Journal of Information Processing Vol. 19 441–450 (Sep. 2011)

Regular Paper

Experimental Analysis of Cheon's Algorithm against Pairing-friendly Curves[⋆1]

Tetsuya Izu,[†1] Masahiko Takenaka[†1] and Masaya Yasuda[†1]

Let $\mathbb{G}$ be an additive group generated by an element G of prime order r. The discrete logarithm problem with auxiliary input (DLPwAI) is a problem to find α on inputs $G, \alpha G, \alpha^d G \in \mathbb{G}$ for a positive integer d dividing $r-1$. The infeasibility of DLPwAI ensures the security of some pairing-based cryptographic schemes. In 2006, Cheon proposed an algorithm for solving DLPwAI which works better than conventional algorithms. In this paper, we report our experimental results of Cheon's algorithm on a pairing-friendly elliptic curve defined over GF(3^{127}). Moreover, based on our experimental results, we estimate the required cost of Cheon's algorithm to solve DLPwAI on some pairing-friendly elliptic curves over a finite field of characteristic 3. Our estimation implies that DLPwAI on a part of pairing-friendly curves can be solved at reasonable cost when the optimal parameter d is chosen.

1. Introduction

Let $\mathbb{G}$ be an additive group generated by an element G of prime order r (We mainly consider the case where the group $\mathbb{G}$ is the Mordell-Weil group of an elliptic curve defined over a finite field). The discrete logarithm problem (DLP) is a problem to find $\alpha \in \mathbb{Z}/r\mathbb{Z}$ on inputs G, $\alpha G \in \mathbb{G}$. In general, the most efficient algorithms for solving DLP require $O(\sqrt{r})$ in time, and DLP is considered to be infeasible when parameters are properly chosen. The security of some cryptographic schemes relies on the infeasibility of DLP.

In 2006, Cheon proposed an algorithm for solving DLP with auxiliary input (DLPwAI). DLPwAI is a problem to find $\alpha \in \mathbb{Z}/r\mathbb{Z}$ on inputs G, $\alpha G, \alpha^d G \in \mathbb{G}$ for a positive integer d dividing $r-1$ [6]. Since the time complexity of Cheon's algorithm is $O(\log r(\sqrt{(r-1)/d} + \sqrt{d}))$ and the space complexity is $O(\max(\sqrt{(r-1)/d},\ \sqrt{d}))$, Cheon's algorithm may work better than conventional algorithms. Especially, it only requires $O(\sqrt[4]{r})$ in time and space when $d \approx \sqrt{r}$. Recently, new (possibly infeasible) problems are introduced to ensure the security of some pairing-based cryptographic schemes, such as the ℓ-WDH problem [17], the ℓ-SDH problem [3], the ℓ-sSDH problem [5], the ℓ-BDHI problem [4] and the ℓ-BDHE problem [5]. Since a common property of these problems is that the elements G, αG, $\alpha^2 G$, …, $\alpha^\ell G \in \mathbb{G}$ are used as inputs, we can use Cheon's algorithm to solve these problems when ℓ is larger than d.

When we implement Cheon's algorithm, we choose either the kangaroo method or the baby-step giant-step (BSGS) method as a subroutine algorithm [6],[7]. Jao, et al. implemented Cheon's algorithm with the kangaroo method [14]. Moreover, Sakemi, et al. implemented Cheon's algorithm with both methods [19]. Note that their works were implemented on elliptic curves over a prime field. We here focus on elliptic curves over a finite field of characteristic 3. In Refs. 12), 13), Izu, et al. implemented Cheon's algorithm with the BSGS method on a pairing-friendly elliptic curve defined over GF(3^{127}) introduced in Ref. 2). In this paper, we report our experimental results on Cheon's algorithm with the kangaroo method on the same curve, and compare the complexity of Cheon's algorithm with both methods. Moreover, based on our experimental results, we estimate the required cost of Cheon's algorithm to solve DLPwAI on some pairing-friendly elliptic curves. As a feedback of our estimation, we can see that it is better to avoid using some pairing-friendly curves when we implement pairing-based cryptographic schemes based on the problems mentioned above.

The rest of this paper is organized as follows: In Sections 2 and 3, we introduce the BSGS method [20] and the kangaroo method [18], respectively, and their combination with Cheon's algorithm. In Section 4, we report our experimental results on Cheon's algorithm with the kangaroo method. In Section 5, we compare the complexity of Cheon's algorithm with both methods. In Section 6, we describe some speeding-up techniques. In Section 7, we estimate the required cost of Cheon's algorithm to solve DLPwAI on some pairing-friendly elliptic curves. Finally, in Section 8, we conclude our work.

†1 FUJITSU LABORATORIES LTD.
⋆1 A part of this paper will be published at the 25th International Conference on Advanced Information Networking and Applications (AINA-2011)

2. Cheon's Algorithm and the BSGS Method

Let $\mathbb{G}$ be an additive group generated by an element G of prime order r. DLP in the group $\mathbb{G}$ is a problem to find $\alpha \in \mathbb{Z}/r\mathbb{Z}$ on inputs $G, \alpha G \in \mathbb{G}$.

2.1 The BSGS Method

The BSGS method of Shanks[20] can be used to solve DLP in $\mathbb{G}$. Instead of finding a solution α directly, the BSGS method searches two integers $i,\ j$ such that $\alpha = i + jm$ and $0 \le i,\ j < m = \lceil \sqrt{r} \rceil$, where $\lceil x \rceil$ denotes the ceiling of a real number x. We note that such integers i, j are uniquely determined. Since $\alpha G = (i + jm)G = iG + jG'$ with $G' = mG$, we have a relation $G_1 - iG = jG'$ with $G_1 = \alpha G$.

