A New Approach to Discrete Logarithm Problem with Auxiliary Inputs

Taechan Kim and Jung Hee Cheon

Department of Mathematics, Seoul National University, Korea

Abstract. Embedding an element of a finite field into auxiliary groups such as elliptic curve groups or extension fields of finite fields has been useful tool for analysis of cryptographic problems such as establishing the equivalence between the discrete logarithm problem and Diffie-Hellman problem or solving the discrete logarithm problem with auxiliary inputs (DLPwAI). Actually, Cheon's algorithm solving the DLPwAI can be regarded as a quantitative version of den Boer and Maurer. Recently, Kim showed in his dissertation that the generalization of Cheon's algorithm using embedding technique including Satoh's [19] is no faster than Pollard's rho algorithm when $d \nmid (p \pm 1)$.

In this paper, we propose a new approach to solve DLPwAI concentrating on the behavior of function mapping between the finite fields rather than using an embedding to auxiliary groups. This result shows the relation between the complexity of the algorithm and the number of absolutely irreducible factors of the substitution polynomials, hence enlightens the research on the substitution polynomials.

More precisely, with a polynomial $f(x)$ of degree d over $\mathbf{F}_p$, the proposed algorithm shows the complexity $O\left(\sqrt{p^2/R}\log^2 d \log p\right)$ group operations to recover α with $g, g^{\alpha}, \ldots, g^{\alpha^d}$, where R denotes the number of pairs $(x, y) \in \mathbf{F}_p \times \mathbf{F}_p$ such that $f(x) - f(y) = 0$. As an example using the Dickson polynomial, we reveal α in $O(p^{1/3}\log^2 d \log p)$ group operations when $d|(p+1)$. Note that Cheon's algorithm requires $g, g^{\alpha}, \ldots, g^{\alpha^d}, \ldots, g^{\alpha^{2d}}$ as an instance for the same problem.

Keywords: discrete logarithm problem, Cheon's algorithm, fast multipoint evaluation, Dickson polynomial, substitution polynomial, point counting, absolutely irreducible polynomial

1 Introduction

Let g be a group element of order p. The Discrete Logarithm Problem (DLP) is a problem to find $\alpha \in \mathbf{F}_p$ when the elements g, g^{α} are given. The DLP is considered as a cryptographic hard problem and guarantees the security of many cryptographic schemes such as encryptions, signatures, and key exchange protocols.

In a recent decade, many variants of DLP such as the Bilinear Diffie-Hellman Problem (BDHP) [3], the ℓ-Strong Diffie-Hellman Problem (ℓ-SDHP) [2], the Bilinear Diffie-Hellman Exponent Problem [4], and the Bilinear Diffie-Hellman Inverse Problem [1] are used to assure the security of many cryptographic schemes such as the ID-based encryption (IBE) [1,3], the short signatures [2] and so on. However the hardness of these variants has not been well-understood. In Eurocrypt 2006, Cheon [7,8] introduced a general version of these problems, so called Discrete Logarithm Problem with Auxiliary Inputs (DLPwAI), to compute a discrete logarithm α when the auxiliary inputs $g, g^{\alpha}, \ldots, g^{\alpha^d}$ are given, and proposed an algorithm to solve this problem more efficiently when d divides either $p-1$ or $p+1$. The $p-1$ case has been dealt with independently in [5].

The idea of Cheon's algorithm is to embed a discrete logarithm $\alpha \in \mathbf{F}_p$ into a finite field $\mathbf{F}_p$ (or $\mathbf{F}_{p^2}$, respectively) for $p-1$ case (or $p+1$ case, respectively). By this algorithm, one can recover the discrete logarithm α in time complexity $O(\sqrt{\frac{p-1}{d}}\log p)$ with the auxiliary inputs $g, g^{\alpha}, g^{\alpha^d}$ when $d < p^{1/2}$ divides $p-1$ and recover α in time $O(\sqrt{\frac{p+1}{d}}\log p)$ with the inputs $g, g^{\alpha}, \ldots, g^{\alpha^d}, \ldots, g^{\alpha^{2d}}$ when $d < p^{1/3}$ divides $p+1$. In fact, Cheon's algorithm can be regarded as a quantitative version of the famous reduction algorithms from the DL problem to the DH problem [10,15,18], and so we may

expect to use the embeddings to other auxiliary groups such as extension fields or elliptic curves in this problem. Satoh [19] generalized Cheon's algorithm when d divides $\varphi_k(p)$ for $k \geq 3$ using an embedding $\mathbf{F}_p$ into $GL(n, \mathbf{F}_p)$, where $\varphi_k(x)$ denotes the k-th cyclotomic polynomial. At last, Kim [13] realized that Satoh's algorithm essentially uses an embedding from $\mathbf{F}_p$ into $\mathbf{F}_{p^n}$ and proved that the algorithm can never be faster than the ordinary algorithm for DLP when $d|\varphi_k(p)$ for $k \geq 3$. However, it is very hard to believe that this DLPwAI is as secure as DLP except only the $p \pm 1$ cases. At this point, we need totally different approaches for this problem. We remark that any small progress in this area is very important because it can result in the attacks of numerous cryptographic schemes and protocols.

Our perspective is focused on the behavior of a function mapping from $\mathbf{F}_p$ to itself rather than an embedding from $\mathbf{F}_p$ into some other groups. We observed that if there exist many points (x, y) in $\mathbf{F}_p \times \mathbf{F}_p$ satisfying the equation $f(x) - f(y) = 0$ for rational functions (or polynomials) $f(x)$ over $\mathbf{F}_p$ then it can be used to obtain the discrete logarithm more efficiently. A bivariate polynomial $f^*(x, y) := f(x) - f(y) \in \mathbf{F}_p[x, y]$ is called the *substitution polynomial* and by Weil's bound the number of points of the curve $f^*(x, y) = 0$ over $\mathbf{F}_p$ is closely related to the number of absolutely irreducible factors, say r. An irreducible polynomial over $\mathbf{F}_p$ is called absolutely irreducible if it is irreducible over algebraic closure $\bar{\mathbf{F}}_p$. At last our algorithm reduces solving DLPwAI to find a polynomial whose substitution polynomial has many absolutely irreducible factors.

