Solving a 676-bit Discrete Logarithm Problem in GF(3^{6n})

Abstract. Pairings on elliptic curves in finite fields are crucial material for constructions of various cryptographic schemes. The η_T pairing on supersingular curves over GF(3^n) is in particular popular since it is efficiently implementable. Taking into account of the MOV attack, the discrete logarithm problems (DLP) in GF(3^{6n}) becomes concerned to the security of cryptosystems using η_T pairings in this case. In 2006, Joux and Lercier proposed a new variant of the function field sieve in the medium prime case, named JL06-FFS. We have, however, not found any practical implementations on JL06-FFS over GF(3^{6n}) up to now. Therefore, we have firstly fulfilled such an implementation and successfully set a new record for solving the DLP in GF(3^{6n}), the DLP in GF($3^{6\cdot 71}$) of 676-bit size. We conclude that $n = 97$ case, where there are many implementations of the η_T pairing, is not recommended in practical use. In addition, we also conduct comparisons between JL06-FFS and an earlier version, named JL02-FFS, by practical experiments. Our results confirm that the former is faster several times than the latter under certain conditions.

Key words: function field sieve, discrete logarithm problems, pairing-based cryptosystems

1 Introduction

Based on pairings, many novel cryptographic protocols are constructed successively, such as identity based encryptions [9], forward-secure cryptosystems, proxy cryptosystems, keyword searchable PKEs [8], and so on. Then, two requirements arise therewith: efficient pairing computation and security parameter selection.

The η_T pairing [5] on supersingular curves over GF(3^n) has been implemented very efficiently both in softwares and in hardwares [6, 13, 14][1]. Along with the speed-up of computation on the η_T pairing, one may ask whether the cryptosystems based on the η_T pairing are still secure. It is well known that a discrete logarithm problem (DLP) on supersingular curves over GF(q) can be converted to a DLP in GF(q^m) (where q is a prime power and m is not larger than 6) [24]. Therefore, the DLP in GF(3^{6n}) is one of the most important problems to analyze the cryptosystems constructed with the η_T pairing on supersingular curves over GF(3^n).

The function field sieve (FFS) is one of the most efficient algorithms for solving the DLP in finite fields of small characteristic. The complexity of the FFS for solving the DLP in GF(3^{6n}) is $L_{3^{6n}}[1/3, c]$ with constant c, where

$$L_{3^{6n}}[1/3, c] = \exp((c + o(1))(\log 3^{6n})^{1/3}(\log\log 3^{6n})^{2/3}).$$

Here $o(1)$ stands for a function that converges to zero as n approaches infinity.

The first FFS was proposed by Adleman [1] in 1994. Five years later, Adleman and Huang proposed an improved FFS (AH-FFS) with $c = (32/9)^{1/3}$ [2]. In 2002, Joux and Lercier proposed a practical improvement of the FFS (JL02-FFS) [16]. Since a definition polynomial of the function field in JL02-FFS is able to choose more flexibly, JL02-FFS is more practical than AH-FFS though its asymptotic complexity is the same with AH-FFS. Furthermore, by using JL02-FFS, Joux and Lercier succeeded

[1] Here, n is a prime number such as $n = 97$, 163, 193, etc [25].

in solving the DLP in $\mathrm{GF}(2^{613})$. This refreshed the top-record of solving the DLP in finite fields of characteristic two in the sense of bit-size [15]. In 2006, Joux and Lercier proposed another new variant of the FFS (JL06-FFS) [18]. JL06-FFS has the same asymptotic complexity with JL02-FFS for solving the DLP in $\mathrm{GF}(3^{6n})$ where n is a prime number[2]. This work implied that JL06-FFS might be efficient to solve the DLP in extension fields of $\mathrm{GF}(3^6)$ of degree n. However, to our knowledge, there are no practical experiments.

Our contributions. We have firstly conducted the experiments on JL06-FFS. Moreover, by our implementation of JL06-FFS using Galois action to reduce required relations, we succeeded in solving the DLP in $\mathrm{GF}(3^{6\cdot 71})$ of 676-bit size with about 19 days computation, which is the new record of solving the DLP in $\mathrm{GF}(3^{6n})$. Our work contributes to selecting of security parameters. Additionally, we compare JL06-FFS [18] with JL02-FFS [16] according to the experiment results, and confirm that JL06-FFS is several times faster than JL02-FFS with $n = 19, 61$.

The rest of the paper is organized as follows. In Section 2, we briefly review the function field sieve algorithm. In Section 3, we compare JL02-FFS with JL06-FFS according to the polynomial selection method and practical experiment results. In Section 4, we describe our implementation on how to solve the DLP in $\mathrm{GF}(3^{6\cdot 71})$ in detail, which is based on JL06-FFS. Concluding remarks are made in Section 5.

2 Outline of the Function Field Sieve

In this section, we describe an overview of the FFS [1], which consists of four steps: polynomial selection, collection of relations, linear algebra, and individual logarithm. We particularly deal with the FFS for solving the DLP in extension fields of $\mathrm{GF}(3^6)$ of degree n and describe the four steps below. For more detail, refer to [1, 12, 16, 18].

Through the description, let γ be a generator of a multiplicative group of $\mathrm{GF}(3^{6n})$ and $\alpha \in \langle\gamma\rangle$, then we try to find the smallest positive integer $\log_\gamma \alpha$ such that $\gamma^{\log_\gamma \alpha} = \alpha$ called the discrete logarithm.

1. Polynomial selection: Select $f \in \mathrm{GF}(3^6)[x]$, such that f is a monic irreducible polynomial of degree n, then $\mathrm{GF}(3^{6n}) \cong \mathrm{GF}(3^6)[x]/(f)$. Next, find a polynomial $H(x, y) \in \mathrm{GF}(3^6)[x, y]$ satisfying the eight conditions proposed by Adleman [1]. Then there is a surjective homomorphism

$$\Phi : \begin{cases} \mathrm{GF}(3^6)[x, y]/(H) \to \mathrm{GF}(3^{6n}) \cong \mathrm{GF}(3^6)[x]/(f) \\ \quad y \qquad\qquad \mapsto \qquad\qquad m, \end{cases}$$

   where m is in $\mathrm{GF}(3^6)[x]$ such that $H(x, m) \equiv 0 \pmod f$. Here we select the smoothness bound B and define a rational factorbase B_R and an algebraic factorbase B_A as follows:

$$B_R = \{\mathfrak{p} \in \mathrm{GF}(3^6)[x] \mid \deg(\mathfrak{p}) \le B, \mathfrak{p} \text{ is irreducible}\},$$
$$B_A = \{\langle \mathfrak{p}, y - t\rangle \in \mathrm{Div}(\mathrm{GF}(3^6)[x, y]/(H)) \mid \mathfrak{p} \in B_R, t \equiv m \pmod{\mathfrak{p}}\},$$

   where $\mathrm{Div}(\mathrm{GF}(3^6)[x, y]/(H))$ is the divisor group of $\mathrm{GF}(3^6)[x, y]/(H)$ and $\langle \mathfrak{p}, y - t\rangle$ is a divisor generated by $\mathfrak{p}$ and $y - t$.

[2] When n is a composite number, this variant may have complexity $L_{3^{6n}}[1/3, 3^{1/3}]$ for solving the DLP in $\mathrm{GF}(3^{6n})$ (When JL06-FFS has complexity $L_{p^n}[1/3, 3^{1/3}]$, then we call it JL06-FFS-2). We do not deal with this case in this paper.