The BSGS method are mainly composed of two steps as follows: In the first step (= baby-step), we compute

$$G_1,\ G_1 - G,\ G_1 - 2G, \ldots, G_1 - (m-1)G$$

successively and store them in a table. In the second step (= giant-step), we compute

$$G',\ 2G', \ldots, (m-1)G'$$

successively and store them in another table. Once we find a collision $G_1 - iG = jG'$ from these tables, we can obtain a solution $\alpha = i + jm$. Since $O(m)$-group operations and $O(m)$-group elements are required in each step, the time and space complexity of the BSGS method are $O(\sqrt{r})$-group operations and $O(\sqrt{r})$-group elements, respectively.

2.2 Cheon's Algorithm with the BSGS Method

DLPwAI is a problem to find α on inputs $G,\ G_1 = \alpha G,\ G_d = \alpha^d G \in \mathbb{G}$ for a positive integer d dividing $r - 1$. Let us describe Cheon's algorithm for solving DLPwAI. Fix a generator ζ of the multiplicative group $(\mathbb{Z}/r\mathbb{Z})^*$. Note that the generator ζ can be found efficiently. The goal of Cheon's algorithm is to find an integer k with $\alpha = \zeta^k$. Such an integer k is uniquely determined up to modulo r. To find k, Cheon's algorithm tries to find two integers $k_1,\ k_2$ such that $k = k_1 + k_2(r-1)/d$ satisfying $0 \le k_1 < (r-1)/d,\ 0 \le k_2 < d$ in the following two steps (The outline of Cheon's algorithm is shown in **Algorithm 1**):

Algorithm 1. Cheon's algorithm with the BSGS method.

```
Input: $G,\ G_1 = \alpha G,\ G_d = \alpha^d G \in \mathbb{G}$
Output: $\alpha \in \mathbb{Z}/r\mathbb{Z}$
1. Find a generator $\zeta \in (\mathbb{Z}/r\mathbb{Z})^*$
2. $\zeta_d \leftarrow \zeta^d,\ d' \leftarrow (r-1)/d$
3. [Step 1] $d_1 \leftarrow \left\lceil \sqrt{d'} \right\rceil$
4.          Find $0 \le u_1,\ v_1 < d_1$ such that $\zeta_d^{-u_1} G_d = \zeta_d^{v_1 d_1} G$.
5.          $k_1 \leftarrow u_1 + v_1 d_1$
6. [Step 2] $d_2 \leftarrow \left\lceil \sqrt{d} \right\rceil$
7.          Find $0 \le u_2,\ v_2 < d_2$ such that $\zeta^{-u_2 d'} G_1 = \zeta^{k_1 + v_2 d_2 d'} G$.
8.          $k_2 \leftarrow u_2 + v_2 d_2$
9. Output $\alpha \leftarrow \zeta^{k_1 + k_2 d'}$
```

Step 1 searches an integer k_1 such that $\alpha^d = \zeta_d^{k_1}$ with $\zeta_d = \zeta^d$, or equivalently, searches two integers $u_1,\ v_1$ such that

$$\alpha^d \zeta_d^{-u_1} = \zeta_d^{v_1 d_1}$$

satisfying

$$0 \le u_1, v_1 < d_1 = \left\lceil \sqrt{(r-1)/d} \right\rceil.$$

Such $u_1,\ v_1$ are uniquely determined. In practice, we search $u_1,\ v_1$ such that $\zeta_d^{-u_1} G_d = \zeta_d^{v_1 d_1} G$. Similarly, Step 2 searches an integer k_2 such that $\alpha = \zeta^{k_1 + k_2(r-1)/d}$, or equivalently, searches integers $u_2,\ v_2$ such that

$$\alpha \zeta^{-u_2 \frac{r-1}{d}} = \zeta^{k_1 + v_2 d_2 \frac{r-1}{d}}$$

satisfying

$$0 \le u_2, v_2 < d_2 = \left\lceil \sqrt{d} \right\rceil.$$

Such $u_2,\ v_2$ are uniquely determined. In practice, we search $u_2,\ v_2$ such that $\zeta^{-u_2(r-1)/d} G_1 = \zeta^{k_1 + v_2 d_2 (r-1)/d} G$. In searching $u_1,\ v_1$ in Step 1 and $u_2,\ v_2$ in Step 2, we can use the BSGS method as a subroutine algorithm.

2.3 Complexity

The complexity of Cheon's algorithm with the BSGS method is as follows[6),7)]:

- Time : $O(\log r \cdot (\sqrt{(r-1)/d} + \sqrt{d}))$-group operations,
- Space : $O(\max(\sqrt{(r-1)/d}, \sqrt{d}))$-group elements.

Here, the term '$\log r$' is due to the complexity of a scalar multiplication on the group $\mathbb{G}$. Kozaki, Kutsuma and Matsuo introduced a precomputated table on fixed points to Cheon's algorithm (KKM method)[16]. Their method can reduce the time complexity to $O(\sqrt{(r-1)/d} + \sqrt{d})$-group operations.

3. Cheon's Algorithm and the Kangaroo Method

Pollard's kangaroo method is one of the most efficient algorithms for solving DLP in general (see Ref. 18) for detail). In Refs. 6), 7), Cheon also proposed a combination with the kangaroo method. Compared with the combination of the BSGS method, Cheon's algorithm with the kangaroo method can reduce the table size, using the distinguished points technique described later.

