Secure and Efficient Pairing at 256-Bit Security Level

Yutaro Kiyomura[1(✉)], Akiko Inoue[2], Yuto Kawahara[1], Masaya Yasuda[3], Tsuyoshi Takagi[3], and Tetsutaro Kobayashi[1]

[1] NTT Secure Platform Laboratories, Musashino, Japan
{kiyomura.yutaro,kawahara.yuto,kobayashi.tetsutaro}@lab.ntt.co.jp
[2] NEC Central Research Laboratories, Kawasaki, Japan
a-inoue@cj.jp.nec.com
[3] Kyushu University, Fukuoka, Japan
{yasuda,takagi}@imi.kyushu-u.ac.jp

Abstract. At CRYPTO 2016, Kim and Barbulescu proposed an efficient number field sieve (NFS) algorithm for the discrete logarithm problem (DLP) in a finite field. The security of pairing-based cryptography (PBC) is based on the difficulty in solving the DLP. Hence, it has become necessary to revise the bitlength that the DLP is computationally infeasible against the efficient NFS algorithms. The timing of the main operations of PBC (i.e. pairing, scalar multiplication on the elliptic curves, and exponentiation on the finite field) generally becomes slower as the bitlength becomes longer, so it has become increasingly important to compute the main operations of PBC more efficiently. To choose a suitable pairing-friendly curve from among various pairing-friendly curves is one of the factors that affect the efficiency of computing the main operations of PBC. We should implement the main operations of PBC and compare the timing among some pairing-friendly curves in order to choose the suitable pairing-friendly curve precisely. In this paper, we focus on the five candidate pairing-friendly curves from the Barreto-Lynn-Scott (BLS) and Kachisa-Schaefer-Scott (KSS) families as the 256-bit secure pairing-friendly curves and show the following two results; (1) the revised bitlength that the DLP is computationally infeasible against the efficient NFS algorithms for each candidate pairing-friendly curve, (2) the suitable pairing-friendly curve by comparing the timing of the main operations of PBC among the candidate pairing-friendly curves using the revised bitlength.

1 Introduction

Many pairing-based cryptography (PBC) have been proposed, e.g., ID-based encryption [8,40], attribute-based encryption [41], and functional encryption [38]. A pairing on the elliptic curve is a non-degenerate bilinear map

A. Inoue—This work was done while the author was the student of Kyushu University.

D. Gollmann et al. (Eds.): ACNS 2017, LNCS 10355, pp. 59–79, 2017.
DOI: 10.1007/978-3-319-61204-1_4

$e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$, where $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_3$ are a group with order r respectively. The security of the pairing is based on the difficulty in solving the discrete logarithm problem (DLP) in $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_3$. The group $\mathbb{G}_1$, $\mathbb{G}_2$ are a subgroup on the elliptic curve, and the DLP on an elliptic curve (ECDLP) in $\mathbb{G}_1$, $\mathbb{G}_2$ must be computationally infeasible against the rho algorithm [16,39]. Hence, we should choose r with a secure bitlength against the rho algorithm. The group $\mathbb{G}_3$ is a subgroup on finite field $\mathbb{F}_{p^k}$, where p is a prime and $k \geq 1$ is an embedding degree, and the DLP on a finite field (FFDLP) in $\mathbb{G}_3$ must be computationally infeasible against the number field sieve (NFS) algorithms. There are various NFS algorithms (e.g. Classical-NFS [25], tower NFS (TNFS) [10,45], and special NFS (SNFS) [10,26]). We should choose p and k with a secure bitlength of p^k against the NFS algorithms.

The recommended bitlength of p^k of the pairing was discussed in the 2013 report of ENISA [15, Table 3.6], where the pairing and RSA have the same recommended bitlength. This was in accordance with a general belief stated, for example, by Lenstra: "An RSA modulus n and a finite field $\mathbb{F}_{p^k}$ therefore offer about the same level of security if n and p^k are of the same order of magnitude" [32, Sect. 5.1]. The recommended bitlength of RSA was derived from the complexity of the NFS algorithm for integer factorization [33]. In other words, the bitlength of p^k of the pairing was estimated considering the complexity of this NFS algorithm.

At CRYPTO 2016, Kim and Barbulescu proposed an efficient NFS algorithm called the extended tower number field sieve (exTNFS) algorithm [28]. This NFS algorithm greatly impacted the security of the mainstream pairing such as optimal ate pairing [47]. The complexity of the exTNFS algorithm was reduced from that of previous NFS algorithms by using the trivial equation $\mathbb{F}_{p^k} = \mathbb{F}_{p^{\eta\kappa}}$, where $\gcd(\eta, \kappa) = 1$. Kim and Barbulescu concluded that the bitlength of the pairing should increase roughly twice [28]. Therefore, we should revise to estimate the secure bitlength of p^k against the exTNFS in detail. Note that Menezes *et al.* estimated the bitlength of p^k for the pairing considering the exTNFS algorithm at 128- and 192-bit security levels [37].

Generally, faster timing of the main operations of PBC (i.e. pairing, scalar multiplication on the elliptic curves, and exponentiation on the finite field) is preferred to implement the PBC. To choose a suitable pairing-friendly curve from among various pairing-friendly curves is one of the factors that affect the efficiency of computing the main operations of PBC. Among the studies conducted before the exTNFS algorithm was proposed, Scott [44] theoretically chose the suitable pairing-friendly curve at each security level based on the bitlength of r, p^k and ρ-value given in Freeman *et al.'s* taxonomy [16]. However, Aranha *et al.* [3] discussed the suitable pairing-friendly curve different from that chosen theoretically by comparing the timing of the pairing among several pairing-friendly curves at 192-bit security level. To choose a suitable pairing-friendly curve at a certain security level, it is important to not only choose theoretically but also compare the timing of the main operations of PBC.

Our Contributions. Our goal with this paper is to obtain a secure and efficient pairing at 256-bit security level. To achieve this, our contribution is to revise the estimation of the bitlength of p^k due to the efficient NFS algorithms (e.g. Special exTNFS, Special TNFS) and choose the suitable pairing-friendly curve for efficiently computing the main operations of PBC. We focus on the Barreto-Lynn-Scott (BLS) [11] and Kachisa-Schaefer-Scott (KSS) [29] families that have high embedding degree and are easy to implement the pairing. We specifically choose the following five candidate pairing-friendly curves at the 256-bit security level; the BLS-k with $k = 24, 42, 48$ and KSS-k with $k = 32, 36$. For these curves, we estimate the secure bitlength p^k in detail against the efficient NFS algorithms by comparing the upper bound of norms of these algorithms using the Kim and Barbulescu's estimation method [28]. Furthermore, based on the revised bitlength of p^k, we search for the specific parameter of each candidate pairing-friendly curve to implement the main operations of PBC, and then compare the timing of these operations among the five candidate pairing-friendly curves. Finally, we show the suitable pairing-friendly curve at 256-bit security level.

2 Overview of Pairing

2.1 Definition and Properties

Let p be a prime and E be an elliptic curve defined over the finite field $\mathbb{F}_p$. Let r be a prime with $\gcd(p, r) = 1$. An embedding degree k is the smallest positive integer with $r \mid p^k - 1$. Let $\mathbb{G}_1$, $\mathbb{G}_2$ be a subgroup on the elliptic curve with order r and $\mathbb{G}_3$ be a subgroup on the finite field $\mathbb{F}_{p^k}$ with order r. A pairing e is defined by $e : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_3 \ ; \ (P, Q) \longmapsto f_{r,P}(Q)^{(p^k-1)/r}$, where the rational function $f_{r,P}$ satisfies $\mathrm{div}(f_{r,P}) = r(P) - r(\mathcal{O})$ for the point at infinity $\mathcal{O}$. For $P \in \mathbb{G}_1$, $Q \in \mathbb{G}_2$ and $a \in \mathbb{Z}$, a pairing e has the following properties;

- bilinearity: $e(aP, Q) = e(P, aQ) = e(P, Q)^a$,
- non-degeneracy: for all $P \in \mathbb{G}_1$, $e(P, Q) = 1$ then $Q = \mathcal{O}$ and for all $Q \in \mathbb{G}_2$, $e(P, Q) = 1$ then $P = \mathcal{O}$,
- efficiently computable: $e(P, Q)$ can be efficiently computed.

