $\mathbb{F}_{p^{12}}$ that arise in BN pairings. Moreover, if p has a special form, as is the case with BN pairings, a further refinement of their algorithm has running time $L_q[\frac{1}{3}, (\frac{32}{9})^{1/3}]$. Also in 2015, experiments conducted by Barbulescu et al. [5] illustrated that the DLP is significantly easier in $\mathbb{F}_{p^2}$ than in prime-order fields $\mathbb{F}_p$.

The aforementioned improvements in algorithms for computing discrete logarithms cast some suspicions on the true intractability of the DLP in extension fields $\mathbb{F}_{p^k}$, especially if the characteristic p is of a special form. On the other hand, these improvements do not apply to the DLP in prime-order fields $\mathbb{F}_p$ provided that the prime p does not have a special form; thus, the fastest general-purpose algorithm known for the DLP in $\mathbb{F}_p$ has running time $L_p[\frac{1}{3}, (\frac{64}{9})^{1/3}]$. Consequently, elliptic curves with embedding degree $k = 1$ would appear to be a conservative choice for implementing pairing-based protocols. The group $\mathbb{G}_T$ in these pairings is the order-n subgroup of the multiplicative group of a prime field $\mathbb{F}_p^*$, whence security is not directly affected by advances in algorithms for computing logarithms in extension fields.

Remark 1. (*DLP in $\mathbb{F}_p$ versus DLP in $\mathbb{F}_{p^k}$*) Since $\mathbb{F}_p$ is a subfield of $\mathbb{F}_{p^k}$ for any $k \geq 2$, the DLP in $\mathbb{F}_p$ can be reduced to the DLP in $\mathbb{F}_{p^k}$. Hence, advances in algorithms for the DLP in $\mathbb{F}_{p^k}$ might be effective for solving the DLP in $\mathbb{F}_p$. For example, an $L_{p^k}[\alpha, c]$ algorithm for solving the DLP in $\mathbb{F}_{p^k}$ (where $k > 1$ is a constant) yields an $L_p[\alpha, ck^\alpha]$ algorithm for the DLP in $\mathbb{F}_p$. Hence, an $L_{p^2}[\frac{1}{3}, c]$ algorithm with $c < (\frac{32}{9})^{1/3}$ for the DLP in $\mathbb{F}_{p^2}$ yields a DLP algorithm for $\mathbb{F}_p$ that is faster than the best currently known algorithm. Similarly, an $L_{p^{12}}[\frac{1}{3}, c]$ algorithm with $c < (\frac{16}{27})^{1/3}$ for the DLP in $\mathbb{F}_{p^{12}}$ yields a faster DLP algorithm for $\mathbb{F}_p$.

Elliptic curves with embedding degree one were first mentioned in [31, 32, 50] and further studied in [36, 30, 53]. These papers consider methods for generating suitable elliptic curves, and study the efficiency of the Weil and Tate pairings. However, none of these papers examine the implementation of pairing-based protocols with these pairings. In this paper we observe that the often-cited classification of pairings into Types 1, 2, 3 and 4 [22] is not applicable to pairings from elliptic curves with embedding degree one. We define three kinds of pairings from these elliptic curves, and discuss some subtleties with using them to implement pairing-based protocols.

The remainder of the paper is organized as follows. The classical Weil and Tate pairings are reviewed in §2. The salient features of elliptic curves with embedding degree one are presented in §3. Three kinds of pairings from elliptic curves with embedding degree one

are introduced in §§4–6. The efficiency of these pairings is considered in §7. We draw our conclusions in §8.

2. Pairings

Let $p > 3$ be a prime, and let E be an elliptic curve defined over the finite field $\mathbb{F}_p$. Let $\pi : (x, y) \mapsto (x^p, y^p)$ be the p-th power Frobenius endomorphism. The *trace* of the Frobenius is $t = p + 1 - \#E(\mathbb{F}_p)$. Let $n \neq p$ be a prime with $n \mid \#E(\mathbb{F}_p)$. The *embedding degree* is the smallest positive integer k satisfying $n \mid (p^k - 1)$. We will assume that $E[n]$, the group of all n-torsion points in $E(\overline{\mathbb{F}}_p)$, is contained in $E(\mathbb{F}_{p^k})$; this is indeed the case whenever $k > 1$ [4].

2.1. **Miller functions.** Let $R \in E(\mathbb{F}_{p^k})$ and let s be a non-negative integer. A *Miller function* $f_{s,R}$ [37] of length s is a function in $\mathbb{F}_{p^k}(E)$ with divisor $(f_{s,R}) = s(R) - (sR) - (s - 1)(\infty)$. Let u_∞ be an $\mathbb{F}_p$-rational uniformizing parameter for ∞. A function $f \in \mathbb{F}_{p^k}(E)$ is said to be *normalized* if $lc_\infty(f) = 1$, where $lc_\infty(f) = (u_\infty^{-t} f)(\infty)$ and t is the order of f at ∞.

Let $P, Q \in E[n] \backslash \{\infty\}$. Miller [37] described Algorithm 1 for evaluating a normalized Miller function $f_{n,P}$ at the point Q. In the algorithm, the line functions ℓ and v are normalized.

Algorithm 1 Miller's algorithm

Input: $P, Q \in E[n] \setminus \{\infty\}$.
Output: $f_{n,P}(Q)$.

1: Write n in binary: $n = \sum_{i=0}^{L-1} n_i 2^i$
2: $f \leftarrow 1$, $g \leftarrow 1$, $T \leftarrow P$
3: **for** $i \leftarrow L - 2$ **downto** 0 **do**
4: Compute the tangent line ℓ through T and the vertical line v through $2T$
5: $T \leftarrow 2T$
6: $f \leftarrow f^2 \cdot \ell(Q)$
7: $g \leftarrow g^2 \cdot v(Q)$
8: **if** $n_i = 1$ **then**
9: Compute the line ℓ through T and P and the vertical line v through $T + P$
10: $T \leftarrow T + P$
11: $f \leftarrow f \cdot \ell(Q)$
12: $g \leftarrow g \cdot v(Q)$
13: **end if**
14: **end for**
15: **return** f/g

Remark 2. (*failure of a Miller function computation*) When Miller's algorithm is used to compute $f_{n,P}(Q)$, one might obtain a value of 0 in the numerator or denominator. This occurs only if Q happens to be a root of one of the line functions ℓ, v encountered in the

computation. Since the roots of ℓ and v must lie in $\langle P \rangle$, we conclude that a Miller function computation can only fail if $Q \in \langle P \rangle$.

2.2. **Weil and Tate pairings.** Let $\mathbb{G}_T$ be the order-n subgroup of $\mathbb{F}_{p^k}^*$. The *Weil pairing* w_n can be defined as follows.

Definition 3 ([37]). *Let $P, Q \in E[n] \setminus \{\infty\}$ with $P \neq Q$. Then the Weil pairing is*

$$w_n(P,Q) = (-1)^n \frac{f_{n,P}(Q)}{f_{n,Q}(P)}, \tag{1}$$

where $f_{n,P}$ and $f_{n,Q}$ are normalized Miller functions. Furthermore, $w_n(P,\infty) = w_n(\infty,P) = w_n(P,P) = 1$ for all $P \in E[n]$.

The (reduced) *Tate pairing* t_n can be defined as follows.

Definition 4 ([49]). *Let $P, Q \in E[n]$. Let $R \in E(\mathbb{F}_{p^k})$ with $R \notin \{\infty, P, -Q, P-Q\}$. Then the Tate pairing is*

$$t_n(P,Q) = \left(\frac{f_{n,P}(Q+R)}{f_{n,P}(R)} \right)^{(p^k-1)/n}. \tag{2}$$

If $f_{n,P}$ is normalized, then

$$t_n(P,Q) = (f_{n,P}(Q))^{(p^k-1)/n} \tag{3}$$

for all $P, Q \in E[n] \setminus \{\infty\}$ with $P \neq Q$.