3.1 Techniques for Cheon's Algorithm with the Kangaroo Method

In order to combine the kangaroo method with Cheon's algorithm, the following three techniques are needed:

3.1.1 Search-space Expansion

In Step 1 of Algorithm 1, the size of the search space is $O(\sqrt{(r-1)/d})$. In the search space, two integers $u_1,\ v_1$ are uniquely determined. Thus we examine the entire search space and store all values. In the kangaroo method, Step 1 is modified as follows: Search $u_1,\ v_1$ such that

$$\alpha^d \zeta_d^{u_1} = \zeta_d^{v_1}$$

satisfying

$$0 \le u_1, v_1 < \lceil (r-1)/d\ \rceil.$$

In this case, the size of the total search space is $O(((r-1)/d)^2)$. However, there are $O\left((r-1)/d\right)$-pairs of u_1, v_1 and the search space for one pair of u_1, v_1 is $O((r-1)/d)$ on average. By using the following random-walk function, a substantive size of the search space is expected to be $O(\sqrt{(r-1)/d})$ by the birthday paradox (This expansion can be similarly applied to Step 2, however, the details are omitted in this paper).

3.1.2 Random-walk Function

In order to reduce the search space by the birthday paradox, elements must be randomly chosen. Consequently, a random-walk function is introduced in Refs. 6), 7). Requirements of the random-walk function for the kangaroo method are as follows: an element is chosen pseudo-randomly, and an element can be calculated using information from previously chosen elements. Additionally, an assumption is required in Cheon's algorithm, that is to describe a chosen element as $\zeta^x G$. The following random-walk function F_s: $\mathbb{G} \to \mathbb{G}$ satisfies these requirements: For $s \mid (r-1)$, we define

$$F_s(G_0) = \zeta_{s'}^{f_s(G_0)} G_0,\ s' = \frac{r-1}{s}\ , \zeta_{s'} = \zeta^{s'},$$

where $f_s : \mathbb{G} \to \mathbb{Z}/s\mathbb{Z}$ is a pseudo-random function. Given an element $F_s^{(0)}(G_0) \in \mathbb{G}$, set

$$F_s^{(i)}(G_0) = \zeta_{s'}^{a} F_s^{(0)}(G_0),\ a = \sum_{t=0}^{i-1} f_s(F_s^{(t)}(G_0)) \qquad (1)$$

for $i \ge 1$.

3.1.3 Distinguished Points Technique

In the kangaroo method, all the subsequent pairs of elements collide after the collision once happens. By using this property, the number of stored data can be considerably reduced. This technique is called the distinguished points technique [11),21)].

When a chosen element has a specified characteristic (e.g., the least significant 6 bits are all zero), the element is stored as a distinguished point. Even if the first pair of points to collide is not stored, subsequent collided pair of distinguished points are stored. Therefore, collided pairs can be searched with reduced data.

By using this technique, the space complexity (also the number of elements) can be reduced by a factor of $1/w$ with arbitrary parameter w. For example, the space complexity is $O(\sqrt{(r-1)/d}/w)$ in Step 1. On the other hand, the time complexity is increased to $O(\log r\sqrt{(r-1)/d} + w)$ (or $O(\sqrt{(r-1)/d} + w)$ with the KKM method). If we choose w to be much smaller than $\sqrt{(r-1)/d}$, then the increase of the time complexity is negligible in general.

3.2 Cheon's Algorithm with the Kangaroo Method

With these techniques, the kangaroo method can be combined with Cheon's algorithm (see **Algorithm 2**). For implementing Cheon's algorithm with the kangaroo method, further improvements are required. Since Algorithm 2 uses the kangaroo method, a collision against own sequence of elements might occur, which we call a 'self-collision'. For example, $F_{d'}^{(i_1)}(G_d) = F_{d'}^{(j_1)}(G_d)$. Once a self-collision occurs, no new points can be subsequently generated by the random-walk function. Therefore, Algorithm 2 should be modified in the following way: Initial point $F_s^{(0)}(G_0)$ of the random-walk function is changed from G_0 to $\zeta_{s'}^{c} G_0$ for a constant value c [7)]. When a self-collision occurs, by changing c to a new value, a

Algorithm 2. Cheon's algorithm with the kangaroo method.

```
Input: $G,\ G_1 = \alpha G,\ G_d = \alpha^d G \in \mathbb{G}$
Output: $\alpha \in \mathbb{Z}/r\mathbb{Z}$
 1. Find a generator $\zeta \in (\mathbb{Z}/r\mathbb{Z})^*$
 2. $\zeta_d \leftarrow \zeta^d,\ \ \zeta_{d'} \leftarrow \zeta^{d'},\ \ d' \leftarrow (r-1)/d$
 3. [Step 1] $F_{d'}^{(0)}(G_d) \leftarrow G_d, F_{d'}^{(0)}(G) \leftarrow G$
 4.          Find $0 \le i_1, j_1$ such that $F_{d'}^{(i_1)}(G_d) = F_{d'}^{(j_1)}(G)$
 5.          $u_1 \leftarrow \sum_{i=0}^{i_1} f_{d'}(F_{d'}^{(i)}(G_d))$ mod $d'$
             $v_1 \leftarrow \sum_{j=0}^{j_1} f_{d'}(F_{d'}^{(j)}(G))$ mod $d'$
 6.          $k_1 \leftarrow v_1 - u_1$ mod $d'$
 7. [Step 2] $G' \leftarrow \zeta^{k_1} G,\ F_d^{(0)}(G_1) \leftarrow G_1,\ F_d^{(0)}(G') \leftarrow G'$
 8.          Find $0 \le i_2, j_2$ such that $F_d^{(i_2)}(G_1) = F_d^{(j_2)}(G')$
 9.          $u_2 \leftarrow \sum_{i=0}^{i_2} f_d(F_d^{(i)}(G_1))$ mod $d$
             $v_2 \leftarrow \sum_{j=0}^{j_2} f_d(F_d^{(j)}(G'))$ mod $d$
10.          $k_2 \leftarrow v_2 - u_2$ mod $d$
11. Output $\alpha = \zeta^{k_1 + k_2 d'}$
```