2. Collection of relations: For $r,\ s \in \mathrm{GF}(3^6)[x]$ of degree not larger than B, find at least $(\#B_R + \#B_A)$ relatively prime pairs $(r,\ s)$, such that

$$\begin{aligned} rm + s &= \prod_{\mathfrak{p}_i \in B_R} \mathfrak{p}_i^{a_i} \\ \langle ry + s \rangle &= \sum_{\langle \mathfrak{p}_j, t_j \rangle \in B_A} b_j \langle \mathfrak{p}_j, y - t_j \rangle. \end{aligned} \tag{1}$$

Such a pair $(r,\ s)$ is called the double smooth pair. Next we define the norm of $\langle ry + s \rangle$ as

$$N_A(r,\ s) = (-r)^d H(x,\ -s/r). \tag{2}$$

When $N_A(r,\ s)$ is factorized into irreducible polynomials of degree not larger than B, called $N_A(r,\ s)$ is B-smooth, then

$$N_A(r,\ s) = \prod_{\langle \mathfrak{p}_j, t_j \rangle \in B_A} \mathfrak{p}_j^{b_j}, \tag{3}$$

where t_j is uniquely determined by r, s and $\mathfrak{p}_j$. Then the b_j in equation (3) is exactly same as one in equation (1). Similarly we define the norm of $rm + s$,

$$N_R(r,\ s) = rm + s. \tag{4}$$

When both $N_R(r,\ s)$ and $N_A(r,\ s)$ are B-smooth, a pair $(r,\ s)$ is the double smooth pair. Eventually, we get the following relation:

$$\sum_{\mathfrak{p}_i \in B_R} a_i \log_\gamma \mathfrak{p}_i \equiv \sum_{\langle \mathfrak{p}_j, t_j \rangle \in B_A} b_j \log_\gamma \kappa_j \pmod{(3^{6n} - 1)/(3^6 - 1)}, \tag{5}$$

where

$$\kappa_j = \Phi(\lambda_j)^{1/h},\ \langle \lambda_j \rangle = h \langle \mathfrak{p}_j\ y - t_j \rangle, \tag{6}$$

for the class number h of the quotient field $\mathrm{GF}(3^6)(x)[y]/(H)$.

3. Linear algebra: For the number R of relations, construct $R \times (\#B_R + \#B_A)$ matrix M from relations in equation (5) and $(\#B_R + \#B_A)$ column vector $\boldsymbol{v}$ as follows:

$$M = \begin{pmatrix} a_1^{(1)} & \dots & a_{\#B_R}^{(1)} & -b_1^{(1)} & \dots & -b_{\#B_A}^{(1)} \\ \vdots & & \vdots & \vdots & & \vdots \\ a_1^{(R)} & \dots & a_{\#B_R}^{(R)} & -b_1^{(R)} & \dots & -b_{\#B_A}^{(R)} \end{pmatrix},\ \boldsymbol{v} = \begin{pmatrix} \log_\gamma \mathfrak{p}_1 \\ \vdots \\ \log_\gamma \mathfrak{p}_{\#B_R} \\ \log_\gamma \kappa_1 \\ \vdots \\ \log_\gamma \kappa_{\#B_A} \end{pmatrix}.$$

Then we solve the linear equation

$$M\boldsymbol{v} \equiv 0 \pmod{(3^{6n} - 1)/(3^6 - 1)}. \tag{7}$$

4. Individual logarithm: Find positive integers e_i, f_j such that

$$\log_\gamma \alpha \equiv \sum_{\mathfrak{p}_i \in B_R} e_i \log_\gamma \mathfrak{p}_i + \sum_{\langle \mathfrak{p}_j, t_j \rangle \in B_A} f_j \log_\gamma \kappa_j \pmod{(3^{6n} - 1)/(3^6 - 1)},$$

then compute the discrete logarithm $\log_\gamma \alpha$. This is done by special-$\mathfrak{q}$ descent method [16, 18].

3 Comparison of the Polynomial Selection on JL02-FFS and JL06-FFS

There are two most efficient variants of the FFS for solving the DLP in $\mathrm{GF}(3^{6n})$, namely JL02-FFS and JL06-FFS. Although their complexities approach asymptotically to the same, there is the considerable difference between them in the fixed extension degree in practical used. The time complexities of JL02-FFS and JL06-FFS depend on the size of each sieving area, which is the number of pairs (r, s), and each size is explained in the following subsections.

3.1 Polynomial Selection of JL02-FFS and Its Sieving Area

At first we describe an outline of the polynomial selection of JL02-FFS, after that we estimate the size of sieving area. In order to distinguish previous section, we set subindex "02" after symbols.

Let $H_{02}(x, y)$ be formed as C_{ab} curves in $\mathrm{GF}(3)[x, y]$ of degree d_{02} in y. Then choose the polynomial $u_1, u_2 \in \mathrm{GF}(3)[x]$ of degree at most $\lfloor 6n/d_{02} \rfloor$ randomly. We try to find an irreducible polynomial $f_{02} = u_2^d H_{02}(x, -u_1/u_2) \in \mathrm{GF}(3)[x]$ of degree $6n$ such that $\gcd(u_2, f_{02}) = 1$, then u_2 is invertible modulo f_{02}. Then, there is a surjective homomorphism

$$\Phi_{02} : \begin{cases} \mathrm{GF}(3)[x,y]/(H_{02}) \to \mathrm{GF}(3^{6n}) \cong \mathrm{GF}(3)[x]/(f_{02}) \\ \quad y \qquad\qquad \mapsto \qquad\quad -u_1/u_2, \end{cases}$$

where $H_{02}(x, y)$ holds $H_{02}(x, -u_1/u_2) \equiv 0 \pmod{f_{02}}$. In this polynomial selection, we need to modify the right side of equation (4) to $su_2 - ru_1$, in other words, we define $N'_R(r, s) = su_2 - ru_1$. Note that r and s are chosen in $\mathrm{GF}(3)[x]$ of degree not larger than B_{02} in JL02-FFS, the size of sieving area in the collection of relation step is

$$3^{B_{02}+1} \cdot 3^{B_{02}+1}. \tag{8}$$

From heuristic analysis by [16], JL02-FFS becomes optimized when we choose the smoothness bound B_{02} as

$$B_{02} = \lceil (4/9)^{1/3} (6n)^{1/3} \log_3 (6n)^{2/3} \rfloor. \tag{9}$$

and the extension degree d_{02} of $H_{02}(x, y)$ as $d_{02} = \lceil \sqrt{6n/(B_{02}+1)} \rfloor$. For example, the extension degree n is chosen in practical used such as $n = 97, 163, 193$, then $(n, B_{02}) = (97, 21), (163, 26), (193, 28)$.

3.2 Polynomial Selection of JL06-FFS and Its Sieving Area

Next we describe an outline of the polynomial selection of JL06-FFS, and we estimate the size of sieving area of JL06-FFS.