Let us describe our result more precisely. Let $f(x)$ be a polynomial of degree d over $\mathbf{F}_p$ and consider a mapping by the polynomial. Define R by the number of solutions in $\mathbf{F}_p \times \mathbf{F}_p$ of the equation $f(x) - f(y) = 0$ and let R_i be the number of images which are mapped by i elements from its domain, then $R = \sum_{i=1}^{d} i^2 R_i$. With the result on the non-uniform birthday problem in [11], for given $g, g^\alpha, \ldots, g^{\alpha^d}$, we can recover α in $O(m \log^2 d \log p)$ group operations where the expected value of m is $\sqrt{\frac{p^2\pi}{R}} + O(p^{1/4})$. Thus if R_d is large *i.e.*, many images are mapped by d elements then we can solve α efficiently. The polynomial $f(x) = x^d$ with $d|(p-1)$ gives a trivial example which has the value $R_d = \frac{p-1}{d}$ and in this case the complexity almost matches with the Cheon's $p-1$ algorithm. As a nontrivial example, the Dickson polynomial with $d|(p+1)$ has $R_1 = (p-1)/2$ and $R_d = (p+1)/2d$, thus our method finds α in $O(\sqrt{p\pi/d} \log^2 d \log p)$ group operations. This method takes only $g, g^\alpha, \ldots, g^{\alpha^d}$ as inputs as in Satoh's [19]. Note that Cheon's algorithm requires $g, g^\alpha, \ldots, g^{\alpha^d}, \ldots, g^{\alpha^{2d}}$ as an instance for the same problem.

As a crucial part for the algorithm, we require the fast multipoint evaluation of the polynomial in blackbox manner. The word "blackbox" means that the function to be evaluated is given by the encrypted coefficients. In other words if we have the values $g^{f_0}, \ldots, g^{f_d}$ where $f(x) = f_0 + f_1 x + \cdots + f_d x^d$, then we can compute the evaluated values of $f(x)$ at many points in the blackboxed form $g^{f(x_0)}, \ldots, g^{f(x_d)}$ efficiently (in time complexity quasi-linear in the degree of $f(x)$). In [17], Mohassel also proposed a similar algorithm when the primitive d-th root of unity exists and applied this to obtain the efficient protocol such as the oblivious polynomial evaluation (OPE) protocols and the private set intersection protocols. By simple observation using Schönhage-Straßen multiplication, we also indicate that the fast multiplication is possible when the primitive root of unity does not exist.

By Weil's theorem, the value $m = O(\sqrt{p^2/R})$ is equal to $O(\sqrt{p/r})$, and m is bounded by $O(\sqrt{p/d}) \leq m \leq O(\sqrt{p})$ since $1 \leq r \leq d$. Mit'kin [16] improved an estimates for the value R, which says that $R \leq (\lfloor d/2 \rfloor + 1)p + c(d)\sqrt{p}$ for some constant $c(n)$ in many cases. On the other hands, in [12] the complete lists of polynomials of degree $d < p^{1/4}$ whose substitution polynomial has at least $d/2$ absolutely irreducible factors were given. For our algorithm to work valuably, it is worthwhile to find a polynomial with large r and until the present there seems to be a room that such polynomials may exist.

Unlike the polynomial cases, for any d, we can obtain a rational function $f(x) = \frac{f_1(x)}{f_2(x)}$ of degree d with $R > \sqrt{d}p$ easily by using the $\sqrt{d}$-th division polynomial. However, the main drawback of using this rational function is that the fast evaluation of $g^{f(x_i)}$ for many x_i's seems hard to achieve.

2 Cheon's algorithm

In [7], Cheon proposed an algorithm to solve DLPwAI for some cases. The result is as follows:

Theorem 1. *Let $\mathbb{G} = \langle g \rangle$ be a cyclic group of prime order p. Then*

1. *Let d be a divisor of $p-1$. Given $g, g^{\alpha}, g^{\alpha^d}$, one can solve α in $O\left(\sqrt{\frac{p-1}{d}} + \sqrt{d}\right)$ group exponentiations.*
2. *Let d be a divisor of $p+1$. Given $g, g^{\alpha}, \ldots, g^{\alpha^d}, \ldots, g^{\alpha^{2d}}$, one can solve α in $O\left(\sqrt{\frac{p+1}{d}} + d\right)$ group exponentiations.*

This algorithm is obtained by observing that the exponent can be embedded into a subgroup of small order of auxiliary groups. More precisely, if d divides $(p-1)$ then $\alpha^d \in \mathbf{F}_p^{\times}$ is an element in a subgroup of order $\frac{p-1}{d}$. Then one can recover α^d and α in Baby Step Giant Step style in the subgroup of small order $\frac{p-1}{d}$. On the other hand, let d be a divisor of $(p+1)$. Since $\mathbf{F}_{p^2}$ has a subgroup H of order $p+1$ and H has subgroup of order $\frac{p+1}{d}$, one can recover α by representing an element in H by rational functions in terms of α and also by using the Baby Step Giant Step technique. In [19], Satoh proposed generalized Cheon's algorithm when d divides $\varphi_k(p)$, where $\varphi_k(\cdot)$ denotes the k-th cyclotomic polynomial. However the efficiency of the algorithm was not well-studied. Finally, Kim [13] showed that Satoh's algorithm essentially uses the embedding from $\mathbf{F}_p$ to $\mathbf{F}_{p^n}$ and this generalization is no faster than time complexity $\sqrt{p}$ when d divides $\varphi_k(p)$ for $k \geq 3$.

3 Non-uniform Birthday Problem

In [11], Galbraith and Holmes considered the sampling of coloured balls and placing them in p urns with non-uniform probability distributions depending on the colour of the ball. They also determined the expected number of trials until two different types of balls are put in the same urn. We describe them briefly in this section. We only consider two types of colours of balls.

Assumption 1 ([11]) Assume that there are two colours of balls. $r_{k,c}$ denotes the probability the k-th ball sampled has colour $c = 1, 2$. Let $q_c := \lim_{n\to\infty} n^{-1} \sum_{k=1}^{n} r_{k,c}$ exists and $q_1 \geq q_2 > 0$. Let $b_{n,c} = q_c - n^{-1} \sum_{k=1}^{n} r_{k,c}$. Assume that there exists a constant K such that $|b_{n,c}| \leq K/p$ for all c.

Assumption 2 ([11]) There are p distinct urns. Let $q_{c,a}$ denote the probability that c coloured ball is put in urn a. Assume there exists $d > 0$ such that for all $c = 1, 2$ and $1 \leq a \leq p$, $0 \leq q_{c,a} \leq d/p$. There exist constants $\gamma, \mu > 0$ such that the set $S_p := \{1 \leq a \leq p : q_{1,a}, q_{2,a} \geq \mu/p\}$ is such that $|S_p| \geq \gamma p$.