Algorithm 2'. Improved Cheon's algorithm with the kangaroo method.

```
Input: $G,\ G_1 = \alpha G,\ G_d = \alpha^d G \in \mathbb{G}$
Output: $\alpha \in \mathbb{Z}/r\mathbb{Z}$
 1. Find a generator $\zeta \in (\mathbb{Z}/r\mathbb{Z})^*$
 2. $\zeta_d \leftarrow \zeta^d,\ \ \zeta_{d'} \leftarrow \zeta^{d'},\ \ d' \leftarrow (r-1)/d$
 3. [Step 1] $\sigma_{d',G_d}^{(0)} \leftarrow c_{1,1},\ \ \sigma_{d',G}^{(0)} \leftarrow c_{1,2}$ ($c_{1,1}, c_{1,2}$ are constant values)
 4.          Find $0 \le i_1, j_1$ such that $F_{d'}^{(i_1)}(G_d) = F_{d'}^{(j_1)}(G)$
 5.          $u_1 \leftarrow \sigma_{d',G_d}^{(i_1)},\ \ \ v_1 \leftarrow \sigma_{d',G}^{(j_1)}$
 6.          $k_1 \leftarrow v_1 - u_1$ mod $d'$
 7. [Step 2] $\sigma_{d,G_1}^{(0)} \leftarrow c_{2,1},\ \ \sigma_{d,G}^{(0)} \leftarrow c_{2,2}$ ($c_{2,1}, c_{2,2}$ are constant values)
 8.          Find $0 \le i_2, j_2$ such that $F_d^{(i_2)}(G_1) = \zeta^{k_1} F_d^{(j_2)}(G)$
 9.          $u_2 \leftarrow \sigma_{d,G_1}^{(i_2)},\ \ \ v_2 \leftarrow \sigma_{d,G}^{(j_2)}$
10.          $k_2 \leftarrow v_2 - u_2$ mod $d$
11. Output $\alpha = \zeta^{k_1 + k_2 d'}$
```

new sequence of elements is generated by the same random-walk function.

In addition, we modified the Eq. (1) as follows:

$$\sigma_{s,G_0}^{(i)} = \sigma_{s,G_0}^{(i-1)} + f_s(F_s^{(i-1)}(G_0)) \bmod s,$$
$$F_s^{(i)}(G_0) = \zeta_{s'}^{\sigma_{s,G_0}^{(i)}} G_0. \tag{2}$$

In the Eq. (1), an element $F_s^{(i-1)}(G_0)$ is scalar multiplied to calculate $F_s^{(i)}(G_0)$. In this case, the element is changed at each scalar multiplication. Therefore, it is difficult to apply the KKM method to Algorithm 2. In the Eq. (2), the element based on the scalar multiplication is not changed, and the KKM method can be easily applied. The improved Cheon's algorithm with the kangaroo method is shown in **Algorithm 2'**.

4. Experimental Results

In this section, we report our experimental results of the improved Cheon's algorithm with the kangaroo method (see Algorithm 2'). Note that we did not use the KKM method in our experiment.

4.1 Parameters

An additive group $\mathbb{G}$ is a subgroup of the Mordell-Weil group on a supersingular elliptic curve defined by

$$E/GF(3^{127}) : y^2 = x^3 - x - 1,$$

which is introduced in Ref. 2) and used for efficient pairing computation in practice. The followings are some parameters related to E:

- $\sharp E(\mathrm{GF}(3^{127})) = 25312879 \cdot 41757061638619 \cdot 399164334498031 \cdot r$ (202-bit),
- $r = 93148560049862239626013 99$ (83-bit prime),
- $r - 1 = 2 \cdot 3 \cdot 127 \cdot 2251 \cdot 5431 \cdot 7485427 \cdot 133582417$,
- $d = 2176458320181 = 3 \cdot 5431 \cdot 133582417$ (41-bit),
- $\zeta = 12$ (a minimum generator of $(\mathbb{Z}/r\mathbb{Z})^*$).

Here $\sharp E(\mathrm{GF}(3^{127}))$ denotes the number of the $\mathrm{GF}(3^{127})$-rational points on the elliptic curve E, and d is chosen optimal so as to minimize the time complexity of Cheon's algorithm. Note that these parameters are the same as in Refs. 12), 13).

In the computation, an element $z \in GF(3^{127}) = GF(3)[t]/(t^{127} + t^8 - 1)$ is represented as a polynomial $z[126]t^{126} + \cdots + z[1]t + z[0]$ ($z[k] \in \{0, \pm 1\}$). In addition, it is stored as a pair of 127-bit sequences

$$z^+ = (z^+[126], \ldots, z^+[1], z^+[0])$$
$$z^- = (z^-[126], \ldots, z^-[1], z^-[0])$$

where $z^+[k]$ and $z^-[k]$ are related to $z[k]$: $(z^+[k], z^-[k]) = (1, 0)$ if $z[k] = 1$, $(z^+[k], z^-[k]) = (0, 1)$ if $z[k] = -1$ and $(z^+[k], z^-[k]) = (0, 0)$ if $z[k] = 0$. The case $(z^+[k], z^-[k]) = (1, 1)$ never occurs.

Moreover, we generated a base-point G in the following manner, exactly the same as Refs. 12), 13). First, we generate a seed-point S such that all $S.x^+[k]$ are 1 and all $S.x^-[k]$ are 0, where $S.x$ is the x-coordinate value of the point S. The y-coordinate value $S.y$ is obtained by $S.x$ and the curve equation. Then we have

$S.x^+ =$ `0x7FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF`,
$S.x^- =$ `0x00000000000000000000000000000000`,
$S.y^+ =$ `0x5404923A08C020F08891241808610860`,
$S.y^- =$ `0x22C2008457299A0025628A60210EB71C`.

Since the order of S equals to the order of the curve itself, a base-point G is obtained by $G = \#E/rS$. Then we have

$G.x^+ =$ `0x0261508200007930A0412800212208D8`,
$G.x^- =$ `0x081C8970DF7A0044149643D502CD9401`,
$G.y^+ =$ `0x0BA754D97DEC657029806214815158B4`,
$G.y^- =$ `0x24500826020108020025056924AEA20B`.

