Subgroup security in pairing-based cryptography

Paulo S. L. M. Barreto[1] * **, Craig Costello[2], Rafael Misoczki[1]*,
Michael Naehrig[2], Geovandro C. C. F. Pereira[1]*, and Gustavo Zanon[1]***

[1] Escola Politécnica, University of São Paulo.
{pbarreto,rmisoczki,geovandro,gzanon}@larc.usp.br
[2] Microsoft Research, USA
{craigco,mnaehrig}@microsoft.com

Abstract. Pairings are typically implemented using ordinary pairing-friendly elliptic curves. The two input groups of the pairing function are groups of elliptic curve points, while the target group lies in the multiplicative group of a large finite field. At moderate levels of security, at least two of the three pairing groups are necessarily proper subgroups of a much larger composite-order group, which makes pairing implementations potentially susceptible to small-subgroup attacks.
To minimize the chances of such attacks, or the effort required to thwart them, we put forward a property for ordinary pairing-friendly curves called *subgroup security*. We point out that existing curves in the literature and in publicly available pairing libraries fail to achieve this notion, and propose a list of replacement curves that do offer subgroup security. These curves were chosen to drop into existing libraries with minimal code change, and to sustain state-of-the-art performance numbers. In fact, there are scenarios in which the replacement curves could facilitate faster implementations of protocols because they can remove the need for expensive group exponentiations that test subgroup membership.
Keywords: Pairing-based cryptography, elliptic-curve cryptography, pairing-friendly curves, subgroup membership, small-subgroup attacks.

1 Introduction

In this paper we propose new instances of pairing-friendly elliptic curves that aim to provide stronger resistance against small-subgroup attacks [41]. A small-subgroup attack can be mounted on a discrete-logarithm-based cryptographic scheme that uses a prime-order group which is contained in a larger group of order divisible by small prime factors. By forcing a protocol participant to carry out an exponentiation of a non-prime-order group element with a secret exponent, an attacker could obtain information about that secret exponent. This is possible if the protocol implementation does not check that the group element being exponentiated belongs to the correct subgroup and thus has large prime order. In the worst case, the user's secret key could be fully revealed although

* Supported by Intel Research grant "Energy-efficient Security for SoC Devices" 2012.
** Supported by CNPq research productivity grant 306935/2012-0.
*** Supported by the São Paulo Research Foundation (FAPESP) grant 2014/09200-5.

the discrete logarithm problem (DLP) in the large prime-order subgroup is computationally infeasible. We start by illustrating the possibility of such attacks in the context of (pairing-based) *digital signature schemes*, many of which are based on the celebrated short signature scheme of Boneh, Lynn and Shacham (BLS) [13][3].

BLS signatures. For both historical reasons and for ease of exposition, authors of pairing-based protocol papers commonly assume the existence of an efficient, *symmetric* bilinear map $e\colon \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, where $\mathbb{G}$ and $\mathbb{G}_T$ are cryptographic groups of large prime order n. Let P be a public generator of $\mathbb{G}$ and let $\mathcal{H}\colon \{0,1\}^* \to \mathbb{G}$ be a suitably defined hash function. Boneh, Lynn and Shacham proposed a simple signature scheme [13] that works as follows. To sign a message $M \in \{0,1\}^*$ with her secret key $a \in \mathbb{Z}_n$, Alice computes $Q = \mathcal{H}(M) \in \mathbb{G}$ and sends the signature $\sigma = [a]Q$ to Bob. To verify this signature, Bob also computes $Q = \mathcal{H}(M)$ and then uses Alice's public key $[a]P$ to assert that $e([a]P, Q) = e(P, \sigma)$. It is shown in [13] that this scheme is a secure signature scheme if the *Gap Diffie-Hellman* (GDH) problem is hard.

Forged system parameters. There are various threat models in which one might wish to think about the possible implications of small-subgroup attacks. In one of them, one assumes that it is possible for an attacker to even forge the public parameters used in the signature system. Such a possibility has for example been discussed for the Digital Signature Standard by Vaudenay in [57]. For BLS signatures, an attacker could forge the system parameters to use a base point P of non-prime order. Thus by means of a small-subgroup attack, Alice reveals information about her private key a to the attacker by simply publishing her public key $[a]P$.

In another example of public parameter manipulation, one might assume that the hash function $\mathcal{H}$ maps into the full composite order group, instead of into the prime order subgroup. Therefore, the hash of a message could be a group element of composite order and the BLS signature could leak information about Alice's private key. Such a faulty hash function might actually be the result of an implementation bug, for example the omission of cofactor exponentiations to move group elements to the right subgroup.

Valid system parameters. Even if the system parameters are valid, there are scenarios in which small subgroup attacks might lead to a security breach. In this setting, we assume that, since P is a fixed public parameter (that is presumably asserted to be in $\mathbb{G}$) and Alice hashes elements into $\mathbb{G}$ herself, Alice's public key and signature are *guaranteed* to lie in $\mathbb{G}$, and are therefore protected by the hardness of the discrete logarithm problem (DLP) in $\mathbb{G}$. There is therefore

[3] We warn the reader that BLS is commonly used to abbreviate two different authorships in the context of pairing-based cryptography: BLS signatures [13] and BLS curves [4].

no threat to Alice's secret key in context of BLS signatures, but this is not necessarily the case in the context of (pairing-based) *blind signatures*, as we discuss below.

Blind signatures. Roughly speaking, blind signatures allow Alice to sign a message that is authored by a third party, Carol. The typical scenarios require that Carol and Alice interact with one another in such a way that Carol learns nothing about Alice's secret signing key and Alice learns nothing about Carol's message. In [12, §5], Boldyreva describes a simple blind signature scheme that follows naturally from BLS signatures. In order to "blindly" sign her message M, Carol computes $Q = \mathcal{H}(M)$ and sends Alice the blinded message $\tilde{Q} = Q + [r]P$, for some random $r \in \mathbb{Z}_n$ of Carol's choosing. Alice uses her secret key $a \in \mathbb{Z}_n$ to return the signed value $[a]\tilde{Q}$ to Carol, who then uses her random value r and Alice's public key $[a]P$ to compute $\sigma = [a]\tilde{Q} - [r]([a]P) = [a]Q$. Carol then sends σ to Bob who can assert that it is a valid BLS signature under Alice's key.

Unlike for the original BLS signatures, where Alice hashed the message into $\mathbb{G}$ herself before signing it, in the above scheme Alice signs the point that Carol sends her. If Carol maliciously sends Alice a point that belongs to a group in which the DLP is easy (e.g. via a small subgroup attack [41]), and if this goes undetected by Alice, then Carol can recover Alice's secret key.

Of course, in a well-designed version of the above protocol, Alice validates that the point she receives is in the correct group before using her secret key to create the signature. However, for the instantiations of bilinear pairings that are preferred in practice, this validation requires a full elliptic curve scalar multiplication. In addition, as is discussed in Remark 1 below, authors of pairing-based protocols often assume that certain group elements belong to the groups that they are supposed to. If these descriptions were translated into real-world implementations *unchanged*, then such instantiations could be susceptible to small subgroup attacks like the example above.

Asymmetric pairings. The original papers that founded pairing-based cryptography [51, 14, 36] assumed the existence of a bilinear map of the form $e\colon \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Such *symmetric* pairings (named so because the two input groups are the same) only exist on supersingular curves, which places a heavy restriction on either or both of the underlying efficiency and security of the protocol; see [30] or [25] for further discussion. It was not long until practical instantiations of asymmetric pairings of the form $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ (with $\mathbb{G}_2 \neq \mathbb{G}_1$) were discovered [27, 5], and were shown to be much more efficient than their symmetric counterparts, especially at high security levels. In recent times this performance gap has been stretched orders of magnitude further, both given the many advances in the asymmetric setting [6, 34, 56, 58, 2], and given that the fastest known instantiations of symmetric pairings are now considered broken [3]. Thus, all modern pairing libraries are built on ordinary elliptic curves in the asymmetric setting.

We note that transferring the above blind signature protocol to the asymmetric setting does not remove the susceptibility of the scheme to small subgroup attacks. Following the asymmetric version of BLS signatures [42, 17], in Boldyreva's simple blind signature scheme Alice's public key could be $([a]P_1, [a]P_2)$ for fixed generators $(P_1, P_2) \in \mathbb{G}_1 \times \mathbb{G}_2$. If $\mathcal{H}\colon \{0,1\}^* \to \mathbb{G}_1$, then Carol can blind the message $Q = \mathcal{H}(M)$ by sending Alice $\tilde{Q} = Q + [r]P_1$. Upon receiving $[a]\tilde{Q}$ back from Alice, Carol removes the blinding factor as before by taking $[a]\tilde{Q} - [r]([a]P_1) = [a]Q$, and sends this result to Bob. Bob then uses Alice's public key to verify that $e([a]Q, P_2) = e(Q, [a]P_2)$. Here, if Alice does not pay the price of a scalar multiplication to assert that $\tilde{Q}$ is in fact in $\mathbb{G}_1$, then Carol could use small subgroup attacks to obtain Alice's private signing key.

In any case, subgroup attacks are inherent to (pairing-based) blind signatures, where signatures are performed *blindly* on points sent by third parties.

The case of the group $\mathbb{G}_2$. In the asymmetric pairing setting, protocols are often designed to perform the bulk of elliptic curve operations in the group $\mathbb{G}_1$, because $\mathbb{G}_1$ can be instantiated as the group of rational points on a pairing-friendly elliptic curve over the base field. Here group operations are more efficient than those in $\mathbb{G}_2$ and group elements have more compact representations. In some cases, the group of rational points even has prime order (i.e. it is equal to $\mathbb{G}_1$) and is thus resistant against subgroup attacks (assuming that valid system parameters are used), while the group $\mathbb{G}_2$ almost always lies in a much larger group with potentially many small prime factors dividing the cofactor. The above translation of the blind signature scheme would thus not be susceptible to the attack because the signed point is in $\mathbb{G}_1$. Only via forged parameters would Alice's public key $[a]P_2 \in \mathbb{G}_2$ leak information about her private key.