2.2 Optimal Ate Pairing

The optimal ate pairing proposed by Vercauteren [47] is the most efficient method of computing the pairing e. There are many implementation results of the optimal ate pairing [3,9,13,36,44]. Let m be an integer such that $r \nmid m$. Let $\lambda = mr$ and write $\lambda = \sum_{i=0}^{\omega} \alpha_i p^i$ where $\omega = \lfloor \log_p \lambda \rfloor$. Let $E[r]$ be an r-torsion subgroup. Define $\mathbb{G}_1 = E[r] \cap \mathrm{Ker}(\pi_p - [1]) = E(\mathbb{F}_p)[r]$, $\hat{\mathbb{G}}_2 = E[r] \cap \mathrm{Ker}(\pi_p - [p]) \subseteq E(\mathbb{F}_{p^k})[r]$ as the subgroup with r. Let E' be a twist of degree d of E with $\psi : E' \to E$ defined over $\mathbb{F}_{p^d}$, and define $\mathbb{G}_2 = \psi^{-1}(\hat{\mathbb{G}}_2)$. Note that d

depends on the pairing-friendly curve and is in $\{2, 3, 4, 6\}$ [24]. An optimal ate pairing a_k is defined by

$$a_k : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_3,\ (P, Q) \longmapsto \left(\prod_{i=0}^{\omega} f_{\alpha_i, Q}^{p^i}(P) \cdot \prod_{i=0}^{\omega-1} \frac{\ell_{[\beta_{i+1}]Q, [\alpha_i p^i]Q}(P)}{v_{[\beta_i]Q}(P)} \right)^{\frac{p^k-1}{r}} \quad (1)$$

where $\beta_i = \sum_{j=i}^{\omega} \alpha_j p^j$, $\ell_{T,T'}$ is the line through T and T', and v_T is the vertical line through T, where T and T' are points on the elliptic curve.

3 Candidate Pairing-Friendly Curves at 256-Bit Security Level

In this section, we choose the five candidate pairing-friendly curves satisfying the security and efficiency from the BLS [11] and KSS [29] families to choose the suitable pairing-friendly curve at 256-bit security level. In this paper, we define $\mathrm{len}(x)$ as the bitlength of x.

3.1 How to Choose Candidate Pairing-Friendly Curves

We show the security against the ECDLP and FFDLP and the efficiency for implementation of the main operations of PBC. An embedding degree k is determined by the chosen pairing-friendly curve, and the primes r and p are represented by the polynomial of a positive integer x.

Security. The parameters r, p, and k should satisfy the complexity of solving the DLP in $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_3$ to achieve the $\mathcal{K}$-bit security level. The definition of ECDLP in $\mathbb{G}_1$ and $\mathbb{G}_2$ is as follows. Given points $G, Y \in \mathbb{G}_1$ (or $\mathbb{G}_2$), find $x \in \mathbb{Z}$ such that $Y = xG$. An efficient algorithm for solving the ECDLP is the rho algorithm [16,39], which has the complexity of $\mathcal{O}(\sqrt{r})$. Therefore, we should choose the bitlength of r with $\mathrm{len}(r) \geq 2\mathcal{K}$. The definition of the FFDLP in $\mathbb{G}_3$ is as follows. Given points $g, y \in \mathbb{G}_3$, find $x \in \mathbb{Z}$ such that $y = g^x$. An efficient algorithm for solving the FFDLP are the STNFS [10,26] and SexTNFS [28] algorithms. We give the bitlength of p^k which the FFDLP is computationally infeasible against these NFS algorithms in Sect. 5.

Efficiency. To efficiently compute the main operations of PBC (i.e. pairing, scalar multiplication in $\mathbb{G}_1$ and $\mathbb{G}_2$, and exponentiation in $\mathbb{G}_3$) with the above security, we consider the following conditions as affecting the efficiency of these operations.

- $\mathrm{len}(r)$ and $\mathrm{len}(p^k)$ are as small as possible.
- The ρ-value is approximately 1 ($\rho = \log p / \log r$).
- Parameter x in polynomials (e.g. $p(x)$, $r(x)$) has a low Hamming weight.
- The embedding degree k has the form $k = 2^i \cdot 3^j$ ($i \in \mathbb{Z}_{\geq 1}, j \in \mathbb{Z}_{\geq 0}$).
- The twist of degree d is 6 ($d = 6$ is maximum of degree).

These conditions are theoretically efficient ones, then the effect of each condition is uncertain in the implementation.

Table 1. Parameters for the five candidate pairing-friendly curves

BLS-24 [16, Construction 6.6]	$k = 24,\ \rho = 1.250,\ \deg(p(x)) = 10,\ \varphi(k) = 8,$ $p(x) = (x-1)^2(x^8 - x^4 + 1)/3 + x,\ r(x) = x^8 - x^4 + 1,\ t(x) = x + 1$
KSS-32 [29, Example 4.4]	$k = 32,\ \rho = 1.063,\ \deg(p(x)) = 18,\ \varphi(k) = 16,$ $p(x) = (x^{18} - 6x^{17} + 13x^{16} + 57120x^{10} - 344632x^9 + 742560x^8$ $+815730721x^2 - 4948305594x + 10604499373)/2970292,$ $r(x) = (x^{16} + 57120x^8 + 815730721)/(2 \cdot 13^8 \cdot 239^2),$ $t(x) = (-2x^9 - 56403x + 3107)/3107$
KSS-36 [29, Example 4.5]	$k = 36,\ \rho = 1.167,\ \deg(p(x)) = 14,\ \varphi(k) = 12,$ $p(x) = (x^{14} - 4x^{13} + 7x^{12} + 683x^8 - 2510x^7 + 4781x^6 + 117649x^2$ $-386569x + 823543)/28749,$ $r(x) = (x^{12} + 683x^6 + 117649)/(7^6 \cdot 37^2),\ t(x) = (2x^7 + 757x + 259)/259$
BLS-42 [16, Construction 6.6]	$k = 42,\ \rho = 1.333,\ \deg(p(x)) = 16,\ \varphi(k) = 12,$ $p(x) = (x-1)^2(x^{14} - x^7 + 1)/3 + x,$ $r(x) = x^{12} + x^{11} - x^9 - x^8 + x^6 - x^4 - x^3 + x + 1,\ t(x) = x + 1$
BLS-48 [16, Construction 6.6]	$k = 48,\ \rho = 1.125,\ \deg(p(x)) = 18,\ \varphi(k) = 16,$ $p(x) = (x-1)^2(x^{16} - x^8 + 1)/3 + x,\ r(x) = x^{16} - x^8 + 1,\ t(x) = x + 1$

* deg(): degree of polynomial, $\varphi()$: Euler function

3.2 Selection of Candidate Pairing-Friendly Curves

In this subsection, we decide the candidate pairing-friendly curves from the BLS [11] and KSS [29] families to choose the suitable pairing-friendly curve at 256-bit security level based on Sect. 3.1. We focus on the BLS and KSS families that have high embedding degree and can be easy to construct the pairing. We specifically choose the following five candidate pairing-friendly curves at the 256-bit security level; the BLS-k with $k = 24, 42, 48$ and the KSS-k with $k = 32, 36$. In the case of the BLS-k, the small $\mathrm{len}(r)$ and $\mathrm{len}(p^k)$ in the BLS-42 and BLS-48 can be choose because these curve have high embedding degree k, and the implementation results in the BLS-24 exists [9,44]. In the case of the KSS-k, the KSS-36 has the small $\mathrm{len}(r)$, the KSS-32 has small ρ-value and simple tower construction for $\mathbb{F}_{p^k}$ since $k = 32 = 2^5$.

The detail of the five candidate pairing-friendly curves are as follows; the curves with $6 \mid k$ are defined by $E/\mathbb{F}_p : y^2 = x^3 + b$, and has the complex multiplication discriminant $D = 3$ and $d = 6$, the curves with $4 \mid k$ is defined by $E/\mathbb{F}_p : y^2 = x^3 + ax$, and has $D = 1$ and $d = 4$. Table 1 shows the parameters $p(x)$, $r(x)$, $t(x)$, k, ρ-value, $\deg(p(x))$, and Euler function $\varphi(k)$ for each curves. The parameters $n(x)$ and $f(x)$ satisfy $n(x) = p(x) + 1 - t(x)$ and $4p(x) - t(x)^2 = Df(x)^2$, respectively.

4 Overview of Number Field Sieve and Its Variants

In this section, we give an overview of the NFS algorithm and its variants to revise the bitlength of the five candidate pairing-friendly curves introduced in Sect. 3.2.

The FFDLP is classified into three cases by size of p: small, medium, or large. In medium and large cases, the NFS algorithms is the most efficient algorithm for solving the FFDLP. To accurately classify p, let $p = L_{p^k}(l_p, c_p)$, where $L_{p^k}(l_p, c_p) = \exp((c_p + o(1))(\log p^k)^{l_p}(\log\log p^k)^{1-l_p})$. $o(1)$ becomes 0 when

$p^k \to \infty$. The prime p is called medium if $1/3 < l_p < 2/3$, large if $2/3 < l_p < 1$, boundary if $l_p = 2/3$.

Note that the above L_{p^k}-notation is just an asymptotic value. If we fix the value of p^k, the L_{p^k}-notation has a constant value c such that $c \times \exp((c_p + o(1))(\log p^k)^{l_p}(\log\log p^k)^{1-l_p})$ and $o(1) \neq 0$. Therefore, when we substitute the concrete value for p^k in L_{p^k}-notation, it is important to evaluate c and $o(1)$.

The NFS algorithms for solving the FFDLP are classified into three types: Classical-NFS, TNFS, and exTNFS, according to their mathematical constructions. The Classical-NFS algorithm was proposed in 2006, and the complexities are $L_{p^k}(1/3, (128/9)^{1/3})$ and $L_{p^k}(1/3, (64/9)^{1/3})$ in the medium and large cases, respectively. The TNFS algorithm was proposed in 1999 and later applied to the large case in 2015, where the complexity of the TNFS algorithm is also $L_{p^k}(1/3, (64/9)^{1/3})$ in the large case. Finally, the exTNFS algorithm proposed in 2015 is the generalization of combining the Classical-NFS and TNFS algorithms, and its complexities in medium and large cases are $L_{p^k}(1/3, (64/9)^{1/3})$.