Theorem 2 ([11, Theorem 1]). *Define $A_p := 2q_1q_2 \sum_{a=1}^{p} q_{1,a}q_{2,a}$. Let Z be the first time that there are two balls with different colours in the same urn under Assumption 1 and 2. Then*

$$E(Z) = \sqrt{\frac{\pi}{2A_p}} + O(p^{1/4})$$

as $p \to \infty$ and the constant in the O depends on $C, q_c, d, K, \gamma, \mu$ but not depend on N or $q_{c,a}$.

Let us apply the result to the following case. Let $f(x)$ be a polynomial of degree d over $\mathbf{F}_p$. Assume that a ball r_i is considered to be coloured with white and s_j is coloured with red. Among p urns, the white ball r_i is considered to be put into an urn numbered by $f(r_i\alpha)$ and the red ball s_j into an urn numbered by $f(s_j)$. In our case $r_{k,1} = \frac{1-(-1)^k}{2}$ and $r_{k,2} = \frac{1+(-1)^k}{2}$. We consider the image $f(x)$ as urns in which the ball x is placed. After the rearrangement, we may assume that $V_1(f) = \{1, 2, \ldots, R_1\}$,

$V_2(f) = \{R_1 + 1, \ldots, R_1 + R_2\}, \ldots, V_d(f) = \{R_1 + \cdots + R_{d-1} + 1, \ldots, R_1 + \cdots + R_d\}$ where $V_i(f)$ denotes the set of images which are mapped by i elements from the domain. The probability that a ball of colour c is put in urn a is denoted by $q_{c,a}$ and in our case, for $c = 1, 2$,

$$q_{c,a} = \begin{cases} \frac{1}{p}, 1 \leq a \leq R_1 \\ \frac{2}{p}, R_1 + 1 \leq a \leq R_1 + R_2 \\ \vdots, \\ \frac{d}{p}, R_1 + \cdots + R_{d-1} + 1 \leq a \leq R_1 + \cdots + R_d. \end{cases}$$

The expected value of the iterations of f until a collision occurs between the sets

$$\{f(r_1\alpha), f(r_2\alpha), \ldots, f(r_m\alpha)\}$$

and

$$\{f(s_1), \ldots, f(s_m)\}$$

is $\sqrt{\frac{p^2\pi}{R}} + O(p^{1/4})$ for $R := \sum_{i=1}^{d} i^2 R_i$ as $p \to \infty$.

4 Multipoint Evaluation in Blackbox Manners

Let g be a group element of order p in $\mathbb{G}$. For a polynomial $a(x) = a_{n-1}x^{n-1} + \cdots + a_0 \in \mathbf{F}_p[x]$ of degree less than n, we denote $g^{a(x)} := (g^{a_0}, g^{a_1}, \ldots, g^{a_{n-1}})$.

In this section we present the fast multipoint polynomial evaluation where the coefficients of the function are given in the blackbox manners. This means that for given $g^{a_i}, 0 \leq i < n$, we can obtain $g^{a(x_i)}$'s for many points x_i in quasi-linear time in degree n. The method is analogous to usual fast multipoint evaluation using the fast Fourier transform (FFT). The key observation of the algorithm is that computing the exponentiation $(g^a)^b$ for given g^a and b corresponds to the multiplication $a \cdot b$ in the finite fields.

The first step is to obtain the fast multiplication algorithm in blackbox manners using the Discrete Fourier Transform (DFT). The DFT_ω of the polynomial $a(x)$ is an n-tuple $\big(a(1), a(\omega), a(\omega^2), \ldots, a(\omega^{n-1})\big)$ in $\mathbf{F}_p^n$ with the n-th primitive root of unity ω.

In [17], they dealt with the case when the primitive n-th root of unity exists, however it can be easily extended to the case that the primitive n-th root of unity does not exist using Schönhage and Straßen's multiplication algorithm.

The following algorithm shows the fast Fourier transform in blackbox manner and Lemma 1 analyzes its complexity.

Algorithm 1 Blackbox Fast Fourier Transform

Input : $n = 2^k \in \mathbb{N}$, $g^{a(x)} := (g^{a_0}, g^{a_1}, \ldots, g^{a_{n-1}}) \in \mathbb{G}^n$ with $a(x) \in \mathbf{F}_p[x]$ and the powers $\omega, \omega^2, \ldots, \omega^{n-1}$ of a primitive n-th root of unity $\omega \in \mathbf{F}_p$

Output : $g^{DFT_\omega(a)} := (g^{a(1)}, g^{a(\omega)}, \ldots, g^{a(\omega^{n-1})}) \in \mathbb{G}^n$

1. If $n = 1$ then return g^{a_0}
2. $g^{r_0(x)} \leftarrow (g^{a_0 + a_{n/2}}, \cdots, g^{a_{n/2-1} + a_{n-1}})$, $g^{r_1(x)} \leftarrow (g^{1 \cdot (a_0 - a_{n/2})}, \cdots, g^{w^{n/2-1}(a_{n/2-1} - a_{n-1})})$
3. call the algorithm recursively to get $g^{DFT_{w^2}(r_0)}$ and $g^{DFT_{w^2}(r_1)}$
4. return $\left(g^{a(1)}, g^{a(\omega)}, \ldots, g^{a(\omega^{n-1})}\right)$

Lemma 1. *Given $g^{a(x)} = (g^{a_0}, g^{a_1}, \ldots, g^{a_{n-1}})$ for a polynomial $a(x) = a_{n-1}x^{n-1} + \cdots + a_0$ of degree $< n$, we compute $g^{DFT_\omega(a)} = \left(g^{a(1)}, g^{a(\omega)}, \ldots, g^{a(\omega^{n-1})}\right)$ in $O(n \log n)$ group operations.*

Proof. Let $S(n)$ and $T(n)$ denote the number of exponentiations and multiplications in $\mathbb{G}$, respectively, that the algorithm requires for input size n. The cost for the individual steps is: In step 2, n multiplications and $n/2$ exponentiations by powers $\omega, \omega^2, \ldots, \omega^{n/2}$ in $\mathbb{G}$, in step 3, $2S(n/2)$ exponentiations and $2T(n/2)$ multiplications.Thus $S(n) = 2S(n/2) + n$, $T(n) = 2T(n/2) + n/2$, and by unfolding the recursions we find that $S(n) = n \log n$ and $T(n) = \frac{1}{2} n \log n$. □

Recall that $DFT_\omega(a * b) = DFT_\omega(a) \cdot DFT_\omega(b)$ where $\cdot$ denotes the pointwise multiplication and $*$ denotes the convolution map $a * b = a(x)b(x) \bmod x^n - 1$. Furthermore $a * b(x) = a(x)b(x)$ when $\deg(a(x)b(x)) < n$. The map $DFT_\omega : a(x) = (a_0, a_1, \ldots, a_{n-1}) \mapsto \big(a(1), a(\omega), \ldots, a(\omega^{n-1})\big)$ can be considered as multiplication by the matrix

$$V_\omega := \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & \omega & \cdots & \omega^{n-1} \\ 1 & \omega^2 & \cdots & \omega^{2(n-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \cdots & \omega^{(n-1)^2} \end{pmatrix}.$$

We can easily verify that $V_\omega \cdot V_{\omega^{-1}} = nI$, where I denotes the $n \times n$ identity matrix. Therefore

$$DFT_\omega^{-1} = V_\omega^{-1} = \frac{1}{n} V_{\omega^{-1}} = \frac{1}{n} DFT_{\omega^{-1}}.$$

Lemma 2 and Algorithm 2 are devoted to compute fast convolution.

