Batch subgroup membership testing on pairing-friendly curves

Dimitri Koshelev[1⋆], Youssef El Housni[2], and Georgios Fotiadis[3]

[1] University of Lleida, Department of Mathematics, Catalonia, Spain
dimitri.koshelev@gmail.com
[2] Linea, New York, United States
youssef.elhousni@consensys.net
[3] SnT, University of Luxembourg, Esch-sur-Alzette, Luxembourg
gfotiadis.crypto@gmail.com

Abstract. A major challenge in elliptic curve cryptosystems consists in mitigating efficiently the small-subgroup attack. This paper explores batch subgroup membership testing (SMT) on pairing-friendly curves, particularly for the Barreto–Lynn–Scott family of embedding degree 12 (BLS12) due to its critical role in modern pairing-based cryptography. Our research introduces a novel two-step procedure for batch SMT to rapidly verify multiple points at once, cleverly combining the already existing tests based on the Tate pairing and a non-trivial curve endomorphism. We clarify why the invented technique is significantly faster (despite a negligible error probability) than testing each point individually. Moreover, it is applicable to prominent curves like BLS12-381 and BLS12-377 being frequently employed in zero-knowledge applications. Nonetheless, to further enhance the speed (or reduce the error probability) of the proposed batch point validation, we have generated two new BLS12 curves that are specifically optimized for this purpose. We also provide an open-source high-speed software implementation in Go, showcasing explicitly significant performance improvements achieved by our work.

Keywords: multi-scalar multiplication · pairing-friendly curves · power residue symbols · subgroup membership testing · Tate pairing

1 Introduction

Pairing-based cryptography is a well-known type of ECC (elliptic curve cryptography), which is widely used in a variety of security mechanisms, starting with Boneh–Franklin's identity-based encryption scheme [13] and Joux's one-round tripartite key exchange [48]. All the three researchers were awarded in 2013 by the Gödel Prize for such a constructive view on pairings. In recent years, pairings have also become an important tool in achieving scalability and privacy

⋆ https://www.linkedin.com/in/dimitri-koshelev

of blockchain technology, among which aggregate digital signatures and ZK-SNARKs (zero knowledge - succinct non-interactive argument of knowledge). Despite the potential quantum threat, the given area of pre-quantum cryptography remains an integral part of the modern cryptographic landscape, since post-quantum analogs are mostly not so performant by the ratio speed/compactness. By the way, NIST also had closely looked at pairings (see the report [60]) until this authoritative organization eventually decided to refuse quantumly weak cryptographic solutions.

Throughout the paper, $E\colon y^2 = x^3 + ax + b$ will denote a pairing-friendly elliptic curve over a finite (usually prime) field $\mathbb{F}_q$ of large characteristic. For simplicity, let the infinity point $\mathcal{O} := (0 : 1 : 0)$ be the identity for the group law on E. By definition, the curve has two subgroups $\mathbb{G}_1$, $\mathbb{G}_2 \subset E(\mathbb{F}_{q^k})$ and a bilinear map $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T \subset \mathbb{F}_{q^k}^*$, where k is a moderate extension degree (so-called embedding degree). As always, k is taken minimal such that $\mathbb{G}_T \subset \mathbb{F}_{q^k}^*$ or, equivalently, $\#\mathbb{G}_T \mid \Phi_k(q)$, where Φ_k is the k-th cyclotomic polynomial. For our purposes, it is sufficient to assume that the pairing is of type III, the most popular choice in practice. This means in particular that all the three groups $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ are of prime order r. Besides, $\mathbb{G}_1 \subset E(\mathbb{F}_q)$ and $\mathbb{G}_2 \hookrightarrow E_2(\mathbb{F}_{q^d})$, where E_2 is a twist of E (of degree k/d) over a subfield $\mathbb{F}_{q^d} \subset \mathbb{F}_{q^k}$. For uniformity, put $\mathbb{H}_1 := E(\mathbb{F}_q)$, $\mathbb{H}_2 := E_2(\mathbb{F}_{q^d})$ and $\mathbb{H}_T := \mu_{\Phi_k(q)}$, the group of all $\Phi_k(q)$-roots of unity. Finally, $h_i := \#\mathbb{H}_i/r$ will stand for the cofactor of $\mathbb{G}_i \subset \mathbb{H}_i$ with $i \in \{1, 2, T\}$.

For a long time, it was believed that the most suitable curves for pairings for the classical 128-bit security level λ are *BN (Barreto–Naehrig) curves* [9] of j-invariant 0 (i.e., the coefficient $a = 0$) with $k = 12$ and $d = 2$. Among them, BN-254 [81] is probably the most famous within the blockchain community. The main feature of BN curves is the primality of $E(\mathbb{F}_q)$, that is, $h_1 = 1$. However, after remarkable improvements (initiated by Kim and Barbulescu [51]) of the number field sieve (NFS) in $\mathbb{F}_{q^k}^*$, BN curves lost several tens of security bits and now have ≈ 100 ones. A naive countermeasure fixing this problem consists in increasing essentially the field $\mathbb{F}_q$ (and so $\mathbb{F}_{q^k}$), most often to $\approx 384 = 6 \cdot 64$ bits. Such a magnitude yields λ being just a little smaller than 128, albeit its true value (also as λ for $\log_2(q) \approx 256$) is a subject of heated debates in the community such as [19,36,41,46]. Nonetheless, the given trade-off is commonly recognized to be satisfactory, as for the classical security level, one needs the non-linearly greater length $\log_2(q) \approx 448 = 7 \cdot 64$.

It is well-known that the cost of pairing computation is directly proportional to $\log_2(r)$. That is why even the slightest growth of $\mathbb{F}_q$ noticeably slows down pairings on BN curves for which $\log_2(r) \approx \log_2(q)$. As a consequence, these curves were principally replaced in the real world by *BLS12 (Barreto–Lynn–Scott) curves* [8]. They have the identical parameters $j = 0$, $k = 12$, $d = 2$ and $\log_2(q) \approx 384$, but the subgroup order is of the optimal length $\log_2(r) \approx 256$ bits, that is, $\log_2(h_1) \approx \log_2(r)/2 \approx 128$ bits (the other cofactors h_2, h_T also grow accordingly). In other words, BN curves have the value $\rho := \lceil \log_2(q) \rceil / \lceil \log_2(r) \rceil$ equal to 1, whereas BLS12 ones enjoy $\rho \approx 1.5$. By the way, a subtle, rigorous and

relatively recent analysis of NFS-type attacks on the BN and BLS12 families is contained in [40, Section 7.2, Tables 8, 10].

At the same time, the non-triviality of h_1 (as well as of h_2, h_T) is a big drawback of BLS12 curves in comparison with BN ones. Indeed, the question arises of how to successfully combat the *small-subgroup attack* [57], one of fault attacks that regularly bothers developers of cryptosystems (see justifications in [4,69,75,76,79]). In a nutshell, the given roguery is nothing but sending to a honest receiver a point from $\mathbb{H}_i \backslash \mathbb{G}_i$. As a rule, *subgroup membership testing* (SMT for laconicity) is not a very cheap primitive, i.e., every its call takes some non-negligible time. The situation is aggravated by the fact that it is often necessary to check membership to $\mathbb{G}_i$ for thousands or even millions of points from $\mathbb{H}_i$ at a time. In the literature there are several ways of SMT to process each input point separately. But surprisingly, there has been until now no (detailed) discussion of *batch subgroup checks*, except for a few too short Internet posts [2,37] without clarifications. Moreover, some of them are even skeptic about usefulness of SMT. Looking ahead, the objective of the present article is to address this topic at the scientific level.

It is emphasized that some BN curves are still maintained for the sake of compatibility with older software. So, testing membership to their subgroups $\mathbb{G}_2$, $\mathbb{G}_T$ partially remains an adequate task. It should also be noted that the cost of $\mathbb{G}_2$ membership testing may be almost free if it is allowed to be executed during pairing computation. Nonetheless, let's continue with the BLS12 family to be definite. As an illustration, we proceed only with the subgroup $\mathbb{G}_1$, although the below theory is readily extended mutatis mutandis to $\mathbb{G}_2$, $\mathbb{G}_T$. Moreover, $\log_2(h_2)$, $\log_2(h_T) \gtrsim \log_2(r)$ and h_2, h_T are irreducible as univariate $\mathbb{Q}$-polynomials equally for the BN and BLS12 families. Therefore, it is feasible to generate (or borrow from [6,7,72,73]) *subgroup-secure curves* also known as $\mathbb{G}_2$ and/or $\mathbb{G}_T$*-strong curves*. Here, strongness means that h_2 and/or h_T do not have divisors smaller than r, which prevents from the small-subgroup attack even if a receiver deals with fraudulent input points. However, the absence of subgroup checks in advanced cryptographic protocols may lead at least to the loss of security proofs or to specific vulnerabilities [78]. Thus, it seems wrong to count subgroup-secure curves absolutely trustworthy.

It is worth remembering that for the BLS12 family, the cofactor $h_1 = 3e_1^2$ (with $e_1 \in \mathbb{N}$) and $E[e_1] \subset E(\mathbb{F}_q)$. Furthermore, $e := 3e_1$ divides $q - 1$. Among other things, there is always in $\mathbb{F}_q$ a primitive cubic root ω of unity. In combination with the primality of q, this is equivalent to the fact that BLS12 curves (as well as BN ones) are ordinary, i.e., non-supersingular. Unless $E[r] \subset E(\mathbb{F}_q)$ and so $k = 1$, there exists a universal (but quite slow) SMT that consists in comparing $rP_i = \mathcal{O}$ given N points $P_i \in E(\mathbb{F}_q)$. We essentially[4] have at our disposal only two state-of-the-art methods of subgroup checks, which are fundamentally

[4] The Pornin method [66] via sequential point halvings in $E(\mathbb{F}_q)$ should be put into a separate category. But among real-world curves, it wins exclusively (to the authors' knowledge) in the case of Curve25519 [26, Section 3.2.2.1]. This effect occurs, because neither of the two other SMT is applicable to this curve (cf. [53, Problem 1]). For

different from each other. The first (whose modification is also relevant for $\mathbb{G}_2$) is grounded on a shorter scalar multiplication in $E(\mathbb{F}_q)$ due to the non-trivial $\mathbb{F}_q$-automorphism $[\omega]\colon (x,y) \mapsto (\omega x, y)$. Meanwhile, the second (not existing de facto for $\mathbb{G}_2$) leverages the reduced *Tate pairing* $t_e\colon E(\mathbb{F}_q)[e] \times E(\mathbb{F}_q) \to \mu_e \subset \mathbb{F}_q^*$, because SMT in a finite field is clearly easier than on an elliptic curve of the same magnitude.

The pairing-free test was initially suggested by Bowe [20] for a concrete BLS12 curve (namely BLS12-381 [18]) and then discussed by Scott [74] for the whole BLS12 family. An extension of this technique to other pairing-friendly families is carried out in [30,33]. In turn, the test via the Tate pairing was proposed by Koshelev [52,53] originally for many plain, i.e., non-pairing-friendly curves of cofactors $\leqslant 16$. The point is that the *power residue symbol* $(\frac{x}{q})_i := x^{(q-1)/i}$ for $x \in \mathbb{F}_q^*$ enjoys fast *Euclidean-type algorithms* [23,24,49] when $i \leqslant 16$ (and $i \mid q-1$ of course). In the follow-up paper [29], Koshelev et al. demonstrated the advantage of the given approach for a series of pairing-friendly curves (two of which are from [28]). They have the smaller value $\rho < 1.5$ than BLS12 curves at the price of the larger embedding degree $k > 12$ to respect a desired security level. For the BLS12 family, the two mentioned tests are comparable in speed: $\rho \approx 1.5$ is a kind of the borderline between them.