2.3. **Type 1, 2, 3 and 4 pairings.** In cryptographic applications, one generally considers the restrictions of the Weil and Tate pairings to a domain $\mathbb{G}_1 \times \mathbb{G}_2$, where $\mathbb{G}_1$ and $\mathbb{G}_2$ are fixed order-n subgroups of $E[n]$.

Suppose now that the embedding degree k is even. Galbraith, Paterson and Smart [22] defined three kinds of pairings $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. The pairing e is of Type 1 (symmetric) if $\mathbb{G}_1 = \mathbb{G}_2$. Now, let $\mathbb{G}_1 = E(\mathbb{F}_p)[n]$. Let d be the order of the automorphism group of E, and suppose that $d \mid k$. Let $e = k/d$. Then there is a unique degree-d twist $\tilde{E}$ of E over $\mathbb{F}_{p^e}$ with $n \mid \#\tilde{E}(\mathbb{F}_{p^e})$ [29]; let $\mu : \tilde{E} \to E$ be the associated twisting isomorphism. Let $\tilde{Q} \in \tilde{E}(\mathbb{F}_{p^e})$ be a point of order n, and let $\mathbb{G}_2 = \langle Q \rangle$ where $Q = \mu(\tilde{Q})$. The group $\mathbb{G}_2$ is called the Trace-0 subgroup of $E[n]$ since it is comprised of all points $P \in E[n]$ for which $\mathrm{Tr}(P) = \sum_{i=0}^{k-1} \pi^i(P) = \infty$. Then $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a Type 3 pairing. Finally, if $\mathbb{G}_2'$ is any order-n subgroup of $E[n]$ different from $\mathbb{G}_1$ and $\mathbb{G}_2$, then $e : \mathbb{G}_1 \times \mathbb{G}_2' \to \mathbb{G}_T$ is a Type 2 pairing.

Type 2 pairings have two properties that distinguish them from Type 3 pairings: an efficient method for hashing onto $\mathbb{G}_2'$ is not known[1], and there is an efficiently-computable isomorphism ψ from $\mathbb{G}_2'$ to $\mathbb{G}_1$ (given by the Trace map). In contrast, hashing onto $\mathbb{G}_2$ can

[1]When we say that "hashing onto a group is not known to be efficient" we mean in such a way that discrete logarithms of hash values are difficult to compute.

be done efficiently and no efficiently-computable isomorphism from $\mathbb{G}_2$ to $\mathbb{G}_1$ is known for Type 3 pairings.

Shacham [47] proposed pairings $e : \mathbb{G}_1 \times E[n] \to \mathbb{G}_T$, subsequently named Type 4 pairings. These pairings have the distinguishing property that the Trace map is an efficiently-computable homomorphism from $E[n]$ onto $\mathbb{G}_1$ with kernel $\mathbb{G}_2$. Furthermore, hashing onto $E[n]$ is relatively efficient [16].

3. Elliptic curves with embedding degree one

For the sake of concreteness, we will only consider a particular trace-2 elliptic curve from [36] in this paper. Elliptic curves with embedding degree one and having trace different from 2 can be generated using the complex multiplication method [14, 30].

Let p be a prime of the form

$$p = A^2 + 1, \text{ where } A \equiv 2 \pmod 4; \tag{4}$$

note that $p \equiv 1 \pmod 4$. Suppose also that $A = hn$ where $n \equiv 3 \pmod 4$ is prime. We assume that p does not have any special properties such as having a sparse binary representation. This is done in order to ensure that the discrete logarithm problem in $\mathbb{F}_p$ does not succumb to variants of the special number field sieve [42]. Then, assuming that the NFS cannot exploit the property of $p-1$ being a perfect square (cf. §3.3), the fastest algorithm known for computing discrete logarithms in $\mathbb{F}_p$ has running time $L_p[\frac{1}{3}, \left(\frac{64}{9}\right)^{1/3}]$.

Consider the elliptic curve

$$E : Y^2 = X^3 - 4X \text{ over } \mathbb{F}_p. \tag{5}$$

Then, as shown in [36], $\#E(\mathbb{F}_p) = p-1$ so E is an ordinary curve with trace 2. Furthermore, $E(\mathbb{F}_p) \cong \mathbb{Z}_A \oplus \mathbb{Z}_A$ so $E[n] \subseteq E(\mathbb{F}_p)$ and $E(\mathbb{F}_p)$ has embedding degree $k = 1$ with respect to n. The endomorphism ring of E is $\mathbb{Z}[i]$, the complex number i being identified with the degree-one endomorphism ψ:

$$\psi : (x, y) \mapsto (-x, Ay). \tag{6}$$

The map ψ is a distortion map on $E[n]$. In other words, if $P \in E[n] \setminus \{\infty\}$ then $\psi(P) \notin \langle P \rangle$. [2] Thus, for any $P \in E[n] \setminus \{\infty\}$, the pair of points $(P, \psi(P))$ generate $E[n]$. The p-th power Frobenius π satisfies $\pi(P) = (1 + Ai)P$ for all $P \in E(\overline{\mathbb{F}}_p)$.

The existence of an efficiently-computable distortion map on $E[n]$ means that the decisional Diffie-Hellman (DDH) problem in any order-n subgroup of $E[n]$ is easy. Recall that the DDH problem in an order-n group $\langle P \rangle$ is the following: given P, $A = aP$, $B = bP$ and Q, where $a, b \in_R [0, n-1]$ and either $Q = abP$ or $Q = cP$ for $c \in_R [0, n-1]$, decide whether or not $Q = abP$. The Weil pairing can be used to solve DDH efficiently since if $Q = abP$ then $w_n(P, Q) = w_n(A, B)$, whereas if $Q = cP$ then $w_n(P, Q) \neq w_n(A, B)$ with overwhelming probability.

[2] If $n \equiv 1 \pmod 4$, then ψ is a distortion map for all but two of the $n+1$ order-n subgroups of $E[n]$. These two exceptions were overlooked in Theorem 3 of [36].

3.1. **Subgroup membership testing.** Let $P, Q \in E[n] \setminus \{\infty\}$. In cryptographic protocols it is usually necessary to have an efficient method for testing whether $Q \in \langle P \rangle$. The alternation and non-degeneracy properties of the Weil pairing imply that $Q \in \langle P \rangle$ if and only if $w_n(P, Q) = 1$. Thus, subgroup membership testing can be done using the Weil pairing.

Subgroup membership testing cannot in general be done in the same manner using the Tate pairing. Indeed, it is no longer guaranteed that $t_n(P, P) = 1$. The following result was stated in Remark 2 of [30]. We include a self-contained proof for completeness.

Lemma 5. *Let E be the trace-2 elliptic curve defined in (5). Then $t_n(P, \psi(P)) = 1$ for all $P \in E[n]$.*

Proof. Let $P \in E[n] \setminus \{\infty\}$. By Theorem X.1.1 of [48], we have $w_n(\pi(Q) - Q, iP) = \beta^{p-1}$, where $Q \in E(\overline{\mathbb{F}}_p)$ with $nQ = P$, and $\beta \in \overline{\mathbb{F}}_p$ with $\beta^n = \hat{t}_n(iP, P)$ where $\hat{t}_n$ is the unreduced Tate pairing. Now, $\beta^{p-1} = (\beta^n)^{(p-1)/n} = \hat{t}_n(iP, P)^{(p-1)/n} = t_n(iP, P)$. And, $\pi(Q) - Q = (1 + Ai)Q - Q = hniQ = hiP$. Thus, $t_n(iP, P) = w_n(hiP, iP) = w_n(iP, iP)^h = 1$.

Now, the dual of the endomorphism i is $-i$. Hence, by part 3 of Theorem IX.9 of [21], we have $1 = t_n(iP, P) = t_n(P, -iP) = t_n(P, iP)^{-1}$, whence $t_n(P, iP) = t_n(P, \psi(P)) = 1$. □

Non-degeneracy of the Tate pairing immediately gives the following result which yields a method for subgroup membership testing using the Tate pairing.

Corollary 6. *Let E be the trace-2 elliptic curve defined in (5). Let $P, Q \in E[n] \setminus \{\infty\}$. Then $Q \in \langle P \rangle$ if and only if $t_n(P, \psi(Q)) = 1$.*

Non-degeneracy of the Tate pairing also gives the following result.