4.1 Extended TNFS and Special-NFS Algorithms

In this subsection, we explain the exTNFS algorithm [28], which is effective for solving the FFDLP. We then give an overview of the Special-NFS algorithm. The NFS algorithms (not specified for exTNFS) are divided into the following four steps: 1. polynomial selection, 2. relation collection, 3. linear algebra, and 4. individual logarithm.

exTNFS Algorithm. We can use the exTNFS algorithm when the extension degree k is composite. Let $k = \eta\kappa$. We select an irreducible polynomial $h(t) \in \mathbb{Z}[t]$ over $\mathbb{Q}$ and $\mathbb{F}_p$ whose degree is η. We construct $\mathbb{Q}(\iota) = \mathbb{Q}[t]/h(t)$ and put $R = \mathbb{Z}[t]/h(t) \subset \mathbb{Q}(\iota)$.

Note that the Classical-NFS algorithm [25] is the case in which $R = \mathbb{Z}$ in the exTNFS algorithm, and the TNFS algorithm [45] is the case in which $\deg h = n$ in the exTNFS algorithm.

Polynomial Selection. We select polynomials f_1 and $f_2 \in R[X]$ that satisfy the condition that $f_1 \mod p$ and $f_2 \mod p$ have a common factor $\varphi(X)$ of degree κ, which is irreducible over $\mathbb{F}_{p^\eta} = R/pR$. In this section, $i \in \{1, 2\}$. Let K_i be the number fields defined by f_i above the fraction field of R and $\mathcal{O}_i$ be the integer ring of K_i. We denote the roots of f_i in $\mathbb{C}$ by θ_i and the degree of f_i by d_i. We then obtain two maps from $R[X]$ to $(R/pR)[X]/\varphi(X) \cong \mathbb{F}_{p^k}$.

Relation Collection. We select smoothness bound $B \in \mathbb{N}$ and define factor base $\mathcal{F}_i$ as follows: $\mathcal{F}_i = \{(\mathfrak{q}, \theta_i - \gamma) : \mathfrak{q} : \text{prime in } \mathbb{Q}(\iota) \text{ lying over a prime } p \leq B, f_i(\gamma) \equiv 0 \mod \mathfrak{q}\}$. We then obtain $a - bX \in R[X]$ by selecting $(a(t), b(t)) \in R^2$. The coefficients of $a(t)$ and $b(t)$ are bounded by A. Let $E = A^\eta$ be the sieve parameter. The norm of $a - b\theta_i$ in K_i is expressed as follows.

$$\mathcal{N}_{K_i/\mathbb{Q}}(a - b\theta_i) = \left| \mathrm{Res}\left(h(t), \sum_{j\in[0,d_i]} f_{i,j} a(t)^j b(t)^{d_i - j} \right) \right|,$$

where $f_{i,j}$ is the coefficient of polynomial $f_i = \sum_{j=0}^{d_i} f_{i,j} X^j$. When $\mathcal{N}_{K_1/\mathbb{Q}}(a - b\theta_1)$ and $\mathcal{N}_{K_2/\mathbb{Q}}(a - b\theta_2)$ are B-smooth, the $(a(t), b(t))$ pair is called a double smooth pair (an integer is B-smooth if the largest prime factor is less than B). When $(a(t), b(t))$ is a double smooth pair, $(a - b\theta_1)$ and $(a - b\theta_2)$ can be factored into the prime ideal in $\mathcal{O}_1$ and $\mathcal{O}_2$ using only the elements of $\mathcal{F}_1$ and $\mathcal{F}_2$, respectively. Therefore, we obtain the following notation: $(a - b\theta_i) = \prod_{\mathfrak{p}\in\mathcal{F}_i} \mathfrak{p}^{\mu_\mathfrak{p}}$ and the following relation up to units.

$$\begin{aligned} \phi_1((a - b\theta_1)) = \phi_2((a - b\theta_2)) \ \text{ in } \ \mathbb{F}_{p^k} &\iff \phi_1\Big(\prod_{\mathfrak{p}\in\mathcal{F}_1} \mathfrak{p}^{\mu_\mathfrak{p}}\Big) = \phi_2\Big(\prod_{\mathfrak{q}\in\mathcal{F}_2} \mathfrak{q}^{\mu_\mathfrak{q}}\Big) \\ &\iff \prod_{\mathfrak{p}\in\mathcal{F}_1} \phi_1(\mathfrak{p})^{\mu_\mathfrak{p}} = \prod_{\mathfrak{q}\in\mathcal{F}_2} \phi_2(\mathfrak{q})^{\mu_\mathfrak{q}} \end{aligned}$$

Thus, this leads to

$$\begin{aligned} &\sum_{\mathfrak{p}\in\mathcal{F}_1} \mu_\mathfrak{p} \log \phi_1(\mathfrak{p}) + \sum_j \lambda_{1,j} \log \Lambda_{1,j} \\ &= \sum_{\mathfrak{q}\in\mathcal{F}_2} \mu_\mathfrak{q} \log \phi_2(\mathfrak{q}) + \sum_j \lambda_{2,j} \log \Lambda_{2,j} \mod p^k - 1, \end{aligned} \tag{2}$$

where $\log \phi_1(\mathfrak{p})$, $\log \phi_2(\mathfrak{q})$, $\log \Lambda_{1,j}$ and $\log \Lambda_{2,j}$ are the unknowns called virtual logarithms [46], and $\lambda_{1,j}$ and $\lambda_{2,j}$ are computable values called character maps to distinguish the difference in units. Let N_λ be the number of character maps. When we collect more than $|\mathcal{F}_1| + |\mathcal{F}_2| + N_\lambda$ double smooth pairs $(a(t), b(t))$, we obtain the relations of (2).

In this section, we collect double smooth pairs (a, b), but it is possible to collect double smooth tupples $(a_1, a_2, \cdots, a_\tau)$. We call parameter τ a sieve dimension.

Linear Algebra. In collecting adequate relations in the previous step, we can construct and solve the simultaneous congruence. We obtain the values of the virtual logarithms $\log \phi_1(\mathfrak{p})$, $\log \phi_2(\mathfrak{q})$, $\log \Lambda_{1,j}$, and $\log \Lambda_{2,j}$.

Individual Logarithm. Finally, we compute the target logarithm x from the values of the virtual logarithms.

Special-NFS Algorithms. We collectively call three NFS algorithms (exTNFS, Classical-NFS, and TNFS) as the General NFS (GNFS) algorithms. The GNFS algorithms can be applied for special polynomial selection when p has a special form. The special cases of the GNFS algorithms are called Special-NFS (SNFS) algorithms. We consider the SNFS algorithms for solving the FFDLP to estimate the security of the pairing where p has a special form.

4.2 Larger Norm Implies Higher Complexity

In this section, we give an overview on the complexity of the NFS algorithms. The main steps to evaluate this complexity are as follows.

– We evaluate the upper bound of norms and probabilities in which the norms are B-smooth.

The smaller the upper bound of norms is, the higher the probability the norms are B-smooth. Therefore, we have to select polynomials so that the upper bounds of norms become small.

– We set the parameters appropriately so that we can collect adequate double smooth pairs from the sieve region.

The sieve region is the region to collect relations. When the sieve degree is τ, the sieve region is E^τ. We set the appropriate parameters to satisfy the inequality that $E^\tau \times$ (the probability of B-smooth of norm's upper bound) $\leq$ (the number of double smooth pairs we collect) $= B^{1+o(1)}$.

– Relation collection and linear algebra have the same complexities.

The whole complexity of NFS algorithms is the sum of the complexities of following two steps: relation collection and linear algebra. In the exTNFS algorithm, the complexity of relation collection is $O(E^2)$. We need to evaluate sizes of parameters because of the trade-off between relation collection and linear algebra.

When the norm is small, the probability that norms are B-smooth becomes high. We can obtain relations with a few trials. The complexity of relation collection becomes small, and the whole complexity becomes small. That is, the decrease in norms implies the reduction in the security of cryptography, which is based on the difficulty of the FFDLP. Therefore, we can estimate bitlengths by comparing the sizes of norms.

4.3 Comparing Norms of NFS Algorithms by Using Kim and Barbulescu's Estimation Method

We refer to the method of comparing norms in [28] to estimate and compare the norms of various NFS algorithms. The norms of each GNFS algorithm are listed in Table 2, and the norms of each SNFS algorithm are listed in Table 3 (part of Table 2 is omitted).