In this article, we will not elaborate on the Koshelev(-type) subgroup check to avoid redundant complications, as it is relatively new and so insufficiently realized in the ECC community. Instead, we will showcase the supremacy (for the pretty large N) of the batch strategy exclusively in the case of the Bowe–Scott subgroup check due to its wide adaptation in cryptographic libraries. Resorting to them, we are in particular able to promptly obtain benchmarks for both batch and non-batch SMT. Nevertheless, the batch Koshelev(-type) method likewise should outperform its non-batch analog. Finally, it is worth bearing in mind that the (non-reduced) Tate pairing may be potentially computed even faster via a novel technique under the name *cubical arithmetic* [58,64,68], at least if E admits the Montgomery $\mathbb{F}_q$-form. All this is a promising subject of further research.

Another naive countermeasure against the small-subgroup attack is just the multiplication by h_1 (or less costly by e). It is especially tempting when the cofactor is modest as for plain curves. However, this is not panacea. First, the given trick often nullifies security proofs of complicated cryptographic schemes or directly leads to unambiguous bugs. Even for more elementary protocols for which clearing h_1 seems harmless, regulators (including NIST [26, Appendix D.1]) explicitly ask to fairly validate input points P_i. Second, the resulting points $h_1 P_i \neq P_i$ (except for $P_i \in E[h_1 - 1]$), hence all the parties of a protocol need to take into account such a modification. It sounds realistic that the total overhead may thereby become much greater, as the additional load may fall not only on a small number of powerful receivers of P_i, but also on numerous low-resource senders. Third, the cofactor multiplication cannot be seamlessly incorporated into a cryptosystem at any stage of its functioning and by any

other practical curves, the Pornin method is much more expensive than the Bowe–Scott(-type) or Koshelev(-type) ones.

party independently of others. So, implementing a more optimized authentic SMT is actually the best solution in all respects.

1.1 New contribution

It is reasonable to diminish somehow the number of SMT calls by treating simultaneously N received points $P_i \in E(\mathbb{F}_q)$. In a nutshell, aggregating P_i into the n sums $S_j := \sum_{i=1}^{N} c_{i,j} P_i$ with random numerical coefficients $c_{i,j}$ underlies the article results. Afterwards, it is suggested to launch the subgroup checks $S_j \in_? \mathbb{G}_1$. Whenever $n \ll N$, one can expect that the given approach is much more performant than the separate checks $P_i \in_? \mathbb{G}_1$. The authors do not pretend at all to the originality of this insight. There exists a well-known similar technique being used for *batch signature verification*. It was invented and thoroughly analyzed already in the previous century [10,22,44,61] in the context of classical signatures RSA and DSA. Batching is also inherent to more modern pairing-based signatures such as the mainstream BLS (Boneh–Lynn–Shacham) one [12,14,15,16].

Note that $\gcd(r, e) = 1$ and $E(\mathbb{F}_q) = \mathbb{G}_1 \times E(\mathbb{F}_q)[e]$, hence the points P_i are uniquely decomposed into the components $Q_i := (e^{-1} \bmod r)eP_i$ and $R_i := P_i - Q_i$ lying in $\mathbb{G}_1$ and $E(\mathbb{F}_q)[e]$, respectively. Inter alia, $P_i \in \mathbb{G}_1$ if and only if $P_i = Q_i$ or, equivalently, $R_i = \mathcal{O}$. Obviously, if all the points P_i belong to $\mathbb{G}_1$, then all the sums S_j also do. The converse statement is not true, because the components R_i may annihilate each other, that is, the equalities $S_j = \sum_{i=1}^{N} c_{i,j} Q_i$ imply nothing from the mathematical point of view. Since a malicious sender of P_i is able to prepare the small-subgroup attack at his discretion, he can try increasing the probability $\mathbb{P}$ of acceptance by a honest receiver of at least one poisoned point P_i. It is natural to assume that $0 \leqslant c_{i,j} < m \leqslant e$ for a certain $m \in \mathbb{N}$. Temporarily, put $n = 1$ and denote by p the smallest prime divisor of e, i.e., just 2 if e is even and 3 otherwise. The probability $\mathbb{P}$ evidently grows when all R_i lie in the same order-p subgroup (and some $R_i \neq \mathcal{O}$ of course). It is readily seen (cf. [10, Lemma 3.1], [37]) that $\mathbb{P} \leqslant 1/\min\{m, p\}$ and this upper bound does not seem to be (significantly) improvable. In particular, there is no any sense for the receiver to slow down computation of S_j by picking $m > p$.

The bound $\mathbb{P} \leqslant 1/3$ (not mention to $\mathbb{P} \leqslant 1/2$) is no doubts too big to rely on such an ingenuous batch SMT. Fortunately, one can mitigate the given trouble analogously as in the paper [52]. Let $\pi := 2^4 \cdot 3^2 \cdot 5 \cdot 7 \cdot 11 \cdot 13$ and $\pi' := \gcd(\pi, e)$. It is always possible to first employ the Koshelev test via the Tate pairing $t_{\pi'}$ with respect to the subgroup $\mathbb{G}'_1 := E(\mathbb{F}_q)[re']$, where $e' := e/\pi'$. Since the symbol $(\frac{\cdot}{q})_{\pi'}$ is "Euclidean-friendly", the bit complexity of each intermediate check $P_i \in_? \mathbb{G}'_1$ is only quadratic in $\log_2(q)$. Consequently, the given operation is an order of magnitude faster than others involved in (batch) SMT. So, we have the right to neglect the corresponding running time, which is adequate at least for scientific papers. To this moment, we can efficiently (although separately) detect the fraudulent points $P_i \notin \mathbb{G}'_1 \supset \mathbb{G}_1$. For the remaining ones, let's redefine the value p, making it often (much) greater than 3. In other words, it now becomes

reasonable to augment the coefficients $c_{i,j}$. By abuse of notation, let p be the minimal prime divisor of e'. Since $\log_2(\pi) \approx 19.5$, it is excluded in practice the scenario when p is not correctly defined, i.e., $e' = 1$. Indeed, this happens only if $e \mid \pi$, but $\log_2(e) \approx 64$ for the quasi-classical λ.

To provide a desired level μ of the error probability $\mathbb{P}$, that is, to guarantee the bound $\mathbb{P} \leqslant 1/2^\mu$, there is an alternative way workable even if $p = 2$. It is enough to repeat n times the described batch SMT. Since the coefficients $c_{i,j}$ are assumed to be independent for different j (and $m \leqslant p$ as we understood), $\mathbb{P} \leqslant 1/m^n$ and hence the probability rapidly converges to zero as $n \to \infty$. Specifically, $n = \lceil \mu / \log_2(m) \rceil$ rounds allow to satisfy the inequality $1/m^n \leqslant 1/2^\mu$. For instance, $n = \lceil \mu \rceil$ with $m = 2$, whereas $n = 1$ with $\mu \leqslant \log_2(m)$. The first extreme case inevitably occurs when $p = 2$, i.e., the number e is highly 2-adic, which basically means for us that the 2-adic valuation $\nu_2(e) \geqslant 5$. It is not a secret that this frequently takes place in the ZK realm, e.g., for the curve BLS12-377 [80] with $\nu_2(e) = 46$ or for Guillevic–Masson's ones [39, Section 4.4] with $\nu_2(e) \geqslant 25$. Thereby, the high 2-adicity of e is a downside in the context of batch SMT. Meanwhile, the second extreme case is optimal in n. It is obviously achieved on $m = \lceil 2^\mu \rceil$. Note that $\log_2(m) \leqslant \log_2(p) \leqslant \log_2(e_1) \approx 62.5$. When $n = 1$, the latter is consequently the maximal possible value of μ. It is not said anywhere (including the seminal sources on batch signature verification) that μ must be $\approx \lambda$. If desired, the given equilibrium can be nevertheless obtained with $n = 2$.

It is shown that the number of rounds n is reduced (with the same parameter μ) when the prime p becomes larger. Unfortunately, the latter is tightly attached to each BLS12 curve. Today's real-world ones were generated in the past without regard to simultaneous subgroup checks and hence to the size of p. In turn, it seems too painful to change curves that underlie already running multi-user cryptosystems. Thus, a by-product of the current paper is to motivate further studies of more performant implementations of the power residue symbols $(\frac{\cdot}{q})_i$, possibly with greater focus on $i \leqslant 16$. Lately, there has been significant progress [3,43,65] in speeding up the Legendre symbol $(\frac{\cdot}{q})_2$, fueled primarily by its utilization in hashing to elliptic curves and point decompression on them. However, higher residue symbols (with $i > 2$) remain outside the attention of the majority of developers, since existing cryptographic libraries support them much more rarely. Even if e_1 does not suffer from small divisors, the ECC community should pay attention at least to the cubic symbol $(\frac{\cdot}{q})_3$ to which the texts [5,31] are devoted. It is an inevitable barrier for the described batch SMT on BLS12 curves because of the structure of their cofactor $h_1 = 3e_1^2$. More generally, the given cubic analog of the Legendre symbol often occurs in various cryptographic primitives on curves of j-invariant 0.

The process of computing S_j is said to be *MSM (multi-scalar multiplication)*. It can be carried out an order of magnitude faster than finding individually each scalar multiplication $c_{i,j}P_i$ and then summing them. The literature on MSM is vast, but the latest important works on this topic are [27,35,42,47,54,67,83]. A new stage of research interest to accelerating MSM is again related to ZK

proofs for which the given primitive is typically a bottleneck. On the one hand, $\log_2(c_{i,j}) \leqslant \log_2(m) \lesssim 62.5$. On the other hand, the Bowe–Scott subgroup check for $\mathbb{G}_1$ requires two sequential scalar multiplications by the seed $z := 1 \pm e$. So, the total length is about $2 \cdot \log_2(e) \approx 128$ bits. In practice, the Hamming weight (of the non-adjacent form) of z is not very far from zero. But even if the coefficients $c_{i,j}$ are weighted considerably, (the shortest) addition chains for them are still not longer than those for z^2. Therefore, the individual tests $P_i \in_? \mathbb{G}_1$ at best have a similar cost as the scalar multiplications $c_{i,j}P_i$. That is why the batch SMT should be noticeably more efficient than its non-batch analog and it actually is as we will see later.

2 BLS12 curves

All the formulas and tables of this section are verified by means of the code [55] in the computer algebra system Magma. For the sake of completeness, let's recall the polynomial expressions in $z \in \mathbb{Z}$ of the main parameters of the BLS12 family:

$$\begin{aligned}
&e_1 = |z-1|/3, \qquad h_1 = 3e_1^2, \qquad r = z^4 - z^2 + 1, \qquad q = h_1 r + z,\\
&h_2 = (z^8 - 4z^7 + 5z^6 - 4z^4 + 6z^3 - 4z^2 - 4z + 13)/9,\\
&h_T = (z^{20} - 8z^{19} + 25z^{18} - 32z^{17} - 8z^{16} + 76z^{15} - 93z^{14} + 36z^{13} + 51z^{12} - 112z^{11}\\
&\quad + 86z^{10} - 16z^9 - 24z^8 + 84z^7 - 90z^6 + 28z^5 - 14z^4 - 38z^3 + 70z^2 - 14z + 73)/81.
\end{aligned}$$

The formulas of the first line are widespread in the literature. In turn, the formula for h_2 is contained in [25, Section 3.1] and that for h_T is immediately established, because $\Phi_{12}(q) = q^4 - q^2 + 1$.