At first, fixed extension degree n, we choose the smallest smoothness bound B_{06} in JL06-FFS satisfied the following condition,

$$(B_{06} + 1) \log(3^6) \geq \sqrt{n/B_{06}} \log(n/B_{06}) \tag{10}$$

For example, n is chosen in practical used such as $n = 97, 163, 193$, then $(n, B_{06}) = (97, 3), (163, 4), (193, 4)$. Next, we choose positive integers d and d' such that $d \approx \sqrt{n/B_{06}}$ and $d' \approx \sqrt{nB_{06}}$, where $dd' \geq n$. After that, we generate $g(y) \in \mathrm{GF}(3^6)[y]$ of degree d randomly and set $H(x, y) = g(y) + x$. Finally we try to find an irreducible polynomial f in $\mathrm{GF}(3^6)[x]$ of degree n which divides $H(x, m)$ where $m \in \mathrm{GF}(3^6)[x]$ of degree d' is chosen randomly. In this polynomial selection, each of leading coefficients

Table 1. The parameters and the sieving area

	n	Polynomial selection in JL02-FFS			Polynomial selection in JL06-FFS		
		$6n$	B_{02}	Size of sieving area	n	B_{06}	Size of sieving area
Experimental class	19	114	10	3.1×10^{10}	19	1	3.9×10^{8}
	31	186	12	2.5×10^{12}	31	2	2.1×10^{14}
	47	282	15	1.9×10^{15}	47	2	2.1×10^{14}
	61	366	17	1.5×10^{17}	61	2	2.1×10^{14}
Practically used class	97	582	21	9.8×10^{20}	97	3	1.1×10^{20}
	163	978	26	5.8×10^{25}	163	4	5.8×10^{25}
	193	1158	28	4.7×10^{27}	193	4	5.8×10^{25}
Futural class	239	1434	30	3.8×10^{29}	239	4	5.8×10^{25}
	313	1878	34	2.5×10^{33}	313	5	3.1×10^{31}
	353	2118	36	2.0×10^{35}	353	5	3.1×10^{31}
	509	3054	42	1.1×10^{41}	509	6	1.6×10^{37}

of $N_R(r, s)$ and $N_A(r, s)$ depends on r, so we avoid to obtain duplicate relations by fixing the leading coefficient of r as a monic polynomial. Therefore the size of sieving area in the collection of relations step is at most

$$(3^6)^{B_{06}+1} \cdot (3^6)^{B_{06}}. \tag{11}$$

3.3 Comparison of the Sieving Area

We compare JL06-FFS with JL02-FFS in the size of sieving area in three classes of extension degree n: *experimental* as {19, 31, 47, 61}, *practically used* as {97, 163, 193}, and *futural* as {239, 313, 353, 509}. Table 1 shows the smoothness bound and the size of sieving area in each variant. For each n, we obtain the smoothness bound B_{02} in equation (9) and B_{06} in equation (10), and estimate the size of sieving area by equation (8) in JL02-FFS and by equation (11) in JL06-FFS.

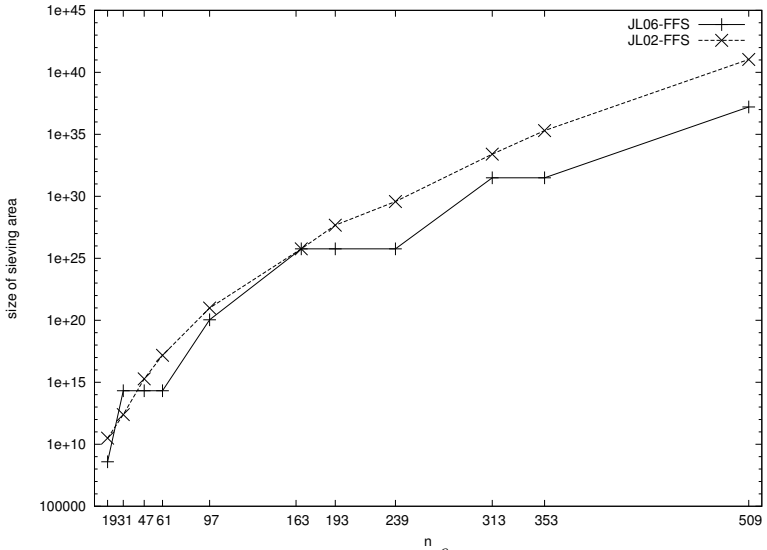

Fig. 1. The size of sieving area over GF(3^{6n}) in JL02-FFS and JL06-FFS

Figure 1 shows the size of required sieving area over GF(3^{6n}). The sieving area in JL06-FFS is much smaller than one in JL02-FFS when $n \neq 31, 167$. Moreover, the

Table 2. The parameters in our experiments

n	Bit size of $\mathrm{GF}(3^{6n})$	Experiments with JL02-FFS			Experiments with JL06-FFS		
		$6n$	B_{02}	$H_{02}(x, y)$	n	B_{06}	$H(x, y)$
19	181	114	10	$y^4 + x$	19	1	$y^5 + x$
31	295	186	12	$y^4 + x$	31	2	$y^4 + x$
47	447	282	15	$y^4 + x$	47	2	$y^5 + x$
61	581	366	17	$y^5 + x$	61	2	$y^6 + x$

differences between the sieving areas in JL06-FFS and in JL02-FFS increase along with the growing in n. The computational cost in the collection of relations step is closely related to the size of the sieving area, so the collection of relations step in JL06-FFS might be several times faster than in JL02-FFS.

We have experimented the collection of relations step in JL02-FFS and JL06-FFS to confirm difference between their computational costs in the collection of relations step. Parameters in JL02-FFS and JL06-FFS are given in Table 2. In our experiments, we use C_{ab} curves of the form $y^d + R(x)$ for the polynomial $H(x, y)$ as in [12]. Note that we have only experimented with experimental class as $n \in \{19, 31, 47, 61\}$, not with practically used and futural class.

In our experiments, we use 96 cores each of which has the same performance about 2.83GHz Xeon. We implement the lattice sieve [26] in JL02-FFS and the polynomial sieve [11] in JL06-FFS, respectively. The detail of our implementation in JL06-FFS is described in Section 4.

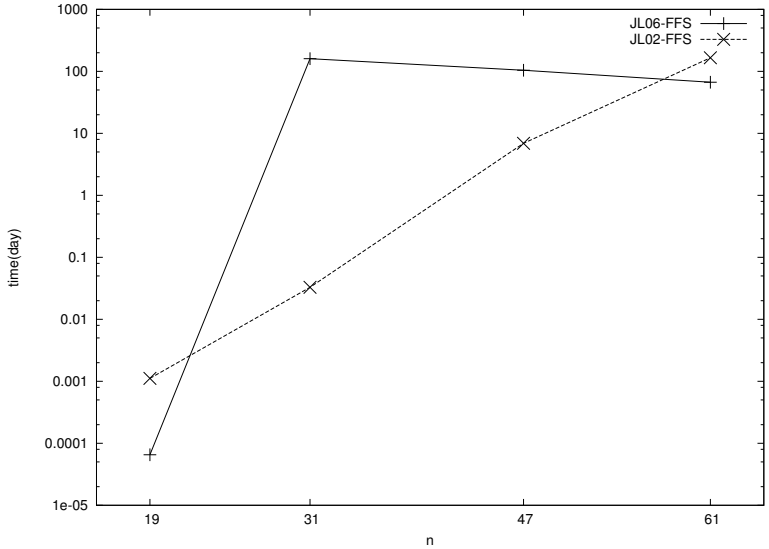

Fig. 2. Estimated time taken to compute whole sieving area in the collection of relations step over $\mathrm{GF}(3^{6n})$ in JL02-FFS and JL06-FFS

Figure 2 shows the time complexity of JL02-FFS and JL06-FFS to compute whole sieving area in the collection of relations step in $\mathrm{GF}(3^{6n})$ with $n = 19, 31, 47, 61$, respectively. Note that we estimate the time when the computation takes over an hour.