Corollary 7. *Let E be the trace-2 elliptic curve defined in (5). Then $t_n(P, P) \neq 1$ for all $P \in E[n] \setminus \{\infty\}$.*

3.2. **Pairings.** For the trace-2 elliptic curve E defined in (5), there are no analogues to $\mathbb{G}_1$ and the Trace-0 group $\mathbb{G}_2$ in the case of Type 2 and Type 3 pairings (see §2.3). Indeed, there is no natural way of distinguishing any order-n subgroup of $E[n]$ from the other order-n subgroups.

There are several types of pairings on order-n groups that can be defined from the Weil pairing (1) and the Tate pairing (3). We present three kinds of pairings, called Type A, Type B and Type C, and discuss their properties in §§4-6. In all these pairings, $\mathbb{G}_T$ denotes the order-n subgroup of $\mathbb{F}_p^*$. Note that no efficient method is known for efficiently and reversibly embedding binary strings from $\{0, 1\}^m$ (for some m) in $\mathbb{G}_T$.

3.3. **Applicability of the SNFS.** Let $d > 1$ and $m = \lfloor p^{1/d} \rfloor$. Suppose also that $p > 2^{d^2}$. Let $f(t) = t^d + f_{d-1}t^{d-1} + \cdots + f_1 t + f_0 \in \mathbb{Z}[t]$, where $m^d + \sum_{i=0}^{d-1} f_i m^i$ is the base-m representation of p. Then f is irreducible over $\mathbb{Z}$ and $f(m) = p \equiv 0 \pmod{p}$. Let $\alpha \in \mathbb{C}$ be a root of f. The NFS for computing discrete logarithms in $\mathbb{F}_p$ [25, 41] seeks to find pairs of integers (a, b) with $0 \leq b < B$ and $0 \leq |a| < B$ such that

$$T = (a - bm) \cdot b^d f(a/b) = (a - bm)(a^d + f_{t-1}a^{t-1}b + \cdots + f_1 ab^{t-1} + f_0 b^t)$$

is smooth. Note that $T \approx (d+1)B^{d+1}p^{2/d}$. The optimal choices for B and d yield an algorithm with running time $L_p[\frac{1}{3}, \left(\frac{64}{9}\right)^{1/3}]$.

In the special NFS (SNFS), p has a special form, e.g., $p = r^e \pm s$ with small r and s. The special form of p yields a monic irreducible polynomial f with small coefficients. Consequently, the bound on T becomes $T \approx (d+1)B^{d+1}p^{1/d}$, resulting in an algorithm with a significantly faster running time $L_p[\frac{1}{3}, \left(\frac{32}{9}\right)^{1/3}]$.

For the case $p = A^2 + 1$, one could take $d = 2$ and $f(t) = t^2 + 1$. But then $T \approx 3B^3p^{1/2}$, which is too large to be effective. A second strategy to exploit the structure of p is to let $m = \lfloor A^{1/d} \rfloor$ and $g(t) = t^d + g_{d-1}t^{d-1} + \cdots + g_1t + g_0$, where $m^d + \sum_{i=0}^{d-1} g_i m^i$ is the base-m representation of A. Then, one can take $f(t) = g^2(t) + 1$ and hope that f is irreducible over $\mathbb{Z}$; note that $f(m) = p$ and $\deg(f) = 2d$. However, the coefficients f_i of f satisfy $f_i \approx m^2$. Indeed, by taking $\tilde{m} = \lfloor p^{1/2d} \rfloor \approx \lfloor A^{1/d} \rfloor$, one could have used the method in the original NFS to obtain a degree-$2d$ irreducible polynomial $\tilde{f}$ with $\tilde{f}(\tilde{m}) = p$ and coefficients $\tilde{f}_i \approx \tilde{m}$. Thus, this second strategy also fails.

An outstanding open problem is to determine whether the special structure of $p = A^2 + 1$ can be exploited to devise an algorithm for computing discrete logarithms in $\mathbb{F}_p$ that has running time significantly smaller than $L_p[\frac{1}{3}, \left(\frac{64}{9}\right)^{1/3}]$. If this can be accomplished, then the keylength and efficiency estimates in §7 will have to be revised accordingly.

4. Type A pairings

4.1. **The pairings.** Let $\mathbb{G}_1$ be an arbitrary order-n subgroup of $E(\mathbb{F}_p)$.

Definition 8. *The Type A Weil pairing* $w_A : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$ *is*

$$w_A(P, Q) = w_n(P, \psi(Q)) = (-1)^n \frac{f_{n,P}(\psi(Q))}{f_{n,\psi(Q)}(P)}, \tag{7}$$

where $P, Q \neq \infty$ *and* $f_{n,P}$ *and* $f_{n,\psi(Q)}$ *are normalized Miller functions. Furthermore,* $w_A(P, \infty) = w_A(\infty, P) = 1$ *for all* $P \in \mathbb{G}_1$.

The pairing w_A is non-degenerate since $\psi(Q) \notin \langle P \rangle$. Note that the Miller function computations in (7) never fail since $\psi(Q) \notin \langle P \rangle$ and $P \notin \langle \psi(Q) \rangle$.

Since $t_n(P, P) \neq 1$, the Tate pairing restricted to $\mathbb{G}_1 \times \mathbb{G}_1$ is a non-degenerate pairing. However, as mentioned in Remark 2, the Miller function computation when computing $t_n(P, Q)$ using (3) can fail since $Q \in \langle P \rangle$. To circumvent this possible failure, we select $R = \psi(P)$ in (2). Then, since $f_{n,P}(\psi(P))^{(p-1)/n} = 1$ when $f_{n,P}$ is normalized, we obtain the following Type A Tate pairing.

Definition 9. *The Type A Tate pairing* $t_A : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$ *is*

$$t_A(P, Q) = \left(f_{n,P}(Q + \psi(P))\right)^{(p-1)/n}, \tag{8}$$

where $f_{n,P}$ *is normalized.*

The Type A pairings w_A and t_A are symmetric in the sense that they are defined on $\mathbb{G}_1 \times \mathbb{G}_1$ where $\mathbb{G}_1$ is a cyclic group of order n. However, these pairings differ from Type 1 symmetric pairings in that no efficient method is known for hashing onto $\mathbb{G}_1$.

4.2. **Protocols.** Let $e_A : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$ denote the Type A Weil pairing w_A or the Type A Tate pairing t_A. The Type A pairing setting provides a protocol designer the same capabilities as in the Type 1 setting provided there is no need of hashing onto $\mathbb{G}_1$ (also known as a map-to-point function). Recall that such a hash function is usually modelled as a random oracle in the security reduction. Hence, all protocols that are originally described in the Type 1 setting and do not require the random oracle assumption for such a map-to-point function can be instantiated in the Type A setting.

Examples of protocols that can be implemented in the Type A setting include Joux's three-party key agreement scheme [31] and Waters' IBE and signature schemes [51]. More generally, all so-called standard model protocols in the Type 1 setting can be implemented in the Type A setting without any difficulty. Note that it is straightforward to cast the corresponding hardness assumptions and security theorems and proofs in the Type A setting. For example, the bilinear Diffie-Hellman problem in the Type A setting and its decisional version can be defined as follows.

Definition 10 (BDH-A)**.** *Bilinear Diffie-Hellman problem in* $(\mathbb{G}_1, \mathbb{G}_T)$*: Given* P*,* aP*,* bP*,* $cP \in \mathbb{G}_1$*, where* $a, b, c \in_R [1, n-1]$*, compute* $e_A(P,P)^{abc}$*.*

Definition 11 (DBDH-A)**.** *Decisional Bilinear Diffie-Hellman problem in* $(\mathbb{G}_1, \mathbb{G}_T)$*: Given* $P, aP, bP, cP \in \mathbb{G}_1$*, where* $a, b, c \in_R [1, n-1]$ *and* $Z \in \mathbb{G}_T$*, decide whether* $Z = e_A(P,P)^{abc}$ *or* Z *is a random element in* $\mathbb{G}_T$*.*

4.2.1. *BLS-1 signature scheme.* To illustrate the problem of implementing protocols that employ a map-to-point function onto $\mathbb{G}_1$, we recall the Boneh-Lynn-Shacham (BLS) signature scheme [13] in the Type 1 setting.