Algorithm 2 Fast Convolution in Blackbox Manners

Input : $g^{a(x)}$ and $b(x)$ where $a(x), b(x) \in \mathbf{F}_p[x]$, and a primitive n-th root of unity $\omega \in \mathbf{F}_p$.
Output : $g^{a(x)b(x)} \in \mathbb{G}^n$.

1. compute $\omega^2, \ldots, \omega^{n-1}$
2. $B \leftarrow DFT_\omega(b) = \big(b(1), b(\omega), \ldots, b(\omega^{n-1})\big)$
3. $g^A \leftarrow g^{DFT_\omega(a)} = \left(g^{a(1)}, g^{a(\omega)}, \ldots, g^{a(\omega^{n-1})}\right)$
4. $g^C = g^{DFT_\omega(a*b)} \leftarrow (g^A)^B$, pointwise exponentiation
5. return $g^{DFT_\omega^{-1}(C)} = \left(g^{DFT_{\omega^{-1}}(C)}\right)^{\frac{1}{n}}$

Lemma 2. *Let ω be a primitive n-th root of unity. Given $g^{a(x)} = (g^{a_0}, g^{a_1}, \ldots, g^{a_{n-1}})$ and $b(x) \in \mathbf{F}_p[x]$ with $\deg(ab) < n$, we can compute $g^{a(x)b(x)}$ in $O(n \log n)$ group exponentiations.*

Proof. The cost for each step becomes: in step 1, $n - 2$ multiplication by $\omega \in \mathbf{F}_p$ and step 2 $O(n \log n)$ operations in $\mathbf{F}_p$. In step 3 and 5, we require $O(n \log n)$ group exponentiations by Lemma 1, in step 4 $O(n)$ exponentiations are needed. □

In the next lemma, the blackbox version of the fast polynomial division algorithm is proposed. We use the Newton iteration method as in the ordinary case. We define the *reversal* of a polynomial $a(x)$ as $\mathrm{rev}_k(a) = x^k a(1/x)$. Note that if $k = \deg(a)$ then the reversal of a is simply the polynomial with the

coefficients of a reversed. Consider the division a by b, we want to find polynomials q and r such that $a = bq + r$ with $\deg(r) < \deg(b)$. By the definition of the reversal we can easily obtain

$$\begin{aligned}\mathrm{rev}_n(a) &= \mathrm{rev}_m(b)\mathrm{rev}_{n-m}(q) + x^{n-m+1}\mathrm{rev}_{m-1}(r) \\ &= \mathrm{rev}_m(b)\mathrm{rev}_{n-m}(q) \bmod x^{n-m+1}\end{aligned}$$

where $\deg(a) = n, \deg(b) = m$. Then $\mathrm{rev}_{n-m}(q) = \mathrm{rev}_n(a)\mathrm{rev}_m(b)^{-1} \bmod x^{n-m+1}$, and we can find $\mathrm{rev}_m(b)^{-1} \bmod x^{n-m+1}$ by using Newton iteration. Consequently we can obtain q and $r = a - bq$. The following algorithm is simply the analogous version of the fast polynomial division using Newton iteration.

Algorithm 3 Blackbox Polynomial Division

Input : $g^{a(x)}$ and $b(x)$ where $a(x), b(x) \in \mathbf{F}_p[x]$ with $\deg(a) = n, \deg(b) = m$ $(n > m)$
Output : $g^{a(x) \bmod b(x)} \in \mathbb{G}^{m-1}$

1. Compute $\mathrm{rev}_m(b)^{-1} \bmod x^{n-m+1}$ using Newton iteration.
2. Call the algorithm 2 to compute $g^{q(x)} = g^{\mathrm{rev}_n(a)\mathrm{rev}_m(b)^{-1} \bmod x^{n-m+1}}$
3. Call the algorithm 2 to compute $g^{q(x)b(x)}$ with inputs $g^{q(x)}$ and $b(x)$
4. Return $g^{r(x)} = g^{a(x)}/g^{q(x)b(x)}$

Lemma 3. *Given $g^{a(x)}$ and $b(x)$, we compute $g^{a(x) \bmod b(x)}$ in $O(n \log n)$ group exponentiations, where $\deg(a) = 2n, \deg(b) = n$.*

Proof. The cost for individual steps becomes: in step 1, $O(m \log m)$ field operations, and $O((n-m)\log(n-m))$ group exponentiations by lemma 2 in step 2, and $O(n \log n)$ group exponentiations in step 3. Finally in step 4 we only require n divisions of the group elements. Especially if $\deg(a) = 2n, \deg(b) = n$, then the total cost becomes $O(n \log n)$ group exponentiations. □

Finally we can propose the fast multipoint evaluation algorithm by building up the sub-product tree of $(x - x_0)(x - x_1) \cdots (x - x_{n-1})$ where $x_0, \ldots, x_{n-1}$ are to be evaluated values. Let $m_i := x - x_i$ and define the recursive relations $M_{0,j} = m_j$, $M_{i+1,j} = M_{i,2j} \cdot M_{i,2j+1}$. From the fact that $f(x_j) = f(x) \bmod m_j$, we can obtain the following algorithm and the lemma is just a direct consequence.