When $n = 19, 61$, our implementation on JL06-FFS is faster than one of JL02-FFS, and we confirm that JL06-FFS is more efficient than JL02-FFS for solving the DLP in $\mathrm{GF}(3^{6n})$. Especially, when $n = 61$, our implementation of JL06-FFS takes

about 66 days for the collection of relations step, but our implementation of JL02-FFS takes about 165 days. Therefore, the former is 2.5 times faster than the later. Accordingly, we expect that JL06-FFS is efficient to solve the DLP in GF(3^{6n}) for bigger n.

4 Solving the DLP in GF($3^{6\cdot 71}$)

In this section, we report that the DLP in GF($3^{6\cdot 71}$) of 676-bit size is solved by improving JL06-FFS. In our implementation, we deal with four practical improvements, namely polynomial sieve, free relation, Galois action, and parallel Lanczos method.

Particularly, by using the polynomial $H(x, y) = y^6 + x$, we only need to find about 1/8 of the originally required relations in the collection of relations step. Furthermore, via the Galois action, the size of the matrix given by the relations is also decreased to 1/6 of the original one. To the best of our knowledge, the size of 676 bits is currently the top-record of solving the DLP in the finite fields.

4.1 Collection of Relations

In the collection of relations step, we collect a lot of double smooth pairs (r, s). The simple idea for collecting them is to factor $N_R(r, s)$ in equation (4) and $N_A(r, s)$ in equation (2) for all pairs (r, s). This is not practical since we have to factor them about $(3^6)^B \times (3^6)^{B+1}$ times. In order to reduce the number of factoring, we use the sieving methods. The idea of the sieving is factoring $N_R(r, s)$ and $N_A(r, s)$ of only the pair (r, s) which may be the double smooth pair with strong possibility, such a pair called a candidate.

The polynomial sieve [11] and the lattice sieve [26] are well-known sieving algorithms. Although the lattice sieve is implemented in some experiments of the FFS such as [12, 15, 16], we implement the polynomial sieve since r is fixed as a monic polynomial by the polynomial sieve in JL06-FFS, whereas neither r nor s is able to be fixed by the lattice sieve.

Polynomial Sieve Here, we describe the polynomial sieve in $N_R(r, s)$. Notice that we can also sieve in $N_A(r, s)$ with the same procedure. Moreover, we discuss the case where s is fixed, and omit the details when r is fixed. From equation (4), by fixed s, we can lead r such that $N_R(r, s)$ is divisible by $\mathfrak{p} \in B_R$ or its power where the degree of $\mathfrak{p}$ is not larger than B. Additionally, $N_R(r, s + k\mathfrak{p})$ with $k \in \mathrm{GF}(3^6)[x]$ is also divisible by $\mathfrak{p}$. Hence, we can obtain all r of degree less than or equal to B such that $N_R(r, s)$ is divisible by $\mathfrak{p}$. After computing such all r for each $\mathfrak{p}$, we can obtain the pair (r, s) such that $N_R(r, s)$ is divisible by some $\mathfrak{p}$. If the summation of degree of all $\mathfrak{p}$ which divide $N_R(r, s)$ reaches to $\deg(N_R(r, s))$, then $N_R(r, s)$ may be B-smooth with strong possibility and the pair (r, s) is a candidate.

In this procedure, the most time-consuming work is to compute $r + k\mathfrak{p}$ for all $k \in \mathrm{GF}(3^6)[x]$ whose degree is not larger than B. In characteristic two, Gordon and McCurley proposed the method using binary gray codes [11] to compute them. Using ternary gray codes, we can also compute them efficiently in characteristic three.

In the polynomial sieve, we sieve with all powers of $\mathfrak{p}$ whose degree is not larger than B. Since B is very small such as 1 or 2 in JL06-FFS, power of $\mathfrak{p}$ is only $\mathfrak{p}^2$ when $\deg(\mathfrak{p}) = 1$. Such polynomials are exceptional since there are 3^6 monic irreducible polynomials of degree 1 in $\mathrm{GF}(3^6)[x]$. In this way, we can obtain only candidates that always generate relation in equation (5) (except that r and s are not relatively prime). Thus, we only check the greatest common divisor of r and s, but not the smoothness of $N_R(r, s)$ and $N_A(r, s)$ using the B-smooth test described in [11].

Free Relation By considering how a divisor $\langle \mathfrak{p} \rangle$ in B_R is factorized into divisors in $\mathrm{GF}(3^6)[x, y]/(H)$, namely, obtaining the following congruent expression that

$$H(x, y) \equiv \prod_{i=1}^{d} (y - t_i) \pmod{\mathfrak{p}}$$

where d is the degree of $H(x, y)$ on y, we can get a relation virtually for free, without the sieving procedure. We call such a relation a free relation.

The number of free relations depends on the degree d of $H(x, y)$ on y and the characteristic of the field treated in FFS. In fact, there are about $\#B_A/d$ free relations in many cases and, furthermore, they increase more when the characteristic is small. For example, in the case of $\mathrm{GF}(3^{6n})$ and $H(x, y) = y^6 + x$, there are about $\#B_A/2$ free relations since $y^6 + x$ is generally factored as $(y - t_1)^3(y - t_2)^3$ modulo $\mathfrak{p}$.

4.2 Linear Algebra

In the linear algebra step, we solve the linear equation depending on the relations. In detail, we construct the matrix from the relations, and reduce it to a much smaller one by Galois action. After that, we solve the reduced linear equation modulo $(3^{6n} - 1)/(3^6 - 1)$, by applying the parallel Lanczos method described as [3]. In this section, we describe Galois action and our ideas about parallel computation of the matrix operation.

Galois Action Here, we consider to reduce unknowns of linear equations, using Galois action which is presented in [18].

Let M' be the matrix given by the relations, whose row $M'_{(i)}$ means the i-th relation and j-th column $M'^{(j)}$ corresponds to the factorbase $\mathfrak{p}_j$. In order to use Galois action, we choose the polynomial $f \in \mathrm{GF}(3^6)[x]$ satisfying that all coefficients of f are in $\mathrm{GF}(3)$ and $\deg f = n$, and so we construct $\mathrm{GF}(3^{6n})$ as $\mathrm{GF}(3^6)[x]/(f)$. Let ϕ be the Frobenius power such that $\phi(\xi) = \xi^{3^n}$. As ϕ fixes the element x in $\mathrm{GF}(3)[x]/(f)$, we also have that $\phi(x) = x$ in $\mathrm{GF}(3^6)[x]/(f)$ by the assumption of f. However, for an element $c \in \mathrm{GF}(3^6) \backslash \mathrm{GF}(3)$, ϕ does not fix c in $\mathrm{GF}(3^6)[x]/(f)$ by the above assumption that n is coprime to 6. The monic irreducible polynomial $\mathfrak{p}_j \in B_R$ of degree not larger than B, and we assume that $B = 1$ for convenience. In fact, $\mathfrak{p}_j$ as $\mathfrak{p}_j = x + c_j$ where $c_j \in \mathrm{GF}(3^6)$ since $B = 1$, and so we have