In Tables 2 and 3, E_G is the sieve parameter of the GNFS algorithms and E_S is the sieve parameter of the SNFS algorithms. In addition, d is the degree of polynomial selected in the step of polynomial selection. Note that d in SNFS algorithm is equal to the degree of p. The τ is the sieve dimension, and others are parameters used in each NFS algorithm. Sieve parameter E depends on the implementations. Kim and Barbulescu [28] used the formula

$$E_G = c_G L_{p^k}\left(\frac{1}{3}, \left(\frac{8}{9}\right)^{1/3}\right), E_S = c_S L_{p^k}\left(\frac{1}{3}, \left(\frac{4}{9}\right)^{1/3}\right).$$

Table 2. Norm sizes of GNFS algorithms

Algorithm	Norm product
NFS-JLSV$_1$	$E_G^{\frac{4k}{\tau}}(p^k)^{\frac{\tau-1}{k}}$
TNFS	$E_G^{\frac{2(d+1)}{\tau}}(p^k)^{\frac{2(\tau-1)}{d+1}}$
exTNFS-Conj	$E_G^{\frac{6\kappa}{\tau}}(p^k)^{\frac{\tau-1}{2\kappa}}$

Table 3. Norm sizes of SNFS algorithms

Algorithm	Norm product
STNFS	$E_S^{\frac{2(d+1)}{\tau}}(p^k)^{\frac{\tau-1}{d}}$
SNFS-JP	$E_S^{\frac{2k(d+1)}{\tau}}(p^k)^{\frac{\tau-1}{kd}}$
SexTNFS	$E_S^{\frac{2\kappa(d+1)}{\tau}}(p^k)^{\frac{\tau-1}{\kappa d}}$

They determined $\log_2 c_G \approx -4.30$ using the results of three implementations [6,7,14]. Similarly, they determined $\log_2 c_S \approx -4.27$ using the results of an implementation [1]. After the values of E_G and E_S are determined, other parameters must be determined. The parameters, except τ, are computed using the theoretical optimal values. Then τ is determined as the best value in their bitsize of the norm.

5 Revise the Bitlength for Candidate Pairing-Friendly Curves

In this section, we revise to estimate the bitlengths for the five pairing-friendly curves (i.e. the BLS-k with $k = 24, 42, 48$ and KSS-k with $k = 32, 36$) at 256-bit security level by using the norms of NFS algorithms in the previous section.

5.1 Revised Estimation of Bitlength for BLS-48

In this subsection, we revised to estimate the bitlength for the BLS-48. We compare the norms of NFS algorithms based on the constants c_G and c_S and estimate the bitlength based on the initial norm of the GNFS algorithms at the 256-bit security level. The estimations for other pairing-friendly curves (i.e. the BLS-24, KSS-32, KSS-36 and BLS-42) are described in the Appendix A.

Determining Constants c_G and c_S. Before plotting norms, the constant values of c_G and c_S must be evaluated. As previously mentioned, c_G and c_S are evaluated from the implementation results. We discuss these values by adding new implementation results. In Kim and Barbulescu's study [28], $\log_2 c_G \approx -4.30$ and $\log_2 c_S \approx -4.27$; however, we evaluate c_G and c_S by adding to new results. First, we evaluate c_G using the result from Kleinjung [31] who solved the DLP in $\mathbb{F}_p$. We extrapolate from the pair ($\log_2 p^k = 768$, $\log_2 E_G \approx 35$) Kleinjung used [31] and obtain $\log_2 c_G \approx -3.26$. The sieve parameter E_G using Kleinjung's result is larger than that Kim and Barbulescu evaluated. Because E_G by Kim and Barbulescu [28] is evaluated more strictly, we plot the norms of the GNFS algorithms using $\log_2 c_G \approx -4.30$ they used. Next, we evaluate c_S using the results by Fried *et al.* [17] and Guillevic *et al.* [22]. We extrapolate from the pair ($\log_2 p^k = 1024$, $\log_2 E_S \approx 31$) used by Fried *et al.* [17] and obtain $\log_2 c_S \approx -3.43$. We also

extrapolate from the pair ($\log_2 p^k = 510$, $\log_2 E_S \approx 26$) used by Guillevic *et al.* [22] and obtain $\log_2 c_S \approx 0.67$. The sieve parameters E_S using the results from Fried *et al.* and Guillevic *et al.* are larger than those Kim and Barbulescu evaluated. As with E_G, because E_S from Kim and Barbulescu [28] is evaluated more strictly, we plot the norms of the SNFS algorithms using $\log_2 c_S \approx -4.27$ they used.

Initial Norm of General NFS Algorithms at 256-Bit Security Level. We define the initial norm as the norm of the GNFS algorithm for integer factorization, which corresponds to the bitlength at 256-bit security level. Let N be a composite number. The norm of this GNFS algorithm is $E_G^{d+1} N^{2/d+1}$, where d is the degree of the polynomial selected in the step of polynomial selection. The recommended bitlength of RSA at 256-bit security level is 15360-bit [5]. When N is 15360-bit, the optimal value of d is 15. We substitute the value of c_G, which we evaluated in the previous section, for the norm, and the initial norm is about 4006-bit.

Fastest Variant of NFS Algorithms for BLS-48. We concretely estimate the bitlengths of the PBC. We give details of the BLS-48, and the other curves are mentioned in the Appendix A. We fix the extension degree to $k = 48$ and examine the fastest NFS algorithm that solves the DLP in $\mathbb{F}_{p^{48}}$, where p is expressed by the BLS-48.

First, we plot the norms of three GNFS algorithms (i.e. NFS-JLSV$_1$, TNFS, and exTNFS-Conj) in Fig. 1. The E_G is as follows;

$$E_G = 2^{-4.3} L_{p^k}\left(\frac{1}{3}, \left(\frac{8}{9}\right)^{\frac{1}{3}}\right).$$

- NFS-JLSV$_1$. The norm of the NFS-JLSV$_1$ algorithm is expressed as $E_G{}^{\frac{4k}{\tau}}(p^k)^{\frac{\tau-1}{k}}$. The optimal value of τ is 9 when the norm is $E_G{}^{\frac{192}{\tau}}(p^k)^{\frac{\tau-1}{48}}$.
- TNFS. The norm of the TNFS algorithm is expressed as $E_G{}^{\frac{2(d+1)}{\tau}}(p^k)^{\frac{2(\tau-1)}{d+1}}$. The d is 15 using the formula $d = \sqrt[3]{3}\left(\log p^k / \log\log p^k\right)^{1/3}$. The optimal value of τ is 2 when the norm is $E_G{}^{\frac{32}{\tau}}(p^k)^{\frac{2(\tau-1)}{16}}$
- exTNFS-Conj. The norm of the exTNFS-Conj algorithm is expressed as $E_G{}^{\frac{6\kappa}{\tau}}(p^k)^{\frac{\tau-1}{2\kappa}}$. Kim and Barbulescu [28] used the case of $\gcd(\eta, \kappa) = 1$. However, Kim and Jeong proposed an algorithm allowing the choosing of η and κ freely from the co-primality condition, and their algorithm has the same complexities as when η and κ are co-prime [30]. Therefore, we use all cases of (η, κ). When $n = 48$, we can consider the cases of $(\eta, \kappa) = (2, 24)$, $(3, 16)$, $(4, 12)$, $(6, 8)$, $(8, 6)$, $(12, 4)$, $(16, 3)$, $(24, 2)$. When $(\eta, \kappa) = (2, 24)$, $(3, 16)$, $(4, 12)$, $(6, 8)$, $(8, 6)$, $(12, 4)$, $(16, 3)$, $(24, 2)$, the optimal value of τ is 7, 5, 4, 3, 2, 2, 2, 2, respectively.

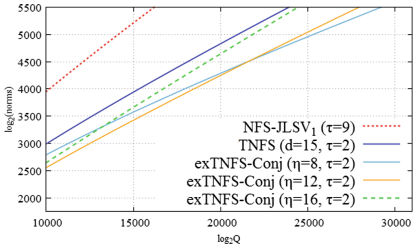

Fig. 1. Norms of GNFS algorithms in $\mathbb{F}_{p^{48}}$

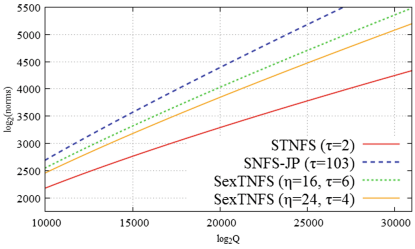

Fig. 2. Norms of SNFS algorithms in $\mathbb{F}_{p^{48}}$

Next, we plot the norms of three SNFS algorithms (i.e. STNFS, SNFS-JP, and SexTNFS) in Fig. 2. In the BLS-48, $\deg(p(x)) = 18$. The E_S is as follows:

$$E_S = 2^{-4.3} L_{p^k}\left(\frac{1}{3}, \left(\frac{4}{9}\right)^{\frac{1}{3}}\right)$$

- STNFS. The norm of the STNFS algorithm is expressed as $E_G{}^{\frac{2(d+1)}{\tau}}(p^k)^{\frac{\tau-1}{d}}$. The optimal value of τ is 2 when the norm is $E_G{}^{\frac{38}{\tau}}(p^k)^{\frac{\tau-1}{18}}$.
- SNFS-JP. The norm of the SNFS-JP algorithm is expressed as $E_G{}^{\frac{2k(d+1)}{\tau}}(p^k)^{\frac{\tau-1}{kd}}$. The optimal value of τ is 103 when the norm is $E_G{}^{\frac{1824}{\tau}}(p^k)^{\frac{\tau-1}{864}}$.
- SexTNFS. The norm of the SexTNFS algorithm is expressed as $E_G{}^{\frac{2\kappa(d+1)}{\tau}}(p^k)^{\frac{\tau-1}{\kappa d}}$. We also use the all case of (η, κ). When $(\eta, \kappa) = (2, 24)$, $(3, 16)$, $(4, 12)$, $(6, 8)$, $(8, 6)$, $(12, 4)$, $(16, 3)$, $(24, 2)$, the optimal value of τ is 51, 34, 26, 17, 13, 9, 6, 4, respectively.