However, there are protocols that use the group $\mathbb{G}_2$ for the signing operation for efficiency reasons. This is indicated in the context of BLS signatures as credentials in the Boneh-Franklin identity-based encryption [14] in [48, Section 2]. An example for which a hash function $\mathcal{H} : \{0,1\}^* \to \mathbb{G}_2$ that is faulty and maps to a larger set of points containing non-prime order group elements, can lead to a subgroup attack is the oblivious signature-based envelope (OSBE) scheme by Li, Du, and Boneh [40]. The original OSBE scheme is described in the symmetric setting, but an asymmetric instantiation would be set up with signatures in $\mathbb{G}_2$. The scheme uses a trusted authority (TA) which hands out Boneh-Franklin credentials, which are BLS signatures on identities. Given an identity M and the master key $x \in \mathbb{Z}_n$, the TA computes and sends $[x]\mathcal{H}(M)$ to the receiver allowed to decrypt messages, which leaks information about x if $\mathcal{H}(M)$ does not have order n.

Subgroup security. The main contribution of this paper is the definition of a new property for pairing-friendly curves, which we call *subgroup security.* A pairing-friendly curve is called *subgroup-secure* if the cofactors of all pairing groups, whenever they are of the same size as the prime group order n or larger, only contain prime factors larger than n. This is a realistic scenario because for curves targeting modern security levels, at least two of the pairing groups have very large cofactors. We slightly relax the condition to allow small inevitable cofactors that are imposed by the polynomial parametrizations in the popular constructions of pairing-friendly curves. This means that this property distinguishes those curves in a given family that provide as much resistance against small-subgroup attacks as possible.

We select subgroup-secure curves for four of the most efficient families of pairing-friendly curves that can replace existing curves in pairing libraries with minimal code change. For example, we find a low NAF-weight Barreto-Naehrig (BN) curve for which no (related) elliptic curve subgroup of order smaller than n exists. Replacing `BN254` with this one could allow implementers to remove certain membership tests (via scalar multiplications). Returning to the blind signature scheme above, this would mean that Alice only needs to check that the point $\tilde{Q}$ is on the right curve before signing, and this is true whether the protocol is arranged such that $\tilde{Q}$ is intended to be in $\mathbb{G}_1$ or in $\mathbb{G}_2$. Even if Carol sends Alice a point that is not in the order n subgroup, Carol's job of recovering a from Q and $[a]Q$ is no easier since, by the application of Definition 1 to the BN family, the smallest prime factor dividing the order of Q will always be at least n.

While existing curves in the literature are not subgroup-secure and may therefore require expensive operations to guarantee discrete-log security in the pairing groups, the curves we propose can, wherever possible, maintain their discete-log security even in the absence of some of the subgroup membership checks. Our performance benchmarks show that replacing existing curves with subgroup-secure curves incurs only a minor performance penalty in the pairing computation; on the other hand, all group operations remain unaffected by this stronger security notion and retain their efficiency.

Related work. The comments made by Chen, Cheng and Smart [19] are central to the theme of this work. We occasionally refer back to the following remark, which quotes [19, §2.2] verbatim.

Remark 1 ([19]). "An assumption is [often] made that all values passed from one party to another lie in the correct groups. Such assumptions are often implicit within security proofs. However, one needs to actually:

(i) check that given message flows lie in the group,
(ii) force the messages to lie in the group via additional computation, or
(iii) choose parameters carefully so as the problem does not arise.

Indeed, some attacks on key agreement schemes, such as the small-subgroup attack [41], are possible because implementors do not test for subgroup membership. For pairing-based systems one needs to be careful whether and how one implements these subgroup membership tests as it is not as clear as for standard discrete logarithm based protocols."

The overall aim of this paper is to explore and optimize option (iii) above.

In the paper introducing small-subgroup attacks [41], Lim and Lee suggest that a strong countermeasure is to ensure that the intended cryptographic subgroup is the smallest subgroup within *the* large group. In the context of pairing-based cryptography, Scott [54] showed a scenario in which a small-subgroup attack could be possible on elements of the third pairing group $\mathbb{G}_T$, the target group, and subsequently he adapted the Lim-Lee solution to put forward the notion of "$\mathbb{G}_T$-strong" curves. Our definition of subgroup security (see Definition 1) applies this solution to all three of the pairing groups, the two elliptic curve input groups $\mathbb{G}_1$, $\mathbb{G}_2$ as well as the target group $\mathbb{G}_T$, and therefore this paper can be seen as an extension and generalization of Scott's idea: while he gave an example of a BN curve that is $\mathbb{G}_T$-strong, we give replacement curves from several families that are both $\mathbb{G}_T$-strong *and* $\mathbb{G}_2$-strong – this is the optimal situation for the families used in practice[4].

2 Pairing groups and pairing-friendly curves

For modern security levels, the most practical pairings make use of an ordinary elliptic curve E defined over a large prime field $\mathbb{F}_p$ whose *embedding degree* (with respect to a large prime divisor n of $\#E(\mathbb{F}_p)$) is k, i.e. k is the smallest positive integer such that $n \mid p^k - 1$. In this case, there exists a pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$, where $\mathbb{G}_1$ is the subgroup $E(\mathbb{F}_p)[n]$ of order n of $E(\mathbb{F}_p)$, $\mathbb{G}_2$ is a specific subgroup of order n of $E(\mathbb{F}_{p^k})$ contained in $E(\mathbb{F}_{p^k}) \setminus E(\mathbb{F}_p)$, and $\mathbb{G}_T$ is the subgroup of n-th roots of unity, in other words, the subgroup of order n of the multiplicative group $\mathbb{F}_{p^k}^{\times}$.

Let $t \in \mathbb{Z}$ be the trace of the Frobenius endomorphism on $E/\mathbb{F}_p$, so that $\#E(\mathbb{F}_p) = p + 1 - t$ and $|t| \leq 2\sqrt{p}$. Write $t^2 - 4p = Dv^2$ for some square-free negative integer D and some integer v. All of the pairing-friendly curves in this paper have $D = -3$ and $6 \mid k$; together, these two properties ensure that we can always write the curves as $E/\mathbb{F}_p : y^2 = x^3 + b$, and that we can make use of a *sextic twist* $E'/\mathbb{F}_{p^{k/6}} : y^2 = x^3 + b'$ of $E(\mathbb{F}_{p^k})$ to instead represent $\mathbb{G}_2$ by the isomorphic group $\mathbb{G}_2' = E'(\mathbb{F}_{p^{k/6}})[n]$, such that coordinates of points in $\mathbb{G}_2'$ lie in the much smaller subfield $\mathbb{F}_{p^{k/6}}$ of $\mathbb{F}_{p^k}$. Henceforth we abuse notation and rewrite $\mathbb{G}_2$ as $\mathbb{G}_2 = E'(\mathbb{F}_{p^{k/d}})[n]$.

For a particular curve $E/\mathbb{F}_q : y^2 = x^3 + b$ with $D = -3$, where $q = p^e$ for some $e \geq 1$, there are six *twists* of E defined over $\mathbb{F}_q$, including E itself. These twists are isomorphic to E when considered over $\mathbb{F}_{q^6}$. The following lemma (cf. [35, §A.14.2.3]) determines the group orders of these twists over $\mathbb{F}_q$, and is used several times in this work.

[4] Our definition of subgroup security incorporates $\mathbb{G}_1$ for completeness, since for curves from the most popular families of pairing-friendly curves the index of $\mathbb{G}_1$ in $E(\mathbb{F}_p)$ is both greater than one and much less than the size of $\mathbb{G}_1$, thereby necessarily containing small subgroups. The only exceptions are the prime order families like the MNT [43], Freeman [24], and BN [6] curve families, for which this index is 1.

Lemma 1. *Let t be the trace of Frobenius of the elliptic curve $E/\mathbb{F}_q\colon y^2 = x^3 + b$, and let $v \in \mathbb{Z}$ such that $t^2 - 4q = -3v^2$. Up to isomorphism, there are at most six curves (including E) defined over $\mathbb{F}_q$ with trace t' such that $t'^2 - 4q = -3v'^2$ for some square-free $v' \in \mathbb{Z}$. The six possibilities for t' are t, $-t$, $(t + 3v)/2$, $-(t + 3v)/2$, $(t - 3v)/2$, and $-(t - 3v)/2$.*

In this work, we focus on four of the most popular families of ordinary pairing-friendly curves: the Barreto-Naehrig (BN) family [6] with $k = 12$ which is favorable at the 128-bit security level; the Barreto-Lynn-Scott (BLS) cyclotomic family [4] with $k = 12$ and the Kachisa-Schaefer-Scott (KSS) family [37] with $k = 18$, both of which are suitable at the 192-bit security level; and the cyclotomic BLS family with $k = 24$, which is well suited for use at the 256-bit security level.

The above examples of pairing-friendly curves are all *parameterized families*. This means that the parameters p, t and n of a specific curve from each family are computed via the evaluation of univariate polynomials $p(u)$, $t(u)$ and $n(u)$ in $\mathbb{Q}[u]$ at some $u_0 \in \mathbb{Z}$. The typical way to find a good curve instance is to search over integer values u_0 of low NAF-weight (i.e. with as few non-zero entries in signed-binary, non-adjacent form (NAF) representation as possible) and of a suitable size, until $p(u_0)$ and $n(u_0)$ are simultaneously prime. Since our curves are all of the form $E/\mathbb{F}_p\colon y^2 = x^3 + b$, and since Lemma 1 states that there are at most 6 isomorphism classes over $\mathbb{F}_p$, the correct curve is quickly found by iterating through small values of b and testing non-zero points $P \neq \mathcal{O}$ on E for the correct order, i.e. testing whether $[p(u_0) + 1 - t(u_0)]P = \mathcal{O}$.

3 Subgroup-secure pairing-friendly curves

In this section we recall small-subgroup attacks and define the notion of *subgroup security*, a property that is simple to achieve in practice and that strengthens the resistance of pairing-friendly curves against *subgroup attacks*. After that, we discuss the four most popular choices of pairing-friendly curve families, BN ($k = 12$), KSS ($k = 18$) and BLS ($k = 12$ and $k = 24$) curves and provide examples of subgroup-secure curves suitable for efficient implementation of optimal pairings at the 128-, 192-, and 256-bit security levels.

3.1 Small-subgroup attacks

Small-subgroup attacks against cryptographic schemes based on the discrete logarithm problem (DLP) were introduced by Lim and Lee [41]. The following is a brief description of the basic idea in a general group setting.