We already know that cryptographers usually prefer the length $\log_2(|z|) \approx 64$. To be more precise, $\log_2(q) \approx 382.5$ in this case rather than ≈ 384. As always, the symbol $\approx$ does not have a strict sense. For concrete BLS12 curves, it would be fairer to say a more exact value of $\log_2(|z|)$, because even a minimal discrepancy in it leads to a larger one in the derived parameters. For instance, the curves BLS12-377 and BLS12-381 have as $\log_2(q)$ slightly smaller values than 382.5 according to their names, because $\log_2(|z|) \approx 63$ and ≈ 63.5, respectively. Nevertheless, 64 is a round number, hence it is taken in our abstract reasoning as the main approximation.

It is worth emphasizing that the inverse operation $-$ on elliptic curves is free, hence it is reasonable to consider the NAF (non-adjacent form) instead of the conventional binary form of the seed z. Hereafter, w will stand for the Hamming weight of this NAF. It should be as small as possible, because the (optimal) Ate pairing $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ on BLS12 curves forces to evaluate at $P \in \mathbb{G}_1$ the Miller function $f_{|z|,Q}$, where $Q \in \mathbb{G}_2$. At the same time, the bottleneck of the Bowe–Scott subgroup check is the scalar multiplication by z^2, i.e., twice by z. In this connection, minimizing w is compatible with versatile acceleration of a pairing-based cryptosystem.

Table 1 exhibits several BLS12 curves from existing cryptographic sources and their indicators primary for us. Some of these curves are even deployed in the real world, whereas the others are rather of theoretical interest. The symbol p_ℓ signifies a certain prime with bit length ℓ. The number of rounds n is meant with respect to the level $\mu = 60$ and $m = p$. This choice for μ, made also in [10, Section 2.1], [22, Section 4.1], will be clear in Section 2.2. For instance, $p = p_{14} = 10177$ if we are talking about the curve BLS12-381. Curiously, the gap between 11 and p_{14} is essential. Without the fast implementation (from [49]) of the symbol $(\frac{\cdot}{q})_{11}$, we get $n = 18$ and thereby the more expensive batch SMT. Last but not least, the row starting with BLS12-(377-383) is dedicated to a set of curves (such that $377 \lesssim \log_2(q) \lesssim 383$) for which there are universal answers in all the entries.

Curve	Reference	w	$e_1 = e/3$	$\pi'/3$	p	n	$\mathbb{G}_2$-strong	$\mathbb{G}_T$-strong
BLS12-377	[80]	7	$2^{46}\cdot 7\cdot 13\cdot p_9$	$2^4\cdot 7\cdot 13$	2	60	no	no
BLS12-380	[71, Section 6]	20	$5\cdot 7\cdot 23\cdot p_{22}\cdot p_{31}$	$5\cdot 7$	23	14		
BLS12-381	[18]	6	$11\cdot p_{14}\cdot p_{20}\cdot p_{26}$	11	p_{14}	5		
BLS12-383	[73, Section 2]	5	$89\cdot p_9\cdot p_{48}$	1	89	10		yes
BLS12-(377-383)	[39, Section 4.4]	$\leqslant 7$	highly 2-adic	0 mod 2^4	2	60		no
BLS12-461	[6, Section 7.5]	7	$3\cdot 5^4\cdot 7\cdot p_{62}$	$3\cdot 5\cdot 7$	5	26	yes	

Table 1: Remarkable BLS12 curves

2.1 High 2-adicity and embedded curves

For the sake of completeness, it was decided to include in the paper Table 2 of embedded curves $E_e/\mathbb{F}_r$ for some BLS12 ones from Table 1. In today's ZK protocols, such elliptic curves are not required to be pairing-friendly, although they still possess subgroups of large prime order r'. Therefore, they unchangeably have a modest cofactor h (including the trivial one $h = 1$) and so the batch SMT (or even SMT itself) for them is useless. Indeed, $r - 1$ is (mostly) highly 2-adic to benefit from the FFT (fast Fourier transform), whereas h is a small power of 2 (typically, $h \leqslant 8$) to enjoy cheaper elliptic curve forms. As a result, the Koshelev subgroup check is an ideal solution in the situation under consideration. Table 2 is thus secondary for the current investigation, but it transparently demonstrates that pairing-friendly curves come into play together with their

plain counterparts. Inter alia, this circumstance should be accounted for during generation of new BLS12 curves. Otherwise, they will have very restricted applications in the modern cryptographic landscape.

BLS12 curve	Embedded curve	Reference	$\nu_2^*(q)$	$\nu_2^*(r)$	$\nu_2^*(r')$	h	$-D$
BLS12-377	no name	[1]	46	47	1	4	$\approx 2^{252}$
BLS12-380		[71, Section 6]	1	38		1	3
BLS12-381	Jubjub	[34]		32		8	$\approx 2^{255}$
	Bandersnatch	[59]			5	4	8
	no name	[39, Section 3.3.1]			2	1	6673027
BLS12-(377-383)		[39, Section 4.4]	$25 \leqslant$			1, 4	3

Table 2: Remarkable embedded curves for the ZK-oriented BLS12 ones from Table 1

In order to save the table space, put $\nu_2^*(u) := \nu_2(u-1)$ for an arbitrary $u \in \mathbb{Z}$. Besides, $D < 0$ stands for the CM (complex multiplication) discriminant of an elliptic curve. The $\mathbb{G}_T$-strong curves BLS12-383 and BLS12-461 were proposed without regard to ZK proofs, hence apparently nobody has never tried to produce for them appropriate embedded curves. We also do not elaborate on this minor problem. If necessary, E_e of huge D can be found with an overwhelming probability by iterating the curve coefficients over $\mathbb{F}_r$ as was done for BLS12-377 and BLS12-381. Since embedded curves are plain, their efficiency does not suffer from the absence of higher-degree twists (when $D < -4$).

In addition, Koshelev–Sanso's fresh result [56] generalizes the famous GLV (Gallant–Lambert–Vanstone) decomposition for accelerating (multi-)scalar multiplication to a much wider class of CM discriminants than previously believed. Nonetheless, the GLV technique continues to be the most impressive on curves of $j = 0$ (along with even-order ones of $j = 1728$). Besides, it is better when E_e is of the prime order r', i.e., $h = 1$. In this scenario, there is one more elliptic curve $E'_e/\mathbb{F}_{r'}$ with which E_e constitutes a 2-cycle, that is, $\#E_e(\mathbb{F}_r) = r'$ and dually $\#E'_e(\mathbb{F}_{r'}) = r$. Otherwise, we can only count on non-closed (2-)chains and hence infinite recursive ZK-SNARKs starting with E_e are impossible. Sought embedded curves of moderate D are derived (if any) with the help of [59, Algorithm 1] or of more optimized [39, Algorithm 1]. As reflected in Table 2, such a search has already been accomplished for BLS12-381. Most likely, the cited algorithms yield positive outcomes similarly for BLS12-377.

Recall that $t = z + 1$ is the Frobenius trace for the BLS12 family. The valuation $\nu_2^*(r)$ is not low for all the table curves without exception. This phenomenon is explicable thanks to the simple decomposition $r - 1 = z^2 \cdot e \cdot |t|$. First, $\nu_2^*(r) = 2 \cdot \nu_2(z)$ provided that z is even. And the 2-adicity of z is easily controlled (as for the seeds of BLS12-380 and BLS12-381). An alternative way to make the higher $\nu_2^*(r)$ consists in increasing $\nu_2^*(z) = \nu_2(e_1) = \nu_2(e)$, which is the case for BLS12-377 and BLS12-(377-383). Moreover, this approach equally augments $\nu_2^*(q)$, which is sometimes no less useful, for example in Zexe [21]. This construction leverages a pairing-friendly curve (such as in [32]) over a new base field and equipped with a subgroup of order q. It is readily verified that

$$q - 1 = e_1 \cdot |z^5 - z^4 - z^3 + z^2 + z + 2|.$$

The second factor is irreducible over $\mathbb{Z}$, so the growth of $\nu_2(e)$ is the unique visible way for that of $\nu_2^*(q)$. However, the high 2-adicity of e radically affects the speed of the batch SMT as already noted in the introduction.

According to [71, Section 5.3], the embedded curves of j-invariant 0 for BLS12 ones admit the prime order r' only if it is parametrized in z by one of the formulas

$$r'_1 := z^4 - 3z^2 + 3, \qquad r'_2 := z^4 + 3, \qquad r'_3 := z^4 - 2z^2 + 4.$$

We have the factorizations

$$r'_1 - 1 = (z^2 - 2) \cdot e \cdot |t|, \qquad r'_2 - 1 = z^4 + 2, \qquad r'_3 - 1 = z^4 - 2z^2 + 3.$$

The last two polynomials are irreducible over $\mathbb{Z}$, hence our eyes are basically on the first one. Whenever the seed z is even, $\nu_2^*(r'_1) = \nu_2^*(r'_2) = 1$ and r'_3 is not even prime. As a consequence, one cannot construct in this case an embedded 2-cycle of $j = 0$ curves over highly 2-adic fields. In turn, $\nu_2^*(r'_1) = \nu_2(e) + 1$ and $\nu_2^*(r'_1) = \nu_2(t) + 1$ once $\nu_2(e) > 1$ and $\nu_2(t) > 1$, respectively. In other words, $\nu_2^*(r)$, $\nu_2^*(r')$ both grow in parallel with $\nu_2(e)$ or $\nu_2(t)$. Fortunately, the highly 2-adic t (unlike e) does not slow down the batch SMT, hence such a trace is preferable.

2.2 New curves

To sum up, we want $z \in \mathbb{Z}$ such that

1. The bit length $\log_2(|z|) \approx 64$;
2. The orders q, r and $r' = r'_1$ are primes;
3. The cofactors h_2, h_T are strong, i.e., do not have divisors less than r;
4. The prime p is as large as possible;
5. The Frobenius trace t is highly 2-adic: Let's say, $\nu_2(t)$ is close to 32;
6. The Hamming weight w is as small as possible.

In fact, the binary logarithms of the orders q, r and r' should not exceed in the strict sense the bounds 384, 256 and 256, respectively. Otherwise, the finite field arithmetic becomes slower in vain if hardware operates with long machine

words, particularly with 64-bit ones. It is thereby desirable for $\log_2(|z|)$ to be closer to 63 than to 64, which is assumed hereafter.

Since $\nu_2(e) = 1$, at best $p = e_1/2 = e/6$ and so $\pi' = 6$. This means that $\log_2(p) \approx 60.5$ does not achieve $\log_2(e_1) \approx 61.5$. Thus, the one-round batch SMT ensures the level $\mu \approx 60.5$ instead of 61.5 (not to mention 62.5 as initially supposed in the introduction). For simplicity and unambiguity, put $\mu = 60$. Formally, n must be doubled, albeit the coefficients $c_{i,2}$ in the second sum S_2 are allowed to be just binary. However, it is not worth the trouble, because the difference is in only one bit. On the other hand, the evenness of e implies that $E[2] \subset E(\mathbb{F}_q)$, that is, the curve E is isomorphic or 2-isogenous over $\mathbb{F}_q$ to a Montgomery/twisted Edwards curve (see, e.g., [38, Theorem 9.12.11]). In the authors' opinion, the high-speed arithmetic on these forms more than compensates for the one-bit loss in estimating the error probability $\mathbb{P}$ and for the necessity of determining the two Legendre symbols (in addition to the cubic one) per point $P_i \in E(\mathbb{F}_q)$.