Finally, the target points $G_1 = \alpha G$ and $G_d = \alpha^d G$ are computed for $\alpha = 3$:

$G_1.x^+ =$ `0x68C314812A8103C102C48AAB02231B8A`,
$G_1.x^- =$ `0x0320A82A9032CC0438284100C504E435`,
$G_1.y^+ =$ `0x532856AA1C8C801140302268E2188786`,
$G_1.y^- =$ `0x2C01A100614346823EC6410210C05040`,
$G_d.x^+ =$ `0x23154CC0519344D10C1D130C18200104`,
$G_d.x^- =$ `0x506AA00D0A6C802660800CB2A301B449`,
$G_d.y^+ =$ `0x02809B34240C8282A0244A7401070C04`,
$G_d.y^- =$ `0x356720C2585250784B43150ADCA0C301`.

Here, $\alpha = 3$ is a solution which is the same as in Refs. 12), 13).

4.2 Our Results

In Step 1 of Algorithm 2', we establish the following two tables:

$$\mathrm{T}_{1,\mathrm{L}} = \left\{ \left(\sigma^{(i)}_{d',G_d},\ F^{(i)}_{d'}(G_d) \right) \right\}$$

and

$$\mathrm{T}_{1,\mathrm{R}} = \left\{ \left(\sigma^{(j)}_{d',G},\ F^{(j)}_{d'}(G) \right) \right\}.$$

Here, $\sigma^{(i)}_{d,G_d}$, $\sigma^{(j)}_{d,G}$ are 42-bit index values of chosen points. These values are calculated from previously chosen points.

In a similar way of Step 1, we establish the following two tables in Step 2:

$$\mathrm{T}_{2,\mathrm{L}} = \left\{ \left(\sigma^{(i)}_{d,G_1},\ F^{(i)}_{d}(G_1) \right) \right\}$$

and

$$\mathrm{T}_{2,\mathrm{R}} = \left\{ \left(\sigma^{(j)}_{d,G},\ F^{(j)}_{d}(G) \right) \right\}.$$

In Cheon's algorithm with the kangaroo method, the total search space is too large to calculate all points. Thus, to establish two tables and to compare them, they have to be concurrently created. Additionally, with the distinguished points technique, we store only the points where the least significant 6-trit (ternary-digit) of the x-axis values are all zero ($w = 3^6$). To save space, each $F^{(i)}_s(G_0)$ is digested as $\mathrm{LSB}_{64}(\mathrm{MD5}(F^{(i)}_s(G_0)))$, and these digested values are stored, which is the same technique as Ref. 12).

In this instance, we establish the two tables with 2 core of 3 GHz Core2Quad, and compared these tables by using another core. For 3 hours, about 1,800 points (28 KByte) were stored in every two tables in Step 1, and collided points are recorded. At this time, $u_1 \leftarrow 1748512041946$, $v_1 \leftarrow 2439349585203$, and $k_1 \leftarrow 2439349585203 - 1748512041946 = 690837543257$. Since $\alpha^d = \zeta_d^{k_1} = 2116715807584984875228271$, this experiment has solved Step 1.

And for 2.75 hours, about 1,600 points (26 KByte) were stored in every two tables in Step 2. At this time, $u_2 \leftarrow 1559364007743$, $v_2 \leftarrow 613947073386$, and $k_2 \leftarrow 613947073386 - 1559364007743 = 1231041385824 \pmod{2176458320181}$.

Finally,

$k \leftarrow k_1 + k_2 \cdot d' = 5268639026442339275434649$,

and

$\alpha' \leftarrow \zeta^k \pmod{r} = 3$.

Since $\alpha' = \alpha$, this experiment has solved Step 2. Therefore, it is confirmed that Cheon's algorithm with the kangaroo method works correctly.

Note that, $1800 \times 3^6 \simeq 2^{20.3}$ and $1600 \times 3^6 \simeq 2^{20.2}$ points were computed for each table in practice respectively. And, in order to search i_1, j_1 such that

$F_{d'}^{(i_1)}(G_d) = F_{d'}^{(j_1)}(G)$ and i_2, j_2 such that $F_d^{(i_2)}(G_1) = F_d^{(j_2)}(G)$, a naive method is used for comparing the two tables. Since the tables are small, the time for table-comparison is negligible. Then, it is necessary to calculate $F_s^{(i)}(G_0)$ from $F_s^{(i-1)}(G_0)$ for 8.2 msec on 1 core 3 GHz Core2Quad in our experiment.

5. Comparison

In this section, we compare Cheon's algorithm with the BSGS method and that with the kangaroo method.

5.1 Previous works

In Refs. 12), 13), Izu, et al. showed their experimental results of Cheon's algorithm with the BSGS method. Note that the curve used in Refs. 12), 13) is same as the one in this paper. We briefly summarize their results [⋆1]: Their experiment in Ref. 13) was conducted on 1 core of 3 GHz Core2Quad using a straightforward implementation.

- In Step 1, total 34 MByte table-making for 7 hours.
- In Step 2, total 23 MByte table-making for 5 hours.

In their experiment, hash values of points are stored in these tables to reduce the total size of the tables. For table-comparison, 1 hour in each step was required.

5.2 The BSGS Method vs. the Kangaroo Method

In **Table 1**, we summarize our experimental results of Cheon's algorithm with the BSGS method and the kangaroo method.

As mentioned above, the required space of Cheon's algorithm with the kangaroo method is more efficient than that with the BSGS method. Additionally, Table 1 shows that the required time of Cheon's algorithm with the kangaroo method is also efficient. The reasons for this are as follows:

- In Ref. 12), it is shown that all points are calculated when using the BSGS method.
- In our experiment, when points collide in both tables, the calculation is stopped at once.
- In our experiment, collided points appeare as expected.