Suppose that $\mathbb{G}$ is a group of prime order n (written additively), which is contained in a larger, finite abelian group $\mathcal{G}$, and let h be the index of $\mathbb{G}$ in $\mathcal{G}$, $|\mathcal{G}| = h \cdot n$. Suppose that the DLP is hard in any subgroup of $\mathcal{G}$ of large enough prime order. In particular, assume that the prime n is large enough such that the DLP is infeasible in $\mathbb{G}$. If the index h has a small prime factor r, then there exists a group element P of order a multiple of r, and if r is small enough, the DLP in $\langle P \rangle$ can be easily solved modulo r. If an attacker manages to force a protocol participant to use P for a group exponentiation involving a secret exponent, instead of using a valid element from $\mathbb{G}$, solving the DLP in $\langle P \rangle$ provides partial information on the secret exponent. If h has several small prime factors, the Pohlig-Hellman attack [50] may be able to recover the full secret exponent.

Such small-subgroup attacks can be avoided by *membership testing*, i.e. by checking that any point P received during a protocol actually belongs to the group $\mathbb{G}$ and cannot have a smaller order (see point (i) in Remark 1). Another way to thwart these attacks is a *cofactor exponentiation* or *cofactor multiplication* (which is a solution to achieve point (ii) in Remark 1). If every received element P is multiplied by the index h, which also means that the protocol needs to be adjusted to work with the point $[h]P$ instead of P, then points of small order are mapped to $\mathcal{O}$ and any small-order component of P is cleared by this exponentiation.

3.2 Subgroup security

If $h > 1$ and it does not contain any prime factors smaller than n, then $\mathbb{G}$ is one of the subgroups in $\mathcal{G}$ with the weakest DLP security. In other words, for any randomly chosen element $P \in \mathcal{G}$, the DLP in the group $\langle P \rangle$ is guaranteed to be at least as hard as the DLP in $\mathbb{G}$, since even if $|\langle P \rangle| = |\mathcal{G}|$, the Pohlig-Hellman reduction [50] requires the solution to a DLP in a subgroup of prime order at least n. Depending on the protocol design, it might be possible to omit membership testing and cofactor multiplication if parameters are chosen such that h does not have prime factors smaller than n: this is one possibility that addresses point (iii) in Remark 1.

One might consider omitting the test as to whether an element belongs to the group $\mathbb{G}$ if this is a costly operation. For example, if testing membership for $\mathbb{G}$ requires a relatively expensive group exponentiation, and testing membership for $\mathcal{G}$ is relatively cheap (i.e. costs no more than a few group operations), one can replace the costly check by the cheaper one given that the index h does not have any small factors. When the group $\mathcal{G}$ is the group of $\mathbb{F}_q$-rational points on an elliptic curve E, and $\mathbb{G}$ is a prime order subgroup, then testing whether a point P belongs to $\mathcal{G}$ is relatively cheap, because it only requires to check validity of the curve equation, while testing whether a point belongs to $\mathbb{G}$ additionally requires either a scalar multiplication $[n]P$ to check whether P has the right order, or a cofactor multiplication $[h]P$ to force the resulting point to have the right order. If the cofactor is small, the latter cost is low, but for large cofactors, it might be more efficient to refrain from carrying out any of the exponentiations when working with suitable parameters.

An attempt to define the notion of subgroup security could be to demand that the index h (if it is not equal to 1) only contains prime factors of size n or larger, in which case both exponentiations are very costly. However, in the case of elliptic curve cryptography (ECC), such a definition does not make sense, since curves are chosen such that the cofactor is equal to 1 or a very small power of 2 (such as 4 or 8) depending on the curve model that is selected for efficiency and security reasons. Although there are good reasons to require cofactor $h = 1$, it would unnecessarily exclude curve models which allow performance gains by having a small cofactor (such as Montgomery [44] or Edwards [23] curve models). Therefore, demanding only large prime factors in h only makes sense if the group inherently has large, unavoidable cofactors by construction. This is the case for some of the groups that arise from pairing-friendly curves.

For the three pairing (sub)groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ defined in Section 2, there are very natural choices of three associated groups $\mathcal{G}_1$, $\mathcal{G}_2$ and $\mathcal{G}_T$ for which testing membership is easy. Namely, we define $\mathcal{G}_1$, $\mathcal{G}_2$, and $\mathcal{G}_T$ as follows:

$$\mathbb{G}_1 \subseteq \mathcal{G}_1 = E(\mathbb{F}_p), \qquad \mathbb{G}_2 \subseteq \mathcal{G}_2 = E'(\mathbb{F}_{p^{k/d}}), \qquad \mathbb{G}_T \subseteq \mathcal{G}_T = G_{\Phi_k(p)},$$

where $G_{\Phi_k(p)}$ is the cyclotomic subgroup[5] of order $\Phi_k(p)$ in $\mathbb{F}_{p^k}^{\times}$. Scott also chose $\mathcal{G}_T$ that way when proposing $\mathbb{G}_T$-strong curves [54]. Note that testing membership in $\mathcal{G}_1$ or $\mathcal{G}_2$ simply amounts to checking the curve equation for $E(\mathbb{F}_p)$ or $E'(\mathbb{F}_{p^{k/d}})$, respectively, and that testing whether an element is in $\mathcal{G}_T$ can also "be done at almost no cost using the Frobenius" [54, §8.3]. We give more details on this check in §5.2, where we also discuss why $\mathcal{G}_T$ is chosen as the cyclotomic subgroup of order $\Phi_k(p)$, rather than the full multiplicative group $\mathbb{F}_{p^k}^{\times}$.

Since $|\mathbb{G}_1| = |\mathbb{G}_2| = |\mathbb{G}_T| = n$, the relevant indices $h_1, h_2, h_T \in \mathbb{Z}$ are defined as

$$h_1 = \frac{|\mathcal{G}_1|}{n}, \qquad h_2 = \frac{|\mathcal{G}_2|}{n}, \qquad h_T = \frac{|\mathcal{G}_T|}{n}.$$

The sizes of these cofactors are determined by the properties of the pairing-friendly curve. For all of the curves in this paper, both $\mathbb{G}_2$ and $\mathbb{G}_T$ are groups of order n in the much larger groups $\mathcal{G}_2$ and $\mathcal{G}_T$ and the cofactors h_2 and h_T are at least of a similar size as n. The group $\mathcal{G}_1$ is typically not that large, and comes closer to the case of a group used in plain ECC. Therefore the cofactor h_1 is smaller than n, and in almost all cases larger than 1.

The next attempt at a definition of subgroup security could demand that for any of the three pairing groups for which the cofactor is of size similar to n or larger, it must not have prime factors significantly smaller than n. This is a more useful definition since it focuses on the case in which large cofactors exist. However, most pairing-friendly curves are instances of parameterized families and their parameters are derived as the evaluation of rational polynomials at an integer value. And for certain families, these polynomials may also necessarily produce small factors in the indices (cf. Remark 2 below).

The following definition of subgroup security accounts for this fact in capturing – for a given polynomial family of pairing-friendly curves – the best that can be achieved within that family. We make use of the fact that, for the parameterized families of interest in this work, the three cofactors above are also parameterized as $h_1(u), h_2(u), h_T(u) \in \mathbb{Q}[u]$.

[5] Here Φ_k denotes the k-th cyclotomic polynomial.

Definition 1 (Subgroup security). *Let $p(u), t(u), n(u) \in \mathbb{Q}[u]$ parameterize a family of ordinary pairing-friendly elliptic curves, and for any particular $u_0 \in \mathbb{Z}$ such that $p = p(u_0)$ and $n = n(u_0)$ are prime, let E be the resulting pairing-friendly elliptic curve over $\mathbb{F}_p$ of order divisible by n. We say that E is* subgroup-secure *if all $\mathbb{Q}[u]$-irreducible factors of $h_1(u)$, $h_2(u)$ and $h_T(u)$ that can represent primes and that have degree at least that of $n(u)$, contain no prime factors smaller than $n(u_0) \in \mathbb{Z}$ when evaluated at $u = u_0$.*

It should be pointed out immediately that the wording of "smaller" in Definition 1 can be relaxed in cases where the difference is relatively close. Put simply, Definition 1 aims to prohibit the existence of any unnecessary subgroups of size smaller than n inside the larger groups for which validation is easy. We note that, for simplicity, Definition 1 says that subgroup security is dependent on the pairing-friendly curve E. However, given that the property is dependent on the three groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$, it would be more precise to say that the property is based on the *pairing* that is induced by E and n.

In Table 1, we have collected popular pairing-friendly curves that have been used in the literature and in pairing implementations because of their efficiency. We have evaluated all such curves according to their subgroup security. This means that we had to (partially) factor the indices h_1, h_2 and h_T. Note that h_1 is quite small in all cases, and since it is smaller than n, there is no need to find its factorization in order to test for subgroup security. To find the (partial) factorizations of h_2 and h_T, we used the implementation of the ECM method[6] [39] in Magma [16]. To illustrate the factorizations, we use p_m and c_m to denote some m-bit prime and some m-bit composite number respectively.

It is important to note that the curves chosen from the literature in Table 1 were not chosen strategically; none of them are subgroup secure, but the chances of a curve miraculously achieving this property (without being constructed to) is extremely small. Thus, these curves are a fair representation of all ordinary pairing-friendly curves proposed in previous works, since we could not find any prior curve that is subgroup secure according to Definition 1 (the closest example being the BN curve given by Scott [54, §9], which is $\mathbb{G}_T$-strong but not $\mathbb{G}_2$-strong).

In §3.3-§3.6, we focus on achieving subgroup security for the four popular parameterized families of pairing-friendly curves mentioned in Section 2. Our treatment of each family follows the same recipe: the polynomial parameterizations of p, n and t immediately give us parameterizations for h_T as $h_T(u) = \Phi_k(p(u))/n(u)$, but in each case it takes some more work to determine the parameterization of the cofactor h_2; this is done in Propositions 1-4. To find a subgroup-secure curve instance from each family, we searched through $u = u_0$ values of a fixed length and of low NAF-weight, increasing the NAF-weight (and exhausting all possibilities each time) until a curve was found with $p(u_0)$, $n(u_0)$, $h_2(u_0)$ and $h_T(u_0)$ all prime. In theory we could have relaxed the search condition of $h_2(u_0)$ and $h_T(u_0)$ being prime to instead having no prime

[6] We tweaked the parameters according to `http://www.loria.fr/~zimmerma/records/ecm/params.html`, until enough factors were found.