In Figs. 1 and 2, the STFNS algorithm has the smallest norm. When the norm is the initial norm of 4006-bit, the bitlength of the STNFS algorithm is 27410-bit. Therefore, we can estimate that the bitlength at 256-bit security level is 27410-bit.

5.2 Revised Bitlength at 256-Bit Security Level

We revise to estimate the bitlength for the five candidate pairing-friendly curves (i.e. the BLS-k with $k = 24, 42, 48$ and KSS-k with $k = 32, 36$) at 256-bit security level based on the Kim and Barbulescu's method [28]. The results are listed in Table 4. According to ENISA [15, Table 3.6], the bitlength of the pairing requires more than 15360-bit to achieve the 256-bit security level [16]. However, our revised estimation shows that it is necessary to increase the bitlength by more than 10000-bit to achieve the 256-bit security level. In other word, it is necessary to approximately multiply the bitlength by 1.7 times to achieve the 256-bit security level.

Table 4. Revised bitlength at 256-bit security level

	BLS-24	KSS-32	KSS-36	BLS-42	BLS-48
$\mathrm{len}(p^k)$	25,990	27,410	28,280	28,150	27,410

6 Comparison of Timing Among Candidate Pairing-Friendly Curves

In this section, we measure and compare the timing of the main operations of PBC among the five candidate pairing-friendly curves using the revised bitlength to show a suitable pairing-friendly curve at 256-bit security level. Our implementation uses the efficient algorithms for computing the main operations of PBC.

6.1 Specific Parameter for Implementation

The specific parameter x_0 for each candidate pairing-friendly curve is required to decide the parameters of each curve in Table 1 and implement the main operations of PBC. The x_0 for the BLS-24 showed in [13] satisfies the revised bitlength of p^k in Table 4, but there are no documents showed the parameter satisfying the revised bitlength of p^k in Table 4 for other curves. Therefore, we should search for x_0 for each KSS-32, KSS-36, BLS-42, and BLS-48.

To efficiently compute the pairing, we search for the specific parameter x_0 with a low Hamming weight and $\mathrm{len}(r) \geq 512$ and $\mathrm{len}(p^k)$ always more than the bitlength in Table 4. Table 6 shows x_0, the bitlength of parameters, and Hamming weight of x_0 for the five candidate pairing-friendly curves. Note that the $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_3$ are better to satisfy subgroup security [12] in order to resist against small-subgroup attacks as an optional security requirement. The information of implementation (i.e. the tower construction, elliptic curve E, twist E' of E, and $\ell_{T,T'}(P)$) are showed in Table 5.

6.2 Our Implemented Algorithms

In this subsection, we give an overview of the implemented efficient algorithms for computing the main operations of PBC. We implement the base field $\mathbb{F}_p$ arithmetic by using the GMP library [18]. Additionally, in the arithmetic of the tower field, we use the lazy reduction technique [42] which can reduce the number of the modulo operations of $\mathbb{F}_p$.

Pairing. The formulas of optimal ate pairing for each candidate pairing-friendly curve are given in Table 7. Note that these formulas can be produced from Eq. (1) and [47, Eq. (9)]. There are two steps involved in computing the optimal ate pairing; the miller loop (ML) $f' = f_{x,Q}(P) \cdot g$, where g is the part other than $f_{x,Q}(P)$ in Table 7, and final exponentiation (FE) $f'^{(p^k-1)/r}$.

Table 5. Information of implementation for the five candidate pairing-friendly curves

BLS-24	Fields	$\mathbb{F}_p \xrightarrow{u^2+1} \mathbb{F}_{p^2} \xrightarrow{v^2+u+1} \mathbb{F}_{p^4} \xrightarrow{w^3+v} \mathbb{F}_{p^{12}} \xrightarrow{z^2+w} \mathbb{F}_{p^{24}}$
	$E,\ E'$	$E/\mathbb{F}_p : y^2 = x^3 + 1,\ E'/\mathbb{F}_{p^4} : y^2 = x^3 - 1/v$
	$\ell_{T,T'}(P)$	$[\underbrace{y_P}_{1} : \overbrace{\underbrace{0}_{w} : \underbrace{0}_{w^2}}^{1} \mid \overbrace{\underbrace{(-\lambda \cdot x_P)u}_{1} : \underbrace{cu}_{w} : \underbrace{0}_{w^2}}^{z}]$
KSS-32	Fields	$\mathbb{F}_p \xrightarrow{u^2+2} \mathbb{F}_{p^2} \xrightarrow{v^2-u} \mathbb{F}_{p^4} \xrightarrow{w^2-v} \mathbb{F}_{p^8} \xrightarrow{z^2-w} \mathbb{F}_{p^{16}} \xrightarrow{s^2-z} \mathbb{F}_{p^{32}}$
	$E,\ E'$	$E/\mathbb{F}_p : y^2 = x^3 + 2x,\ E'/\mathbb{F}_{p^8} : y^2 = x^3 + 2x/w$
	$\ell_{T,T'}(P)$	$[\overbrace{\underbrace{y_P}_{1} : \underbrace{0}_{z}}^{1} \mid \overbrace{\underbrace{-\lambda \cdot x_P}_{1} : \underbrace{c}_{z}}^{s}]$
KSS-36	Fields	$\mathbb{F}_p \xrightarrow{u^2+1} \mathbb{F}_{p^2} \xrightarrow{v^3+u+1} \mathbb{F}_{p^6} \xrightarrow{w^3+v} \mathbb{F}_{p^{18}} \xrightarrow{z^2+w} \mathbb{F}_{p^{36}}$
	$E,\ E'$	$E/\mathbb{F}_p : y^2 = x^3 + 2,\ E'/\mathbb{F}_{p^6} : y^2 = x^3 - 2/v$
	$\ell_{T,T'}(P)$	$[\overbrace{\underbrace{y_P}_{1} : \underbrace{0}_{w} : \underbrace{0}_{w^2}}^{1} \mid \overbrace{\underbrace{(-\lambda \cdot x_P)u}_{1} : \underbrace{cu}_{w} : \underbrace{0}_{w^2}}^{z}]$
BLS-42	Fields	$\mathbb{F}_p \xrightarrow{u^7+2} \mathbb{F}_{p^7} \xrightarrow{v^3+u-1} \mathbb{F}_{p^{21}} \xrightarrow{w^2-v} \mathbb{F}_{p^{42}}$
	$E,\ E'$	$E/\mathbb{F}_p : y^2 = x^3 + 1,\ E'/\mathbb{F}_{p^6} : y^2 = x^3 + 1/(1-u)$
	$\ell_{T,T'}(P)$	$[\overbrace{\underbrace{y_P}_{1} : \underbrace{0}_{v} : \underbrace{0}_{v^2}}^{1} \mid \overbrace{\underbrace{-\lambda \cdot x_P}_{1} : \underbrace{c}_{v} : \underbrace{0}_{v^2}}^{w}]$
BLS-48	Fields	$\mathbb{F}_p \xrightarrow{u^2+1} \mathbb{F}_{p^2} \xrightarrow{v^2+u+1} \mathbb{F}_{p^4} \xrightarrow{w^2+v} \mathbb{F}_{p^8} \xrightarrow{z^3+w} \mathbb{F}_{p^{24}} \xrightarrow{s^2+z} \mathbb{F}_{p^{48}}$
	$E,\ E'$	$E/\mathbb{F}_p : y^2 = x^3 + 1,\ E'/\mathbb{F}_{p^8} : y^2 = x^3 - 1/w$
	$\ell_{T,T'}(P)$	$[\overbrace{\underbrace{y_P}_{1} : \underbrace{0}_{z} : \underbrace{0}_{z^2}}^{1} \mid \overbrace{\underbrace{(-\lambda \cdot x_P)u}_{1} : \underbrace{cu}_{z} : \underbrace{0}_{z^2}}^{s}]$

In the ML, the rational function $f_{x,Q}(P)$ can be computed using Miller's algorithm [34]. The computational cost of the ML is affected by bitlength x_0 and the Hamming weight of x_0. We can reduce the computational cost of the multiplication on $\mathbb{F}_{p^k}$ by using the sparse multiplication technique [35]. We use the affine pairing [2] since the computation of the inversion in $\mathbb{G}_2$ is fast.