Therefore, it is concluded that the time complexity of these two methods are almost the same.

Table 1 The complexity of Cheon's algorithm with both methods.

Subroutine algorithm	Step	Time	Space	Calculating points
The BSGS method [12)]	1	8 hours	32,325 KByte	$2^{22.0}$
	2	6 hours	23,051 KByte	$2^{21.5}$
The kangaroo method	1	6 hours	56 KByte	$2^{21.3}$
	2	5.5 hours	52 KByte	$2^{21.2}$

⋆1 In Ref. 12), they showed their experimental results only on Step 1 of Cheon's algorithm with the BSGS method. Moreover, experimental results on Step 1 and Step 2 were shown in Ref. 13). In this paper, we used the results in Ref. 13).

6. Speeding-up Techniques

In this section, we describe some speeding-up techniques for Cheon's algorithm with the kangaroo method. Note that these techniques were not used in our experiment. As mentioned in Section 2.3, the KKM method can reduce the time complexity by a factor of $1/\log r$. Besides the KKM method, we introduce the following two techniques:

6.1 Using Automorphisms

Let E be an elliptic curve defined over GF(3). For a positive integer m, set $\mathbb{G} = E(\mathrm{GF}(3^m))$. The group $\mathbb{G}$ has the Negation map and the Frobenius map as fast computable automorphisms. Note that the order of the Negation map (resp. the Frobenius map) is equal to 2 (resp. m). Let ϕ be the Negation map or Frobenius map, and let n denote the order of ϕ. We denote by $\mathbb{G}/\sim_\phi$ the set of equivalence classes of $\mathbb{G}$ defined by ϕ (see Ref. 11) [Chapter 4, pp.161–163] for details). Fix a random-walk function F on $\mathbb{G}$ for the kangaroo method. If the function F is well defined on $\mathbb{G}/\sim_\phi$, the kangaroo method can be sped-up by a factor of $\sqrt{n}$ (see Refs. 9), 11)). Hence, using both the Negation map and the Frobenius map, the kangaroo method can be sped-up by a factor of $\sqrt{2m}$.

With respect to Cheon's algorithm with the kangaroo method, the algorithm is composed of two steps and each step is based on the kangaroo method. Since our random-walk function is of the form

$$F_s(G) = \zeta_{s'}^a G$$

with randomly chosen $a \in \mathbb{Z}/s\mathbb{Z}$ (see Section 3.1 for details), our function is well-defined on $\mathbb{G}/\sim_\phi$ if the condition $n \mid s$ is satisfied. Since $n \mid (r-1) = s \cdot s'$, we see that the automorphisms technique can be applied at least in either of two steps of the algorithm. For a random-walk function F of additive type such as Teske's adding walk [22] or the function proposed by Ref. 9), the function F on $\mathbb{G}/\sim_\phi$ can fall into short cycles, which are called "fruitless cycles", and hence the optimal speed-up cannot be expected in general [8),9)]. However, since our random-walk function is of multiplicative type as above, our function on $\mathbb{G}/\sim_\phi$ rarely falls into fruitless cycles. Therefore, using both the Negation map and the Frobenius map, the time complexity of the algorithm can be reduced by a factor of about $\sqrt[4]{2m}$ in the case $d \approx \sqrt{r}$.

6.2 Parallel Computing

In order to parallelize Cheon's algorithm, plural computers (cores) have to compute points in the group $\mathbb{G}$ independently. As mentioned in Section 3.2, constant values $c_{1,1}, c_{1,2}$ in Step 1 and $c_{2,1}, c_{2,2}$ in Step 2 are used in Algorithm 2' to change initial points for the random-walk function. By changing the initial points, the points calculated by the random-walk function constitute independent sequence. Therefore, Algorithm 2' can be parallelized by controlling these constant values. With parallel computing, the time complexity can be reduced to $1/M$ with M computers.

Note: with parallel computing, two computers occasionally calculate collided point in a table ($\mathrm{T}_{1,L}$, $\mathrm{T}_{1,R}$, $\mathrm{T}_{2,L}$, or $\mathrm{T}_{2,R}$). At that time, the collision is detected and one of these computer is re-initialize with new constant values using the same procedure for self-collisions (see Section 3.2).

7. Cost Estimations

In this section, we estimate the required cost of Cheon's algorithm to solve DLPwAI on some pairing-friendly curves. Here we consider pairing-friendly elliptic curves on $\mathrm{GF}(3^m)$ defined by

$$E_b : y^2 = x^3 - x + b, \quad b \in [-1, 1].$$

Since the equation of E_b is defined over GF(3), we can use the automorphisms technique (see Section 6.1).

7.1 Equation for the Computation Cost

Since the kangaroo method has an advantage in the space complexity, we can solve DLPwAI with large size by using Cheon's algorithm with the kangaroo method (see Section 5). Here we estimate the required cost of Cheon's algorithm with the kangaroo method (We do not consider parallel computing here). Based on our experimental results, we assume the followings:

- Parameter $d \approx \sqrt{r}$ (We focus on the optimal case for Cheon's algorithm).
- Since the cost of a scalar multiplication on $\mathrm{GF}(3^{127})$ using a straightforward implementation is 8.2 msec (see Section 4.2), we assume that the cost of a scalar multiplication on $\mathrm{GF}(3^m)$ is equal to $8.2 \times (m/127)^2$ msec. Moreover, the cost can be reduced by a factor of $1/\log r$ using the KKM method.
- We assume that the cost of Cheon's algorithm with the kangaroo method is 2.8 times of that with the BSGS-method (see Ref. 13)[Section 3.5] for details). Moreover, the cost can be reduced by a factor of $\sqrt[4]{2m}$ by using the automorphisms technique.