Table 1. Subgroup security for pairing-friendly curves previously used in the literature, considering curves from the Barreto-Naehrig (BN) family [6] with $k = 12$; the Barreto-Lynn-Scott (BLS) cyclotomic families [4] with $k = 12$ and $k = 24$ and the Kachisa-Schaefer-Scott (KSS) family [37] with $k = 18$. The columns for p and n give the bitsizes of these primes. The column marked "where?" provides reference to the literature in which the specific curves have been used in implementations. The column $\mathrm{wt}(u_0)$ displays the NAF-weight of the parameter u_0. The symbols p_m and c_m in the columns that display factors of the indices h_1, h_2, and h_T are used to denote an unspecified prime of size m bits or a composite number of size m bits, respectively.

sec.	family	p	n	Curve choices					sub.
level	k	(bits)	(bits)	where?	$\mathrm{wt}(u_0)$	h_1	h_2	h_T	sec.?
128	BN 12	256	256	[46]	23	1	$c_{17}p_{239}$	$c_{74}c_{692}$	no
		254	254	[47, 2, 49, 53, 59]	3	1	$c_{96}p_{158}$	$c_{79}c_{681}$	no
		254	254	Example 1	6	1	p_{254}	p_{762}	yes
192	BLS 12	638	427	[1]	4	c_{212}	$c_{48}c_{802}$	$c_{58}c_{2068}$	no
		635	424	[15]	4	c_{211}	$c_{15}c_{831}$	$c_{33}c_{2082}$	no
		635	425	Example 2	6	c_{211}	p_{845}	p_{2114}	yes
192	KSS 18	511	378	[53]	8	c_{133}	$c_{50}c_{1106}$	$c_{26}c_{2660}$	no
		508	376	[1]	4	c_{133}	$c_{85}c_{1063}$	$c_{15}c_{2656}$	no
		508	376	Example 3	9	c_{133}	$3p_{1146}$	p_{2671}	yes
256	BLS 24	639	513	[22]	4	c_{127}	2^2c_{2040}	$c_{41}c_{4556}$	no
		629	505	[53]	4	c_{125}	$2^2p_{69}c_{1940}$	$c_{132}c_{4392}$	no
		629	504	Example 4	8	c_{125}	p_{2010}	p_{4524}	yes

factors smaller than n, but finding or proving such factorizations requires an effort beyond the efforts of current factorization records. The fixed length of u_0 was chosen so that the parameter sizes closely match the sizes of curves already in the literature and in online libraries; we also aimed to make sure the parameters matched in terms of efficient constructions of the extension field towerings. In order to compare to previous curves, we have included the subgroup-secure curves found in each family in Table 1.

3.3 BN curves with $k = 12$

The Barreto-Naehrig (BN) family [6] of curves is particularly well-suited to the 128-bit security level. BN curves are found via the parameterizations $p(u) = 36u^4+36u^3+24u^2+6u+1$, $t(u) = 6u^2+1$, and $n(u) = 36u^4+36u^3+18u^2+6u+1$.

In this case $\#E(\mathbb{F}_p) = n(u)$, so $\mathbb{G}_1 = \mathcal{G}_1 = E(\mathbb{F}_p)$, meaning $h_1(u) = 1$. The cofactor in $\mathcal{G}_T = G_{\Phi_{12}(p)}$ is parameterized as $h_T(u) = (p(u)^4 - p(u)^2+1)/(n(u))$. The following proposition gives the cofactor $h_2(u)$.

Proposition 1. *With parameters as above, the correct sextic twist $E'/\mathbb{F}_{p^2}$ for a BN curve has group order $\#E'(\mathbb{F}_{p^2}) = h_2(u) \cdot n(u)$, where*

$$h_2(u) = 36u^4 + 36u^3 + 30u^2 + 6u + 1.$$

Proof. [45, Rem. 2.13] says that BN curves always have $h_2(u) = p(u) - 1 + t(u)$. □

Example 1. The BN curve $E/\mathbb{F}_p\colon y^2 = x^3+5$ with $u_0 = 2^{62}+2^{59}+2^{55}+2^{15}+2^{10}-1$ has both $p = p(u_0)$ and $n = n(u_0) = \#E(\mathbb{F}_p)$ as 254-bit primes. A model for the correct sextic twist over $\mathbb{F}_{p^2} = \mathbb{F}_p[i]/(i^2+1)$ is $E'/\mathbb{F}_{p^2}\colon y^2 = x^3+5(i+1)$, and its group order is $\#E'(\mathbb{F}_{p^2}) = h_2 \cdot n$, where $h_2 = h_2(u_0)$ is also a 254-bit prime. Thus, once points are validated to be in $\mathcal{G}_1 = E(\mathbb{F}_p)$ or $\mathcal{G}_2 = E'(\mathbb{F}_{p^2})$, no cofactor multiplications are required to avoid subgroup attacks on this curve, i.e. there are no points of order less than n in $E(\mathbb{F}_p)$ or $E'(\mathbb{F}_{p^2})$. Furthermore, the group $\mathcal{G}_T$ has order $|\mathcal{G}_T| = h_T \cdot n$, where $h_T = h_T(u_0)$ is a 762-bit prime, so once $\mathbb{F}_{p^{12}}$ elements are validated to be in $\mathcal{G}_T = G_{\Phi_{12}(p)}$, no further cofactor multiplications are necessary for discrete log security here either. For completeness, we note that $\mathbb{F}_{p^{12}}$ can be constructed as $\mathbb{F}_{p^6} = \mathbb{F}_{p^2}[v]/(v^3-(i+1))$ and $\mathbb{F}_{p^{12}} = \mathbb{F}_{p^6}[w]/(w^2-v)$; E and E' are then isomorphic over $\mathbb{F}_{p^{12}}$ via $\Psi\colon E' \to E$, $(x',y') \mapsto (x'/v, y'/(vw))$.

3.4 BLS curves with $k = 12$

The Barreto-Lynn-Scott (BLS) family [4] with $k = 12$ was shown to facilitate efficient pairings at the 192-bit security level [1]. This family has the parameterizations $p(u) = (u-1)^2 \cdot (u^4-u^2+1)/3+u$, $t(u) = u+1$, and $n(u) = u^4-u^2+1$.

Here $\#E(\mathbb{F}_p) = h_1(u) \cdot n(u)$ with $h_1(u) = (u-1)^2/3$, so there is always a cofactor that is much smaller than n in $\mathcal{G}_1$. Again, the cofactor in $\mathcal{G}_T = G_{\Phi_{12}(p)}$ is $h_T(u) = (p(u)^4 - p(u)^2 + 1)/(n(u))$. The following proposition gives the cofactor $h_2(u)$.

Proposition 2. *With parameters as above, the correct sextic twist $E'/\mathbb{F}_{p^2}$ for a $k = 12$ BLS curve has group order $\#E'(\mathbb{F}_{p^2}) = h_2(u) \cdot n(u)$, where*

$$h_2(u) = (u^8 - 4u^7 + 5u^6 - 4u^4 + 6u^3 - 4u^2 - 4u + 13)/9.$$

Proof. Write $\#E(\mathbb{F}_{p^2}) = p_2 + 1 - t_2$, where $p_2 = p^2$ and $t_2 = t^2 - 2p$ [11, Corollary VI.2]. The CM equation for $E(\mathbb{F}_{p^2})$ is $t_2^2 - 4p_2 = -3v_2^2$, which gives $v_2 = (x-1)(x+1)(2x^2-1)/3$. Lemma 1 reveals that $t' = (t_2 - 3v_2)/2$ gives rise to the correct sextic twist $E'/\mathbb{F}_{p^2}$ with $n \mid \#E'(\mathbb{F}_{p^2}) = p^2+1-t'$, and the cofactor follows as $h_2 = (p^2+1-t')/n$. □

Example 2. The $k = 12$ BLS curve $E/\mathbb{F}_p\colon y^2 = x^3 - 2$ with $u_0 = -2^{106} - 2^{92} - 2^{60} - 2^{34} + 2^{12} - 2^9$ has $p = p(u_0)$ as a 635-bit prime and $\#E(\mathbb{F}_p) = h_1 \cdot n$, where $n = n(u_0)$ is a 425-bit prime and the composite cofactor $h_1 = h_1(u_0)$ is 211 bits. A model for the correct sextic twist over $\mathbb{F}_{p^2} = \mathbb{F}_p[i]/(i^2+1)$ is $E'/\mathbb{F}_{p^2}\colon y^2 = x^3 - 2/(i+1)$, and its group order is $\#E'(\mathbb{F}_{p^2}) = h_2 \cdot n$, where $h_2 = h_2(u_0)$ is an 845-bit prime. Furthermore, the group $\mathcal{G}_T$ has order $|\mathcal{G}_T| = h_T \cdot n$, where $h_T = h_T(u_0)$ is a 2114-bit prime. Thus, once elements are validated to be in either $\mathcal{G}_2 = E'(\mathbb{F}_{p^2})$ or $\mathcal{G}_T = G_{\Phi_{12}(p)}$, no cofactor multiplications are required to avoid subgroup attacks. On the other hand, a scalar multiplication (by either h_1 or n) may be necessary to ensure that points in $E(\mathbb{F}_p)$ have the requisite discrete log security, and this is unavoidable across the $k = 12$ BLS family. For completeness, we note that $\mathbb{F}_{p^{12}}$ can be constructed as $\mathbb{F}_{p^6} = \mathbb{F}_{p^2}[v]/(v^3 - (i+1))$ and $\mathbb{F}_{p^{12}} = \mathbb{F}_{p^6}[w]/(w^2 - v)$; E and E' are then isomorphic over $\mathbb{F}_{p^{12}}$ via $\Psi\colon E' \to E$, $(x', y') \mapsto (x' \cdot v, y' \cdot vw)$.

3.5 KSS curves with $k = 18$

The Kachisa-Schaefer-Scott (KSS) family [37] with $k = 18$ is another family that is suitable at the 192-bit security level. This family has the parameterizations $p(u) = u^8 + 5u^7 + 7u^6 + 37u^5 + 188u^4 + 259u^3 + 343u^2 + 1763u + 2401$, $t(u) = (u^4 + 16u + 7)/7$, and $n(u) = (u^6 + 37u^3 + 343)/7^3$.

Here $\#E(\mathbb{F}_p) = h_1(u) \cdot n(u)$ with $h_1(u) = (49u^2 + 245u + 343)/3$, so again there is always a cofactor much smaller than n in $\mathcal{G}_1$. The cofactor in $\mathcal{G}_T = G_{\Phi_{18}(p)}$ is $h_T(u) = (p(u)^6 - p(u)^3 + 1)/(n(u))$. The proposition below gives the cofactor $h_2(u)$.