In the FE, the equation $f'^{(p^k-1)/r}$ can be broken down into three components by using cyclotomic polynomial Φ_k as follows [43].

$$(p^k - 1)/r = \underbrace{[(p^{k/2} - 1)] \cdot [(p^{k/2} + 1)/\Phi_k(p)]}_{\text{easy part}} \cdot \underbrace{[\Phi_k(p)/r]}_{\text{hard part}}.$$

The computation of the easy part $m = f'^{(p^{k/2}-1)\cdot((p^{k/2}+1)/\Phi_k(p))}$ requires one conjugation, one inversion, some Frobenius operations and some multiplications on $\mathbb{F}_{p^k}$, so the computational cost of the easy part hardly affects that of the whole FE. The hard part can be computed by using the base p representation of $\Phi_k(p)/r$ as

Table 6. x_0, bitlength of parameters and Hamming weight of x_0

	x_0	$\mathrm{len}(x_0)$	$\mathrm{HW}(x_0)$	$\mathrm{len}(p^k)$	$\mathrm{len}(p^{k/d})$	$\mathrm{len}(p)$	$\mathrm{len}(r)$
BLS-24	$-1+2^{65}-2^{75}+2^{109}$	109	4	26122	4354	1089	872
KSS-32	$-1-2^2-2^{12}+2^{14}+2^{18}-2^{30}+2^{49}$	49	7	27536	6884	861	738
KSS-36	$2^5+2^{34}+2^{40}+2^{45}-2^{58}$	58	5	28699	4784	798	669
BLS-42	$-1+2^2-2^8+2^{43}$	43	4	28830	4805	687	516
BLS-48	$-1+2^7-2^{10}-2^{30}-2^{32}$	33	5	27851	4642	581	518

* HW() : Hamming weight

Table 7. Formulas for computing optimal ate pairing $a_k(P,Q)$

BLS-k with $k=24,42,48$	$(f_{x,Q}(P))^{(p^k-1)/r}$
KSS-32	$\left(f_{x,Q}(P)\cdot f_{2,Q}^{p^9}(P)\cdot\left(\overline{f_{3,Q}(P)}\right)^p\cdot\ell_{xQ,-3pQ}(P)\right)^{(p^{32}-1)/r}$
KSS-36	$\left(f_{x,Q}(P)\cdot f_{2,Q}^{p^7}(P)\cdot\left(\overline{f_{3,Q}(P)}\right)^p\cdot\ell_{xQ,-3pQ}(P)\right)^{(p^{36}-1)/r}$

* $\overline{t}$: conjugation of t in $\mathbb{F}_{p^k}$

$$m^{\Phi_k(p)/r}=(m^{\lambda_0})\cdot(m^{\lambda_1})^p\cdots(m^{\lambda_{s-1}})^{p^{s-1}}\cdot(m^{\lambda_s})^{p^s}, \tag{3}$$

where λ_i is the polynomial by x and $s=\varphi(k)-1$. For computing the Eq. (3), in the BLS-k, we can compute it with essentially just exponentiation by x since the BLS-k has a very convenient way to compute the each m^{λ_i} [13]. In the KSS-k, we apply the addition chain technique with Ghammam *et al.*'s λ_i-representation to compute the each m^{λ_i} efficiently since the coefficients of λ_i are dozens bits [23]. Additionally, to efficiently compute the exponentiation by x, we use the Karabina squaring technique [27] in the case of the curve with $d=6$, and Granger-Scott squaring technique [19] in the case of the curve with $d=4$ respectively. Hence, the computation of the hard part requires exponentiations by x $\deg(p(x))-1$ times, Frobenius operations s times, and squarings/multiplications on $\mathbb{F}_{p^k}$.

Scalar Mult. in $\mathbb{G}_1$ *and* $\mathbb{G}_2$. To efficiently compute the scalar multiplication in $\mathbb{G}_1$ and $\mathbb{G}_2$, we use the Gallant-Lambert-Vanstone (GLV) [20] and Galbraith-Lin-Scott (GLS) [21] which are the scalar decomposition methods By using the GLV/GLS, for given a scalar u and $P\in\mathbb{G}_1$ (or $\mathbb{G}_2$), the scalar u is decomposed into t scalars $u_1,u_2,\ldots,u_t$ with roughly the size $\mathrm{len}(u)/t$, then we convert the multi-scalar multiplication $uP=u_1P+u_2\psi(P)+\cdots+u_t\psi^{t-1}(P)$ by using an efficient endomorphism ψ [9], where $t=2$ in $\mathbb{G}_1$ and $t=\varphi(k)$ in $\mathbb{G}_2$ respectively. The number of the doubling in $\mathbb{G}_1$ and $\mathbb{G}_2$ can reduce to roughly $1/t$. Moreover, by using the width-w non adjacent form (w-NAF) [45], the number of the addition in $\mathbb{G}_1$ and $\mathbb{G}_2$ can be reduced in the computing the each scalar multiplication $u_i\psi^{i-1}(P)$. Note that we chose the optimal window size w for the scalars u_i. Let $\mathtt{I}_i$ and $\mathtt{M}_i$ be the cost of inversion and multiplication in $\mathbb{F}_{p^i}$ respectively. We use the Jacobian coordinates in $\mathbb{G}_1$ since $\mathtt{I}_1\approx 17.7\mathtt{M}_1$. We use the affine coordinates in $\mathbb{G}_2$ since $\mathtt{I}_{k/d}\approx 3.3\mathtt{M}_{k/d}$ for BLS-24, KSS-32, KSS-36, BLS-48, and the Jacobian coordinates in $\mathbb{G}_2$ since $\mathtt{I}_7\approx 17.6\mathtt{M}_7$ for BLS-42.

Table 8. Timing of computing pairing (ML, FE), scalar multiplication in $\mathbb{G}_1$, $\mathbb{G}_2$ and exponentiation in $\mathbb{G}_3$ (M clk: million clocks)

		BLS-24	KSS-32	KSS-36	BLS-42	BLS-48
Pairing	ML	53.80	32.04	37.61	36.36	**20.48**
	FE	**89.84**	197.26	147.49	100.95	96.36
	Total	143.64	229.30	185.10	137.31	**116.84**
Scalar Mult. in $\mathbb{G}_1$		12.00	8.31	6.13	3.94	**3.56**
Scalar Mult. in $\mathbb{G}_2$		38.87	53.32	35.01	49.43	**25.18**
Exp. in $\mathbb{G}_3$		71.00	62.88	77.30	**56.00**	63.46

* Scalar Mult. in $\mathbb{G}_2$ of BLS-42 only used the Jacobian coordinates because of $\mathtt{I}_7 \approx 17.6\mathtt{M}_7$.

Exp. in $\mathbb{G}_3$. To efficiently compute the exponentiation in $\mathbb{G}_3$, we can use the GLS method and w-NAF since $\mathbb{G}_3 \subseteq \mathbb{G}_{\Phi_k(p)} = \{\alpha \in \mathbb{F}_{p^k} \mid \alpha^{\Phi_k(p)} = 1\}$ and the inversion in $\mathbb{G}_3$ can be efficiently computed by the conjugation [4].

6.3 Timing and Comparison

In this subsection, we show the timing of the main operations of PBC and compare those among the five candidate pairing-friendly curves.

Environment. We implement in C language, and its compiler is gcc 6.2.0 with -O3 option. We also measure on an Intel Core i7-6700 @ 3.4 GHz, RAM: 32 GB and OS: Ubuntu 16.04 (64-bit).

Results. Table 8 shows the timing of computing the pairing (ML, FE), scalar multiplication in $\mathbb{G}_1$, $\mathbb{G}_2$, and exponentiation in $\mathbb{G}_3$ for each candidate pairing-friendly curve. We discuss the timing of these operations among the five candidate pairing-friendly curves.

Pairing. The computational cost of the ML is affected by the bitlength and Hamming weight of x_0. The effect on the timing of the ML by the Hamming weight of x_0 is small since the Hamming weight of x_0 of each five candidate pairing-friendly curves is sufficiently small and much the same. As the embedding degree k increases, the bitlength of x_0 decrease, so the timing of the ML in the BLS-48 can be computed most efficiently.

The computational cost of the FE is affected by the coefficient of λ_i of Eq. (3), $\varphi(k)$, degree of $p(x)$, the bitlength of x_0 and p^k. The timings of the FE in the KSS-k are slower than that of the BLS-k since the computational cost of addition chain is required in addition to the exponentiation by x to compute the each m^{λ_i} in Eq. (3) of KSS-k. In the BLS-k, the coefficient of each λ_i of Eq. (3) is a few bits. Hence, the cost to compute all m^{λ_i} in the BLS-k is affected by the

bitlength of p^k. The timing of the FE in the BLS-24 is faster than other BLS-k since the bitlength of x_0 of the BLS-24 is smaller.

Consequently, the timing of the pairing in the BLS-48 is faster than other pairing-friendly curves. The multi-pairing requires computing multiple MLs and one FE. Hence, the pairing-friendly curve with fast calculation of the ML has a significant effect on the efficiency of the computing the multi-pairing.

Scalar Mult. in $\mathbb{G}_1$. The input of scalar multiplication in $\mathbb{G}_1$ is a random element $P \in \mathbb{G}_1$ and a random scalar value of less than r. The computational cost is affected by the bitlength of r and the point addition/doubling in $\mathbb{G}_1$ affected by the bitlength of p. As k increases, the bitlength of p and r decrease. Hence, the timing of the scalar multiplication in $\mathbb{G}_1$ in the BLS-48 is faster than other pairing-friendly curves.