It will not hurt to repeat that $\nu_2^*(r) = \nu_2^*(r') = \nu_2(t) + 1$. In order to obtain the fields $\mathbb{F}_r$, $\mathbb{F}_{r'}$ of the same high 2-adicity v, it is sufficient to sequentially iterate an auxiliary seed z' of length $\approx 63 - v$, putting $z := 2^v z' - 1$, until a desired BLS12 curve is met. For transparency, z' must start with a canonical value such as $\pm 2^{63-v}$. Two new curves BLS12-376 and BLS12-377 were produced in this way. Their parameters (along with the generation Sage script) can be found in [55] or in Table 3. By construction, the curves are well suited for ZK applications (see Table 4 as a confirmation). Furthermore, both of them have $e_1 = 2p$ and the prime cofactor h_2. The cofactor h_T is also prime for BLS12-377, but at least $73 \mid h_T$ for BLS12-376. Therefore, one of the curves satisfies all the properties imposed above. However, this happens at the price of skewed values of v, w. In turn, the second curve possesses more optimal ones. To be honest, the (quasi-)primality of h_T is much less important than that of e_1, h_2, as in the majority[5] of pairing-based protocols, entities usually do not receive from a communication channel field elements unlike curve points.

Curve	z	w	$e_1 = e/3$	$\pi'/3$	p	n	$\mathbb{G}_2$-strong	$\mathbb{G}_T$-strong
BLS12-376	$-1008476051 \cdot 2^{33} - 1$	10	$2 \cdot p_{61}$	2	p_{61}	1	yes	no
BLS12-377	$-555684677407 \cdot 2^{24} - 1$	15						yes

Table 3: New BLS12 curves

[5] A quite exotic exception is the scheme [70] proposed by the Ethereum Foundation research team before it has been recently decided to carry over the ecosystem to post-quantum cryptography. In addition, testing membership to $\mathbb{G}_T$ is indispensable in the scope of pairing delegation (see, e.g., the fresh paper [50]), although the authors are not sure about the spread of such a mechanism in the real world.

BLS12 curve	$\nu_2^*(q)$	$\nu_2^*(r)$	$\nu_2^*(r')$	h	$-D$
BLS12-376	1	34		1	3
BLS12-377		25			

Table 4: New embedded curves for the ZK-oriented BLS12 ones from Table 3

3 Subgroup checks and their cost

The batch SMT for the subgroup $\mathbb{G}_1$ of BLS12 curves is formalized in Algorithm 1. As before, we are given N points $P_i \in E(\mathbb{F}_q)$ and hence their n sums $S_j = \sum_{i:=1}^{N} c_{i,j} P_i$ with respect to random uniform coefficients $c_{i,j} \in \mathbb{Z}/m$. Instead of the maximal allowable $m = p$, it is more convenient in practice to take $m = 2^{\lfloor \log_2(p) \rfloor}$. Fortunately, such a choice should not affect the values μ and n once p is at least moderate.

Whenever N is small, the *Straus algorithm* [77] should be the best for finding S_j. In turn, we will resort to the *Pippenger (bucket) algorithm* for the large N. It is asymptotically optimal (see, e.g., [45, Section 1]) and its various optimizations have occurred in recent years. Since for the modest N conventional non-batch tests are competitive with (or even better if $n \geqslant N$ than) their batch versions, we are (only) interested in the truly multi-dimensional setting. Besides, note that the coefficients $c_{i,j}$ are zero with probability $1/m$. Consequently, the sums S_j consist on average of $N' := N(m-1)/m$ terms. This is a useful remark in the case of the small m, since it permits to reduce (extremely for $m = 2$) the running time of an MSM algorithm.

The symbols $\mathcal{A}$, $\mathcal{D}$ will stand for the general addition and doubling operations in $E(\mathbb{F}_q)$, respectively. Let's fix a window size $c \in \mathbb{N}$ not exceeding $\log_2(m)$. Recall that the Pippenger algorithm costs

$$\approx \mathcal{P} := \frac{\log_2(m)}{c}(N' + 2^{c+1} - 3)\cdot\mathcal{A} + \big(\log_2(m) - c\big)\cdot\mathcal{D}$$

according to [27, Section 2.3] for example. In particular, if we deal with the scenario $m = 2$, then necessarily $c = 1$ and the algorithm boils down to computing a sum with approximately $N' = N/2$ terms, that is, $\mathcal{P} \approx N'\cdot\mathcal{A}$. Meanwhile, the Bowe–Scott subgroup check requires

$$\approx \mathcal{BS} := 2w\cdot\mathcal{A} + \frac{\log_2(r)}{2}\cdot\mathcal{D}$$

if the scalar multiplication by z^2 is carried out via the standard double-add-subtract method. This is reasonable, because the weight w is chosen to be

Algorithm 1: Batch subgroup membership testing

Data: a BLS12 curve $E/\mathbb{F}_q$ and its subgroup $\mathbb{G}_1$ with order r and cofactor $h_1 = 3e_1^2$;
the intermediate subgroup $\mathbb{G}_1' = E(\mathbb{F}_q)[re']$, where $e' = e/\pi'$;
the number $m \leqslant p$, the level μ and the number of rounds $n = \lceil \mu / \log_2(m) \rceil$;
N points $P_i \in E(\mathbb{F}_q)$.
Result: if all $P_i \in \mathbb{G}_1$, then 1; otherwise, 0 with error probability $\mathbb{P} \leqslant 1/2^\mu$.
begin
  for $i := 1$ **to** N **do**
    if $P_i \notin \mathbb{G}_1'$ **then**
      return 0.
    end
  end
  for $j := 1$ **to** n **do**
    $c_{i,j} :=$ Random$(\mathbb{Z}/m)$;
    $S_j := \sum_{i:=1}^{N} c_{i,j} P_i$;
    if $S_j \notin \mathbb{G}_1$ **then**
      return 0.
    end
  end
  return 1.
end

as minimal as possible and hence there are no substantially shorter addition-subtraction chains. Furthermore, it is understandable that the Pippenger and Bowe–Scott algorithms are not prone to precomputation, as the curve points are variables. As a result, the overall overhead of performing the batch SMT at worst amounts to $\approx n(\mathcal{P} + \mathcal{BS})$ versus N separate group checks, i.e., $\approx N \cdot \mathcal{BS}$.

Asymptotically, there is no difference (subject to the same N and μ) between computing n sums S_j with shorter coefficients $c_{i,j}$ and finding one sum with longer $c_{i,j}$. Indeed, the corresponding complexity is equal to

$$\Theta\left(\frac{nN\log_2(m)}{\log_2\big(nN\log_2(m)\big)}\right) = \Theta\left(\frac{\mu N}{\log_2(\mu N)}\right)$$

curve additions/doublings (cf. the formula of $\mathcal{P}$) by virtue of [45, Section 1] and by the definition of n. Consequently, the given quantity is in fact independent of n and $\log_2(m)$ and hence it is difficult to see why (and if) the growth of the parameter m is really a winning strategy (at least for the moderate N). The fundamental reason lies in the fact that n is equally the number of calls of the Bowe–Scott test. Nevertheless, the benchmarks of the next section are paramount to dispel all doubts in this regard.

4 Implementation and benchmarks

In this section we provide an open-source high-speed software implementation in Golang, using the `gnark-crypto` library [17]. We chose to implement Algorithm 1 for BLS12-381 and BLS12-377, two pairing-friendly elliptic curves widely used in production. We also fully implemented in `gnark-crypto` the two new BLS12 curves presented in Section 2.2, alongside the new batch SMT for them. All implementations are available in [55]:

`https://github.com/yelhousni/batch-subgroup-membership-testing`.

We benchmarked the new batch SMT algorithm against the naive separate checks, namely the multi-time Bowe–Scott membership test. All benchmarks were run on a AWS `z1d.large` machine (Intel(R) Xeon(R) Platinum 8151 CPU @ 3.40GHz) with hyperthreading, turbo and frequency scaling disabled. In the sequel, we present the results derived from a single-threaded implementation. Although we also explored code parallelism using Goroutines, with the corresponding implementation accessible in [55], our primary focus remains on the sequential execution. This decision facilitates a more straightforward analysis within the scope of this paper and avoids confounding factors introduced by diverse parallelism strategies.

The BLS12-381 case: This $\mathbb{F}_q$-curve of equation $E\colon y^2 = x^3 + 4$ has the seed $z = \texttt{-0xd201000000010000}$ and thus $\pi' = 3{\cdot}11$ and $e' = 10177{\cdot}859267{\cdot}52437899$. Given N points $P_i \in E(\mathbb{F}_q)$ purported to be in $\mathbb{G}_1$, Algorithm 1 is a two-step procedure:

- *Step 1*: We check that $P_i \in \mathbb{G}_1' = E[re'] \supset \mathbb{G}_1$ via the Koshelev test. For this, we need to compute the N Tate pairings $t_3(Q_3, P_i)$ together with the $2N$ Tate pairings $t_{11}(Q_{11}, P_i)$ and $t_{11}(Q'_{11}, P_i)$. We then check that they are all equal to 1. Here, the point $Q_3 := (0, 2)$ is of order 3 and hence the Miller function $f_{3,Q_3} = y - 2$ is elementary. We exponentiate by $(q-1)/3$, using a short addition chain of 375 squarings and 79 multiplications. In turn, the points Q_{11} and Q'_{11} constitute a basis of $E[11]$ (see Appendix A). The Miller function for Q_{11} is of the form

$$f_{11,Q_{11}} = \frac{\ell_{1,1}^4 \cdot (\ell_{4,1} \cdot \ell_{2,2})^2 \cdot \ell_{5,5}}{v_2^4 \cdot (v_5 \cdot v_4)^2},$$

  where

$$v_i := x - x_{[i]Q_{11}}, \qquad \ell_{i,j} := (y - y_{[i]Q_{11}}) - \lambda_{i,j} \cdot v_i$$

  and

$$\lambda_{i,j} := \begin{cases} \dfrac{y_{[i]Q_{11}} - y_{[j]Q_{11}}}{x_{[i]Q_{11}} - x_{[j]Q_{11}}} & \text{if} \quad i \neq j, \\ \dfrac{3 \cdot x_{[i]Q_{11}}^2}{2 \cdot y_{[i]Q_{11}}} & \text{otherwise.} \end{cases}$$

Since Q_{11} is fixed, we precompute all the 7 lines $\ell_{i,j}$, v_i and only evaluate them at P_i during the Tate pairing computation. Instead of inverting the denominator $v_2^4 \cdot (v_5 \cdot v_4)^2$, we raise it to the power 10, since $(\frac{1/v}{q})_k = (\frac{v^{k-1}}{q})_k$ for arbitrary $k \mid q-1$ and $v \in \mathbb{F}_q^*$. The same applies to computing $f_{11,Q'_{11}}$. Finally, we exponentiate by $(q-1)/11$, using a short addition chain of 372 squarings and 77 multiplications.