Proposition 3. *With parameters as above, the correct sextic twist $E'/\mathbb{F}_{p^3}$ for a $k = 18$ KSS curve has group order $\#E'(\mathbb{F}_{p^3}) = h_2(u) \cdot n(u)$, where*

$$h_2(u) = (u^{18} + 15u^{17} + 96u^{16} + 409u^{15} + 1791u^{14} + 7929u^{13} + 27539u^{12} + 81660u^{11} + 256908u^{10} + 757927u^9 + 1803684u^8 + 4055484u^7 + 9658007u^6 + 19465362u^5 + 30860595u^4 + 50075833u^3 + 82554234u^2 + 88845918u + 40301641)/27.$$

Proof. Write $\#E(\mathbb{F}_{p^3}) = p_3 + 1 - t_3$, where $p_3 = p^3$ and $t_3 = t^3 - 3pt$ [11, Corollary VI.2]. The CM equation for $E(\mathbb{F}_{p^3})$ is $t_3^2 - 4p_3 = -3v_3^2$, which gives $v_3 = (x^4 + 7x^3 + 23x + 119)(5x^4 + 14x^3 + 94x + 259)(4x^4 + 7x^3 + 71x + 140)/3087$. Lemma 1 reveals that $t' = (t_3 + 3v_3)/2$ gives rise to the correct sextic twist $E'/\mathbb{F}_{p^3}$ with $n \mid \#E'(\mathbb{F}_{p^3}) = p^3 + 1 - t'$, and the cofactor follows as $h_2 = (p^3 + 1 - t')/n$. □

Remark 2. The KSS parameterization requires $u \equiv 14 \bmod 42$. Under this condition, it is straightforward to see that $h_2(u) \equiv 0 \bmod 3$. Thus, there is always a factor of 3 in the cofactor of $\mathcal{G}_2$ in this family.

Example 3. The $k = 18$ KSS curve $E/\mathbb{F}_p\colon y^2 = x^3 + 2$ with $u_0 = 2^{64} + 2^{47} + 2^{43} + 2^{37} + 2^{26} + 2^{25} + 2^{19} - 2^{13} - 2^7$ has $p = p(u_0)$ as a 508-bit prime and $\#E(\mathbb{F}_p) = h_1 \cdot n$, where $n = n(u_0)$ is a 376-bit prime and the composite cofactor $h_1 = h_1(u_0)$ is 133 bits. A model for the correct sextic twist over $\mathbb{F}_{p^3} = \mathbb{F}_p[v]/(v^3 - 2)$ is $E'/\mathbb{F}_{p^3}: y^2 = x^3 + 2/v$, and its group order is $\#E'(\mathbb{F}_{p^3}) = 3 \cdot h_2 \cdot n$ (see Remark 2), where $h_2 = h_2(u_0)$ is a 1146-bit prime. Thus, once points are validated to be in $E'(\mathbb{F}_{p^3})$, it may be necessary to multiply points by 3 to clear this cofactor. Furthermore, a scalar multiplication by h_1 or n may be necessary to ensure that random points in $E(\mathbb{F}_p)$ are in $\mathbb{G}_1 = E(\mathbb{F}_p)[n]$ before any secret scalar multiplications take place. On the other hand, once points are validated to be in $\mathcal{G}_T = G_{\Phi_{18}(p)}$, no cofactor multiplications are required to avoid subgroup attacks since $h_T = h_T(u_0)$ is a 2671-bit prime in this case. For completeness, we note that $\mathbb{F}_{p^{18}}$ can be constructed as $\mathbb{F}_{p^9} = \mathbb{F}_{p^3}[v]/(w^3 - v)$ and $\mathbb{F}_{p^{18}} = \mathbb{F}_{p^9}[z]/(z^3 - w)$; E and E' are then isomorphic over $\mathbb{F}_{p^{18}}$ via $\Psi\colon E' \to E$, $(x', y') \mapsto (x' \cdot w, y' \cdot wz)$.

3.6 BLS curves with $k = 24$

The Barreto-Lynn-Scott (BLS) family [4] with $k = 24$ is well suited to the 256-bit security level. This family has the parameterizations $p(u) = (u-1)^2 \cdot (u^8 - u^4 + 1)/3 + u$, $t(u) = u + 1$, and $n(u) = u^8 - u^4 + 1$.

Here $\#E(\mathbb{F}_p) = h_1(u) \cdot n(u)$ with $h_1(u) = (u-1)^2/3$, so once more there is always a cofactor which is much smaller than n in $\#\mathcal{G}_1$. Here the cofactor for $\mathcal{G}_T = G_{\Phi_{24}(p)}$ is $h_T(u) = (p(u)^8 - p(u)^4 + 1)/(n(u))$. The following proposition gives the cofactor $h_2(u)$.

Proposition 4. *With parameters as above, the correct sextic twist $E'/\mathbb{F}_{p^4}$ for a $k = 24$ BLS curve has group order $\#E'(\mathbb{F}_{p^4}) = h(u) \cdot n(u)$, where*

$h_2(u) = (u^{32} - 8u^{31} + 28u^{30} - 56u^{29} + 67u^{28} - 32u^{27} - 56u^{26} + 160u^{25} - 203u^{24} + 132u^{23} + 12u^{22} - 132u^{21} + 170u^{20} - 124u^{19} + 44u^{18} - 4u^{17} + 2u^{16} + 20u^{15} - 46u^{14} + 20u^{13} + 5u^{12} + 24u^{11} - 42u^{10} + 48u^9 - 101u^8 + 100u^7 + 70u^6 - 128u^5 + 70u^4 - 56u^3 - 44u^2 + 40u + 100)/81.$

Proof. Write $\#E(\mathbb{F}_{p^4}) = p_4 + 1 - t_4$, where $p_4 = p^4$ and $t_4 = t^4 - 4pt^2 + 2p^2$ [11, Corollary VI.2]. The CM equation for $E(\mathbb{F}_{p^4})$ is $t_4^2 - 4p_4 = -3v_4^2$, which gives $v_4 = (x-1)(x+1)(2x^4-1)(2x^{10} - 4x^9 + 2x^8 - 2x^6 + 4x^5 - 2x^4 - x^2 - 4x - 1)/9$. Lemma 1 reveals that $t' = (t_4 + 3v_4)/2$ gives rise to the correct sextic twist $E'/\mathbb{F}_{p^4}$ with $n \mid \#E'(\mathbb{F}_{p^3}) = p^4 + 1 - t'$, and the cofactor follows as $h_2 = (p^4 + 1 - t')/n$. □

Example 4. The $k = 24$ BLS curve $E/\mathbb{F}_p \colon y^2 = x^3 + 1$ with $u_0 = -(2^{63} - 2^{47} - 2^{31} - 2^{26} - 2^{24} + 2^8 - 2^5 + 1)$ has $p = p(u_0)$ as a 629-bit prime and $\#E(\mathbb{F}_p) = h_1 \cdot n$, where $n = n(u_0)$ is a 504-bit prime and the composite cofactor h_1 is 125 bits. If $\mathbb{F}_{p^4}$ is constructed by taking $\mathbb{F}_{p^2} = \mathbb{F}_p[i]/(i^2+1)$ and $\mathbb{F}_{p^4} = \mathbb{F}_{p^2}[v]/(v^2 - (i+1))$, then a model for the correct sextic twist is $E'/\mathbb{F}_{p^4} \colon y^2 = x^3 + 1/v$, and its group order is $\#E'(\mathbb{F}_{p^4}) = h_2 \cdot n$, where $h_2 = h_2(u_0)$ is a 2010-bit prime. Furthermore, the group $\mathcal{G}_T$ has order $|\mathcal{G}_T| = h_T \cdot n$, where $h_T = h_T(u_0)$ is a 4524-bit prime. Thus, once elements are validated to be in either $\mathcal{G}_2 = E'(\mathbb{F}_{p^4})$ or $\mathcal{G}_T = G_{\Phi_{24}(p)}$, no cofactor multiplications are required to avoid subgroup attacks. On the other hand, once random points are validated to be on $E(\mathbb{F}_p)$, a scalar multiplication by h_1 or n is required to ensure points are in $\mathbb{G}_1$. In this case we note that $\mathbb{F}_{p^{24}}$ can be constructed as $\mathbb{F}_{p^{12}} = \mathbb{F}_{p^4}[w]/(w^3 - v)$ and $\mathbb{F}_{p^{24}} = \mathbb{F}_{p^{12}}[z]/(z^2 - w)$; E and E' are then isomorphic over $\mathbb{F}_{p^{24}}$ via $\Psi \colon E' \to E$, $(x', y') \mapsto (x' \cdot w, y' \cdot wz)$.

4 Performance comparisons: the price of subgroup security

As we saw in Table 1, subgroup-secure curves are generally found with a search parameter of larger NAF-weight than non-subgroup-secure curves because of the additional (primality) restrictions imposed in the former case[7]. Thus, pairings

[7] We note that Scott [54, §9] hinted at this "negative impact" when discussing a $\mathbb{G}_T$-strong curve.

computed with the subgroup-secure curves will naturally be more expensive. In this section, we give performance numbers that provide a concrete comparison between our subgroup-secure curves and the speed-record curves that have appeared elsewhere in the literature. Table 2 shows the approximate factor slowdowns incurred by choosing subgroup-secure curves. We note that these slowdowns are only reported in the computation of the pairing; optimal methods for group exponentiations are unrelated to the search parameter and will therefore remain unchanged when using a subgroup-secure curve of the same size in the same curve family. On the other hand, operations like hashing to $\mathbb{G}_2$ [55, 26] do benefit from a low hamming-weight and will also experience a slowdown when using subgroup secure curves.

We used a pairing library written in C to obtain the performance numbers in Table 2, benchmarked on an Intel Xeon E5-2620 clocked at 2.0GHz. We note that our library does not perform as fast as some other pairing libraries as it was written entirely in C without using any assembly-level optimizations. Nevertheless, it uses all of the state-of-the-art high-level optimizations such as the optimal ate pairing [58] with a fast final exponentiation [56], as well as taking advantage of efficient extension field towerings [7] and enhanced operations in the cyclotomic subgroups [33]. Moreoever, comparison to speed-record implementations in the literature is immaterial; our point here is to compare the price of a pairing on a subgroup-secure curve to the price of a pairing on one of the popular curves used in the literature, using the same implementation in order to get a fair performance ratio. The pairing functions use the NAF representation of the loop parameter u_0 for the Miller loop as well as the final exponentiation. The implementation computes the runtime for pairings on the subgroup-secure curves by only changing the value for u_0 in the Miller loop and final exponentiation in the implementation of the original curves, all other parameters remain the same. We note that the fastest implementations of field arithmetic for ordinary pairing-friendly curves, e.g. [2], do not take advantage of the NAF-weight of the prime p. The results therefore provide a gauge as to the relative slowdown one can expect in the pairing when employing a subgroup-secure curve, indicating that, in the worst case, a slowdown factor of 1.13 can be expected.