$$\phi(\mathfrak{p}_j) = \phi(x + c_j) = x + \phi(c_j)$$

in $\mathrm{GF}(3^6)[x]/(f)$. If c_j is not in $\mathrm{GF}(3)$, it is clear that $c_j \neq \phi(c_j)$ in $\mathrm{GF}(3^6)[x]/(f(x))$. This fact implies that there are ordinarily many unknowns of linear equations, which can be rewritten by the other one via Galois action. Clearly, for such $\mathfrak{p}_j$, there exists $\mathfrak{p}_{j'}$ satisfying that

$$\log_\gamma \mathfrak{p}_{j'} = \log_\gamma \phi(\mathfrak{p}_j) = 3^n \log_\gamma \mathfrak{p}_j$$

where $\mathfrak{p}_j \neq \mathfrak{p}_{j'}$, and so we can remove the j'-th column $M'^{(j')}$ and set the j-th column $M'^{(j)}$ as $M'^{(j)} + 3^n M'^{(j')}$. Then we denote the new matrix M^* as the reduced M'. Notice that this technique is also used for algebraic factorbase. Consequently, the number of unknowns is about 1/6 of the original one, thus the number of relations is reduced to about 1/6. By this technique, the collection of relations step is about 6 times faster and the linear algebra step is about 6^2 times faster.

The parallel Lanczos method The reduced matrix M^* is reconstructed to optimize firstly, and then we apply the parallel Lanczos method to it. Before explaining the reconstruction, we begin with the explanation of the parallel computation. Assume that there are 4 nodes written as $N_{1,1}, N_{1,2}, N_{2,1}, N_{2,2}$ and each node has 4 or 8 cores. As the Figure 3, we partition the reconstructed matrix M into 4 matrices $M_{i,j}$, and each $M_{i,j}$ is alloted to the node $N_{i,j}$ respectively. The given vector $\boldsymbol{v}$ is also shared into $\boldsymbol{v}_1, \boldsymbol{v}_2$, and $\boldsymbol{v}_j$ is given to the node $N_{i,j}, N_{i',j}$ where $i \neq i'$. Moreover $M_{i,j}$ is shared into L matrices A_ℓ, when $N_{i,j}$ has L cores.

Fig. 3. Sharing the M into 4 matrices $M_{i,j}$, and an $M_{i,j}$ into L matrices A_ℓ.

$$M\boldsymbol{v} = \left(\begin{array}{c|c} M_{1,1} & M_{1,2} \\ \hline M_{2,1} & M_{2,2} \end{array}\right)\left(\begin{array}{c} \boldsymbol{v}_1 \\ \hline \boldsymbol{v}_2 \end{array}\right). \qquad M_{i,j}\boldsymbol{v_j} := A\boldsymbol{b} = \left(\begin{array}{c} A_1 \\ \hline A_2 \\ \hline \vdots \\ \hline A_L \end{array}\right)\boldsymbol{b}.$$

Here, we give the notation of the Lanczos method. The Lanczos method can operate only a symmetric matrix, however the given matrix M is usually a non-symmetric matrix. Therefore, we try to solve the linear equation of the form $M^T M\boldsymbol{v} = \boldsymbol{\alpha}$, where $\boldsymbol{v}$ is an unknown column vector consists of the logarithms of factorbase and $\boldsymbol{\alpha}$ is the given column vector. Note that computing $M^T M$ is not efficient, and so we compute the vector $\boldsymbol{u} = M\boldsymbol{v}$ and $M^T\boldsymbol{u}$. For more details about this computation is written in [20].

After partitioning M, we perform the parallel computation for $\boldsymbol{u} := M\boldsymbol{v}$ and $\boldsymbol{w} := M^T\boldsymbol{u}$ with $M_{i,j}$. Let $\boldsymbol{v}_1$, $\boldsymbol{v}_2$, $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ be the partitioned vectors such that $\boldsymbol{v} = \boldsymbol{v}_1 \oplus \boldsymbol{v}_2$ and $\boldsymbol{u} = \boldsymbol{u}_1 \oplus \boldsymbol{u}_2$. Via Algorithm 1, we obtain the partitioned vector $\boldsymbol{w}_i$ such that $\boldsymbol{w} = \boldsymbol{w}_i \oplus \boldsymbol{w}_{i'}$ in the node $N_{i,j}$, where $i \in \{1, 2\}$ and $i' = 3 - i$. The symbol j' also means that $j' = 3 - j$ for $j \in \{1, 2\}$.

Algorithm 1 (Computation by the node $N_{i,j}$.)
Input : the partitioned matrix $M_{i,j}$ and the short partitioned $\boldsymbol{v}_j$.
Output : the partitioned vector $\boldsymbol{w}_j$ such that $\boldsymbol{w}_1 \oplus \boldsymbol{w}_2 = M^T M\boldsymbol{v}$, where i is equal to 1 or 2.
[Step for the computation of $u := Mv$]
1. $\boldsymbol{u}_{i,j} := M_{i,j}\boldsymbol{v}_j$.
2. Give $\boldsymbol{u}_{i,j}$ to the Node $N_{i,j'}$, and receive $\boldsymbol{u}_{i,j'}$ from $N_{i,j'}$.
3. $\boldsymbol{u}_j := \boldsymbol{u}_{i,j} + \boldsymbol{u}_{i,j'}$.

[Step for the computation of $w := M^T u$]
4. $\boldsymbol{w}_{i,j} := M_{i,j}^T\boldsymbol{u}_i$.
5. Give $\boldsymbol{w}_{i,j}$ to the Node $N_{i',j}$, and receive $\boldsymbol{w}_{i',j}$ from $N_{i',j}$.
6. $\boldsymbol{w}_j := \boldsymbol{w}_{i,j} + \boldsymbol{w}_{i',j}$.

Line 4, 5 and 6 describe the computation of $M^T u$. Note that, in each node $N_{i,j}$, by regarding the column of $M_{i,j}$ as the row of $M_{j,i}^T$, we do not have to trade $M_{i,j}$ with $M_{j,i}^T$, namely, we can cut unnecessary operations off.

We have discussed the parallel computations among nodes, and now we move on to the parallel computations among cores in one node. Via Algorithm 2, we can easily obtain $A_\ell\boldsymbol{b}$, and then we set the new vector $\boldsymbol{c} := (A_1\boldsymbol{b}, ..., A_L\boldsymbol{b})^T$ where L is number of cores in the same node. We compute $A^T\boldsymbol{c}$ by regarding the column of A_ℓ as the row of A_ℓ^T. Algorithm 2 describes the computation among L cores in the same node.

Algorithm 2 (Parallel computation among L cores in the same node.)
Input : the small matrix A whose size is $s \times t$ and the partitioned t-vector v.
Output : the partitioned vector $\boldsymbol{d}$ such that $w = A^T A\boldsymbol{b}$.

[Step for the computation of $\boldsymbol{c} := A\boldsymbol{b}$]

1. $u := \oplus_{\ell=1}^{L} A_\ell \boldsymbol{b}$. // This is a parallel computation.

[Step for the computation of $\boldsymbol{d} := A^T\boldsymbol{c}$]

2. Partition u into small vectors $\boldsymbol{c}_1, ..., \boldsymbol{c}_L$ such that the size of $\boldsymbol{c}_\ell$ is equal to s_ℓ, where A_ℓ is the $s_\ell \times t$ matrix.
3. $\boldsymbol{d} = \sum_{\ell=1}^{L} A_\ell^T \boldsymbol{c}_\ell$ // This is a parallel computation.