– *Step 2*: After detecting some fraudulent points $P_i \notin \mathbb{G}'_1$, we compute n MSMs $S_j = \sum_{i=1}^{N} c_{i,j} P_i$ with random coefficients $c_{i,j} \in \mathbb{Z}/m$ (of course, $c_{i,j} = 0$ when $P_i \notin \mathbb{G}'_1$). We then check that $S_j \in \mathbb{G}_1$, using the Bowe–Scott test. For BLS12-381, $p = 10177$ is the smallest prime divisor of e'. Consequently, for the error probability $\mathbb{P} \leqslant 1/2^{60}$, we chose $m = 2^{13} < p < 2^{14}$ and the number of rounds $n = \lceil 60/\log_2(m) \rceil = 5$ (as well as in Table 1 when m is replaced by p).

Figure 1 reports the execution times of the new batch SMT compared to the collection of the naive separate checks. The benchmarks were run on a set of $N \in \{2^{3+2k}\}_{k=1}^{9}$ random points $P_i \in \mathbb{G}_1$, albeit correctness of the batch strategy was verified for non-restricted $P_i \in E(\mathbb{F}_q)$. Since Step 2 becomes even more efficient when a part of P_i are outside the subgroup $\mathbb{G}'_1$, the graph in reality showcases the worst-case running time of Algorithm 1. In other words, its speed should be more impressive on average if there is no goal to catch painful points from $\mathbb{G}'_1 \setminus \mathbb{G}_1$. It is normal that the naive method is faster for the small $N = 32$, since we compare 32 Bowe–Scott tests to 5 ones, 5 MSMs and $3 \cdot 32 = 96$ long exponentiations (for determining the 3-rd and 11-th power residue symbols). For the greater N, the new method is slightly faster as can be seen by zooming the figure. Notably, if the computational overhead of Step 1 is disregarded, the new method achieves an order of magnitude improvement in speed. This observation highlights the potential for significant performance gain with optimized implementations of the residue symbols.

The BLS12-377 case: This $\mathbb{F}_q$-curve of equation $E\colon y^2 = x^3 + 1$ has the seed $z = \texttt{0x8508c00000000001}$ and thus $\pi' = 2^4 \cdot 3 \cdot 7 \cdot 13$ and $e' = 2^{42} \cdot 499$. The latter number is highly 2-adic and thereby the first step of the batch SMT is useless ($p = 2$). We proceed directly to the second step as follows:

– *Step 2*: We compute the n MSMs S_j with random coefficients $c_{i,j} \in \{0, 1\}$. We implement this, using mixed additions of the points P_i in affine coordinates and the running sum in extended Jacobian coordinates. We then check that $S_j \in \mathbb{G}_1$ through the Bowe–Scott test. For the error probability $\mathbb{P} \leqslant 1/2^{60}$, we obviously need $n = 60$ sums.

Figure 2 exhibits the results for the BLS12-377 curve. It is again normal that the new method is slower for $N = 32$, since we compare 32 Bowe–Scott tests to 60 ones and 60 sums. For the larger N, the new method is between 1.35× to 4.11× faster than the naive one. The performance gain is more pronounced

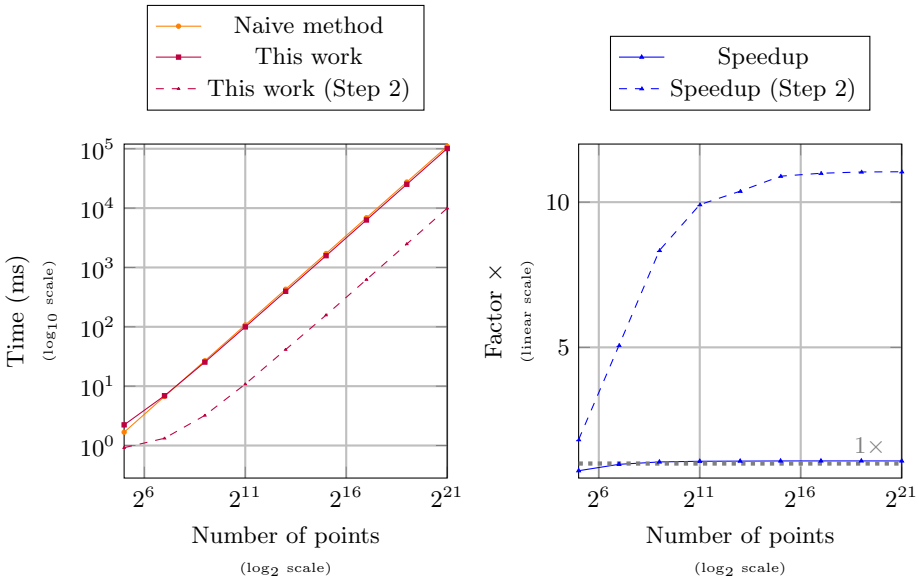

Fig. 1: Comparison of execution time and speedup between the naive method and this work for $\mathbb{P} \leqslant 1/2^{60}$ in the case of the curve BLS12-381

for greater values of N, since we only keep the running sum in memory contrary to the general Pippenger algorithm, which is more memory-heavy. We also note that for the large N we might speed up the approach by using affine additions and implementing Montgomery's batch inversion technique to invert all the denominators at the cost of a single inversion.

The new BLS12-377 case: This $\mathbb{F}_q$-curve of equation $E : y^2 = x^3 + 1$ has the seed $z =$ `-0x816163471f000001` and thus $\pi' = 2 \cdot 3$ and $e' = p = 155380697679125981 9$. Following Algorithm 1, we proceed as follows:

- *Step 1*: We check that $P_i \in \mathbb{G}'_1$ by checking that $t_3(Q_3, P_i) = 1$ and $t_2(Q_2, P_i) = t_2(Q'_2, P_i) = 1$. Here, the point $Q_3 := (0, 1)$ is of order 3 and hence $f_{3,Q_3} = y - 1$. We exponentiate by $(q-1)/3$, using a short addition chain of 370 squarings and 81 multiplications. In turn, the points $Q_2 := (-1, 0)$ and $Q'_2 := (-\omega, 0)$, with $\omega = z^5 - 3z^4 + 3z^3 - z + 1 \pmod q$ being a cubic root of unity in $\mathbb{F}_q$, are of order 2 and hence $f_{2,Q_2} = x + 1$ and $f_{2,Q'_2} = x + \omega$. To evaluate the Legendre symbol, we do not exponentiate banally by $(q-1)/2$, using instead Pornin's variant [65] of the binary GCD and quadratic reciprocity. This is an order of magnitude faster than the exponentiation even with the shortest addition chains.
- *Step 2*: We compute the single MSM $S_1 = \sum_{i=1}^{N} c_{i,1} P_i$ with random scalars $c_{i,1} \leq 2^{60} < p < 2^{61}$ and test if $S_1 \in_? \mathbb{G}_1$. This gives the error probability $\mathbb{P} \leqslant 1/2^{60}$.

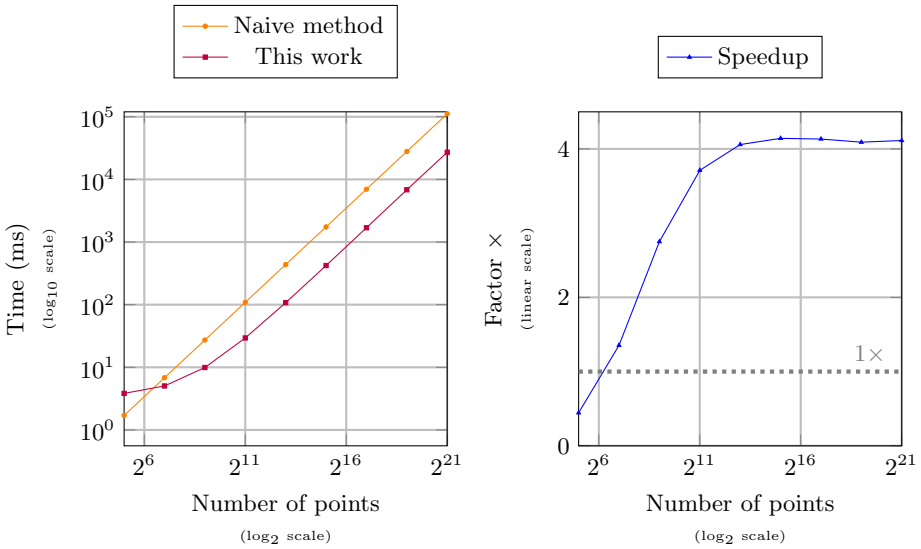

Fig. 2: Comparison of execution time and speedup between the naive method and this work for $\mathbb{P} \leqslant 1/2^{60}$ in the case of the curve BLS12-377

Figure 3 shows that the new method is marginally faster for $N = 32$ than the naive one. However, the former demonstrates more substantial performance improvements for larger values of N, achieving speedup that ranges from 1.6× to 2.3×.

The new BLS12-376 case: This $\mathbb{F}_q$-curve of equation $E : y^2 = x^3 + 1$ has the seed $z =$ `-0x78383f2600000001` and thus $\pi' = 2 \cdot 3$ and $e' = p = 1443790552614742699$. We proceed similarly to the case of the new BLS12-377 curve. Similar results can be seen in Figure 4. The speedup ranges from 1.7× to 2.4×. Since we propose the two new curves as batch-SMT-friendly replacements to BLS12-381, we also benchmark and compare common operations on these three curves (see Appendix B).

Note 1. The benchmarks can be further enhanced by implementing efficiently Euclidean-type algorithms to compute $(\frac{\cdot}{q})_k$ for $k \in \{3, 11\}$. We implemented in [55] the cubic symbol, using the Eisenstein integers $\mathbb{Z}[\omega]$ arithmetic and cubic reciprocity in line with the algorithms of [5,31], but it was slower than the exponentiation by $(q-1)/3$. This is primarily due to two main reasons. First, Eisenstein arithmetic requires arbitrary-precision integer operations (arithmetic in $\mathbb{Z}$), which introduces a computational overhead related to Go's `big.Int`. These operations, in contrast to the fixed-size modular operations (arithmetic in $\mathbb{F}_q$) needed for the exponentiation, involve numerous temporary variable allocations, imposing a significant load on Go's garbage collector. Second, to mimic the binary GCD algorithm, which only requires addition/subtraction and division by

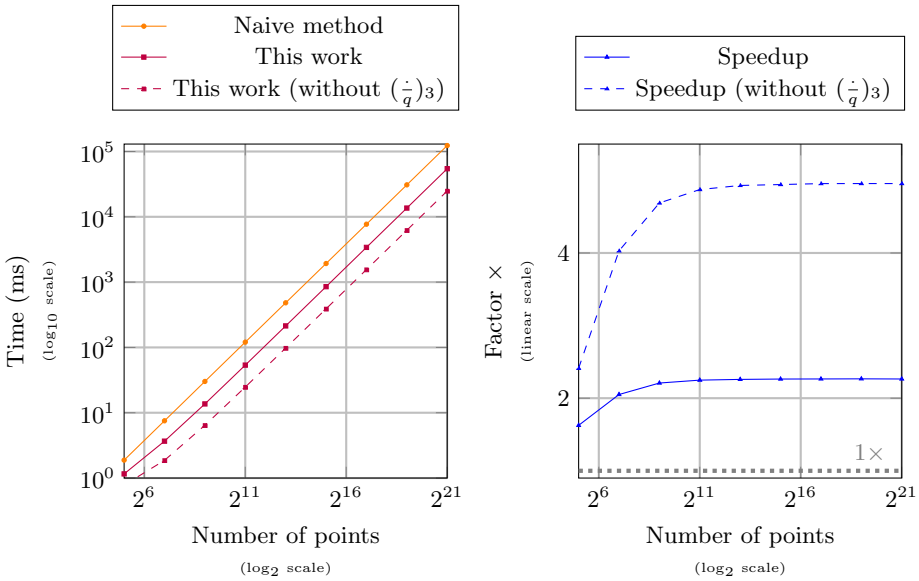

Fig. 3: Comparison of execution time and speedup between the naive method and this work for $\mathbb{P} \leqslant 1/2^{60}$ in the case of the new curve BLS12-377