Let $e_1 : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$ be a Type 1 pairing where $\mathbb{G}_1 = \langle P \rangle$. Let $H : \{0,1\}^* \to \mathbb{G}_1$ be a hash function. The public parameters are e_1, P and H.

Alice's private key is an integer $a \in_R [1, n-1]$, while her public key is $A = aP$. To sign a message $m \in \{0,1\}^*$, Alice computes $M = H(m)$ and $\sigma = aM$. Her signature on m is σ. To verify the signed message (m, σ), Bob computes $M = H(m)$, verifies that $\sigma \in \langle \mathbb{G}_1 \rangle$, and accepts if and only if $e_1(P, \sigma) = e_1(A, M)$.

Observe that the BLS signature scheme *cannot* be instantiated in the Type A setting simply because there is no known efficient method for hashing onto $\mathbb{G}_1$. A similar obstruction is present for several other protocols including the original Boneh-Franklin IBE scheme [11].

5. Type B pairings

5.1. **The pairings.** Let $\mathbb{G}_1$ be an arbitrary order-n subgroup of $E(\mathbb{F}_p)$ and let $\mathbb{G}_2 = E[n]$. Then we have the following restriction of the Weil or Tate pairings:

$$e_B : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T. \tag{9}$$

We can now hash to $\mathbb{G}_2$ — by hashing to a random point in $E(\mathbb{F}_p)$ and then multiplying the resulting point by the cofactor $A/n = h$. However, as before, we cannot hash to any particular order-n subgroup of $\mathbb{G}_2$. We will henceforth assume that the group $\mathbb{G}_1$ is the same as the one in the definition of e_A; thus the restriction of e_B to $\mathbb{G}_1 \times \mathbb{G}_1$ is equal to e_A.

Since $\mathbb{G}_2$ is not an order-n group, e_B is not a Type 1, 2 or 3 pairing. The pairing e_B is reminiscent of a Type 4 pairing (see §2.3). However, unlike the case of Type 4 pairings, no efficiently-computable homomorphism is known from $\mathbb{G}_2$ onto $\mathbb{G}_1$.

Remark 12. (*failure of a Miller function computation*) The Type B Weil pairing

$$w_B(P,Q) = (-1)^n \frac{f_{n,P}(Q)}{f_{n,Q}(P)} \tag{10}$$

is degenerate if $Q \in \langle P \rangle$. Moreover, the computation of (10) can fail when $Q \in \langle P \rangle$ or when $P \in \langle Q \rangle$ (both these conditions are implied by $Q \in \langle P \rangle$). Thus protocols that use the Type B pairing w_B should ensure that $Q \notin \langle P \rangle$, or that this occurs only with negligible probability.

The Type B Tate pairing

$$t_B(P,Q) = (f_{n,P}(Q))^{(p-1)/n} \tag{11}$$

is degenerate if $Q \in \langle \psi(P) \rangle$. Moreover, the computation of (11) can fail when $Q \in \langle P \rangle$. Thus, protocols that use the Type B pairing t_B should ensure that $Q \notin \langle P \rangle$ and $Q \notin \langle \psi(P) \rangle$, or that this occurs only with negligible probability.

5.2. **Protocols.**

5.2.1. *BLS-B signature scheme.* Let E be an elliptic curve with embedding degree one, and let $e_B : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a Type B pairing where $\mathbb{G}_1 = \langle P \rangle$. Let $H : \{0,1\}^* \to E[n]$ be a hash function. The public parameters are E, P, e_B and H.

Alice's private key is an integer $a \in_R [1, n-1]$, while her public key is $A = aP$. To sign a message m, Alice computes $M = H(m)$ and $\sigma = aM$. Her signature on m is σ. To verify the signed message (m, σ), Bob computes $M = H(m)$, verifies that $\sigma \in \langle M \rangle$, and accepts if and only if $e_B(P, \sigma) = e_B(A, M)$.

Correctness of the verification algorithm follows because

$$e_B(P, \sigma) = e_B(P, aM) = e_B(aP, M) = e_B(A, M).$$

Note that since M is randomly selected from $E[n]$, the probability that $M \in \langle A \rangle$ or $M \in \langle \psi(A) \rangle$ is negligible (cf. Remark 12).

Recall that the Diffie-Hellman problem (DHP) in a cyclic group $\mathbb{G}_1 = \langle P \rangle$ is the following: Given P, $A = aP$ where $a \in_R [1, n-1]$, and $M \in_R \mathbb{G}_1$, compute $\sigma = aM$. Security of BLS-B can be proven under the assumption that the following variant of DHP is intractable.

Definition 13. *The DHP-B problem is the following: Given $P \in \mathbb{G}_1$, $A = aP$ where $a \in_R [1, n-1]$, and $M \in_R E[n]$, compute $\sigma = aM$.*

It is easy to see that DHP polynomial-time reduces to DHP-B. Namely, suppose we are given an instance (P, A, M) of DHP and a DHP-B oracle. We compute $Q = \psi(P)$ and $M' = M + rQ$ where $r \in_R [0, n-1]$. We present (P, A, M') to the oracle and obtain $\sigma' = aM'$. Since

$$\sigma' = aM' = a(M + rQ) = aM + r(aQ) = \sigma + r\psi(A),$$

we can efficiently compute the DHP solution $\sigma = \sigma' - r\psi(A)$.

Security of BLS-B can be proven with respect to intractability of DHP-B. The following result omits the tightness gap in the reduction.

Theorem 14. *Suppose that the DHP-B problem is intractable and H is a random oracle. Then BLS-B is existentially unforgeable against adaptive chosen-message attacks.*

Proof. Suppose we are given a DHP-B instance (P, A, M^*) and need to compute $\sigma^* = aM^*$ where $A = aP$. We compute $Q = \psi(P)$ and $B = \psi(A) = aQ$, where ψ is the distortion map (6). We then run the forger with input (P, A). The forger makes queries to a random oracle H and to a signing oracle. We assume that the forger always queries H with a message m before it queries the signing oracle with m. We also assume that the forger queries H with m before producing its forgery (m, σ). We respond to all but one randomly selected hash query on m with $M = H(m) = r_1P + r_2Q$ where $r_1, r_2 \in_R [0, n-1]$; note that M is indeed distributed uniformly at random in $E[n]$, so the response to the hash query is valid. For the randomly selected hash query m^*, we respond with $H(m^*) = M^*$. We respond to signing queries on messages $m \neq m^*$ with $\sigma = r_1A + r_2B$. This is a valid signature since

$$\sigma = r_1A + r_2B = r_1(aP) + r_2(aQ) = a(r_1P + r_2Q) = aH(m) = aM.$$

If the forger asks for the signature on m^*, we abort. We also abort if the forger does not output a valid signature on m^*. Otherwise, if the forger outputs a valid signed message (m^*, σ^*), then σ^* is the solution to the DHP-B challenge (P, A, M^*). □

Note that in the Type B setting we can hash onto $E[n]$ but not onto $\mathbb{G}_1$. Hence, protocols that require a map-to-point function such that at most one argument of the pairing function is expressed in terms of such a point can be instantiated in the Type B pairing setting. Apart from the BLS signature scheme, some other prominent examples are the Boneh-Franklin IBE scheme [11] and the Boneh-Gentry-Lynn-Shacham aggregate signature scheme [12]. Note also that the protocols that can be instantiated in the Type A setting can be instantiated in the Type B setting without any difficulty.

The main difference between the Type A and the Type B settings is that $E[n]$ is not a cyclic group of prime order. This necessitates an appropriate modification in the underlying hardness assumption as well as the protocol description and its security reduction. We have already illustrated this in the context of the BLS signature scheme. Next we briefly describe the bilinear Diffie-Hellman (BDH) problem in the Type B setting and its relation with the corresponding problem in the Type A setting.