By the parallel computations of $M_{i,j}\boldsymbol{v}_j$ and so on, we obtain the vector $M^T M\boldsymbol{v}$ via Algorithm 1 and Algorithm 2. Therefore, we need to reconstruct M so that each node has the balanced calculation amount of computing $M_{i,j}\boldsymbol{v}_j$ and so on. It is clear that the calculation amount depends on the number of non-zero elements in allotted matrix, and the distribution of non-zero elements in M is not uniformity. In fact, the number of non-zero elements in a column of M is not balanced, but the one in a row is balanced. Thus, we reconstruct the new matrix M so that the number of non-zero elements in $M_{1,1}$ and $M_{2,1}$ is almost equal to the one in $M_{1,2}$ and $M_{2,2}$, by sorting columns of M^* defined in the section of Galois action. We perform the similar strategy as above for the parallel computation among cores in the same node, namely, A is shared into 4 or 8 smaller matrices A_ℓ so that each A_ℓ has the almost same number of non-zero elements.

4.3 The Computation Result

In this section, we describe our computation result of the 676-bit size DLP in $\mathrm{GF}(3^{6\cdot 71})$ which contains a multiplicative subgroup whose order is a 112-bit prime. We construct $\mathrm{GF}(3^6)$ as $\mathrm{GF}(3)[z]/(z^6+2z+2)$, and define a mapping $\psi : \mathbb{Z} \to \mathrm{GF}(3^6)[x]$, such that $\psi^{-1} : z \mapsto 3, x \mapsto 3^6$, in order to represent the element in $\mathrm{GF}(3^{6\cdot 71})$.

In the polynomial selection step, we set $H(x, y) = y^6 + x$ in order to use Galois action. Moreover, we select $m \in \mathrm{GF}(3^6)[x]$ such that all its coefficients are in $\mathrm{GF}(3)$, to construct f whose coefficients are also in $\mathrm{GF}(3)$. By an easy computation, we obtain proper m and f as follows,

$$
\begin{aligned}
m = \psi\,(&\texttt{0x456bc 60e76c11 1e679735 c929fc55})\\
f = \psi\,(&\texttt{\quad 0x9 2d3e5daf 5ac01130 4e6909f7 09cc8833 baa757d3}\\
&\texttt{17dc6f99 9c8b98b5 ab8baa01 d68ec151 aec39e2e ed081c79}\\
&\texttt{d851066b 3ffb2a4f a3e19c1e cef46675 0918a26d 9c7cacd4}\\
&\texttt{8d74ccfe 2c1d3b79 e81e6138 ab06aef4}).
\end{aligned}
$$

Then, $\mathrm{GF}(3^{6n})$ is constructed as $\mathrm{GF}(3^6)[x]/(f)$ where $\mathrm{GF}(3^6) \cong \mathrm{GF}(3)[z]/(z^6+2z+2)$. When we set the smoothness bound $B = 2$, there are 266,085 elements in the rational factorbase and 265,721 elements in the algebraic factorbase, so we need to collect at least 531,806 relations. However, the size of sieving area when $B = 2$ is too small to collect enough relations.

We settle this problem by using Galois action, since we can considerably reduce the number of required elements in the factorbase described as Section 4.2. In fact, we need only 88,674 relations, and so this number is about 1/6 of the number of the originally required relations.

Moreover, we deal with free relations which are obtained without sieving. If we choose $H(x, y)$ as y^6+x, then it is fortunately factored as $(y-t_1)^3(y-t_2)^3 \pmod{\mathfrak{p}}$ for most of elements $\mathfrak{p}$ in factorbase, and so there are 132,860 ($\approx \#B_A/2$) free relations. Even if we delete many duplicates which come out by using Galois action, 22,155 free relations are remained. Thus, we only have to find at least 66,519 relations in the collection of relations step, and 66,519 is about 1/8 of the number of originally required relations.

In the collection of relations step, we use the polynomial sieve described in Section 4.1, and compute relations using five nodes each of which has Intel Quad-Core Xeon E5440 (2.83GHz) × 2 CPUs, 16GB RAM, a node has Intel Quad-Core Xeon X5355 (2.66GHz) × 2 CPUs, 16GB RAM, and twelve nodes each of which has Intel Quad-Core Xeon L5420 (2.33GHz) × 1 CPU, 4GB RAM, total 96 cores. With 18 days computation, after removing duplicates, we find 66,646 relations. Thus, totally we obtain 88,801 relations which are enough to solve the linear equation in equation (7).

Since $(3^{6n}-1)$ can be factored as $(3^{2n}+3^n+1)(3^{2n}-3^n+1)(3^n+1)(3^n-1)$, we work modulo the product of over 30-bit prime factors of each cofactor in order to avoid failing in the Lanczos method, and solve in parallel in the linear algebra step. Using a cluster with four nodes each of which has Intel Quad-Core Xeon E5440 (2.83GHz) × 2 CPUs, 16GB RAM, and three clusters with four nodes each of which has Intel Quad-Core Xeon L5420 (2.33GHz) × 1 CPU, 4GB RAM, we compute via the parallel Lanczos method described in Section 4.2 about 12 hours and solve unknowns modulo each product. With Chinese remainder theorem and Galois action of ϕ, we solved discrete logarithms of elements in factorbase modulo the product of over 30-bit prime factors of $(3^{6\cdot 71})-1$. Some examples of the relation and discrete logarithms of elements in factorbase are given in Appendix.

In the individual logarithm step, our target of computing the logarithm is the element

$$\begin{aligned}\pi(x) &= \psi(\lfloor \pi \times 10^{202} \rfloor) \\ &= (z^4+z^3+2z^2+1)x^{70} + \cdots + (z^5+2z^4+2z^3+z^2+2)\end{aligned}$$

in basis $\gamma = \psi(\mathtt{456})$. We choose the representation of $\pi(x)$ as a product of elements of degree at most 7, and compute the logarithms of $\pi(x)$ in basis g using special-$\mathfrak{q}$ descent technique [16, 18].

Unfortunately, we have not finished the individual logarithm step yet, however, we expect that about 2 weeks computation using five nodes each of which has Intel Quad-Core Xeon E5440 (2.83GHz) × 2 CPUs, 16GB RAM and a node each of which has Intel Quad-Core Xeon X5355 (2.66GHz) × 2 CPUs, 16GB RAM, is required in this step, and about 50 percents of the computation have already finished when we posted this paper.

4.4 For Larger Extension Degrees

We have solved the DLP in GF(3^{6n}) for n in the experimental class, where the smoothness bound B (i.e., B_{06}) is less than or equal to 2 (ref. Table 1). Note that the size of sieving area rises $(3^6)^2$-fold if the smoothness bound B grows by one (see equation (11)). However, we expect that, if we set $B = 3$, the DLP in GF($3^{6\cdot 97}$) might be computed for some years by using dozens of our computational resources through some techniques: large prime variation, block sieving and sieving via bucket sort [29, 4], SIMD implementation, and so on.