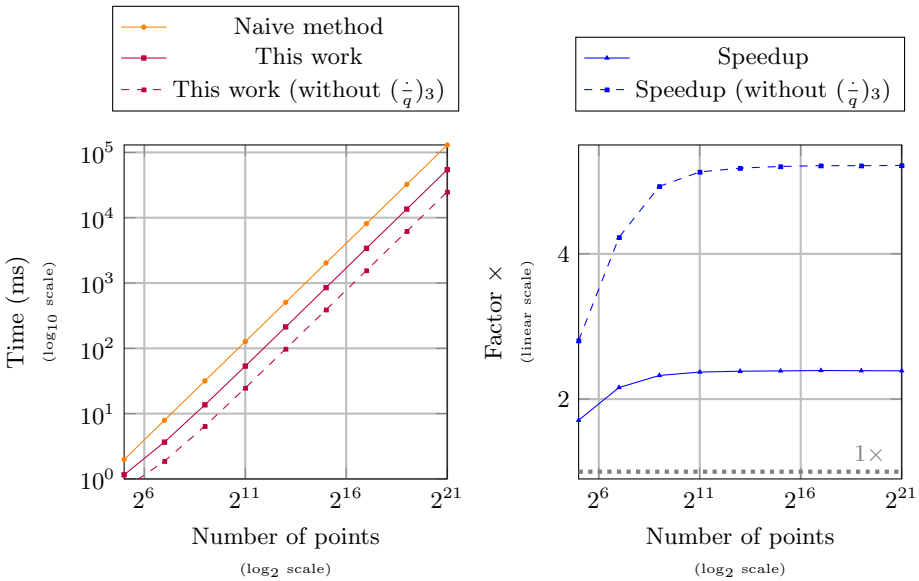

Fig. 4: Comparison of execution time and speedup between the naive method and this work for $\mathbb{P} \leqslant 1/2^{60}$ in the case of the new curve BLS12-376

2, we need to divide by 3, which is not as CPU-friendly as right-shifts (divisions by 2). The role of 2 is taken by the number $1-\omega$, which is a prime of norm 3 in $\mathbb{Z}[\omega]$.

Note 2. It seems that for the moderate number N of points, maximizing the MSM coefficients size and minimizing the number n of rounds always yields the best performance, and we expect it to be even more pronounced with optimized residue symbols. For the huge N, it seems that minimizing the MSM coefficients to $\{0,1\}$ and maximizing n to 60 is conversely better. This is because point additions scale better than the Pippenger algorithm memory-wise. However, all the speedup figures plateau as N increases. This is likely because for the large N, the dominant factor becomes iterating through extensive in-memory point arrays. We believe that a tailored parallelism strategy could alleviate this behavior.

5 Conclusion

The present paper has analyzed the possibility of batch SMT with accent on the subgroup $\mathbb{G}_1$ of a BLS12 curve $E/\mathbb{F}_q$. It has been shown that treating input points $P_i \in E(\mathbb{F}_q)$ in combination truly works much faster (with negligible error probability) than validating independently every point P_i. And whenever the cofactor $h_1 = 3e_1^2$ (or the value e_1) is less smooth, this tendency is clearer. Moreover, the greater the number of points, the more essential the performance gain. Furthermore, two new BLS12 curves have been generated for which $e_1/2$ (along with the other cofactor h_2) is prime. These curves are thus (almost) etalon for using the (re)invented batch SMT. Bearing in mind potential adaptation of the curves in future blockchain projects, each of them also enjoys an embedded 2-cycle consisting of two plain prime-order curves of j-invariant 0 over highly 2-adic fields.

The proposed batch SMT does not have visible disadvantages when all input $\mathbb{F}_q$-points P_i on E belong to the subgroup $\mathbb{G}_1$ or do not belong to the intermediate one $\mathbb{G}'_1$. The second situation is malleable for the new test, since P_i are still not summed in its first stage. In turn, pulling $P_i \in \mathbb{G}'_1 \setminus \mathbb{G}_1$ from massive point collections is a task far from trivial. Taking into account this state of affairs, simultaneous point validation may lose to separate ones in the overall efficiency if the frequency of fraudulent points is not low. In this work, we have not touched at all what to do exactly if the batch SMT fails. Naturally, a similar task has already appeared (under the name *batch forgery identification*) for batch signature verification. Resolution strategies of the given task are elaborated on, for example, in the sources [11,62,63,82]. Their analysis in our context is beyond the scope of the current article, as it deserves a standalone R&D project.

Basically, the batch SMT is generalized to any pair $\mathbb{G} \subset \mathbb{H}$ of groups and to any basic algorithm answering the question $P \in_? \mathbb{G}$ given an element $P \in \mathbb{H}$. The most relevant subgroups are $\mathbb{G}_2$, $\mathbb{G}_T$ of pairing-friendly curves and that of Curve25519 with cofactor 8 (the non-batch Koshelev test works well for other

real-world curves). Certainly, there are peculiarities in each concrete case. For instance, even if we are talking about the BLS12 family (not to mention alternative curve families), one cannot guarantee that $f \mid q^2 - 1$ for a number $f \leqslant 16$ such that $f^2 \mid h_2$. The reason is in the realizable situation $E_2[f] \not\subset E_2(\mathbb{F}_{q^2})$. As a consequence, the preliminary Koshelev test, i.e., Step 1 is not applicable and hence one can only be content with the parameter $p \leqslant f$. On the other hand, the problem of such a type obviously does not exist in the case of $\mathbb{G}_T$. Lastly, the state-of-the-art Pornin test [66] for Curve25519 is batchable with $p = 2$ analogously to the curve BLS12-377.

Acknowledgements. The authors express their gratitude to Antonio Sanso for attracting attention to the problem of batch subgroup membership testing and to Gautam Botrel for helping with software optimizations.

This research is a result of the strategic project "Advances in post-quantum cryptography applied to the development of a coupon system" (C039/24), resulting from an agreement between the Spanish National Cybersecurity Institute (INCIBE) and University of Lleida. This initiative is carried out in the scope of the funds from the Recovery, Transformation and Resilience Plan, funded by the European Union (Next Generation). The paper is also part of the R&D+i project PID2021-124613OB-I00 funded by MICIU/AEI/10.13039/501100011033 and FEDER, EU. Georgios Fotiadis acknowledges the financial support from the Luxembourg National Research Fund (FNR) via the CORE project Real-World Implementation and Human-Centered Design of PAKE Technologies - ImPAKT (C21/IS/16221219/ImPAKT) and the CORE project Privacy-Preserving Tokenisation of Artworks - PABLO (C21/IS/16326754/PABLO),

References

1. The decaf377 group, `https://protocol.penumbra.zone/main/crypto/decaf377.html`
2. EIP-2537 (BLS12 precompile) discussion thread, `https://ethereum-magicians.org/t/eip-2537-bls12-precompile-discussion-thread/4187`
3. Aranha, D.F., Salling Hvass, B., Spitters, B., Tibouchi, M.: Faster constant-time evaluation of the Kronecker symbol with application to elliptic curve hashing. In: CCS 2023: ACM SIGSAC Conference on Computer and Communications Security. pp. 3228–3238. Association for Computing Machinery, New York (2023), `https://doi.org/10.1145/3576915.3616597`
4. Aumasson, J.P., Nguyen, Q.T.M., Sanso, A.: Security of BLS batch verification, `https://ethresear.ch/t/security-of-bls-batch-verification/10748`
5. Bach, E., Sandlund, B.: On Euclidean methods for cubic and quartic Jacobi symbols (2018), `https://arxiv.org/abs/1807.07719`
6. Barbulescu, R., Duquesne, S.: Updating key size estimations for pairings. Journal of Cryptology **32**(4), 1298–1336 (2019), `https://doi.org/10.1007/s00145-018-9280-5`
7. Barreto, P.S.L.M., Costello, C., Misoczki, R., Naehrig, M., Pereira, G.C.C.F., Zanon, G.: Subgroup security in pairing-based cryptography. In: Lauter, K.,

Rodríguez-Henríquez, F. (eds.) Progress in Cryptology – LATINCRYPT 2015. Lecture Notes in Computer Science, vol. 9230, pp. 245–265. Springer, Cham (2015), https://doi.org/10.1007/978-3-319-22174-8_14
8. Barreto, P.S.L.M., Lynn, B., Scott, M.: Constructing elliptic curves with prescribed embedding degrees. In: Cimato, S., Persiano, G., Galdi, C. (eds.) Security in Communication Networks. SCN 2002. Lecture Notes in Computer Science, vol. 2576, pp. 257–267. Springer, Berlin, Heidelberg (2003), https://doi.org/10.1007/3-540-36413-7_19
9. Barreto, P.S.L.M., Naehrig, M.: Pairing-friendly elliptic curves of prime order. In: Preneel, B., Tavares, S. (eds.) Selected Areas in Cryptography. SAC 2005. Lecture Notes in Computer Science, vol. 3897, pp. 319–331. Springer, Berlin, Heidelberg (2006), https://doi.org/10.1007/11693383_22
10. Bellare, M., Garay, J.A., Rabin, T.: Fast batch verification for modular exponentiation and digital signatures. In: Nyberg, K. (ed.) Advances in Cryptology – EUROCRYPT 1998. Lecture Notes in Computer Science, vol. 1403, pp. 236–250. Springer, Berlin, Heidelberg (1998), https://doi.org/10.1007/BFb0054130
11. Bernstein, D.J., Doumen, J., Lange, T., Oosterwijk, J.J.: Faster batch forgery identification. In: Galbraith, S., Nandi, M. (eds.) Progress in Cryptology – INDOCRYPT 2012. Lecture Notes in Computer Science, vol. 7668, pp. 454–473. Springer, Berlin, Heidelberg (2012), https://doi.org/10.1007/978-3-642-34931-7_26
12. Boneh, D., Drijvers, M., Neven, G.: Compact multi-signatures for smaller blockchains. In: Peyrin, T., Galbraith, S.D. (eds.) Advances in Cryptology – ASIACRYPT 2018. Lecture Notes in Computer Science, vol. 11273, pp. 435–464. Springer, Cham (2018), https://doi.org/10.1007/978-3-030-03329-3_15
13. Boneh, D., Franklin, M.K.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) Advances in Cryptology – CRYPTO 2001. Lecture Notes in Computer Science, vol. 2139, pp. 213–229. Springer, Berlin, Heidelberg (2001), https://doi.org/10.1007/3-540-44647-8_13
14. Boneh, D., Gentry, C., Lynn, B., Shacham, H.: Aggregate and verifiably encrypted signatures from bilinear maps. In: Biham, E. (ed.) Advances in Cryptology – EUROCRYPT 2003. Lecture Notes in Computer Science, vol. 2656, pp. 416–432. Springer, Berlin, Heidelberg (2003), https://doi.org/10.1007/3-540-39200-9_26
15. Boneh, D., Gorbunov, S., Wahby, R.S., Wee, H., Wood, C.A., Zhang, Z.: BLS signatures (2022), https://datatracker.ietf.org/doc/draft-irtf-cfrg-bls-signature
16. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the Weil pairing. Journal of Cryptology **17**(4), 297–319 (2004), https://doi.org/10.1007/s00145-004-0314-9
17. Botrel, G., Piellard, T., El Housni, Y., Tabaie, A., Gutoski, G., Kubjas, I.: Consensys/gnark-crypto: v0.15.0 (2025). https://doi.org/10.5281/zenodo.5815453
18. Bowe, S.: BLS12-381: New zk-SNARK elliptic curve construction (2017), https://electriccoin.co/blog/new-snark-curve
19. Bowe, S.: Switch from BN254 to BLS12-381 (2017), https://github.com/zcash/zcash/issues/2502
20. Bowe, S.: Faster subgroup checks for BLS12-381 (2019), https://eprint.iacr.org/2019/814
21. Bowe, S., Chiesa, A., Green, M., Miers, I., Mishra, P., Wu, H.: Zexe: enabling decentralized private computation. In: 2020 IEEE Symposium on Security and