5 How to use subgroup-secure curves

In this section we discuss the implications of working with a subgroup-secure pairing-friendly curve and point out possible efficiency improvements. As we have seen in Section 4, there is a small performance penalty in the pairing algorithm when switching from the currently used “speed-record curves” to subgroup-secure curves, which is incurred by the increase in the NAF-weight of the parameter u_0. Note that this penalty only affects the pairing computation; it does not have any consequences for elliptic curve or finite field arithmetic in the groups themselves.

Table 2. Benchmarks of the optimal ate pairing on non-subgroup-secure pairing-friendly curves used previously compared to subgroup-secure curves (according to Definition 1) from the same family. Timings show the rounded average over 10000 measurements for which Turbo Boost and Hyperthreading were disabled. The experiments only reflect the difference in the NAF-weight of the parameter u_0 leading to an increased number of Miller steps in the Miller loop and multiplications in the final exponentiation. All other parameters are kept the same.

sec.	family	p	u	NAF-weight	where?	optimal ate	subgroup
level	k	(bits)	(bits)	of param. u_0		($\times 10^6$ clock cycles)	secure
128	BN	254	63	3	$[47, 2, 49, 53, 59]$	7.68	no
	12	254	63	6	Example 1	8.20	yes
approximate slowdown factor						**1.07**	
192	BLS	635	106	4	[15]	51.00	no
	12	635	107	6	Example 2	51.98	yes
approximate slowdown factor						**1.02**	
192	KSS	508	65	4	[1]	85.10	no
	18	508	65	9	Example 3	94.06	yes
approximate slowdown factor						**1.11**	
256	BLS	629	63	3	[53]	123.79	no
	24	629	63	8	Example 4	139.37	yes
approximate slowdown factor						**1.13**	

As we discussed earlier, an important subtlety that is rarely[8] factored into pairing-based protocol papers is the notion of (testing) subgroup membership [19, §2.2]. Naturally then, the cost for performing these checks is often not reflected in the pairing literature. When using a subgroup-secure curve, there is the potential to reduce the cost for these checks as hinted to in §3.2, and possibly to mitigate the performance penalty.

5.1 Reducing the cost of subgroup membership checks

We emphasize that we do not recommend skipping subgroup membership checks. What we do recommend, though, is that if such checks are in place to guarantee DLP security, then protocols should be examined to see if these checks can be replaced by less expensive measures, such as omitting costly scalar multiplications in the presence of a subgroup-secure curve. Next, we discuss the different possibilities.

For the group $\mathcal{G}_1$, the index h_1 of $\mathbb{G}_1$ is typically much smaller than n, which means that we cannot select the parameters to avoid prime factors smaller than n in $|\mathcal{G}_1|$. Therefore, one must carry out either a scalar multiplication by n to check for the correct order or by the cofactor h_1 to force points to have the right order. Let $\#E(\mathbb{F}_p) = h_1 \cdot n$ for some *cofactor* h_1 and recall that $\log_2 h_1 \ll \log_2 n$ for all of the above curve families. Thus, for a random point $P \in E(\mathbb{F}_p)$, it is faster to compute $R = [h_1]P$ to guarantee that $R \in \mathbb{G}_1$ than it is to check

[8] Menezes and Chatterjee recently pointed out another interesting example of this [18].

whether $[n]P = \mathcal{O}$ and was in $\mathbb{G}_1$ to begin with. However, this solution requires the protocol to allow the point R to replace the orginal point P, and this might require slight changes to the protocol; for example, it may require more than one party to perform the same scalar multiplication by h_1 such that it would have been less expensive (overall) for a single party to check that $[n]P = \mathcal{O}$.

In the group $\mathcal{G}_2$, the picture is different. Let $\#E'(\mathbb{F}_{p^{k/6}}) = h_2 \cdot n$ for some cofactor h_2, and recall from §3.2 that $h_2 > n$ for the families in this paper. In this case, guaranteeing that a point is in the order n subgroup $\mathbb{G}_2$ through a naive cofactor multiplication by h_2 seems to be at least as costly as checking that a point was in $\mathbb{G}_2$ to begin with; in particular, for the $k = 18$ and $k = 24$ families above, the bit length of h_2 is around 3 and 4 times that of n, respectively. However, the work by Scott et al. [55] and improvements by Fuentes-Castañeda et al. [26] show that cofactor multiplication with h_2 in $\mathcal{G}_2$ is significantly faster than multiplication by the group order n. As in $\mathcal{G}_1$, for a curve that is not subgroup-secure and if the protocol allows, it is thus cheaper to move a point $Q' \in E'(\mathbb{F}_{p^{k/6}})$ to $\mathbb{G}_2$ by computing $[h_2]Q'$ than it is to check the condition $[n]Q' = \mathcal{O}$. On the other hand, if the curve is subgroup-secure, one could check the curve equation to ensure that $Q' \in E'$ and omit either check at no risk of compromising the DLP security. However, not every protocol might allow working with points of order different than n. Thus the application of this optimization needs to be evaluated in every specific protocol. For example, the pairing implementation might not be bilinear when applied to points of order other than n, or the subgroup membership check may be in place for a reason different than discrete log security.

In the context of the Tate pairing, Scott [52, §4.4] pointed out that during the pairing computation, one can check whether the first input $P \in \mathcal{G}_1$ to the pairing function actually has order n, i.e. whether it is in $\mathbb{G}_1$. This is possible because the Miller loop in the Tate pairing inherently computes $[n]P$ alongside the pairing value, so there is no additional effort required to assert that $[n]P = \mathcal{O}$. However, when using optimal pairings [58], this is not true anymore. Due to the shortening of the Miller loop and the swapping of the input groups, optimal pairings only compute $[\lambda]Q'$ for λ much smaller than n, and for $Q' \in \mathcal{G}_2$. The trick outlined by Scott can therefore only help to save part of the exponentiation $[n]Q'$.

Elliptic curve scalar multiplications in both $\mathcal{G}_1$ and $\mathcal{G}_2$ can benefit from GLV/GLS decompositions [32, 31, 28]. In $\mathcal{G}_1$, one can use precomputed 2-dimensional GLV decompositions to speed up the scalar multiplications by h_1 and n. In $\mathcal{G}_2$, one can use even higher-dimensional GLV+GLS decompositions of the scalar n. In both cases, since n and h_1 are fixed system parameters, their decomposition can be computed offline. Moreover, these fixed multiplications are not by secret scalars and therefore need not be implemented in *constant time*.

Finally, the index of $\mathbb{G}_T$ in $\mathcal{G}_T = G_{\Phi_k(p)}$ is $h_T = \Phi_k(p)/n$, which is at least three times larger than n for the families in this paper. Thus, for a subgroup-secure curve, h_T is prime (up to possibly small factors given by the polynomial parameterization) and a subgroup membership test for $\mathbb{G}_T$ may be replaceable by a cheap membership test for $\mathcal{G}_T$ (see §5.2 below). Again, this is contingent on the ability of the protocol to allow $\mathcal{G}_T$-elements of order other than n. If membership tests can not be avoided, then the fixed exponentiation by n can take advantage of several techniques that accelerate arithmetic in the cyclotomic subgroup $\mathcal{G}_T = G_{\Phi_k(p)}$; these include cyclotomic squarings [33], exponent decompositions [31], and trace-based methods (cf. [54, §8.1]).

5.2 Checking membership in $\mathcal{G}_T$

We elaborate on Scott's observation [54, §8.3] concerning the ease of checking membership in $\mathcal{G}_T = G_{\Phi_k(p)}$. For the $k = 12$ BN and BLS families, checking that $g \in \mathcal{G}_T$ amounts to asserting $g^{p^4-p^2+1} = 1$, i.e. asserting that $g^{p^4} \cdot g = g^{p^2}$. Here the required Frobenius operations are a small cost compared to the multiplication that is needed, so this check essentially costs one multiplication in $\mathbb{F}_{p^{12}}$. Similarly, the tests for the $k = 18$ KSS and $k = 24$ BLS families check that $g^{p^6} \cdot g = g^{p^3}$ and $g^{p^8} \cdot g = g^{p^4}$ respectively, which also cost around one extension field multiplication.

The reason we take $\mathcal{G}_T$ to be the subgroup of order $\Phi_k(p)$, rather than the full multiplicative group $\mathbb{F}_{p^k}^{\times}$, is because it is extremely difficult to achieve subgroup security in $\mathbb{F}_{p^k}^{\times}$. As Scott points out when $k = 12$, the number of elements in $\mathbb{F}_{p^{12}}^{\times}$ factors as $p^{12}-1 = (p-1)\cdot(p^2+1)\cdot(p^2+p+1)\cdot(p^2-p+1)\cdot(p+1)\cdot((p^4-p^2+1)/n)\cdot n$, so here there are 6 factors (excluding n) that we would need to be almost prime if we were to deem $\mathbb{F}_{p^k}^{\times}$ as subgroup-secure. Even if it were possible to find a u_0 value such that these 6 factors were almost prime, it would certainly no longer have a sparse NAF representation, and the resulting loss in pairing efficiency would be drastic. On the other hand, taking $\mathcal{G}_T = G_{\Phi_k(p)}$ means that we can search for only one additional factor (i.e. $(p^4 - p^2 + 1)/n)$) being almost prime, meaning that sparse u_0 values (and therefore state-of-the-art performance numbers) are still possible and the cost of asserting membership in $\mathcal{G}_T$ remains negligible.

Acknowledgements

We are grateful to Melissa Chase and Greg Zaverucha for their interest in this work, and for their help pointing out implications for pairing-based protocols. We also thank Francisco Rodríguez-Henríquez for his suggestions to improve the paper.

References

1. Diego F. Aranha, Laura Fuentes-Castañeda, Edward Knapp, Alfred Menezes, and Francisco Rodríguez-Henríquez. Implementing pairings at the 192-bit security level. In Michel Abdalla and Tanja Lange, editors, *Pairing*, volume 7708 of *Lecture Notes in Computer Science*, pages 177–195. Springer, 2012.
2. Diego F. Aranha, Koray Karabina, Patrick Longa, Catherine H. Gebotys, and Julio López. Faster explicit formulas for computing pairings over ordinary curves. In Kenneth G. Paterson, editor, *EUROCRYPT*, volume 6632 of *Lecture Notes in Computer Science*, pages 48–68. Springer, 2011.