Algorithm 4 Blackbox Multipoint Evaluation Algorithm

Input : $g^{a(x)}$ with $a(x) \in \mathbf{F}_p[x]$ of degree $< n = 2^k$ for some $k \in \mathbb{N}$ and $x_0, \ldots, x_{n-1} \in \mathbf{F}_p$
Output : $g^{a(x_0)}, \ldots, g^{a(x_{n-1})} \in \mathbb{G}$

1. Compute the subproduct $M_{i,j}$
2. Call the algorithm 3, $g^{R_0} \leftarrow g^{a(x) \bmod M_{k-1,0}}$, $g^{R_1} \leftarrow g^{a(x) \bmod M_{k-1,1}}$
3. Call the algorithm recursively to compute $g^{R_0(x_0)}, \ldots, g^{R_0(x_{n/2-1})}$
4. Call the algorithm recursively to compute $g^{R_1(x_{n/2})}, \ldots, g^{R_1(x_{n-1})}$
5. Return $g^{R_0(x_0)}, \ldots, g^{R_0(x_{n/2-1})}, g^{R_1(x_{n/2})}, \ldots, g^{R_1(x_{n-1})}$

Lemma 4. *Given $g^{a(x)} = (g^{a_0}, g^{a_1}, \ldots, g^{a_{n-1}})$, we can compute $g^{a(x_0)}, \cdots, g^{a(x_{n-1})}$ in $O(n \log^2 n)$ exponentiations in group $\mathbb{G}$.*

Let $a(x) = a_{n-1}x^{n-1} + \cdots + a_0$ and $b(x) = b_{n-1}x^{n-1} + \cdots + b_0$ be polynomials over $\mathbf{F}_p$ with $\deg(ab) < n = 2^k$. The blackbox Schönhage-Straßen multiplication outputs $g^{a(x)b(x)} = (g^{c_0}, g^{c_1}, \ldots, g^{c_{n-1}})$ with inputs $g^{a(x)} = (g^{a_0}, g^{a_1}, \ldots, g^{a_{n-1}})$ and $b(x) = (b_0, \ldots, b_{n-1})$.

Let us first explain the non-blackbox version of the method. Let $m = 2^{\lfloor k/2 \rfloor}$ and $t = n/m$. Write down the polynomial as $a(x) = A_0 + A_1 x^m + \cdots + A_{t-1} x^{m(t-1)}$ where $A_i \in \mathbf{F}_p[x]$ with degree less than m and let $a'(x, y) = A_0 + A_1 y + \cdots + A_{t-1} y^{t-1} \in \mathbf{F}_p[x, y]$ so that $a'(x, x^m) = a(x)$. Consider a ring $D := \mathbf{F}_p[x]/(x^{2m} + 1)$ and let $\zeta \in D$ be an element corresponding to x in $\mathbf{F}_p[x]/(x^{2m} + 1)$. Then we can view $a^*(y) = a'(\zeta, y) = A_0(\zeta) + A_1(\zeta) y + \cdots + A_{t-1}(\zeta) y^{t-1} \in D[y]$. The goal is to obtain $a(x)b(x) \bmod x^n + 1$ which is equivalent to $a^*(y)b^*(y) \bmod y^t + 1$. However since $\zeta^{2m} = -1$ and $\zeta^{4m} = 1$, ζ is a $4m$-th primitive root of unity in D, thus $\eta = \zeta^2$ if $t = m$ and $\eta = \zeta$ if $t = 2m$ is a primitive $2t$-th root of unity in D. Now we want to compute $a^*(\eta y)b^*(\eta y) \bmod (\eta y)^t + 1$ or $a^*(\eta y)b^*(\eta y) \bmod y^t - 1$, this can be done by fast multiplication using the discrete Fourier transform with the t-th primitive root of unity $\omega = \eta^2$ in D. The multiplication in D can be done recursively with polynomial degree less than $2m$. In blackbox version of the algorithm, we simply write $g^{a(x)} = (g^{a_0}, g^{a_1}, \ldots, g^{a_{n-1}}) = (g^{A_0}, \ldots, g^{A_{t-1}})$ where g^{A_i} means $(g^{a_{mi}}, g^{a_{mi+1}}, \ldots, g^{a_{mi+(m-1)}})$.

Finally, we give the blackbox version of fast blackbox Schönhage and Straßen multiplication in the following algorithm.

Algorithm 5 Blackbox Schönhgae-Straßen Multiplication

Input : $n = 2^k \in \mathbb{N}$, $g^{a(x)} := (g^{a_0}, g^{a_1}, \ldots, g^{a_{n-1}}) \in \mathbb{G}^n$ and $b(x) = b_{n-1}x^{n-1} + \cdots + b_0$ where $a(x), b(x) \in \mathbf{F}_p[x]$ with $\deg(ab) < n$
Output : $g^{a(x)b(x)} := (g^{c_0}, g^{c_1}, \ldots, g^{c_{n-1}}) \in \mathbb{G}^n$

1. $m \leftarrow 2^{\lfloor k/2 \rfloor}$, $t \leftarrow n/m$
   let $g^{a(x)} = (g^{A_0}, \ldots, g^{A_{t-1}})$ and $b(x) = (B_0, \ldots, B_{t-1})$ so that $a(x) = \sum_{i=0}^{t-1} A_i x^{mi}$, $b(x) = \sum_{i=0}^{t-1} B_i x^{mi}$ where $\deg A_i, \deg B_j < m$
2. let $D = \mathbf{F}_p[x]/(x^{2m} + 1)$ and $\zeta \leftarrow x \bmod (x^{2m} + 1)$
   if $t = 2m$ then $\eta \leftarrow \zeta$, otherwise $\eta \leftarrow \zeta^2$ so that η is a primitive $2t$-th root of unity
   call the algorithm 2 with $\omega = \eta^2$ to compute $g^{c^*(\eta y)} = g^{a^*(\eta y)b^*(\eta y) \bmod (y^t - 1)}$ using algorithm 5 recursively for the multiplication in D
3. return $g^{c^*(y)} = (g^{C_0}, \ldots, g^{C_{t-1}})$

Lemma 5. *Let $a(x), b(x) \in \mathbf{F}_p[x]$ with $\deg(ab) < n$. Given $g^{a(x)}$ and $b(x)$, we can compute $g^{a(x)b(x)}$ in $O(n \log n \log\log n)$ group operations.*

5 New Approach to DLP with Auxiliary Inputs using Function Mapping

In this section, we show that if a function $f(x)$ defined over $\mathbf{F}_p$ has many collisions by its mapping, then it can be used to solve DLPwAI efficiently. Our function can also be a rational function, however for a while we let our function to be a polynomial of degree d.

Let $f(x) := f_0 + f_1 x + \cdots + f_d x^d \in \mathbf{F}_p[x]$. Define the value set of $f(x)$ by $V(f) := \{f(x) : x \in \mathbf{F}_p\} = \{a_1, \ldots, a_t\}$ where t is the size of the value set. Since f is a polynomial of degree d, we have $q_\ell := \#f^{-1}(a_\ell) \le d$ for $1 \le \ell \le t$. Thus $p = \sum_{\ell=1}^{t} q_\ell \le d \cdot t$ which is equivalent to $t \ge \lfloor \frac{p-1}{d} \rfloor + 1$. If a function $f(x)$ has $V(f)$ with size $t = \lfloor \frac{p-1}{d} \rfloor + 1$, we say that the function has a minimal value set.