Therefore, the equation for the computation cost is assumed to be as follows:

$$\mathrm{Cost} = 2.8 \times 4 \times \sqrt[4]{r} \times 8.2/\log r \times (m/127)^2/\sqrt[4]{2m} \ \text{msec}.$$

7.2 Cost Prediction

In this subsection, we show the required cost of Cheon's algorithm with the kangaroo method to solve DLPwAI on some curves.

(1) The case $m = 89, b = 1$: Ref. 10)

- $\sharp E(\mathrm{GF}(3^m)) = 7 \cdot 1069 \cdot 2137 \cdot r$ (142-bit),
- $r = 181932967220635112252099081035759771$ (118-bit),
- $r - 1 = 2 \cdot 3 \cdot 5 \cdot 7 \cdot 89 \cdot 8087 \cdot 4139171 \cdot 130108711 \cdot 2235089484239$.

$$\begin{aligned}\mathrm{Cost} &= 2.8 \times 4 \times \sqrt[4]{r} \times 8.2/118 \times (89/127)^2/\sqrt[4]{178} \ \text{msec} \\ &\simeq 22 \ \text{hours}.\end{aligned}$$

(2) The case $m = 97, b = 1$: Refs. 1), 2), 10), 15)

- $\sharp E(\mathrm{GF}(3^m)) = 7 \cdot r$ (154-bit),
- $r = 2726865189058261010774960798134976187171462721$ (151-bit),
- $r{-}1 = 26{\cdot}3{\cdot}5{\cdot}13{\cdot}17{\cdot}41{\cdot}43{\cdot}73{\cdot}97{\cdot}193{\cdot}577{\cdot}739{\cdot}769{\cdot}6481{\cdot}59632043{\cdot}42094721833$.

$$\begin{aligned}\mathrm{Cost} &= 2.8 \times 4 \times \sqrt[4]{r} \times 8.2/151 \times (97/127)^2/\sqrt[4]{194} \ \text{msec} \\ &\simeq 254 \ \text{days}.\end{aligned}$$

(3) The case $m = 103, b = 1$: Ref. 2)

- $\sharp E(\mathrm{GF}(3^m)) = 7 \cdot 524683 \cdot r$ (164-bit),
- $r = 3788734765226760304517052348776005195050727$ (142-bit),

Table 2 Cost prediction of Cheon's algorithm to solve DLPwAI in $d \approx \sqrt{r}$.

	m	b	$\sharp E$	r	Cost	Cost without speed-up*
(1)	89	1	142-bit	118-bit	22 hours	395 days
(2)	97	1	154-bit	151-bit	254 days	392 years
(3)	103	1	164-bit	142-bit	63 days	92 years
(4)	127	−1	202-bit	83-bit	8 minutes	44 hours**

* We list the cost prediction of our implementation.
** In our experiment, it required 11.5 hours (see Table 1 for details).

- $r - 1 = 2 \cdot 3 \cdot 7 \cdot 17 \cdot 97 \cdot 103 \cdot 696239 \cdot \cdot 12353064397 \cdot 61752419775112892603$,

$$\begin{aligned}\text{Cost} &= 2.8 \times 4 \times \sqrt[4]{r} \times 8.2/142 \times (103/127)^2/\sqrt[4]{206} \ \text{ msec} \\ &\simeq 63 \ \text{ days}.\end{aligned}$$

(4) The case $m = 127, b = -1$: Ref. 2) (same parameter as our experiment)

- $\sharp E(\mathrm{GF}(3^m)) = 25312879 \cdot 41757061638619 \cdot 399164334498031 \cdot r$ (202-bit),
- $r = 93148560049862239626013 99$ (83-bit),
- $r - 1 = 2 \cdot 3 \cdot 127 \cdot 2251 \cdot 5431 \cdot 7485427 \cdot 133582417$,

$$\begin{aligned}\text{Cost} &= 2.8 \times 4 \times \sqrt[4]{r} \times 8.2/83 \times (127/127)^2/\sqrt[4]{254} \ \text{ msec} \\ &\simeq 8 \ \text{ minutes}.\end{aligned}$$

In **Table 2**, we summarize the above results. The costs are predicted using a single-core CPU environment. As described in Section 6.2, Cheon's algorithm with the kangaroo method can be applied for parallel computing. For example, when 100 cores of CPU are used, the costs can be reduced by a factor of 100. Thus, when the optimal parameter d is chosen, DLPwAI on these pairing-friendly curves can be solved by Cheon's algorithm at reasonable cost.

8. Concluding Remarks

In this paper, we reported our experimental results of Cheon's algorithm with the kangaroo method on a pairing-friendly curves defined over $\mathrm{GF}(3^{127})$. Moreover, based on our experimental results, we estimated the required cost of Cheon's algorithm to solve DLPwAI on some pairing-friendly curves defined over a finite field of characteristic 3. Our estimation showed that DLPwAI on a part of pairing-friendly curves can be solved at reasonable cost when the optimal parameter d is chosen (see Table 2 for details). If we implement the pairing-based cryptographic schemes based on the problems related with DLPwAI such as the ℓ-WDH problem, we should avoid using such weak parameters.

Here we only considered the case where the parameter d is optimal for Cheon's algorithm. Hence our future work is to estimate the required cost of Cheon's algorithm with d as a parameter.