Scalar Mult. in $\mathbb{G}_2$. The input of scalar multiplication in $\mathbb{G}_2$ is a random element of $P \in \mathbb{G}_2$ and a random scalar value of less than r. Its computational cost is affected by the degree of twist d, and the bitlength r and $p^{k/d}$. The group $\mathbb{G}_2$ is a subgroup on $E'(\mathbb{F}_{p^{k/d}})$, and the bitlength of $p^{k/d}$ can be small when d is large. The bitlength of $p^{k/d}$ in the KSS-32 with $d = 4$ is about 2000-bit larger than that in the BLS-24, KSS-32, KSS-36, and BLS-48 with $d = 6$. Hence, the timing of the scalar multiplication in $\mathbb{G}_2$ in the KSS-32 is slower than other curves. Among the BLS-24, KSS-36 and BLS-48, as k increases, the bitlength of r decrease. Hence, the timing of the scalar multiplication in $\mathbb{G}_2$ in the BLS-48 is faster than other curves.

Exp. in $\mathbb{G}_3$. The input of exponentiation in $\mathbb{G}_3$ is a random element of $g \in \mathbb{G}_3$ and a random scalar value of less than r. The group $\mathbb{G}_3$ is a subgroup on finite field $\mathbb{F}_{p^k}$, and then the computational cost of the exponentiation in $\mathbb{G}_3$ is affected by the bitlength of r and the multiplication/squaring in $\mathbb{F}_{p^k}$. The BLS-42 and BLS-48 are theoretically better to compute the exponentiation in $\mathbb{G}_3$ efficiently since the bitlength of r is small rather than other curves. The number of the squaring in $\mathbb{F}_{p^k}$ in the BLS-48 is less than that in the BLS-42 because of the decomposition size. To compute the multi-exponentiation after the decomposition, the number of the multiplication in $\mathbb{F}_{p^k}$ in the BLS-42 is more reduced rather than in the BLS-48 because the bigger window size can be use in BLS-42 by w-NAF. As the result, the timing of the exponentiation in $\mathbb{G}_3$ in the BLS-42 is fastest in all candidate curves.

6.4 Impact on Timing by Revised Bitlength

In this subsection, we show the impact on the timing of the main operations of PBC by revised bitlength of p^k at 256-bit security level by comparing between our and previous implementations. Note that it is difficult to directly compare the timing of the main operations of PBC between our and previous implementations since the implemented algorithms and techniques are different.

In previous implementations, Scott [44] showed that the timing of the pairing is 88.8M clk, and Bos *et al.* [9] showed that the timing of the scalar multiplication in $\mathbb{G}_1$ and $\mathbb{G}_2$, and exponentiation in $\mathbb{G}_3$ are 5.2, 27.6, 47.1M clk respectively. These implemented in BLS-24 with about 15000-bit p^k and 500-bit r.

We compare the timing between our BLS-48 implementation and the previous BLS-24 implementations. Our BLS-48 implementation of the scalar multiplication in $\mathbb{G}_1$ is approximately 1.5 times faster than the previous BLS-24 implementations of that because the bitlength of r is the same between these implementations. Then, our BLS-48 implementation of other operations is approximately 1.0–1.3 times slower than the previous BLS-24 implementations of that due to the effect of the efficient NFS algorithms.

7 Conclusion

We give for the first time the revised bitlength which the DLP is computationally infeasible against the efficient NFS algorithms (e.g. SexTNFS, STNFS), and the timing of the main operations of PBC for the five candidate pairing-friendly curves (i.e. the BLS-k with $k = 24, 42, 48$, KSS-k with $k = 32, 36$) at 256-bit security level. On the security side, we show that it is necessary to increase bitlengths by more than 10000-bit from the previous estimation to achieve the 256-bit security level. On the implementation side, we show that the BLS-48 curve is the suitable curve at the 256-bit security level by comparing the timing of the main operations of PBC among the five candidate pairing-friendly curves with revised bitlengths. For more speeding up, we should implement $\mathbb{F}_p$-arithmetic in assembly, apply other efficient algorithms, etc.

A Norm Plots of BLS-24, KSS-32, KSS-36 and BLS-42

In this appendix, we show the norm plots of the GNFS and SNFS algorithms for the BLS-24, KSS-32, KSS-36, and BLS-42 in order to revise the bitlength of the these curves using the same method discussed in Sect. 5 (Figs. 3, 4, 5, 6, 7, 8, 9 and 10).

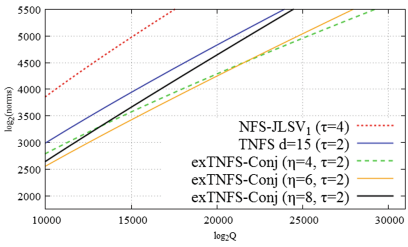

Fig. 3. Norms of GNFS algorithm in $\mathbb{F}_{p^{24}}$

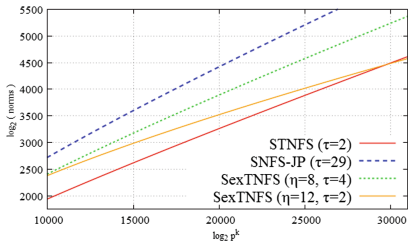

Fig. 4. Norms of SNFS algorithm in $\mathbb{F}_{p^{24}}$

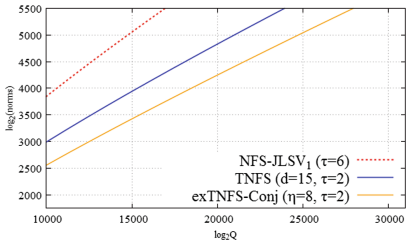

Fig. 5. Norms of GNFS algorithm in $\mathbb{F}_{p^{32}}$

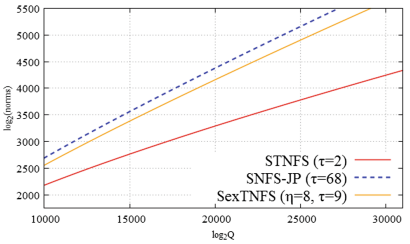

Fig. 6. Norms of SNFS algorithm in $\mathbb{F}_{p^{32}}$

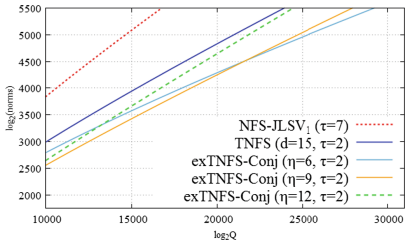

Fig. 7. Norms of GNFS algorithm in $\mathbb{F}_{p^{36}}$

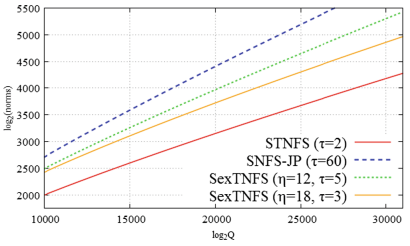

Fig. 8. Norms of the SNFS algorithm in $\mathbb{F}_{p^{36}}$

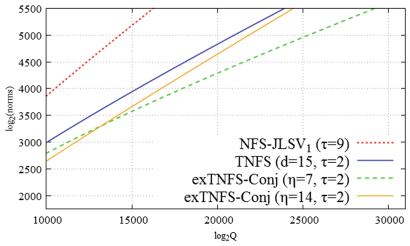

Fig. 9. Norms of GNFS algorithm in $\mathbb{F}_{p^{42}}$

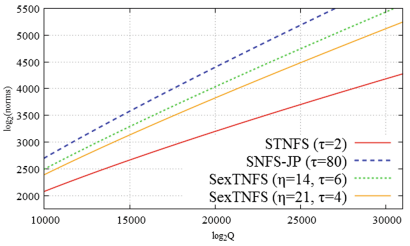

Fig. 10. Norms of SNFS algorithm in $\mathbb{F}_{p^{42}}$

References

1. Aoki, K., Franke, J., Kleinjung, T., Lenstra, A.K., Osvik, D.A.: A kilobit special number field sieve factorization. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 1–12. Springer, Heidelberg (2007). doi:10.1007/978-3-540-76900-2_1
2. Acar, T., Lauter, K., Naehrig, M., Shumow, D.: Affine pairings on ARM. In: Abdalla, M., Lange, T. (eds.) Pairing 2012. LNCS, vol. 7708, pp. 203–209. Springer, Heidelberg (2013). doi:10.1007/978-3-642-36334-4_13