5.1 New Approach with Function Mappings

Let R_i denote the number of the set $S_i := \{a_\ell \in V(f) : |f^{-1}(a_\ell)| = i\}$, *i.e.*, the number of images which are mapped by i elements. We also define R to be the size of the set $\{(x, y) \in \mathbf{F}_p \times \mathbf{F}_p : f(x) = f(y)\}$. Then we have

$$p = \sum_{i=1}^{d} iR_i, |V(f)| = \sum_{i=1}^{d} R_i, \text{ and } R = \sum_{i=1}^{d} i^2 R_i.$$

For randomly chosen $r \in \mathbf{F}_p$, the probability that $f(r)$ (or $f(r\alpha)$, resp.) is equal to a_ℓ in S_i for some i is $\frac{i}{p}$. In this case we shall say that a ball $r \in \mathbf{F}_p$ colored by 1 (or 2, resp.) is put into a urn $a_\ell \in S_i$. In this manner we want to find a collision, that is, two balls of different types are put into a same urn. Naturally the expected number of trials until a collision occurs is related to the number of solutions of the equation $f(x) = f(y)$.

Definition 1. *Let $f(x)$ be any function defined over $\mathbf{F}_p$. Consider the algorithm which receives $g, g^\alpha, \ldots, g^{\alpha^d}$ and the description of $f(x)$ as inputs and returns $g^{f(r_1\alpha)}, g^{f(r_2\alpha)}, \ldots, g^{f(r_m\alpha)}$ for random $r_i \in \mathbf{F}_p$ (or $g^{f(s_1)}, g^{f(s_2)}, \ldots, g^{f(s_m)}$ for random $s_i \in \mathbf{F}_p$ resp.). The number of required group operations of the algorithm is defined by $T_1(m)$ (or $T_2(m)$ resp.). We simply define $T(m) := T_1(m) + T_2(m)$.*

Theorem 3. *Let $g, g^\alpha, \cdots, g^{\alpha^d}$ with $d \leq p^{1/3}$ be given and let $f(x) := f_0 + f_1 x + \cdots + f_d x^d$ be a polynomial of degree d over $\mathbf{F}_p$. Let R and R_i be defined as aforementioned. Then we can find α with $T_1(m)+O(m+d)$ group operations and $O(m \log^2 d + d \log p)$ $\mathbf{F}_p$-operations, where the expected value of m is $\sqrt{\frac{p^2\pi}{R}} + O(p^{1/4})$. The group operation includes group multiplications and exponentiations.*

Proof. Suppose we choose random elements r_i and s_j from $\mathbf{F}_p$ alternatively and compute $f(r_i\alpha)$ and $f(s_j)$ sequentially. Let m be the number of iterations until a collision occurs between the following two lists

$$\{f(r_1\alpha), f(r_2\alpha), \ldots, f(r_m\alpha)\}$$

and

$$\{f(s_1), \ldots, f(s_m)\}.$$

The expected value of m is $\sqrt{\frac{p^2\pi}{R}} + O(p^{1/4})$ as described in Section 3.

After a collision occurs, solving an equation $f(r_i\alpha) = f(s_j)$ with respect to the intermediate α gives candidates for α. However actually the value α is secret thus we cannot compute $f(r_i\alpha)$. Thus we shall find a collision in the exponentiated forms instead of the original values. Computing $g^{f(r_i\alpha)}$ and $g^{f(s_j)}$ with the description of $f(x)$ requires $T(m)$ group operations by the definition 1.

The value of $T_2(m)$ can be determined by the fast multipoint polynomial evaluation. Evaluating $f(x)$ at randomly chosen points $s_1, \ldots, s_m$ requires $O(m \log^2 d)$ operations in $\mathbf{F}_p$. Then computing the list $\left\{g^{f(s_1)}, \ldots, g^{f(s_m)}\right\}$ takes $O(m)$ group exponentiations. After we have a collision, one solves the equation $f(r_i\alpha) = f(s_j)$ with respect to the intermediate α. The solutions of the equation consist of $f^{-1}(f(s_j))$. Thus we have at most d possibilities of $r_i\alpha$. We can find roots in $\mathbf{F}_p$ of polynomial of degree d over $\mathbf{F}_p$ using an expected number of $\tilde{O}(d \log p)$ operations in $\mathbf{F}_p$ [25]. If the number of solutions for the equation does not exceed d elements, then we raise these solutions to the power with base g in $O(d)$ group exponentiations and compare these values with g^α. □

Remark 1. As shown in Section 4, $T_1(m)$ is equal to $O(m \log^2 d)$ for $m \geq d$ if a given function f is a polynomial of degree d. In this case, the total complexity of the algorithm is upper bounded by $O(m \log^2 d + d)$ group operations.

Remark 2. For any function, not a polynomial, $f(x)$ defined over $\mathbf{F}_p$, we can show by analogous argument using non-uniform birthday problem that the value m is equal to $O\left(\sqrt{p^2/R}\right)$ where R is the number of $\mathbf{F}_p$-points on the curve $f(x) - f(y) = 0$. Thus for a constant map $f(x)$, R attains the maximum value p^2 which induces $m = O(1)$. It trivially means that finding the collision is easy, but in this case the number of the candidates for α becomes p, which obstructs finding the solutions by exhaustive trials.

5.2 Substitution Polynomial and Value Set of Polynomial

In our algorithm, it is crucial to find a polynomial with sufficiently large R. Since $R = \sum_{i=1}^{d} i^2 R_i$ and $p = \sum_{i=1}^{d} iR_i$, it is easily seen that $p \leq R \leq dp$ inducing $O(\sqrt{p/d}) \leq m \leq O(\sqrt{p})$. In the case $m = O(\sqrt{p})$, the algorithm performs like the ordinary DL-solving algorithms such as Pollard's rho algorithm. In the case $m = O(\sqrt{p/d})$, the proposed algorithm collides with Cheon's algorithm and the value is optimal in the sense of Shoup's [21] lower bound in generic group model.

If $|V(f)| \leq \frac{p}{c}$ for some $1 \leq c \leq d$, then $R \geq cp$ by Cauchy-Schwartz inequality

$$\left(\sum_{i=1}^{d} i^2 R_i\right)\left(\sum_{i=1}^{d} R_i\right) \geq \left(\sum_{i=1}^{d} iR_i\right)^2.$$

Thus the polynomial with small value sets would give an improvement in the complexity of the algorithm.