5 Concluding Remarks

In this study, we implemented the new variant of the FFS in GF(3^{6n}) (n is a prime), proposed by Joux and Lercier in 2006 [18], and compared it with the earlier variant which is also proposed by Joux and Lercier in 2002 [16] by practical experiments. In solving the DLP in GF(3^{6n}), these two variants of the FFS have the same asymptotic complexity, but the new variant was expected more efficient than the earlier one in some extension degree n. By our experiment result, we confirmed this forecast when the extension degree $n = 19, 61$. Moreover, with our implementations, we computed

Table 3. The top-record of solving the DLP in finite fields

Finite Fields	GF(p)	GF(2^n)	GF(p^3)	GF(p^{30})	**GF(3^{6n})**
Reference	[21]	[15]	[19]	[18]	**This Work**
Date	Feb. 5, 2007	Sep. 22, 2005	Aug. 23, 2006	Nov. 9, 2005	**Nov. 12, 2009**
Algorithm	NFS*	JL02-FFS	JLSV06-NFS†	JL06-FFS-2‡	**JL06-FFS**
Collection of Relations	Many CPUs¶	4 nodes of 16 Itanium 2 (1.3GHz)	16 Alpha processors (1.15GHz)	16 Alpha processors (1.15GHz)	**Xeon (2.83GHz) 96 cores in total**
Linear Algebra	12–24 Xeon (3.2GHz)	4 nodes of 16 Itanium 2 (1.3GHz)	16 Alpha processors (1.15GHz)	16 Alpha processors (1.15GHz)	**Xeon (2.83GHz) 80 cores in total**
Timing	33 days	17 days	19 days	12 hours	**19 days**
Bit Size	532	613	394	556	**676**

*NFS: Number Field Sieve [10, 17]. †JLSV06-NFS: NFS in the medium prime case [19].
‡See footnote [2] in page 2. ¶No detailed description about computational resources is in [21].

the discrete logarithm of elements in factorbase in GF($3^{6\cdot 71}$) of 676-bit size with about 19 days computation.

We have experimented the DLP in GF(3^{6n}) required for pairing-based cryptosystems. The security of pairing-based cryptosystems relies on the hardness of the DLP in various finite fields, for example, GF(2^{4n}), GF(p^{12}), etc. Table 3 presents the current top-record of solving the DLP in various finite fields. All the DLPs used for pairing-based cryptosystems have not examined yet. It is an open problem to analyze the hardness of the DLP with practical key sizes in such finite fields.

References

1. L. M. Adleman. The function field sieve. *ANTS-I*, LNCS 877, pp. 108–121, 1994.
2. L. M. Adleman and M.-D. A. Huang. Function field sieve method for discrete logarithms over finite fields. *Inform. and Comput.*, Vol. 151, pp. 5–16, 1999.
3. K. Aoki, T. Shimoyama, and H. Ueda. Experiments on the linear algebra step in the number field sieve. *IWSEC 2007*, LNCS 4752, pp. 58–73, 2007.
4. K. Aoki and H. Ueda. Sieving using bucket sort. *ASIACRYPT 2004*, LNCS 3329, pp. 92–102, 2004.
5. P. S. L. M. Barreto, S. Galbraith, C. Ó hÉigeartaigh, and M. Scott. Efficient pairing computation on supersingular abelian varieties. *Des., Codes Cryptogr.*, Vol. 42, No. 3, pp. 239–271, 2007.
6. J.-L. Beuchat, N. Brisebarre, J. Detrey, E. Okamoto, M. Shirase, and T. Takagi. Algorithms and arithmetic operators for computing the η_T pairing in characteristic three. *IEEE Trans. Comput.*, Vol. 57, No. 11, pp. 1454–1468, 2008.
7. B. L. Bender and C. Pomerance. Rigorous discrete logarithm computations in finite fields via smooth polynomials. *AMS/IP*, Vol. 7, pp. 221–232, 1998.
8. D. Boneh, D. Crescenzo, R. Ostrovsky and G. Persiano. Public key encryption with keyword search. *EUROCRYPT 2004*, LNCS 3027, pp. 506–522, 2004.
9. D. Boneh and M. Franklin. Identity based encryption from the Weil pairing. *SIAM J. Comput.*, Vol. 32, No. 3, pp. 586–615, 2003.
10. D. M. Gordon. Discrete logarithms in GF(p) using the number field sieve. *SIAM J. Discrete Math.*, vol. 6, no. 1, pp. 124-138, 1993.
11. D. M. Gordon and K. S. McCurley. Massively parallel computation of discrete logarithms. *CRYPTO' 92*, LNCS 740, pp. 312–323, 1992.
12. R. Granger, A. J. Holt, D. Page, N. P. Smart, and F. Vercauteren. Function field sieve in characteristic three. *ANTS-VI*, LNCS 3076, pp. 223–234, 2004.
13. R. Granger, D. Page, and M. Stam. Hardware and software normal basis arithmetic for pairing-based cryptography in characteristic three. *IEEE Trans. Comput.*, Vol. 54, No. 7, pp. 852–860, 2005.

14. D. Hankerson, A. Menezes, and M. Scott. Software Implementation of Pairings. In *Identity-Based Cryptography*, pp. 188–206, 2009.
15. A. Joux et al. Discrete logarithms in GF(2^{607}) and GF(2^{613}). Posting to the Number Theory List, available at `http://listserv.nodak.edu/cgi-bin/wa.exe?A2=ind0509&L=nmbrthry&T=0&P=3690`, 2005.
16. A. Joux and R. Lercier. The function field sieve is quite special. *ANTS-V*, LNCS 2369, pp. 431–445, 2002.
17. A. Joux and R. Lercier. Improvements to the general number field sieve for discrete logarithms in prime fields. A comparison with the Gaussian integer method. *Math. Comp.*, Vol. 72, No. 242, pp. 953–967, 2002.
18. A. Joux and R. Lercier. The function field sieve in the medium prime case. *EUROCRYPT 2006*, LNCS 4004, pp. 254–270, 2006.
19. A. Joux, R. Lercier, N. P. Smart, and F. Vercauteren. The number field sieve in the medium prime case. *CRYPTO 2006*, LNCS 4117, pp. 326–344, 2006.
20. B. A. LaMacchia and A. M. Odlyzko. Solving large sparse linear systems over finite fields. *CRYPTO' 90*, LNCS 537, pp. 109–133, 1991.
21. T. Kleinjung et al. Discrete logarithms in GF(p) - 160 digits. Posting to the Number Theory List, available at `http://listserv.nodak.edu/cgi-bin/wa.exe?A2=ind0702&L=nmbrthry&T=0&P=194`, 2007.
22. C. Lanczos. Solution of systems of linear equations by minimized iterations. *J. Res. Nat. Bur. Stand.*, vol. 49, pp. 33–53, 1952.
23. R. Matsumoto. Using C_{ab} curves in the function field sieve. *IEICE Trans. Fundamentals*, Vol. E82-A, pp. 551–552, 1999.
24. A. J. Menezes, T. Okamoto and S. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Trans. Inform. Theory*, Vol. 39, No. 5, pp. 1639–1646, 1993.
25. D. Page, N. P. Smart, and F. Vercauteren. A Comparison of MNT Curves and Supersingular Curves. *Appl. Algebra Engrg. Comm. Comput.*, Vol. 17, No. 5, pp. 379–392, 2006.
26. J. Pollard. The lattice sieve. In *The Development of the Number Field Sieve*, pp. 43–49, 1991.
27. C. Pomerance and J. W. Smith. Reduction of huge, sparse matrices over finite fields via created catastrophes. *Experiment. Math.*, Vol. 1, No. 2, pp. 89–94, 1992.
28. O. Schirokauer. The special function field sieve. *SIAM J. Discrete Math.*, Vol. 16, No. 1, pp. 81–98, 2003.
29. G. Wambach and H. Wettig. Block sieving algorithms. Technical Report 190, Informatik, Universität zu Köln, 1995.
30. D. H. Wiedemann. Solving sparse linear equations over finite fields. *IEEE Trans. Inform. Theory*, Vol. 32, No. 1, pp. 54–62, 1986.