Privacy (SP). pp. 947–964. IEEE, New York (2020), https://doi.org/10.1109/SP40000.2020.00050
22. Boyd, C., Pavlovski, C.: Attacking and repairing batch verification schemes. In: Okamoto, T. (ed.) Advances in Cryptology – ASIACRYPT 2000. Lecture Notes in Computer Science, vol. 1976, pp. 58–71. Springer, Berlin, Heidelberg (2000), https://doi.org/10.1007/3-540-44448-3_5
23. Brier, E., Géraud-Stewart, R., Joye, M., Naccache, D.: Primary elements in cyclotomic fields with applications to power residue symbols, and more (2021), https://eprint.iacr.org/2021/1106
24. Brier, E., Naccache, D.: The thirteenth power residue symbol (2019), https://eprint.iacr.org/2019/1176
25. Budroni, A., Pintore, F.: Efficient hash maps to $\mathbb{G}_2$ on BLS curves. Applicable Algebra in Engineering, Communication and Computing **33**(3), 261–281 (2022), https://doi.org/10.1007/s00200-020-00453-9
26. Chen, L., Moody, D., Regenscheid, A., Robinson, A., Randall, K.: Recommendations for discrete logarithm-based cryptography: Elliptic curve domain parameters (NIST Special Publication 800-186) (2023), https://csrc.nist.gov/publications/detail/sp/800-186/final
27. Chen, Y., Peng, C., Dai, Y., Luo, M., He, D.: Load-balanced parallel implementation on GPUs for multi-scalar multiplication algorithm. IACR Transactions on Cryptographic Hardware and Embedded Systems **2024**(2), 522–544 (2024), https://doi.org/10.46586/tches.v2024.i2.522-544
28. Clarisse, R., Duquesne, S., Sanders, O.: Curves with fast computations in the first pairing group. In: Krenn, S., Shulman, H., Vaudenay, S. (eds.) Cryptology and Network Security. CANS 2020. Lecture Notes in Computer Science, vol. 12579, pp. 280–298. Springer, Cham (2020), https://doi.org/10.1007/978-3-030-65411-5_14
29. Dai, Y., He, D., Koshelev, D., Peng, C., Yang, Z.: Revisiting subgroup membership testing on pairing-friendly curves via the Tate pairing (2024), https://eprint.iacr.org/2024/1790
30. Dai, Y., Lin, K., Zhao, C.A., Zhou, Z.: Fast subgroup membership testings for $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ on pairing-friendly curves. Designs, Codes and Cryptography **91**(10), 3141–3166 (2023), https://doi.org/10.1007/s10623-023-01223-7
31. Damgård, I.B., Frandsen, G.S.: Efficient algorithms for the gcd and cubic residuosity in the ring of Eisenstein integers. Journal of Symbolic Computation **39**(6), 643–652 (2005), https://doi.org/10.1016/j.jsc.2004.02.006
32. El Housni, Y., Guillevic, A.: Optimized and secure pairing-friendly elliptic curves suitable for one layer proof composition. In: Krenn, S., Shulman, H., Vaudenay, S. (eds.) Cryptology and Network Security. CANS 2020. Lecture Notes in Computer Science, vol. 12579, pp. 259–279. Springer, Cham (2020), https://doi.org/10.1007/978-3-030-65411-5_13
33. El Housni, Y., Guillevic, A., Piellard, T.: Co-factor clearing and subgroup membership testing on pairing-friendly curves. In: Batina, L., Daemen, J. (eds.) Progress in Cryptology – AFRICACRYPT 2022. Lecture Notes in Computer Science, vol. 13503, pp. 518–536. Springer, Cham (2022), https://doi.org/10.1007/978-3-031-17433-9_22
34. Electric Coin Company: What is Jubjub?, https://bitzecbzc.github.io/technology/jubjub
35. Fan, X., Kuchta, V., Sica, F., Xu, L.: Speeding up multi-scalar multiplications for pairing-based zkSNARKs. Journal of Cryptology **38**(2), article 21 (2025), https://doi.org/10.1007/s00145-025-09540-x

36. Faraoun, K.M.: Why not BLS12-461 with equivalent performance? (2019), https://github.com/zcash/zcash/issues/4065
37. Gabizon, A.: Possible improvements in subgroup testing, https://github.com/zcash/zcash/issues/3470
38. Galbraith, S.D.: Mathematics of public key cryptography. Cambridge University Press, New York (2012)
39. Guillevic, A., Masson, S.: Embedded curves and embedded families for SNARK-friendly curves (2024), https://eprint.iacr.org/2024/1737
40. Guillevic, A., Singh, S.: On the alpha value of polynomials in the tower number field sieve algorithm. Mathematical Cryptology **1**(1), 1–39 (2021), https://journals.flvc.org/mathcryptology/article/view/125142
41. Haboeck, U.: Security estimates of pairing-friendly curves (2020), https://github.com/arkworks-rs/algebra/issues/732
42. Haddaji, W., Ghammam, L., El Mrabet, N., Ben Abdelghani, L.: On computing the multidimensional scalar multiplication on elliptic curves (2024), https://eprint.iacr.org/2024/038
43. Hamburg, M.: Computing the Jacobi symbol using Bernstein–Yang (2021), https://eprint.iacr.org/2021/1271
44. Harn, L.: Batch verifying multiple RSA digital signatures. Electronics Letters **34**(12), 1219–1220 (1998), https://doi.org/10.1049/el:19980833
45. Henry, R.: Pippenger's multiproduct and multiexponentiation algorithms (2010), https://cacr.uwaterloo.ca/techreports/2010/cacr2010-26.pdf
46. Hopwood, D.E.: Understand the concrete security level of the BN_128 curve in libsnark (2024), https://github.com/zcash/zcash/issues/714
47. Jiang, R., Peng, C., Luo, M., Chen, R., He, D.: SimdMSM: SIMD-accelerated multi-scalar multiplication framework for zkSNARKs. IACR Transactions on Cryptographic Hardware and Embedded Systems **2025**(2), 681–704 (2025), https://doi.org/10.46586/tches.v2025.i2.681-704
48. Joux, A.: A one round protocol for tripartite Diffie–Hellman. Journal of Cryptology **17**(4), 263–276 (2004), https://doi.org/10.1007/s00145-004-0312-y
49. Joye, M., Lapiha, O., Nguyen, K., Naccache, D.: The eleventh power residue symbol. Journal of Mathematical Cryptology **15**(1), 111–122 (2021), https://doi.org/10.1515/jmc-2020-0077
50. Keilty, A.P., Aranha, D.F., Pagnin, E., Rodríguez-Henríquez, F.: That's AmorE: amortized efficiency for pairing delegation (2025), https://eprint.iacr.org/2025/542
51. Kim, T., Barbulescu, R.: Extended tower number field sieve: a new complexity for the medium prime case. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology – CRYPTO 2016. Lecture Notes in Computer Science, vol. 9814, pp. 543–571. Springer, Berlin, Heidelberg (2016), https://doi.org/10.1007/978-3-662-53018-4_20
52. Koshelev, D.: Subgroup membership testing on elliptic curves via the Tate pairing. Journal of Cryptographic Engineering **13**(1), 125–128 (2023), https://doi.org/10.1007/s13389-022-00296-9
53. Koshelev, D.: Correction to: Subgroup membership testing on elliptic curves via the Tate pairing. Journal of Cryptographic Engineering **14**(1), 127–128 (2024), https://doi.org/10.1007/s13389-023-00331-3
54. Koshelev, D.: Application of Mordell–Weil lattices with large kissing numbers to acceleration of multiscalar multiplication on elliptic curves. Journal of Mathematical Cryptology **19**(1), article 20240034 (2025), https://doi.org/10.1515/jmc-2024-0034

55. Koshelev, D., El Housni, Y., Fotiadis, G.: Code (2025), https://github.com/yelhousni/batch-subgroup-membership-testing
56. Koshelev, D., Sanso, A.: Endomorphisms for faster cryptography on elliptic curves of moderate CM discriminants (2024), https://eprint.iacr.org/2024/1985
57. Lim, C.H., Lee, P.J.: A key recovery attack on discrete log-based schemes using a prime order subgroup. In: Kaliski, B.S. (ed.) Advances in Cryptology – CRYPTO 1997. Lecture Notes in Computer Science, vol. 1294, pp. 249–263. Springer, Berlin, Heidelberg (1997), https://doi.org/10.1007/BFb0052240
58. Lin, J., Robert, D., Zhao, C.A., Zheng, Y.: Biextensions in pairing-based cryptography (2025), https://eprint.iacr.org/2025/670
59. Masson, S., Sanso, A., Zhang, Z.: Bandersnatch: a fast elliptic curve built over the BLS12-381 scalar field. Designs, Codes and Cryptography **92**(12), 4131–4143 (2024)
60. Moody, D., Peralta, R., Perlner, R., Regenscheid, A., Roginsky, A., Chen, L.: Report on pairing-based cryptography. Journal of Research of the National Institute of Standards and Technology **120**, 11–27 (2015), http://doi.org/10.6028/jres.120.002
61. Naccache, D., M'Raïhi, D., Vaudenay, S., Raphaeli, D.: Can D.S.A. be improved? Complexity trade-offs with the digital signature standard. In: De Santis, A. (ed.) Advances in Cryptology – EUROCRYPT 1994. Lecture Notes in Computer Science, vol. 950, pp. 77–85. Springer, Berlin, Heidelberg (1995), https://doi.org/10.1007/BFb0053426
62. Pastuszak, J., Michałek, D., Pieprzyk, J., Seberry, J.: Identification of bad signatures in batches. In: Imai, H., Zheng, Y. (eds.) Public Key Cryptography. PKC 2000. Lecture Notes in Computer Science, vol. 1751, pp. 28–45. Springer, Berlin, Heidelberg (2000), https://doi.org/10.1007/978-3-540-46588-1_3
63. Pastuszak, J., Pieprzyk, J., Seberry, J.: Codes identifying bad signatures in batches. In: Roy, B., Okamoto, E. (eds.) Progress in Cryptology – INDOCRYPT 2000. Lecture Notes in Computer Science, vol. 1977, pp. 143–154. Springer, Berlin, Heidelberg (2000), https://doi.org/10.1007/3-540-44495-5_13
64. Pope, G., Reijnders, K., Robert, D., Sferlazza, A., Smith, B.: Simpler and faster pairings from the Montgomery ladder (2025), https://eprint.iacr.org/2025/672
65. Pornin, T.: X25519 implementation for ARM Cortex-M0/M0+ (2020), https://github.com/pornin/x25519-cm0
66. Pornin, T.: Point-halving and subgroup membership in twisted Edwards curves (2022), https://eprint.iacr.org/2022/1164
67. Pottier, X., de Ruijter, T., Bertels, J., Legiest, W., Van Beirendonck, M., Verbauwhede, I.: OPTIMSM: FPGA hardware accelerator for zero-knowledge MSM. IACR Transactions on Cryptographic Hardware and Embedded Systems **2025**(2), 489–510 (2025), https://doi.org/10.46586/tches.v2025.i2.489-510
68. Robert, D.: Fast pairings via biextensions and cubical arithmetic (2024), https://eprint.iacr.org/2024/517
69. Sanso, A.: Fast $\mathbb{G}_2$ subgroup check in BN254 (2022), https://ethresear.ch/t/fast-mathbb-g-2-subgroup-check-in-bn254/13974
70. Sanso, A.: The return of torus based cryptography: Whisk and Curdleproof in the target group (2023), https://ethresear.ch/t/the-return-of-torus-based-cryptography-whisk-and-curdleproof-in-the-target-group/16678