5.2.2. *BDH problem in Types A and B.*

Definition 15 (BDH-B)**.** *Bilinear Diffie-Hellman problem in* $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$*: Given* P*,* aP*,* $bP \in \mathbb{G}_1$*, where* $a, b \in_R [1, n-1]$*, and* $R \in_R \mathbb{G}_2$*, compute* $e_B(P, R)^{ab}$*.*

It is easy to see that the BDH problem in the Type A setting (BDH-A) polynomial-time reduces to BDH-B. Namely, suppose we are given an instance (P, aP, bP, cP) of BDH-A and access to a BDH-B oracle. We compute $Q = \psi(P)$ and $B = \psi(bP) = bQ$. We further compute $R = cP + rQ$ where $r \in_R [1, n-1]$. We run the BDH-B oracle with input (P, aP, bP, R). The oracle returns $e_B(P, R)^{ab} = e_A(P, P)^{abc} e_B(aP, bQ)^r$ from which one can easily extract the BDH-A solution, namely $e_A(P, P)^{abc}$.

5.2.3. *Sakai-Oghishi-Kasahara id-based key agreement.* We recall the Sakai-Oghishi-Kasahara id-based Non-Interactive Key Agreement (SOK-NIKA) scheme in the Type 1 setting [40].

Let $e_1 : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$ be a Type 1 pairing where $\mathbb{G}_1 = \langle P \rangle$. Let $H : \{0,1\}^* \to \mathbb{G}_1$ be a hash function. The key generation center (KGC) chooses her master secret $s \in_R [1, n-1]$ and the corresponding public parameters are e_1, P, sP and H.

An entity with identity id obtains her corresponding private key from the KGC as $d_{\mathsf{id}} = sQ_{\mathsf{id}}$ where $Q_{\mathsf{id}} = H(\mathsf{id})$. Two parties having identities respectively id and id' can compute a shared secret without any interaction as follows. The party with identity id computes $K = e_1(d_{\mathsf{id}}, H(\mathsf{id}'))$, whereas the party with identity id' computes $K' = e_1(d_{\mathsf{id}'}, H(\mathsf{id}))$. Both parties compute the same shared secret because

$$K = e_1(d_{\mathsf{id}}, H(\mathsf{id}')) = e_1(sQ_{\mathsf{id}}, Q_{\mathsf{id}'}) = e_1(Q_{\mathsf{id}}, sQ_{\mathsf{id}'}) = e_1(H(\mathsf{id}), d_{\mathsf{id}'}) = e_1(d_{\mathsf{id}'}, H(\mathsf{id})) = K'.$$

Note that SOK-NIKA cannot be implemented in the Type B (or Type A) setting because there is no known way to hash onto a fixed order-n group $\mathbb{G}_1$.

6. Type C pairings

6.1. **The pairings.** Let $\mathbb{G}_2 = E[n]$. Then the full Weil and Tate pairings defined in (1) and (3) are bilinear pairings from $\mathbb{G}_2 \times \mathbb{G}_2$ to $\mathbb{G}_T$. We say that these pairings are of Type C and denote them by e_C. Note that one can efficiently hash onto $\mathbb{G}_2$. Hence, protocols that require a map-to-point function for either of the two arguments of the pairing function can be considered for implementation with Type C pairings. For any fixed order-n subgroup $\mathbb{G}_1$, the restriction of e_C to $\mathbb{G}_1 \times \mathbb{G}_1$ gives the Type A pairing e_A. Similarly, the restriction of e_C to $\mathbb{G}_1 \times \mathbb{G}_2$ gives the Type B pairing e_B.

6.2. **Protocols.**

6.2.1. *Sakai-Oghishi-Kasahara id-based key agreement in Type C.* Let $e_C : \mathbb{G}_2 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a Type C pairing, and let $P \in \mathbb{G}_2$. Let $H : \{0,1\}^* \to \mathbb{G}_2$ be a hash function. The key generation center (KGC) chooses her master secret $s \in_R [1, n-1]$ and the corresponding public parameters are e_C, P, sP and H.

An entity with identity id obtains her corresponding private key from the KGC as $d_{\mathsf{id}} = sQ_{\mathsf{id}}$ where $Q_{\mathsf{id}} = H(\mathsf{id})$. Two parties having identities respectively id and id' can compute a

shared secret without any interaction as follows. Suppose that id is lexicographically smaller than id'. The party with identity id computes $K = e_C(d_{\mathsf{id}}, H(\mathsf{id}'))$, whereas the party with identity id' computes $K' = e_C(H(\mathsf{id}), d_{\mathsf{id}'})$. Both parties compute the same shared secret because

$$K = e_C(d_{\mathsf{id}}, H(\mathsf{id}')) = e_C(sQ_{\mathsf{id}}, Q_{\mathsf{id}'}) = e_C(Q_{\mathsf{id}}, sQ_{\mathsf{id}'}) = e_C(H(\mathsf{id}), d_{\mathsf{id}'}) = K'.$$

A Key Derivation Function (KDF) $\tilde{H} : \mathbb{G}_T \to \{0,1\}^k$ can be applied to the shared secret to produce a k-bit secret key.

In the Type 1 pairing setting it has been shown [20, 38] that when H and $\tilde{H}$ are modeled as random oracles, the security of SOK-NIKA is based on the hardness of BDH in the Type 1 setting. The original security argument can be easily modified for the above variant whence the security depends on the following version of the BDH problem in the Type C setting.

Definition 16 (BDH-C)**.** *Bilinear Diffie-Hellman problem in* $(\mathbb{G}_2, \mathbb{G}_T)$*: Given* $P \in \mathbb{G}_2$*,* $Q = aP$ *where* $a \in_R [1, n-1]$*, and* $R, S \in_R \mathbb{G}_2$*, compute* $e_C(R,S)^a$*.*

If $\tilde{H}$ is not included in the protocol description, then one can argue the security of SOK-NIKA based on the following version of the decisional Bilinear Diffie-Hellman problem in the Type C setting.

Definition 17 (DBDH-C)**.** *Decisional Bilinear Diffie-Hellman problem in* $(\mathbb{G}_2, \mathbb{G}_T)$*: Given* $P \in \mathbb{G}_2$*,* $Q = aP$ *where* $a \in_R [1, n-1]$*,* $R, S \in_R \mathbb{G}_2$*, and* $Z \in \mathbb{G}_T$*, decide whether* $Z = e_C(R,S)^a$ *or* Z *is a random element in* $\mathbb{G}_T$*.*

As in the case of variants of the BDH problem, it is easy to see that DBDH-A polynomial-time reduces to DBDH-C. Namely, suppose we are given an instance (P, aP, bP, cP, Z) of DBDH-A and access to a DBDH-C oracle. We compute $Q = \psi(P)$, $A = \psi(aP) = aQ$, $B = \psi(bP) = bQ$, and $C = \psi(cP) = cQ$. We further compute $R = bP + rQ$ and $S = cP + sQ$ where $r, s \in_R [1, n-1]$ and $Z' = Z \cdot e_C(bP, aQ)^s \cdot e_C(aQ, cP)^r \cdot e_C(aQ, Q)^{rs}$. We run the DBDH-C oracle with input (P, aP, R, S, Z') and return whatever bit the oracle returns. Clearly, when $Z = e_A(P,P)^{abc}$ then $Z' = e_C(R,S)^a$. On the other hand, if Z is a random element of $\mathbb{G}_T$ then so is Z'.

In the proof sketch below for the security of SOK-NIKA, we follow the IND-SK security model proposed in [38]. In this model the adversary is allowed to make H-oracle queries, key extraction queries on identities of its choice, shared key reveal queries on any pair of identities of its choice, and a challenge (Test) query on a pair of identities subject to the natural restriction that no key extraction query is made on the challenge identities, nor is there any shared-key reveal query on the pair of challenge identities. In response to the challenge query the adversary is given with equal probability either the shared key of the two challenge identities or a random element from the key space. IND-SK security is achieved if the adversary cannot distinguish between the two with probability significantly greater than $\frac{1}{2}$. The following result omits the tightness gap in the reduction.