Appendix: Some Solutions of the DLP in GF($3^{6\cdot 71}$)

We present some solutions (discrete logarithms) in factorbase used in our implementation for solving the DLP in GF($3^{6\cdot 71}$). We have found 66,646 relations satisfying equation (5). We give one of them as an example,

$$\sum_{i=0}^{6} \log_\gamma \mathfrak{p}_i \equiv \sum_{j=0}^{4} 3 \log_\gamma \kappa_j \pmod{(3^{6\cdot 71}-1)/(3^6-1)}, \tag{12}$$

where each $\mathfrak{p}_i$ is in rational factorbase,

$$\mathfrak{p}_0 = \psi(\texttt{0x2d9}),\ \mathfrak{p}_1 = \psi(\texttt{0x90581}),\ \mathfrak{p}_2 = \psi(\texttt{0x9ea2b}),\ \mathfrak{p}_3 = \psi(\texttt{0xb1a07}),$$
$$\mathfrak{p}_4 = \psi(\texttt{0xb942e}),\ \mathfrak{p}_5 = \psi(\texttt{0xcada1}),\ \mathfrak{p}_6 = \psi(\texttt{0xd6d36}),$$

and each κ_j corresponding to an element in algebraic factorbase by equation (6) is given as follows

$$\kappa_0 : \langle \psi(\texttt{0x3c3}), y - \psi(\texttt{0x175}) \rangle, \ \kappa_1 : \langle \psi(\texttt{0x3c4}), y - \psi(\texttt{0x200}) \rangle,$$
$$\kappa_2 : \langle \psi(\texttt{0x533}), y - \psi(\texttt{0x258}) \rangle, \ \kappa_3 : \langle \psi(\texttt{0xda9c2}), y - \psi(\texttt{0x4cc58}) \rangle,$$
$$\kappa_4 : \langle \psi(\texttt{0xed6e4}), y - \psi(\texttt{0x387b6}) \rangle.$$

Let N be the product of prime factors of $(3^{6 \cdot 71} - 1)$, where those prime factors are not larger than 55,126,531 (Note that N is a 602-bit integer). Equation (12) also holds modulo N instead of $(3^{6 \cdot 71} - 1)/(3^6 - 1)$, and so we obtain the following solutions of equation (12) modulo N except for $\log_\gamma \mathfrak{p}_6$ and $\log_\gamma \kappa_4$, after performing the linear algebra step:

$$\log_\gamma \mathfrak{p}_0 \equiv \texttt{0x8 9e0c0faa 4190baa5 c885e3b7 308ae498 eb2d4a03 0dfab3d9 16437d96 bfd4e2b9 014f5402 90aa2f83 7b9cc76b 16ae97ef dcc9c319 670f0f9c 47e8ea96 4754cfbf 1529c311},$$
$$\log_\gamma \mathfrak{p}_1 \equiv \texttt{0x2 e8b84752 70de651a b03ae702 e3268e86 77179013 0c9edab5 31d2ac5b 2a23da92 2e8352c5 321832bf ff36a8d5 2d16c9e5 ae47c6fc 2ba7a1c5 cc990233 34c3d6da 25e08d52},$$
$$\log_\gamma \mathfrak{p}_2 \equiv \texttt{0x7 b565cae8 39dc8d83 415b0b9e 164c7b55 6e57ad98 80b8f232 7cf30ebe 972ac1fb 2d1133be 5cdd9604 c9ea6e83 c1c8c9f3 2f9fa4c6 51d65ded 33d2e4c7 8ff8d162 3a5408c9},$$
$$\log_\gamma \mathfrak{p}_3 \equiv \texttt{0x6 ae81aef6 7c0fddcf 7c23e69e c3f18e07 bf546751 8df9d1ad 78113a85 9a2578c8 36764402 2598160b 5c055ed4 7d412a42 17c987c0 14aafff7 03ef6fa4 c6771dfd 150b88f2},$$
$$\log_\gamma \mathfrak{p}_4 \equiv \texttt{0x7 2e418546 92ba2b75 8d0831df 1d5ca5c0 f6d8a05d 0528c97d 16c4f782 d9b59ce7 d55deefe bf85390a 23113680 b184d203 d1d3b6a4 e9d9263a 8544acd7 5afc9974 78a4498a},$$
$$\log_\gamma \mathfrak{p}_5 \equiv \texttt{0x1 c35f26bf 717ed338 cfd71243 b86c024b 98b18342 4710450a d9aaf2e3 557ce5ed debbc870 0fc840f2 19aca778 2ba931a2 cdd2cb53 a2dafcaa 28a5176e a378bf8c 9a6cd33c},$$
$$\log_\gamma \kappa_0 \equiv \texttt{0x 92671082 6cf3288f 1c83edcf 66fb9041 9bb2239c 10cd8445 820d975e 6f9730fb f4ca3005 279a500d b2fc0f60 b4425edb 65991a31 629d54e7 84ae64b6 080828b3 0fc6ba0b},$$
$$\log_\gamma \kappa_1 \equiv \texttt{0x7 06c2cfcd 7fb4f7c8 386ea65b c0c259c3 f14888ec ccda75ee 77ddddf4 065a7da6 981af728 98699166 c52484c6 73bbefbd a4660135 1244b297 42f3cf76 fdab7cad 3d01e8a1},$$
$$\log_\gamma \kappa_2 \equiv \texttt{0x5 4623bf43 0ede6e43 bbe3cb8b a79c1400 97f7ac1e 2320c70e 5a700159 4460b073 e5c670c5 d19921ea 59f4f9c6 41ce8203 28edb204 94bd322f 3551d5ee 472cf59b d58d0bd0},$$
$$\log_\gamma \kappa_3 \equiv \texttt{0x6 e063f01c 43624c96 30712701 2223edf3 95ddfdc2 aa1dd9f6 dd3636ef 12d9260f 555a2101 c0e94fe5 9a524c5b c2c1d768 1499d7b6 41b71d4f b13566b3 b39794c5 90ff78cb},$$

mod N. Finally, by Galois action, we obtain that

$$\log_\gamma \mathfrak{p}_6 \equiv 3^{3 \cdot 71} \log_\gamma \mathfrak{p}_3 \equiv \texttt{0x7 3deb8075 ee684576 073761e2 974c4eba 72df97ce 299f9e46 87ae3f70 b6cd8b50 1c65ccb3 e9ed8f80 08387efe 9326eea8 7302c1a5 1f0671b5 22e32949 81250923 9b072989},$$
$$\log_\gamma \kappa_4 \equiv 3^{3 \cdot 71} \log_\gamma \kappa_3 \equiv \texttt{0x4 0473a949 4056ac7c 76677e6f a284977a 2a2e539f 751d5e0b ee628ca8 63e7f732 a02886c2 0711d445 0006c79a 778c6fbf abb923e7 e89deb8d 0c7f5508 2d797bd2 2414eaa1},$$

mod N.