71. Sanso, A., El Housni, Y.: Families of prime-order endomorphism-equipped embedded curves on pairing-friendly curves. Journal of Cryptology **37**(4), article 37 (2024), https://doi.org/10.1007/s00145-024-09514-5
72. Scott, M.: Unbalancing pairing-based key exchange protocols (2013), https://eprint.iacr.org/2013/688
73. Scott, M.: Pairing implementation revisited (2019), https://eprint.iacr.org/2019/077
74. Scott, M.: A note on group membership tests for $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ on BLS pairing-friendly curves (2021), https://eprint.iacr.org/2021/1130
75. Shlomovits, O.: Baby sharks: injecting small order points to threshold EdDSA (2020), https://medium.com/zengo/baby-sharks-a3b9ceb4efe0
76. Spagni, R.: Disclosure of a major bug in CryptoNote based currencies (2017), https://www.getmonero.org/2017/05/17/disclosure-of-a-major-bug-in-cryptonote-based-currencies.html
77. Straus, E.G.: Addition chains of vectors (problem 5125). American Mathematical Monthly **71**(7), 806–808 (1964)
78. Teruya, T.: A note on subgroup security in discrete logarithm-based cryptography. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **E104-A**(1), 104–120 (2021), https://doi.org/10.1587/transfun.2020CIP0019
79. Valenta, L., Adrian, D., Sanso, A., Cohney, S., Fried, J., Hastings, M., Halderman, J.A., Heninger, N.: Measuring small subgroup attacks against Diffie–Hellman (2016), https://eprint.iacr.org/2016/995
80. Vlasov, A.: EIP-2539: BLS12-377 curve operations (2020), https://eips.ethereum.org/EIPS/eip-2539
81. Wang, J.: BN254 for the rest of us (2024), https://hackmd.io/@jpw/bn254
82. Zaverucha, G.M., Stinson, D.R.: Group testing and batch verification. In: Kurosawa, K. (ed.) Information Theoretic Security. ICITS 2009. Lecture Notes in Computer Science, vol. 5973, pp. 140–157. Springer, Berlin, Heidelberg (2010), https://doi.org/10.1007/978-3-642-14496-7_12
83. Zhu, X., He, H., Yang, Z., Deng, Y., Zhao, L., Hou, R.: Elastic MSM: A fast, elastic and modular preprocessing technique for multi-scalar multiplication algorithm on GPUs. IACR Transactions on Cryptographic Hardware and Embedded Systems **2024**(4), 258–284 (2024), https://doi.org/10.46586/tches.v2024.i4.258-284

A The Miller functions $f_{11,Q_{11}}$ and $f_{11,Q'_{11}}$

Consider the points $Q_{11} := (x_{11}, y_{11})$ and $Q'_{11} := (x'_{11}, y'_{11})$ on BLS12-381 with the coordinates

x_{11} = 0x19b3e2c8c6bbf59d3c326b531fc1e639d29200c28624ac604f251a12908c9b7f735318617f625954cc71cdf03229b1ef,

y_{11} = 0x42fcc94d6c6440d3c0a01177616b72eb6972d90e36e88b5981a05a52ab48ce3e22fa88d10f6de85a7d386aad66c0ca8,

x'_{11} = 0xb9529a7b23788075a6c33c7b77b3dcf4da4f58af5310f32e739a6c653a5a8f7cf7f19a297bd6a8f3f19ea82cf9419,

y'_{11} = 0x2ecc645926cbd45f215b3fa17df0d7a50e5814f9631c502f2b2c2457926089a452bd11bf89ee72baa1981f99f88acb2.

They were generated by clearing the cofactor $er/11$ for $(4, 2\sqrt{17})$ and $(5, \sqrt{129})$, the curve points with the smallest x-coordinates and whose orders are divided by 11. Moreover, the points Q_{11}, Q'_{11} form a basis of $E[11]$, since their Weil pairing turns out to be non-trivial.

The precomputed lines $\ell_{i,j}$, v_i related to Q_{11} are as follows:

$\ell_{1,1} =$ 0x52084a813637c846209a6bd185637a4f3064ed2a6842bf0bd880b1f59b0c20bd849bfe974d911a8298449f291164bdd$\cdot x + y +$ 0x177945f53bd9ccccf92efbde018f0231394f35d08d5ec12f3ada6582e07b503ffbd3ef06b5897d39ebb8659179e2d724,

$\ell_{4,1} =$ 0x13c891a2eb0efb385efd6c30a970b8ba2476e5e336ca503655a3f4639afa20ee93a66bc4e3292af389c08375b7f7ca11$\cdot x + y +$ 0x12fe29f8433353a1efbb127d92e74b33ef62de088fd63e32e371dc7fdb8fe8c441f43f2e23f272ee147fe8159079192c,

$\ell_{2,2} =$ 0x1629665baebce7a5e15fd940a207d30b892abbf11839e597e88111ed040ef83e228aa158afea8d4b940e1a333d1a330f$\cdot x + y +$ 0x18afe2760ca5e1821765ba9f63fb695f6141bf348af097618f548b48bc3664304750705638722 5d782990d74698b2245,

$\ell_{5,5} =$ 0xc56f430391b7b504419357c8377df95e6852c5c4dde312c11da4caa0fbb894aa9ae785edebb1da4425ad3738fc71d9e$\cdot x + y +$ 0xe6dd0ee7e55161d51c168b1a4dc84a7cde75831ceec4bf58416295c6b483568bbaa2a4d5c4f2f7fc984c70320156fec,

$v_2 = x +$ 0x175870c9f78d5b3c8080769771854a65aee01f4ce1c7bab7a8e07219383b0dd70bf3b6672721aa6c965c809e3129aef2,

$v_4 = x +$ 0x12f69088512349670c11c7ce2b1917bc1a8e981440a99f9f235bc0f987fd00e719a4d68437eb80c91ace7614fa08578d,

$v_5 = x +$ 0x29cbec7fa11916ea27cafdf2e570b2821d1457fa9fe5f4a06cbd0d7964735f364f671311150a5d43a425438208a5534.

In turn, the precomputed lines $\ell_{i,j}$, v_i related to Q'_{11} are as follows:

$\ell_{1,1} =$ 0xe86fb44c035add8d5bd0c1bbc00746b8f5948f53d19e6f1463e1c2116aee7d8e1e5e52f4243d92e30d81016615a4d8f$\cdot x + y +$ 0x19e29d77082445f62426e681d2164313f0938233dc212252d18bd59fa843866de644e19c23fdca39d4caf5a21b6218ef,

$\ell_{4,1} =$ 0x143a9f359b2e91fde880b224bb02eabe251998d5b9fac7e99344a5f591be3c87307b0d8d34066ed312a406df7a3a7de2$\cdot x + y +$ 0xdea366ab1f6dad2b4e86d4e092d3c60a78cb23409b43afe97572f7c8ea087b37861e527e1074263ceb03442c00c83a5,

$\ell_{2,2} =$ 0xd90c951049c94a6ed1bd28a9478fbf9832e538012959dc03beda2f2db93b8d2ad8f118632faa008092f1bd63dbe79ad$\cdot x + y +$ 0x8a7c2874b1186be6a04eaa25f518a805eee85220d647ca6ca7b55367e283de6f5d2827f8bc139d33d399d7b5eaface,

$\ell_{5,5} =$ 0xe7ffc6cda09c9a1836d575029f4f18ef073d1f62da3e9966afd7d4c8d6393c67d0ed253d72414922367c7096429c8ca$\cdot x + y +$ 0x861494ad62ca8ff789de6c6510352d4f507d3f52aa71432b6d2675f600318b237ce30295749ac0525a141e0b38b2ea1,

$v_2 = x + \texttt{0x1980d2dd2f8125ffb528b941ae1970c1a315247876b662c4e1c9e95f30452703b0396aa158a04cd163b11cdf62987e06}$,

$v_4 = x + \texttt{0xe1636df6805a24e45371392a93bd29875131d336ed3525c9bc2a5d2992412918c898f91b13450ebd925f8d6d2e36887}$,

$v_5 = x + \texttt{0x1314a66019d8da69008ff230b23b95e951d0ae4ea9dca5a58f5678b10105345fa00b065f5c465df9d71d7ea84d702b13}$.

B General comparison of BLS12-381 and the new BLS12 curves

In Table 5 we compare the new curves BLS12-377 and BLS12-376 to the widely-used one BLS12-381 with respect to common elliptic curve operations. All benchmarks were run on a AWS `z1d.large` machine (Intel(R) Xeon(R) Platinum 8151 CPU @ 3.40GHz) with hyperthreading, turbo and frequency scaling disabled. The performance on the two new curves are quite on par with BLS12-381, except for pairings because of the very small Hamming weight of the BLS12-381 seed.

Operation	BLS12-381	new BLS12-377	new BLS12-376
$\mathbb{G}_1$ GLV scalar multiplication	0.079	0.081	0.078
$\mathbb{G}_2$ GLV scalar multiplication	0.198	0.283	0.239
$\mathbb{G}_1$ MSM ($N = 2^5$)	1.049	1.053	1.041
$\mathbb{G}_1$ MSM ($N = 2^{21}$)	9443.791	9357.049	9363.302
$\mathbb{G}_2$ MSM ($N = 2^5$)	3.049	3.393	3.008
$\mathbb{G}_2$ MSM ($N = 2^{21}$)	25120.589	27640.164	25136.455
Pairing	0.704	0.876	0.812
Pairing product ($N = 2$)	0.836	1.165	1.032
Pairing product ($N = 4$)	1.213	1.463	1.344
Pairing product ($N = 10$)	2.363	3.430	2.703

Table 5: Benchmarks (in milliseconds) of common operations on the curve BLS12-381 and on the new ones BLS12-377, BLS12-376