3. Razvan Barbulescu, Pierrick Gaudry, Antoine Joux, and Emmanuel Thomé. A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT*, volume 8441 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2014.
4. Paulo S. L. M. Barreto, Ben Lynn, and Michael Scott. Constructing elliptic curves with prescribed embedding degrees. In Stelvio Cimato, Clemente Galdi, and Giuseppe Persiano, editors, *SCN*, volume 2576 of *Lecture Notes in Computer Science*, pages 257–267. Springer, 2002.
5. Paulo S. L. M. Barreto, Ben Lynn, and Michael Scott. Efficient implementation of pairing-based cryptosystems. *J. Cryptology*, 17(4):321–334, 2004.
6. Paulo S. L. M. Barreto and Michael Naehrig. Pairing-friendly curves of prime order. In *Selected Areas in Cryptography – SAC'2005*, volume 3897 of *Lecture Notes in Computer Science*, pages 319–331. Springer, 2006.
7. Naomi Benger and Michael Scott. Constructing tower extensions of finite fields for implementation of pairing-based cryptography. In M. Anwar Hasan and Tor Helleseth, editors, *Arithmetic of Finite Fields, Third International Workshop, WAIFI 2010, Istanbul, Turkey, June 27-30, 2010. Proceedings*, volume 6087 of *Lecture Notes in Computer Science*, pages 180–195. Springer, 2010.
8. Daniel J. Bernstein. Curve25519: New Diffie-Hellman speed records. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *Public Key Cryptography - PKC 2006, 9th International Conference on Theory and Practice of Public-Key Cryptography, New York, NY, USA, April 24-26, 2006, Proceedings*, volume 3958 of *Lecture Notes in Computer Science*, pages 207–228. Springer, 2006.
9. Daniel J. Bernstein and Tanja Lange. Explicit-formulas database. `http://www.hyperelliptic.org/EFD`.
10. Ingrid Biehl, Bernd Meyer, and Volker Müller. Differential fault attacks on elliptic curve cryptosystems. In Mihir Bellare, editor, *Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2000, Proceedings*, volume 1880 of *Lecture Notes in Computer Science*, pages 131–146. Springer, 2000.
11. Ian F Blake, Gadiel Seroussi, and Nigel Smart. *Elliptic curves in cryptography*, volume 265. Cambridge university press, 1999.
12. Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the Gap-Diffie-Hellman-group signature scheme. In Yvo Desmedt, editor, *Public Key Cryptography - PKC 2003, 6th International Workshop on Theory and Practice in Public Key Cryptography, Miami, FL, USA, January 6-8, 2003, Proceedings*, volume 2567 of *Lecture Notes in Computer Science*, pages 31–46. Springer, 2003.
13. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. In *Advances in Cryptology – Asiacrypt 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532. Springer, 2002.
14. Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Kilian [38], pages 213–229.
15. Joppe W. Bos, Craig Costello, and Michael Naehrig. Exponentiating in pairing groups. In Tanja Lange, Kristin E. Lauter, and Petr Lisonek, editors, *Selected Areas in Cryptography*, volume 8282 of *Lecture Notes in Computer Science*, pages 438–455. Springer, 2013.
16. Wieb Bosma, John Cannon, and Catherine Playoust. The Magma algebra system. I. The user language. *J. Symbolic Comput.*, 24(3-4):235–265, 1997. Computational algebra and number theory (London, 1993).

17. Sanjit Chatterjee, Darrel Hankerson, Edward Knapp, and Alfred Menezes. Comparing two pairing-based aggregate signature schemes. *Des. Codes Cryptography*, 55(2-3):141–167, 2010.
18. Sanjit Chatterjee and Alfred Menezes. Type 2 structure-preserving signature schemes revisited. Cryptology ePrint Archive, Report 2014/635, 2014. `http://eprint.iacr.org/`.
19. Liqun Chen, Zhaohui Cheng, and Nigel P. Smart. Identity-based key agreement protocols from pairings. *Int. J. Inf. Sec.*, 6(4):213–241, 2007.
20. Henri Cohen, Atsuko Miyaji, and Takatoshi Ono. Efficient elliptic curve exponentiation using mixed coordinates. In Kazuo Ohta and Dingyi Pei, editors, *ASIACRYPT*, volume 1514 of *Lecture Notes in Computer Science*, pages 51–65. Springer, 1998.
21. Craig Costello, Tanja Lange, and Michael Naehrig. Faster pairing computations on curves with high-degree twists. In Phong Q. Nguyen and David Pointcheval, editors, *Public Key Cryptography*, volume 6056 of *Lecture Notes in Computer Science*, pages 224–242. Springer, 2010.
22. Craig Costello, Kristin Lauter, and Michael Naehrig. Attractive subfamilies of BLS curves for implementing high-security pairings. In Daniel J. Bernstein and Sanjit Chatterjee, editors, *INDOCRYPT*, volume 7107 of *Lecture Notes in Computer Science*, pages 320–342. Springer, 2011.
23. Harold M Edwards. A normal form for elliptic curves. *Bulletin of the American Mathematical Society*, 44(3):393–422, 2007.
24. David Freeman. Constructing pairing-friendly elliptic curves with embedding degree 10. In Florian Hess, Sebastian Pauli, and Michael E. Pohst, editors, *Algorithmic Number Theory, 7th International Symposium, ANTS-VII, Berlin, Germany, July 23-28, 2006, Proceedings*, volume 4076 of *Lecture Notes in Computer Science*, pages 452–465. Springer, 2006.
25. David Freeman, Michael Scott, and Edlyn Teske. A taxonomy of pairing-friendly elliptic curves. *J. Cryptology*, 23(2):224–280, 2010.
26. Laura Fuentes-Castañeda, Edward Knapp, and Francisco Rodríguez-Henríquez. Faster hashing to G_2. In Ali Miri and Serge Vaudenay, editors, *Selected Areas in Cryptography - 18th International Workshop, SAC 2011, Toronto, ON, Canada, August 11-12, 2011, Revised Selected Papers*, volume 7118 of *Lecture Notes in Computer Science*, pages 412–430. Springer, 2011.
27. Steven D. Galbraith, Keith Harrison, and David Soldera. Implementing the Tate pairing. In Claus Fieker and David R. Kohel, editors, *Algorithmic Number Theory, 5th International Symposium, ANTS-V, Sydney, Australia, July 7-12, 2002, Proceedings*, volume 2369 of *Lecture Notes in Computer Science*, pages 324–337. Springer, 2002.
28. Steven D. Galbraith, Xibin Lin, and Michael Scott. Endomorphisms for faster elliptic curve cryptography on a large class of curves. *J. Cryptology*, 24(3):446–469, 2011.
29. Steven D. Galbraith and Kenneth G. Paterson, editors. *Pairing-Based Cryptography - Pairing 2008, Second International Conference, Egham, UK, September 1-3, 2008. Proceedings*, volume 5209 of *Lecture Notes in Computer Science*. Springer, 2008.
30. Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.
31. Steven D. Galbraith and Michael Scott. Exponentiation in pairing-friendly groups using homomorphisms. In Galbraith and Paterson [29], pages 211–224.

32. Robert P. Gallant, Robert J. Lambert, and Scott A. Vanstone. Faster point multiplication on elliptic curves with efficient endomorphisms. In Kilian [38], pages 190–200.

33. R. Granger and M. Scott. Faster squaring in the cyclotomic subgroup of sixth degree extensions. In P. Q. Nguyen and D. Pointcheval, editors, *Public Key Cryptography*, volume 6056 of *LNCS*, pages 209–223. Springer, 2010.

34. Florian Hess, Nigel P. Smart, and Frederik Vercauteren. The eta pairing revisited. *IEEE Transactions on Information Theory*, 52(10):4595–4602, 2006.

35. IEEE P1363 Working Group. *Standard Specifications for Public-Key Cryptography – IEEE Std 1363-2000*, 2000.

36. Antoine Joux. A one round protocol for tripartite Diffie-Hellman. *J. Cryptology*, 17(4):263–276, 2004.

37. Ezekiel J. Kachisa, Edward F. Schaefer, and Michael Scott. Constructing Brezing-Weng pairing-friendly elliptic curves using elements in the cyclotomic field. In Galbraith and Paterson [29], pages 126–135.

38. Joe Kilian, editor. *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*. Springer, 2001.

39. Hendrik W Lenstra Jr. Factoring integers with elliptic curves. *Annals of mathematics*, pages 649–673, 1987.

40. Ninghui Li, Wenliang Du, and Dan Boneh. Oblivious signature-based envelope. In Elizabeth Borowsky and Sergio Rajsbaum, editors, *Proceedings of the Twenty-Second ACM Symposium on Principles of Distributed Computing, PODC 2003, Boston, Massachusetts, USA, July 13-16, 2003*, pages 182–189. ACM, 2003.

41. Chae Hoon Lim and Pil Joong Lee. A key recovery attack on discrete log-based schemes using a prime order subgroupp. In Burton S. Kaliski Jr., editor, *CRYPTO*, volume 1294 of *Lecture Notes in Computer Science*, pages 249–263. Springer, 1997.

42. Alfred Menezes. Asymmetric pairings. Talk at ECC2009. Slides at `http://math.ucalgary.ca/ecc/files/ecc/u5/Menezes_ECC2009.pdf`.

43. Atsuko Miyaji, Masaki Nakabayashi, and Shunzou Takano. New explicit conditions of elliptic curve traces for FR-reduction. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 84(5):1234–1243, 2001.

44. Peter L Montgomery. Speeding the Pollard and elliptic curve methods of factorization. *Mathematics of computation*, 48(177):243–264, 1987.

45. M. Naehrig. *Constructive and computational aspects of cryptographic pairings*. PhD thesis, Eindhoven University of Technology, May 2009.

46. Michael Naehrig, Ruben Niederhagen, and Peter Schwabe. New software speed records for cryptographic pairings. In Michel Abdalla and Paulo S. L. M. Barreto, editors, *LATINCRYPT*, volume 6212 of *Lecture Notes in Computer Science*, pages 109–123. Springer, 2010.

47. Yasuyuki Nogami, Masataka Akane, Yumi Sakemi, Hidehiro Katou, and Yoshitaka Morikawa. Integer variable chi-based ate pairing. In Galbraith and Paterson [29], pages 178–191.