Theorem 18. *Suppose that the DBDH-C problem is intractable and* H *is a random oracle. Then SOK-NIKA is a secure key-agreement scheme in the IND-SK model.*

Proof. Suppose we are given a DBDH-C instance $(P, A = aP, R, S, Z)$ and need to decide whether $Z = e_C(R,S)^a$ or Z is random. We compute $Q = \psi(P)$ and $B = \psi(A) = aQ$, where ψ is the distortion map (6). We then run the IND-SK adversary against SOK-NIKA with input (P, A). The adversary makes queries to a random oracle H as well as key extraction and shared-key reveal queries. We assume that the adversary always queries H with id before it queries for key-extraction or shared-key involving that identity. We respond to all but two randomly selected hash queries on id with $Q_{\mathsf{id}} = H(\mathsf{id}) = r_1P + r_2Q$ where $r_1, r_2 \in_R [0, n-1]$; note that Q_{id} is indeed distributed uniformly at random in $\mathbb{G}_2$, so the response to the hash query is valid. For the randomly selected hash query id_1^* and id_2^*, we respond with $H(\mathsf{id}_1^*) = R$ and $H(\mathsf{id}_2^*) = S$. We respond to key-extraction queries on identity $\mathsf{id} \notin \{\mathsf{id}_1^*, \mathsf{id}_2^*\}$ with $d_{\mathsf{id}} = r_1A + r_2B$. This is a valid private key for id since

$$d_{\mathsf{id}} = r_1A + r_2B = r_1(aP) + r_2(aQ) = a(r_1P + r_2Q) = aH(\mathsf{id}) = aQ_{\mathsf{id}}.$$

We abort if $\mathsf{id} \in \{\mathsf{id}_1^*, \mathsf{id}_2^*\}$.

Next, suppose that the adversary asks for the shared key for $(\mathsf{id}_i, \mathsf{id}_j)$. We abort if both of them are from $\{\mathsf{id}_1^*, \mathsf{id}_2^*\}$. If not, let's assume without loss of generality,that $\mathsf{id}_i \notin \{\mathsf{id}_1^*, \mathsf{id}_2^*\}$. We return $e_C(d_{\mathsf{id}_i}, H(\mathsf{id}_j))$ as the shared key. When the adversary makes the challenge query we abort if the challenge identities are not $\{\mathsf{id}_1^*, \mathsf{id}_2^*\}$. Else, we return Z as response. It is easy to see that the above strategy provides a perfect simulation of IND-SK security if we do not abort during the simulation.

Finally, we return the adversary's response ("real" or "random") to the challenge query as our solution to the DBDH-C instance. □

Some other SOK-type protocols that can be implemented with Type C pairings (but not Type A or Type B) are Scott's key agreement protocol [43] and the hierarchical id-based key-agreement of Gennaro et al. [23].

7. Efficiency

This section provides cost estimates for arithmetic operations on the $k = 1$ elliptic curve E defined in (5). The cost estimates are in terms of the number of $\mathbb{F}_p$ multiplications. We do not count $\mathbb{F}_p$ additions and subtractions, so the estimates are somewhat optimistic. Nonetheless, they serve to demonstrate the practicality of implementing pairing-based protocols with $k = 1$ elliptic curves at the 128-bit security level.

We focus on the operations that are needed to implement the BLS-B signature scheme at the 128-bit security level. To achieve this security level against NFS attacks on the discrete logarithm problem, one needs to select a 3072-bit prime p. One method for selecting such a p is to randomly select a 1535-bit prime n with $n \equiv 3 \pmod 4$ until $A^2 + 1$, where $A = 2n$, is a 3072-bit prime; here the cofactor is $h = 2$. We are assuming here that the special form of p, namely $p = A^2 + 1$ where A is twice a randomly selected 1535-bit prime, cannot be exploited to speed up the NFS.

In order to reduce the number of iterations in Miller's algorithm, an alternative is to select a 256-bit prime n with $n \equiv 3 \pmod 4$ and a randomly selected 1280-bit cofactor $h = A/n$. A

smaller n also speeds up subgroup membership testing and point multiplication since scalars are now only 256 bits in length. The resulting speedups are partially offset by an increase in the time to perform hashing and the final exponentiation in the Tate pairing computation. Note that Pollard's rho attack [39] on the discrete logarithm problem in an order-n subgroup of $E(\mathbb{F}_p)$ takes roughly $n^{1/2}$ steps, so selecting a 256-bit n preserves the 128-bit security level. In order to decrease the number of addition steps in Miller's algorithm (steps 9–12 in Algorithm 1), one can select n so that its binary representation has low Hamming weight. Then, under the heuristic assumption that A is a uniformly distributed 1536-bit number, it is reasonable to assume that the special form of n cannot be exploited to speed up attacks on the discrete logarithm problem.

In §7.1 we give the cost estimates for arithmetic operations; see Table 1. We consider two cases, one where n is a randomly-selected 1535-bit prime with Hamming weight approximately 768, and the other where n is a 256-bit prime of Hamming weight 3 [3] with 1280-bit cofactor h of Hamming weight approximately 640. In both cases, $n \equiv 3 \pmod 4$ and p is assumed to have Hamming weight approximately 1536. We denote the cost of multiplication, squaring and inversion in $\mathbb{F}_p$ by M, S, and I, respectively. For the sake of simplicity, we will assume that $S \approx M$. In §7.2 we give clock cycle counts for BLS-B signature generation and signature verification. These counts indicate that the BLS-B signature scheme at the 128-bit security level is reasonably efficient on desktop computers.

Operation	1535-bit n	256-bit n
Tate pairing	$42957M$	$8726M$
Weil pairing	$81306M$	$9252M$
Point multiplication	$16277M$	$2863M$
Hashing	$4617M$	$18238M$
Subgroup membership testing	$59234M$	$11589M$

TABLE 1. Cost estimates for operations on the $k = 1$ elliptic curve (5) with 1535-bit n and 256-bit n. The cost of a multiplication in $\mathbb{F}_p$ is denoted by M.

7.1. **Cost estimates.**

7.1.1. *Elliptic curve representation.* To avoid expensive inversions in $\mathbb{F}_p$, we represent points in $E(\mathbb{F}_p)$ using modified Jacobian coordinates, where a point $(X, Y, Z, W = Z^2)$ corresponds to the point (x, y) in affine coordinates with $x = X/Z^2$ and $y = Y/Z^3$ [18]. With modified Jacobian coordinates, the cost of a point addition is $A = 7M + 4S \approx 11M$ and the cost of a point doubling is $D = 1M + 8S \approx 9M$.

[3]One can also use primes whose signed binary representations have low Hamming weight. Among such primes are $2^{255} + 2^{96} - 1$, $2^{255} + 2^{176} - 1$, $2^{255} + 2^{232} - 1$, $2^{256} - 2^{76} - 1$ and $2^{256} - 2^{194} - 1$.

7.1.2. *Pairing computation.* Section 4.2 of [30] gives formulas for the doubling step of Miller's algorithm (steps 4–7 of Algorithm 1) and the addition step (steps 9–12 of Algorithm 1). The costs are $8M + 10S$ and $12M + 5S$, respectively. Now, the cost of Miller's algorithm with n of bitlength ℓ and Hamming weight v is $(\ell-1)\cdot(8M+10S)+(v-1)\cdot(12M+5S)+2M+I$. Ignoring the cost of I, we get a total cost of $40653M$ in the case of 1535-bit n, and $4626M$ in the case of 256-bit n (with Hamming weight $v = 3$).

In the case of the Tate pairing, we must perform a final exponentiation by the power $(p-1)/n$. Suppose we use the repeated-square-and-multiply method for the exponentiation. Then the final exponentiation cost is $2304M$ in the case of 1535-bit n, and $4100M$ in the case of 256-bit n.

Thus, the total cost of a Tate pairing is $42957M$ in the case of 1535-bit n, and $8726M$ in the case of 256-bit n. The cost of the Weil pairing in the two cases is $81306M$ and $9252M$.