It has been long-standing issue to determine the value sets of polynomials. Carlitz *et al.* [9] showed that the polynomial with minimal value sets exists if and only if $d|(p-1)$ and $f(x)$ is of form $a(x-b)^d+c$ for some constants a, b and c in $\mathbf{F}_p$. Wan *et al.* [24] showed that there exists no polynomial whose value set size is between $\lfloor \frac{p-1}{d} \rfloor + 1$ and $\lfloor \frac{p-1}{d} \rfloor + \frac{2(p-1)}{d^2}$. Furthermore, Gomez-Calderon and Madden [12] gave a complete list of polynomials with $V(f) \leq \frac{2p}{d}$. Unfortunately, it is shown that the mean value of $|V(f)|$ over all polynomials of degree $d < p$ is about $c_d p + O(1)$ where $c_d = 1 - \frac{1}{2!} + \frac{1}{3!} + \cdots + (-1)^{d-1}\frac{1}{d!} \approx 1/e$ by Uchiyama [23], so we can deduce that polynomials with small value sets are rare.

More precisely, the value of R for a polynomial $f(x)$ is definitely related to the substitution polynomial $f^*(x, y) = f(x) - f(y) \in \mathbf{F}_p[x, y]$. It is widely used to determine the value set of polynomials.

Weil's bound says that if $f^*(x, y)$ has r absolutely irreducible factors (that is, irreducible factors over $\mathbf{F}_p$ which are irreducible over algebraic closure of $\mathbf{F}_p$), then the value of R is $rp + O(d^2\sqrt{p})$. With this observation, Mit'kin [16] estimated the value of R, which says that $R < (\lfloor d/2 \rfloor + 1)p + c(d)p^{1/2}$ in most cases for a constant $c(d)$ depending on d. He also showed that for a polynomial $\varphi(x, y)$ of degree t, being square-free and having no factor $(x - y)$ and $Res_y(\varphi(x, y), \varphi(z, y)) = c(x - z)^t(\varphi(x, z))^{t-1}$, $c \neq 0$, there exists a polynomial $f(x)$ such that $f^*(x, y) = (x - y)\varphi(x, y)$. Here, $Res_y(\cdot, \cdot)$ denotes the resultant with respect to y. From this, he derived the existence of $f(x)$ satisfying

$$f^*(x, y) = (x - y) \prod_{k=1}^{(d-1)/2} \left(x^2 - (\zeta^k + \zeta^{-k})xy + y^2 + a(\zeta^{2k} + \zeta^{-2k} - 2))\right)$$

for $a \in \mathbf{F}_p^*$, where $\zeta \in \mathbf{F}_{p^2}$ is a primitive d-th root of unity and $d|(p^2 - 1)$. This polynomial is exactly the substitution polynomial of the Dickson polynomial. This result induces that $R = \frac{d+1}{2}p + O(d^2)$ and in Section 6.1 we give the exact value of R by virtue of [6].

In particular, Gomez-Calderon and Madden [12] gave a complete list of polynomials of degree $d < p^{1/4}$ which has $r \geq \lfloor d/2 \rfloor$. The list contains polynomials of following forms:

1. $f(x) = (x + a)^d + b$, where $d|p - 1$,
2. $f(x) = ((x + a)^{d/2} + b)^2 + c$, where $(d/2)|(p - 1)$,
3. $f(x) = ((x + a)^2 + b)^{d/2} + c$, where $(d/2)|(p - 1)$, or
4. $f(x) = D_{d,a}(x + b) + c$, where $D_{d,a}(x)$ is the Dickson polynomial of degree d such that $d|(p^2 - 1)$ and a is a 2^k-th power in $\mathbf{F}_{p^2}$, where $d = 2^k r$ for odd r.

By Weil's bound, we can write $m = O\left(\sqrt{p^2/R}\right)$ in terms of $O\left(\sqrt{p/r}\right)$. The first polynomial of the list trivially factors into d linear factors, assuming $a = 0$, $f(x) - f(y) = (x - \zeta^0 y) \cdots (x - \zeta^{d-1} y)$ where ζ is a primitive d-th root of unity in $\mathbf{F}_p$, hence has d absolutely irreducible factors and

$m = O(\sqrt{p/d})$. Second polynomial factors into $d/2$ linear factors and one irreducible factor, hence $m = O(\sqrt{2p/d})$. The third and forth have one or two linear factor(s) and all others quadratic factors, thus $r \approx \lfloor d/2 \rfloor$.

Overall, the existence of polynomials with not so small $r < d/2$ may not be clear, however it seems that analyzing the substitution polynomial may give a new direction to solve DLPwAI.

5.3 Application with Rational Functions

Let $f(x) = \frac{f_1(x)}{f_2(x)}$ be a rational function defined over $\mathbf{F}_p$ where $f_1(x)$ and $f_2(x)$ are relatively prime. The degree of $f(x)$ is defined to be the maximum of the degrees of the numerator and the denominator. If the degree of $f(x)$ is d, then the solutions of the equation $f(x) = a$ are equal to the solutions of $f_1(x) - af_2(x) = 0$ with $\deg(f_1 - af_2) \leq d$. By simple observation, Theorem 3 also holds for the rational functions.

Consider an elliptic curve $E(\mathbf{F}_p)$ with a divisor k of its order and a rational function $f(x) = \frac{\phi_k(x)}{\psi_k^2(x)}$ where $\psi_k(x)$ denotes the k-th division polynomial. Then we know that $f(x)$ denotes the x-coordinate of the point $[k](x, y) \in E(\mathbf{F}_p)$ for $(x, y) \in E(\mathbf{F}_p)$. If we define $S = \{x \in \mathbf{F}_p : (x, y) \in E(\mathbf{F}_p)\}$ then $f(S) = x[H]$ where $H \subset E(\mathbf{F}_p)$ is a subgroup of order $\frac{\#E(\mathbf{F}_p)}{k}$ and $x[H]$ denotes the set of x-coordinates of all points in H. It means that the mapping by f is almost k-to-1 mapping with domain S. Since $|S| \approx p/2$, we have $R \geq k^2 R_k \approx kp/2$ thus the value $m = \sqrt{p^2/R} \leq O(\sqrt{p/k})$. Consequently, to solve DLPwAI, we require at least $T(m)$ group operations, however it remains open to compute $g^{f(r_1\alpha)}, g^{f(r_2\alpha)}, \ldots, g^{f(r_m\alpha)}$ in quasi-linear in terms of m.

The remarkable thing is that for any d, we can obtain a rational function $f(x) = \frac{f_1(x)}{f_2(x)}$ of degree d with $R \geq \sqrt{d}p$ easily by using the $\sqrt{d}$-th division polynomial unlike polynomial cases.