48. Dan Page, Nigel P. Smart, and Frederik Vercauteren. A comparison of MNT curves and supersingular curves. *IACR Cryptology ePrint Archive*, 2004:165, 2004.

49. Geovandro C. C. F. Pereira, Marcos A. Simplício Jr., Michael Naehrig, and Paulo S. L. M. Barreto. A family of implementation-friendly BN elliptic curves. *Journal of Systems and Software*, 84(8):1319–1326, 2011.

50. Stephen C. Pohlig and Martin E. Hellman. An improved algorithm for computing logarithms over GF(p) and its cryptographic significance. *Information Theory, IEEE Transactions on*, 24(1):106–110, 1978.
51. Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. Cryptosystems based on pairing. In *The 2000 Symposium on Cryptography and Information Security, Okinawa, Japan*, pages 135–148, 2000.
52. Michael Scott. Computing the Tate pairing. In Alfred Menezes, editor, *Topics in Cryptology – CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 293–304. Springer, 2005.
53. Michael Scott. On the efficient implementation of pairing-based protocols. In Liqun Chen, editor, *IMA Int. Conf.*, volume 7089 of *Lecture Notes in Computer Science*, pages 296–308. Springer, 2011.
54. Michael Scott. Unbalancing pairing-based key exchange protocols. Cryptology ePrint Archive, Report 2013/688, 2013. `http://eprint.iacr.org/2013/688`.
55. Michael Scott, Naomi Benger, Manuel Charlemagne, Luis J. Dominguez Perez, and Ezekiel J. Kachisa. Fast hashing to G_2 on pairing-friendly curves. In Hovav Shacham and Brent Waters, editors, *Pairing-Based Cryptography - Pairing 2009, Third International Conference, Palo Alto, CA, USA, August 12-14, 2009, Proceedings*, volume 5671 of *Lecture Notes in Computer Science*, pages 102–113. Springer, 2009.
56. Michael Scott, Naomi Benger, Manuel Charlemagne, Luis J. Dominguez Perez, and Ezekiel J. Kachisa. On the final exponentiation for calculating pairings on ordinary elliptic curves. In *Pairing-Based Cryptography - Pairing 2009, Third International Conference, Palo Alto, CA, USA, August 12-14, 2009, Proceedings*, pages 78–88, 2009.
57. Serge Vaudenay. Hidden collisions on DSS. In Neal Koblitz, editor, *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, volume 1109 of *Lecture Notes in Computer Science*, pages 83–88. Springer, 1996.
58. Frederik Vercauteren. Optimal pairings. *IEEE Transactions on Information Theory*, 56(1):455–461, 2010.
59. Eric Zavattoni, Luis J. Dominguez Perez, Shigeo Mitsunari, Ana H. Sánchez-Ramírez, Tadanori Teruya, and Francisco Rodríguez-Henríquez. Software implementation of an attribute-based encryption scheme, 2015.

A Twist security

In this appendix, we briefly look at the notion of twist security, since it bears resemblance to the notion of subgroup security described in this work; we discuss the similarities and differences between the two in §A.3. Twist security has previously only been considered in the context of elliptic curve cryptography (ECC); we give a brief overview in §A.1. In §A.2 we put twist security in the context of pairing-based cryptography (PBC), where we discuss why it may or may not be relevant, why it is difficult to achieve in practice, and an alternative countermeasure.

A.1 Twist security in ECC

Bernstein [8] proposed the notion of twist security as a means to allow certain checks to be omitted during cryptographic scalar multiplications using the Montgomery ladder [44]. On elliptic curves of the form $E/K\colon By^2 = x^3 + Ax^2 + x$, the Montgomery ladder can efficiently compute scalar multiplications using only the x-coordinates of points. If $x(P)$ denotes the x-coordinate of a point $P \in E$, then the Montgomery ladder is essentially a function f that computes $x([k]P) \leftarrow f(x(P), k, A)$ for some scalar $k \in \mathbb{Z}$. The key point is that, so long as A is fixed, the function f correctly computes scalar multiplications independently of the curve constant B. However, there are only two isomorphism classes that can be obtained by varying B, depending on whether or not B is a square in K; these two classes correspond to the *quadratic twists* – E and E' – of one another. Since all $x' \in K$ correspond to the x-coordinate of a point lying on either[9] twist, Bernstein's solution is to ensure that E and E' are both cryptographically strong. In the context of elliptic curve Diffie-Hellman, this allows one to omit the check to which twist any particular $x' \in K$ corresponds to, and to successfully establish a secure shared secret regardless [8].

A.2 Twist security in PBC

Twist security only offers a concrete and practical advantage when the possibility of x-coordinate-only arithmetic is available, for if one has access to both the x-and-y coordinates, then checking curve pertinence is a negligible computation. In this regard there are then two main reasons why twist security is not likely to be as relevant in the context of PBC. Firstly, the most popular constructions do not have the cofactor 4 that is required to facilitate the Montgomery model (cf. [15, Table 2]), meaning that x-only arithmetic is rarely an option; for example, BN curves can never have a Montgomery representation[10]. And secondly, even if a Montgomery model is an option, pairing-friendly curves typically facilitate scalar decompositions [31], those of which are best performed via multiexponentiations that use both coordinates.

Nevertheless, since having a curve with a strong quadratic twist does not necessarily come at a price, one might consider employing this property in the context of PBC anyway[11]. Moreover, Lemma 1 shows that all of the curves in this paper have six twists over the ground field (rather than just two), so to be on the very safe side, one might try to find instances in which all six of the twists have (almost-)prime order. In fact, in the context of fault attacks, which can even pose a threat in the presence of point validation checks [10], protecting all six twists is a desirable property. To wit, we point out that just like the Montgomery ladder function f in §A.1 that did not distinguish between twists, typical scalar multiplication routines on the pairing-friendly curves in this paper will also work identically for all six twists. This is because the standard formulas for scalar multiplication are independent of the constant b, so for any

[9] There actually are a few points lying on both, e.g. the point $(0, 0)$, but this is unimportant here.

[10] For $E(\mathbb{F}_p)$, $n(u_0) \not\equiv 0 \bmod 4$ is obvious, and the same argument for $E'(\mathbb{F}_{p^2})$ follows from Proposition 1.

[11] As an aside, we note that the BN curve used to fool Alice in Section 1 was twist-secure.

pair $(\tilde{x}, \tilde{y}) \in \mathbb{F}_p^2$, and for a general scalar $k \in \mathbb{Z}$, the scalar multiplication routine will correctly compute the multiple $[k]P$ of the point $P = (\tilde{x}, \tilde{y})$ on the curve $\tilde{E}/\mathbb{F}_p \colon y^2 = x^3 + \tilde{b}$ with $\tilde{b} = \tilde{y}^2 - \tilde{x}^3 \in \mathbb{F}_p$. Since there are only six possible group orders for $\tilde{E}/\mathbb{F}_p \colon y^2 = x^3 + \tilde{b}$ as $\tilde{b}$ ranges over $[0, p)$, a fault attack that tries to prey on the correctness of the scalar multiplication routine for weak curves could be thwarted completely if all six group orders were strong.

Unfortunately, for all of the families in this paper, the parameterized versions of the six possible group orders in Lemma 1 reveal that at least one of the six twists will always have a weak group order. Even if five of the six twists are cryptographically strong, a sophisticated fault attack [10] has a good chance of producing an altered point with coordinates on the weak twist, and therefore a good chance of success.

Of course, the fault attack would have to be sophisticated indeed, if it were able to get around point pertinence checks at both the beginning and end of a scalar multiplication routine. Nevertheless, such an attack is not an impossibility, so we now discuss one potential countermeasure. We propose employing explicit formulas that *do* distinguish between the six twists. Recall from above that the affine schoolbook formulas for arithmetic on $E \colon y^2 = x^3 + b$ are independent of the constant b, and therefore of any particular twist. Thus, it makes sense that the fastest projective versions of these formulas are also independent of b. These formulas use Jacobian coordinates [9] and require[12] $2\mathbf{M}+5\mathbf{S}$ for point doublings, $11\mathbf{M}+5\mathbf{S}$ for projective additions and $7\mathbf{M}+4\mathbf{S}$ for a projective-and-affine (a.k.a. "mixed") addition. On the other hand, the projective formulas for arithmetic on E in homogenous coordinates do incorporate b; these require $3\mathbf{M} + 5\mathbf{S}$ for point doublings [21, §5], $12\mathbf{M} + 2\mathbf{S}$ for projective additions and $9\mathbf{M} + 2\mathbf{S}$ for mixed additions [20] (see also [9]). In this case point doublings are $1\mathbf{M}$ slower than in Jacobian coordinates, but the performance penalty here will be very minor given the competitive homogenous addition formulas, and the higher density of such additions in scalar multiplications exploiting decompositions – see [31, 15]. The incentive is that this set of formulas only computes scalar multiplications correctly for the particular curve they are intended for. This means that any fault attack that alters the input point $(x, y) \in E$ to $(\tilde{x}, \tilde{y}) \in \tilde{E}$ will almost certainly be returned a point that is neither on E or $\tilde{E}$, and even in the case where the returned point is on a twist isomorphic to E or $\tilde{E}$, it will not correspond to a multiplication by the secret scalar.

A.3 Subgroup security vs. twist security

In this section we briefly compare the notion of twist security in the context of ECC and that of subgroup security in the context of PBC. Indeed, while neither of these properties are absolutely necessary, they are both intended to maintain DLP security in certain scenarios when checks are omitted for the sake of efficiency. In the case of x-coordinate-only ECC, twist security comes at no price (a well-chosen twist-secure curve introduces no overhead), while in the case

[12] Here $\mathbf{M}$ and $\mathbf{S}$ denote a field multiplication and field squaring respectively.

of PBC, achieving subgroup security introduces a small but noticeable overhead in the pairing (see Section 4). On the other hand, the potential savings offered by subgroup-secure curves are far greater in the context of PBC; here we can possibly save large elliptic curve or finite field group exponentiations, while twist security for Montgomery curves saves a relatively inexpensive Legendre symbol computation. Just like twist security in ECC, subgroup security in PBC removes possible points of failure in practice. We believe that the minor overhead in the pairing is a small price to pay for the assurance that all elements which are asserted to be in $E'(\mathbb{F}_{p^{k/d}})$ or $G_{\Phi(k)(p)}$ are guaranteed to have large prime order.