7.1.3. *Point multiplication.* Let $P \in E(\mathbb{F}_p)$ and let r be an ℓ-bit integer. Then point multiplication rP can be performed using the w-NAF method at a cost of $(2^{w-2}-1)A + (\ell/(w+1))A + (\ell+1)D$ (see [28, Algorithm 3.36]). Thus the cost of the point multiplication when $P \in E[n]$ and $r \in_R [1, n-1]$ is $223A + 1536D = 16277M$ in the case of 1535-bit n and $w = 7$, and $50A + 257D = 2863M$ in the case of 256-bit n and $w = 5$.

7.1.4. *Hashing.* Hashing onto $\mathbb{G}_2$ can be performed by first using a standard hash function to map an arbitrary string to the x-coordinate of a point in $E(\mathbb{F}_p)$, then solving a quadratic equation over $\mathbb{F}_p$ to find the corresponding y-coordinate, and finally multiplying the resulting point by the cofactor h to obtain an n-torsion point. Since $p \equiv 1 \pmod 4$, one can select parameters so that $p \equiv 5 \pmod 8$. Then the square root of a quadratic residue $a \in \mathbb{F}_p$ can be computed with one exponentiation by $(p-5)/8$ and a small number of multiplications; see Algorithm 3 in [2]. The exponentiation can be performed using the repeated-square-and-multiply algorithm at a cost of approximately $4608M$. The cost of the w-NAF point multiplication method with an ℓ-bit scalar is $(2^{w-2}-1)A + (\ell/(w+1))A + (\ell+1)D$ (see [28, Algorithm 3.36]). Thus the cost of the point multiplication by h is $1D = 9M$ in the case $h = 2$, and approximately $191A + 1281D = 13630M$ in the case of 1280-bit h and $w = 7$. This gives a hashing cost of $4617M$ for $h = 2$, and $18238M$ for 1280-bit h.

7.1.5. *Subgroup membership testing.* Let $P \in E[n]$. By Corollary 6 a point Q is in $\langle P \rangle$ if and only if (i) Q satisfies the defining equation of E; (ii) $nQ = \infty$; and (iii) $t_n(P, \psi(Q)) = 1$. Thus the cost of subgroup membership testing is approximately the sum of the costs of a point multiplication and a Tate pairing computation.

7.2. **BLS-B signature scheme.** We employ an elliptic curve with 256-bit n, and the Tate pairing. The dominant operations in BLS-B signature generation (see §5.2.1) are a hashing and a point multiplication, for a total cost of $21101M$. The dominant operations in BLS-B signature verification are a hashing, a subgroup membership testing, and two pairings, for a total cost of $47279M$.

Now, the 3072-bit operands $a, b \in \mathbb{F}_p$ each require forty-eight 64-bit words. Beginning with the Haswell microarchitecture, Intel introduced the MULX instruction that performs a 64-bit word product leaving the arithmetic flags untouched. This feature permits the smooth combination of the MULX and add-with-carry instructions. In this way, the Karatsuba multiplier approach as described in [52, 45] can be efficiently implemented on a Haswell processor by computing the 6144-bit integer product $t = a \cdot b$ in roughly 5,200 clock cycles. The field product $c = t \bmod p$ can then be performed using Barrett reduction [10] at an extra cost of roughly 1.5 integer multiplications. Hence, the total cost of computing the field multiplication is approximately 13,000 clock cycles.

The 13,000 clock cycle estimate for a field multiplication yields the estimates of 113.4 million clock cycles for computing the Tate pairing, 274.3 million clock cycles for BLS-B signature generation, and 614.6 million clock cycles for BLS-B signature verification. These numbers suggest that $k = 1$ pairings have acceptable performance for applications that run on desktop computers and high-end mobile devices where clock rates of 2.0 GHz are common.

Remark 19. (*speed comparisons with pairings of even embedding degree*) We did not attempt to optimize our cost estimates, e.g., by exploiting parallelism or combining operations when computing a product of pairings [44]. Nonetheless, the $k = 1$ pairings are expected to be significantly slower than pairings derived from elliptic curves of even embedding degree because of the inapplicability of many optimizations available in the latter such as reducing the number of iterations in Miller's algorithm [49], denominator elimination [8], and fast exponentiation [46]. For example, [3] report a speed of 18.7 million clock cycles for computing a BLS12 pairing at the 192-bit security level on an Intel Core i5 Nehalem machine. Still, $k = 1$ pairings are not expected to be drastically slower than pairings derived from supersingular elliptic curves with embedding degree 2 as standardized in [15, 27].

8. Concluding remarks

We have defined three types of pairings with elliptic curves of embedding degree one. These three pairings do not fit within the general classification of pairings derived from elliptic curves of even embedding degree. Our study of pairing-based protocols indicates that any protocol that can be implemented with a Type A pairing can also be implemented with Types B and C; however the converse is not true. Similarly, any protocol that can be implemented with a Type B pairing can also be implemented with Type C, but the converse is not true.

Finally, we note that there are some pairing-based protocols that cannot be implemented in either Types A, B or C. Protocols have been designed based on the Symmetric eXternal Diffie-Hellman assumption (SXDH) in the Type 3 setting (which says that the decisional Diffie-Hellman (DDH) problem is hard in the pairing groups $\mathbb{G}_1$ and $\mathbb{G}_2$). Examples of such protocols can be found in [17, 33]. Since DDH is *easy* in the pairing groups defined over elliptic curves with embedding degree one, these protocols cannot be securely instantiated in either Types A, B or C. Neither will it be possible to implement protocols that require

reversible embedding of messages in $\mathbb{G}_T$; among such protocols are Waters' IBE scheme in its original form [51] and Gentry's IBE scheme [24] (see also [34]).

Acknowledgements

We thank Razvan Barbulescu for answering our questions on the Number Field Sieve, and Neal Koblitz for several helpful suggestions.

References

[1] G. Adj, A. Menezes, T. Oliveira and F. Rodríguez-Henríquez, "Computing discrete logarithms in $\mathbb{F}_{3^{6\cdot 137}}$ and $\mathbb{F}_{3^{6\cdot 163}}$ using Magma", *International Workshop on Arithmetic of Finite Fields — WAIFI 2014*, LNCS 9061 (2015), 3–22.

[2] G. Adj and F. Rodríguez-Henríquez, "Square root computation over even extension fields", *IEEE Transactions on Computers*, 63 (2014), 2829–2841.

[3] D. Aranha, L. Fuentes-Castañeda, E. Knapp, A. Menezes and F. Rodríguez-Henríquez, "Implementing pairings at the 192-bit security level", *Pairing-Based Cryptography — Pairing 2012*, LNCS 7708 (2013), 177–195.

[4] R. Balasubramanian and N. Koblitz, "The improbability that an elliptic curve has subexponential discrete log problem under the Menezes-Okamoto-Vanstone algorithm", *Journal of Cryptology*, 11 (1998), 141–145.

[5] R. Barbulescu, P. Gaudry, A. Guillevic and F. Morain, "Improving NFS for the discrete logarithm problem in non-prime finite fields", *Advances in Cryptology — EUROCRYPT 2015*, LNCS 9056 (2015), 129–155.

[6] R. Barbulescu, P. Gaudry, A. Joux and E. Thomé, "A quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic: Improvements over FFS in small to medium characteristic", *Advances in Cryptology — EUROCRYPT 2014*, LNCS 8441 (2014), 1–16.

[7] P. Barreto, S. Galbraith, C. Ó'hÉigeartaigh and M. Scott, "Efficient pairing computation on supersingular abelian varieties", *Designs, Codes and Cryptography*, 42 (2007), 239–271.

[8] P. Barreto, B. Lynn and M. Scott, "On the selection of pairing-friendly group", *Selected Areas in Cryptography — SAC 2003*, LNCS 3006 (2004), 17–25.

[9] P. Barreto and M. Naehrig, "Pairing-friendly elliptic curves of prime order", *Selected Areas in Cryptography — SAC 2005*, LNCS 3897 (2006), 319–331.