3. Aranha, D.F., Fuentes-Castañeda, L., Knapp, E., Menezes, A., Rodríguez-Henríquez, F.: Implementing pairings at the 192-bit security level. In: Abdalla, M., Lange, T. (eds.) Pairing 2012. LNCS, vol. 7708, pp. 177–195. Springer, Heidelberg (2013). doi:10.1007/978-3-642-36334-4_11
4. Aranha, D.F., Karabina, K., Longa, P., Gebotys, C.H., López, J.: Faster explicit formulas for computing pairings over ordinary curves. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 48–68. Springer, Heidelberg (2011). doi:10.1007/978-3-642-20465-4_5
5. Barker, E.B., Barker, W.C., Burr, W.E., Polk, W.T., Smid, M.E.: Recommendation for key management - part 1: General (Revision 4). NIST SP 800-57 (2016)
6. Barbulescu, R., Gaudry, P., Guillevic, A., Morain, F.: Improving NFS for the discrete logarithm problem in non-prime finite fields. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9056, pp. 129–155. Springer, Heidelberg (2015). doi:10.1007/978-3-662-46800-5_6
7. Bouvier, C., Gaudry, P., Imbert, L., Jeljeli, H., Thom, E.: Discrete logarithms in GF(p) — 180 digits. Announcement available at the NMBRTHRY archives, item 004703 (2014)
8. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001). doi:10.1007/3-540-44647-8_13
9. Bos, J.W., Costello, C., Naehrig, M.: Exponentiating in pairing groups. In: Lange, T., Lauter, K., Lisoněk, P. (eds.) SAC 2013. LNCS, vol. 8282, pp. 438–455. Springer, Heidelberg (2014). doi:10.1007/978-3-662-43414-7_22
10. Barbulescu, R., Gaudry, P., Kleinjung, T.: The tower number field sieve. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015. LNCS, vol. 9453, pp. 31–55. Springer, Heidelberg (2015). doi:10.1007/978-3-662-48800-3_2
11. Barreto, P.S.L.M., Lynn, B., Scott, M.: Constructing elliptic curves with prescribed embedding degrees. In: Cimato, S., Persiano, G., Galdi, C. (eds.) SCN 2002. LNCS, vol. 2576, pp. 257–267. Springer, Heidelberg (2003). doi:10.1007/3-540-36413-7_19
12. 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.) LATINCRYPT 2015. LNCS, vol. 9230, pp. 245–265. Springer, Cham (2015). doi:10.1007/978-3-319-22174-8_14
13. Costello, C., Lauter, K., Naehrig, M.: Attractive subfamilies of BLS curves for implementing high-security pairings. In: Bernstein, D.J., Chatterjee, S. (eds.) INDOCRYPT 2011. LNCS, vol. 7107, pp. 320–342. Springer, Heidelberg (2011). doi:10.1007/978-3-642-25578-6_23
14. Danilov, S.A., Popovyan, I.A.: Factorization of RSA-180, Cryptology ePrint Archive, Report 2010/270 (2010)
15. European Union Agency of Network and Information Security (ENISA): Algorithms, key sizes and parameters report, 2013 recommandations, version 1.0, October 2013
16. Freeman, D., Scott, M., Teske, E.: A taxonomy of pairing-friendly elliptic curves. J. Cryptol. **23**, 224–280 (2010)
17. Fried, J., Gaudry, P., Heninger, N., Thomé, E.: A kilobit hidden SNFS discrete logarithm computation. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017. LNCS, vol. 10210, pp. 202–231. Springer, Cham (2017). doi:10.1007/978-3-319-56620-7_8
18. The GNU Multiple Precision Arithmetic Library. https://gmplib.org/

19. Granger, R., Scott, M.: Faster squaring in the cyclotomic subgroup of sixth degree extensions. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 209–223. Springer, Heidelberg (2010). doi:10.1007/978-3-642-13013-7_13
20. Gallant, R.P., Lambert, R.J., Vanstone, S.A.: Faster point multiplication on elliptic curves with efficient endomorphisms. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 190–200. Springer, Heidelberg (2001). doi:10.1007/3-540-44647-8_11
21. Galbraith, S.D., Lin, X., Scott, M.: Endomorphisms for faster elliptic curve cryptography on a large class of curves. J. Crypto **24**, 446–469 (2011)
22. Guillevic, A., Morain, F., Thomé, E.: Solving discrete logarithms on a 170-bit MNT curve by pairing reduction, arXiv preprint arXiv:1605.07746 (2016)
23. Ghammam, L., Fouotsa, E.: Adequate elliptic curves for computing the product of n pairings. In: Duquesne, S., Petkova-Nikova, S. (eds.) WAIFI 2016. LNCS, vol. 10064, pp. 36–53. Springer, Cham (2016). doi:10.1007/978-3-319-55227-9_3
24. Hess, F., Smart, N., Vercauteren, F.: The eta pairing revisited. IEEE Trans. Inf. Theory **52**(10), 4595–4602 (2006)
25. Joux, A., Lercier, R., Smart, N., Vercauteren, F.: The number field sieve in the medium prime case. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 326–344. Springer, Heidelberg (2006). doi:10.1007/11818175_19
26. Joux, A., Pierrot, C.: The special number field sieve in $\mathbb{F}_{p^n}$. In: Cao, Z., Zhang, F. (eds.) Pairing 2013. LNCS, vol. 8365, pp. 45–61. Springer, Cham (2014). doi:10.1007/978-3-319-04873-4_3
27. Karabina, K.: Squaring in cyclotomic subgroups. Math. Comput. **82**, 555–579 (2013)
28. Kim, T., Barbulescu, R.: Extended tower number field sieve: a new complexity for the medium prime case. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9814, pp. 543–571. Springer, Heidelberg (2016). doi:10.1007/978-3-662-53018-4_20
29. Kachisa, E.J., Schaefer, E.F., Scott, M.: Constructing brezing-weng pairing-friendly elliptic curves using elements in the cyclotomic field. In: Galbraith, S.D., Paterson, K.G. (eds.) Pairing 2008. LNCS, vol. 5209, pp. 126–135. Springer, Heidelberg (2008). doi:10.1007/978-3-540-85538-5_9
30. Kim, T., Jeong, J.: Extended tower number field sieve with application to finite fields of arbitrary composite extension degree. In: Fehr, S. (ed.) PKC 2017. LNCS, vol. 10174, pp. 388–408. Springer, Heidelberg (2017). doi:10.1007/978-3-662-54365-8_16
31. Kleinjung, T.: Discrete Logarithms in GF(p) – 768 bits. Announcement available at the NMBRTHRY archives, item **004917** (2016)
32. Lenstra, A.K.: Unbelievable security *matching AES security using public key systems*. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 67–86. Springer, Heidelberg (2001). doi:10.1007/3-540-45682-1_5
33. Lenstra, A.K., Lenstra, H.W. (eds.): The Development of the Number Field Sieve. LNM, vol. 1554. Springer, Heidelberg (1993). doi:10.1007/BFb0091534
34. Miller, V.S.: The weil pairing, and its efficient calculation. J. Cryptol. **17**, 235–261 (2004)
35. Mori, Y., Akagi, S., Nogami, Y., Shirase, M.: Pseudo 8–sparse multiplication for efficient ate–based pairing on barreto–naehrig curve. In: Cao, Z., Zhang, F. (eds.) Pairing 2013. LNCS, vol. 8365, pp. 186–198. Springer, Cham (2014). doi:10.1007/978-3-319-04873-4_11
36. Mitsunari, S.: A fast implementation of the optimal ate pairing over BN curve on intel haswell processor, Cryptology ePrint Archive, Report 2013/362 (2013)

37. Menezes, A., Sarker, P., Singh, S.: Challenges with assessing the impact of NFS advances on the security of pairing-based cryptography, Cryptology ePrint Archive, Report 2016/1102 (2016)
38. Okamoto, T., Takashima, K.: Fully secure functional encryption with general relations from the decisional linear assumption. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 191–208. Springer, Heidelberg (2010). doi:10.1007/978-3-642-14623-7_11
39. Pollard, J.: Monte Carlo methods for index computation (mod p). Math. Comput. **32**(143), 918–924 (1978)
40. Sakai, R., Ohgishi, K., Kasahara, M.: Cryptosystems based on pairing. In: SCIS 2000, C-20, pp. 26–28 (2000)
41. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005). doi:10.1007/11426639_27
42. Devegili, A.J., Scott, M., Dahab, R.: Implementing cryptographic pairings over barreto-naehrig curves. In: Takagi, T., Okamoto, T., Okamoto, E., Okamoto, T. (eds.) Pairing 2007. LNCS, vol. 4575, pp. 197–207. Springer, Heidelberg (2007). doi:10.1007/978-3-540-73489-5_10
43. Scott, M., Benger, N., Charlemagne, M., Dominguez Perez, L.J., Kachisa, E.J.: On the final exponentiation for calculating pairings on ordinary elliptic curves. In: Shacham, H., Waters, B. (eds.) Pairing 2009. LNCS, vol. 5671, pp. 78–88. Springer, Heidelberg (2009). doi:10.1007/978-3-642-03298-1_6
44. Scott, M.: On the efficient implementation of pairing-based protocols. In: Chen, L. (ed.) IMACC 2011. LNCS, vol. 7089, pp. 296–308. Springer, Heidelberg (2011). doi:10.1007/978-3-642-25516-8_18
45. Schirokauer, O.: Using number fields to compute logarithms in finite fields. Math. Comp. **69**, 1267–1283 (2000)
46. Schirokauer, O.: Virtual logarithms. J. Algorithms **57**, 140–147 (2005)
47. Vercauteren, F.: Optimal pairings. IEEE Trans. Inf. Theory **56**(1), 455–461 (2010)