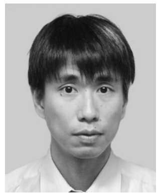
References

1) Baretto, P., Kim, H., Lynn, B. and Scott, M.: Efficient Algorithms for Pairing-based Cryptosystems, *CRYPTO 2002*, LNCS 2442, pp.354–368 (2002).
2) Beuchat, J., Brisebarre, N., Detrey, J., Okamoto, E. and Rodríguez-Henríquez, F.: Comparison between Hardware Accelerators for the Modified Tate Pairing over $\mathbb{F}_{2^m}$ and $\mathbb{F}_{3^m}$, *Pairing 2008*, LNCS 5209, pp.297–315 (2008).
3) Boneh, D. and Boyen, X.: Short Signatures without Random Oracles, *EUROCRYPT 2004*, LNCS 3027, pp.56–73 (2004).
4) Boneh, D. and Boyen, X.: Efficient Selective-ID Secure Identity-based Encryption without Random Oracles, *CRYPTO 2004*, LNCS 3152, pp.223–238 (2004).
5) Boneh, D., Boyen, X. and Goh, E.: Hierarchical Identity-based Encryption with Constant Size Ciphertext, *EUROCRYPT 2005*, LNCS 3494, pp.440–456 (2005).
6) Cheon, J.H.: Security Analysis of Strong Diffie-Hellman Problem, *EUROCRYPT 2006*, LNCS 4004, pp.1–11 (2006).
7) Cheon, J.H.: Discrete Logarithm Problems with Auxiliary Inputs, *Journal of Cryptology*, Vol.23, No.3, pp.457–476 (2010).
8) Galbraith, S.D. and Ruprail, R.S.: Using Equivalence Classes to Accelerate Solving the Discrete Logarithm Problem in a Short Interval, *PKC 2010*, LNCS 6056, pp.368–386 (2009).
9) Gallant, R., Lambert, R. and Vanstone, S.: Improving the Parallelized Pollard Lambda Seach on Binary Anomalous Curves, *Math. Comp.*, Vol.69, pp.1699–1705 (2000).
10) Grabher, P. and Page, D.: Hardware Acceleration of the Tate Pairing in Characteristic Three, *CHES 2005*, LNCS 3659, pp.398–411 (2005).
11) Hankerson, D., Menezes, A. and Vanstone, S.: *Guide to Elliptic Curve Cryptography*, Springer Professional Computing (2004).
12) Izu, T., Takenaka, M. and Yasuda, M.: Experimental Results on Cheon's Algorithm, *ARES 2010*, pp.625–628, IEEE Computer Society (2010).
13) Izu, T., Takenaka, M. and Yasuda, M.: Experimental Results on Cheon's Algorithm, *SCIS 2010*, Pre-proceedings, 3A3-5 (2010).
14) Jao, D. and Yoshida, K.: Boneh-Boyen Signatures and the Strong Diffie-Hellman Problem, *Pairing 2009*, LNCS 5671, pp.1–16, Springer (2009).

15) Kerins, T., Marnane, W., Popovici, E. and Barreto, P.: Efficient Hardware for the Tate Pairing Calculation in Characteristic Three, *CHES 2005*, LNCS 3659, pp.412–426 (2005).
16) Kozaki, S., Kutsuma, T. and Matsuo, K.: Remarks on Cheon's Algorithms for Pairing-related Problems, *Pairing 2007*, LNCS 4575, pp.302–316 (2007).
17) Mitsunari, S., Sakai, R. and Kasahara, M.: A New Traitor Tracing, *IEICE Trans. Fund.*, Vol.E85-A, No.2, pp.481–484 (2002).
18) Pollard, J.: Monte Carlo Methods for Index Computation (mod p), *Math. Comp.*, Vol.32, pp.918–924 (1978).
19) Sakemi, Y., Izu, T., Takenaka, M. and Yasuda, M.: Solving DLP with Auxiliary Input over an Elliptic Curve Used in TinyTate Library, to appear in *Proc. WISTP 2011*, Springer (2011).
20) Shanks, D.: Class Number, a Theory of Factorization, and Genera, *Proc. Symp. Math. Soc.*, Vol.20, pp.415–440 (1971).
21) Teske, E.: Speeding up Pollard's Rho Method for Computing Discrete Logarithms, *ANTS III*, LNCS 1423, pp.541–554 (1998).
22) Teske, E.: On random walks for Pollard's rho method, *Math. Comp.*, Vol.70, pp.809–825 (2001).

(Received November 30, 2010)
(Accepted June 3, 2011)
(Released September 7, 2011)

Tetsuya Izu received his B.S. and M.S. degrees in mathematics from the University of Tokyo in 1992, and from Rikkyo University in 1994, respectively. He received his Ph.D. in engineering from the University of Electro-Communications in 2007. He has been engaged in research on cryptography and information security at FUJITSU LABORATORIES Ltd. and FUJITSU Ltd. since 1997. He was a visiting researcher at the University of Waterloo, Canada, in 2001. He received the Paper Prizes of the Symposium on Cryptography and Information Security (SCIS) in 1999 and the Computer Security Symposium (CSS) in 2002. He was awarded the Young Scientists' Prize by the Minister of Education, Culture, Sports, Science and Technology Research on side channel attacks and countermeasures in information security in 2007. He was awarded the IPSJ Kiyasu Special Industrial Achievement Award on the development of the dedicated integer factoring devices in 2008. He is a member of IACR, IEICE and IPSJ.



Masahiko Takenaka received his B.E. and M.E. degrees in electronic engineering in 1990, 1992 respectively from Osaka University, Osaka, Japan. He received his Ph.D. in engineering from University of Tsukuba in 2009. Since 1992, He has been engaged in research and development on cryptography and information security at FUJITSU LABORATORIES LTD. He is currently a senior researcher. He was awarded the CSS Paper Prize in 2002, and the OHM Technology Award in 2005. He is a member of IEICE.

Masaya Yasuda received his B.S. degree in mathematics from Kyoto University in 2002. He received his M.S. degree and Ph.D. in mathematical sciences from the University of Tokyo in 2004 and 2007, respectively. He has been engaged in research on cryptography at FUJITSU LABORATORIES Ltd. and FUJITSU Ltd. since 2007. His recent interest includes elliptic curve cryptography and homomorphic encryption schemes.