[10] P. Barrett, "Implementing the Rivest Shamir and Adleman public key encryption algorithm on a standard digital signal processor", *Advances in Cryptology — CRYPTO '86*, LNCS 263 (1987), 311–323.

[11] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing", *SIAM Journal on Computing*, 32 (2003), 586–615.

[12] D. Boneh, C. Gentry, B. Lynn and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps", *Advances in Cryptology — EUROCRYPT 2003*, LNCS, 2656 (2003), 416–432.

[13] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the Weil pairing", *Journal of Cryptology*, 17 (2004), 297–319.

[14] D. Boneh, K. Rubin and A. Silverberg, "Finding composite order ordinary elliptic curves using the Cocks-Pinch method", *Journal of Number Theory*, 131 (2011), 832-841.

[15] X. Boyen and L. Martin, "Identity-based cryptography standard (IBCS) #1: Supersingular curve implementations of the BF and BB1 cryptosystems", IETF RFC 5091, December 2007.

[16] S. Chatterjee, D. Hankerson and A. Menezes, "On the efficiency and security of pairing-based protocols in the Type 1 and Type 4 settings", *International Workshop on Arithmetic of Finite Fields — WAIFI 2010*, LNCS 6087 (2010), 114–134.

[17] J. Chen and H. Wee, "Fully, (almost) tightly secure IBE and dual system groups, *Advances in Cryptology — CRYPTO 2013*, LNCS 8043 (2013), 435–460.
[18] D. Chudnovsky and G. Chudnovsky, "Sequences of numbers generated by addition in formal groups and new primality and factoring tests", *Advances in Applied Mathematics*, 7 (1987), 385–434.
[19] D. Coppersmith, "Fast evaluation of logarithms in fields of characteristic two", *IEEE Transactions on Information Theory*, 30 (1984), 587–594.
[20] R. Dupont and A Enge, "Provably secure non-interactive key distribution based on pairings", *Discrete Applied Mathematics*, 154 (2006), 270–276.
[21] S. Galbraith, "Pairings", Chapter IX of I. Blake, G. Seroussi, and N. P. Smart, eds., *Advances in Elliptic Curve Cryptography*, Vol. 2, Cambridge University Press, 2005.
[22] S. Galbraith, K. Paterson and N. Smart, "Pairings for cryptographers", *Discrete Applied Mathematics*, 156 (2008), 3113-3121.
[23] R. Gennaro, S. Halevi, H. Krawczyk, T. Rabin, S. Reidt and S. Wolthusen, "Strongly-resilient and non-interactive hierarchical key-agreement in MANETs", *13th European Symposium on Research in Computer Security — ESORICS 2008*, LNCS 5283 (2008), 49–65.
[24] C. Gentry, "Practical identity-based encryption without random oracles", *Advances in Cryptology — EUROCRYPT 2006*, LNCS 4404 (2006), 445–464.
[25] D. Gordon, "Discrete logarithms in $GF(p)$ using the number field sieve", *SIAM Journal on Discrete Mathematics*, 6 (1993), 124–138.
[26] R. Granger, T. Kleinjung and J. Zumbrägel, "Breaking '128-bit secure' supersingular binary curves (or how to solve discrete logarithms in $\mathbb{F}_{2^{4\cdot 1223}}$ and $\mathbb{F}_{2^{12\cdot 367}}$)", *Advances in Cryptology — CRYPTO 2014*, LNCS 8617 (2014), 126–145.
[27] M. Groves, "Sakai-Kasahara key encryption (SAKKE)", IETF RFC 6508, February 2012.
[28] D. Hankerson, A. Menezes and S. Vanstone, *Guide to Elliptic Curve Cryptography*, Springer, 2004.
[29] F. Hess, N. Smart and F. Vercauteren, "The eta pairing revisited" *IEEE Transactions on Information Theory*, 52 (2006), 4595–4602.
[30] Z. Hu, L. Wang, M. Xu and G. Zhang, "Generation and Tate pairing computation of ordinary elliptic curves with embedding degree one", *ICICS 2013*, LNCS 8233 (2013), 393-403.
[31] A. Joux, "A one round protocol for tripartite Diffie-Hellman", *Journal of Cryptology*, 17 (2004), 263–276.
[32] A. Joux and K. Nguyen, "Separating Decision Diffie-Hellman from Computational Diffie-Hellman in cryptographic groups", *Journal of Cryptology*, 16 (2003), 239–247.
[33] C. Jutla and A. Roy, "Shorter quasi-adaptive NIZK proofs for linear subspaces", *Advances in Cryptology — ASIACRYPT 2013*, LNCS 8269 (2013), 1–20.
[34] S. Chatterjee, K. Karabina and A. Menezes, "Fault attacks on pairing-based protocols revisited", *IEEE Transactions on Computers*, 64 (2015), 1707–1714.
[35] T. Kim and R. Barbulescu, "Extended tower number field sieve: A new complexity for medium prime case", available at http://eprint.iacr.org/2015/1027.
[36] N. Koblitz and A. Menezes, "Pairing-based cryptography at high security levels", *Cryptography and Coding: 10th IMA International Conference*, LNCS 3796 (2005), 13–36.
[37] V. Miller, "The Weil pairing, and its efficient calculation', *Journal of Cryptology*, 17 (2004), 235–261.
[38] K. Paterson and S. Srinivasan, "On the relations between non-interactive key distribution, identity-based encryption and trapdoor discrete log groups", *Designs, Codes and Cryptography*, 52 (2009), 219–241.
[39] J. Pollard, "Monte Carlo methods for index computation mod p", *Mathematics of Computation*, 32 (1978), 918–924.
[40] R. Sakai, K. Oghishi and M. Kasahara, "Cryptosystems based on pairing over elliptic curve" (in Japanese), *The 2000 Symposium on Cryptography and Information Security*, 2000.
[41] O. Schirokauer, "Discrete logarithms and local units", *Philosophical Transactions of the Royal Society London A*, 345 (1993), 409–423.

[42] O. Schirokauer, "The number field sieve for integers of low weight", *Mathematics of Computation*, 79 (2010), 583-602.
[43] M. Scott, "Authenticated id-based key exchange and remote log-in with simple token and PIN number", available at http://eprint.iacr.org/2002/164.
[44] M. Scott, "Computing the Tate pairing", *Topics in Cryptology — CT-RSA 2005*, LNCS 3376 (2005) 300–312.
[45] M. Scott, "Missing a trick: Karatsuba revisited", available at http://eprint.iacr.org/2015/1247.
[46] M. Scott, N. Benger, M. Charlemagne, L. Dominguez Perez and E. Kachisa, "On the final exponentiation for calculating pairings on ordinary elliptic curves", *Pairing-Based Cryptography — Pairing 2009*, LNCS 5671 (2009), 78–88.
[47] H. Shacham, "New paradigms in signature schemes", Ph.D. thesis, Stanford University, 2005.
[48] J. Silverman, *The Arithmetic of Elliptic Curves*, Springer, 1986.
[49] F. Vercauteren, "Optimal pairings", *IEEE Transactions on Information Theory*, 56 (2010), 455–461.
[50] E. Verheul, "Evidence that XTR is more secure than supersingular elliptic curve cryptosystems", *Journal of Cryptology*, 17 (2004), 277–296.
[51] B. Waters, "Efficient identity-based encryption without random oracles", *Advances in Cryptology — EUROCRYPT 2005*, LNCS 3494 (2005), 114–127.
[52] A. Weimerskirch and C. Paar, "Generalizations of the Karatsuba algorithm for efficient implementation", available at http://eprint.iacr.org/2006/224.
[53] C. Zhao, F. Zhang and D. Xie, "Faster computation of self-pairings", *IEEE Transactions on Information Theory*, 58 (2012), 3266–3272.

Department of Computer Science and Automation, Indian Institute of Science
E-mail address: sanjit@csa.iisc.ernet.in

Department of Combinatorics & Optimization, University of Waterloo
E-mail address: ajmeneze@uwaterloo.ca

Computer Science Department, CINVESTAV-IPN
E-mail address: francisco@cs.cinvestav.mx