6 Examples

We shall give a concrete example by using the Dickson polynomial. The example solves α in $O\left(\sqrt{\frac{p\pi}{d}} \log^2 d\right)$ group operations given $g, g^{\alpha}, \ldots, g^{\alpha^d}$ when $d|(p+1)$, otherwise one requires $g, g^{\alpha}, \ldots, g^{\alpha^d}, \ldots, g^{\alpha^{2d}}$ using Cheon's algorithm. Satoh's generalization [19] also solves the problem with $g, g^{\alpha}, \ldots, g^{\alpha^d}$ for $p+1$ case. The value set of the Dickson polynomial is not small, so it shows that the value sets of the polynomials are not necessarily small.

6.1 Application with the Dickson Polynomials

For $a \in \mathbf{F}_p^*$ and $d \in \mathbb{Z}_{\geq 1}$, let

$$D_d(x, a) = \sum_{k=0}^{\lfloor d/2 \rfloor} \frac{d}{d-k} \binom{d-k}{k} (-a)^k x^{d-2k}$$

denote the Dickson polynomial of degree d over $\mathbf{F}_p$. In this section, we explain the cardinality of R_i's of the Dickson polynomial without proof. Refer to [6] for details.

Let p be an odd prime. Write $2^u || v$ if $2^u | v$ and $2^{u+1} \nmid v$. Suppose that $2^u || (p^2 - 1)$. Recall that R_i for the polynomial $f(x)$ denotes the number of the set $\{a_\ell \in V(f) : \#f^{-1}(a_\ell) = i\}$. The followings are direct consequence of Theorem 9 in [6].

Case 1. $2^u||d$. Then

$$R_i = \begin{cases} \frac{p-1-i}{2i}, i = (d, p-1) \\ \frac{p+1-i}{2i}, i = (d, p+1) \\ 1, \quad i = \frac{(d,p-1)+(d,p+1)}{2} \\ 0, \quad \text{otherwise.} \end{cases}$$

Case 2. $2^{u-1}||d$. If a is a quadratic residue in $\mathbf{F}_p$, then R_i's are same as in Case 1. Otherwise, if a is a quadratic non-residue in $\mathbf{F}_p$, then

$$R_i = \begin{cases} \frac{p-1-i}{2i}, i = (d, p-1) \\ \frac{p+1-i}{2i}, i = (d, p+1) \\ 1, \quad i = \frac{(d,p-1)}{2} \text{ or } \frac{(d,p+1)}{2} \\ 0, \quad \text{otherwise.} \end{cases}$$

Case 3. $2^v||d$ with $1 \le v \le u-2$. Suppose that $\frac{p-1}{(d,p-1)}$ is odd and $\frac{p+1}{(d,p+1)}$ is even. If a is a quadratic residue, then

$$R_i = \begin{cases} \frac{p-1-i}{2i}, \ i = (d, p-1) \\ \frac{p+1-2i}{2i}, i = (d, p+1) \\ 1, \quad i = \frac{(d,p-1)+(d,p+1)}{2} \text{ or } \frac{(d,p+1)}{2} \\ 0, \quad \text{otherwise.} \end{cases}$$

If a is a quadratic non-residue, then

$$R_i = \begin{cases} \frac{p-1-i}{2i}, i = (d, p-1) \\ \frac{p+1}{2i}, \quad i = (d, p+1) \\ 1, \quad i = \frac{(d,p-1)}{2} \\ 0, \quad \text{otherwise.} \end{cases}$$

The case where $\frac{p-1}{(d,p-1)}$ is even and $\frac{p+1}{(d,p+1)}$ is odd is similar.

Case 4. d is odd. If a is a quadratic residue, then

$$R_i = \begin{cases} \frac{p-1-2i}{2i}, i = (d, p-1) \\ \frac{p+1-2i}{2i}, i = (d, p+1) \\ 2, \quad i = \frac{(d,p-1)+(d,p+1)}{2} \\ 0, \quad \text{otherwise.} \end{cases}$$

Otherwise,

$$R_i = \begin{cases} \frac{p-1}{2i}, i = (d, p-1) \\ \frac{p+1}{2i}, i = (d, p+1) \\ 0, \quad \text{otherwise.} \end{cases}$$

In any case if some index i of R_i coincides with the other j, then we consider R_i as $R_i + R_j$. For example, in Case 1, if $(d, p-1) = (d, p+1) = 1$ then

$$R_1 = \frac{p-1-(d,p-1)}{2(d,p-1)} + \frac{p+1-(d,p+1)}{2(d,p+1)} + 1 = p.$$

Thus $D_d(x, a)$ is a permutation if $(d, p-1) = (d, p+1) = 1$.

Consider the Dickson polynomial of an odd degree d with quadratic non-residue a. This is the second case in Case 4. Assume that d divides $p+1$, then trivially $(d, p-1) = 1$ since d is odd.

Thus we have $R_1 = \frac{p-1}{2}$, $R_d = \frac{p+1}{2d}$ and $R \approx \frac{dp}{2}$ for $D_d(x, a)$. By applying our method, we can recover α in $O(\sqrt{p/d}\log^2 d)$ group operations. If $d \approx p^{1/3}$ then it becomes $O(p^{1/3}\log^2 d)$. With the Dickson polynomial $D_d(x, a)$ of degree d, we only require $g, g^{\alpha}, \ldots, g^{\alpha^d}$ to solve DLPwAI, not $g, g^{\alpha}, \ldots, g^{\alpha^d}, \ldots, g^{\alpha^{2d}}$.

7 Conclusion and Open Problems

In this paper, we proposed a new approach to solve DLPwAI focusing on the behavior of the function mapping. For a given polynomial over $\mathbf{F}_p$, our algorithm recovers α with $g, g^{\alpha}, \ldots, g^{\alpha^d}$ in $\tilde{O}(\sqrt{p^2/R} + d)$ operations in the base group and $\mathbf{F}_p$ where R denotes the number of zeros in $\mathbf{F}_p$ of the substitution polynomial $f^*(x, y) = f(x) - f(y)$. The value of R is determined by the number of absolutely irreducible factors of $f^*(x, y)$ by Weil's theorem. As a result, we reduced solving DLPwAI into finding a polynomial whose substitution polynomial has many absolutely irreducible factors. It would be enlightening to study further on the substitution polynomial.

Unlike polynomial cases, it is easy to find a rational function of degree d with $R \approx p\sqrt{d}$ using the $\sqrt{d}$-th division polynomial. In this case, the complexity becomes $\tilde{O}\left(\sqrt{p/\sqrt{d}} + d\right)$, assuming $T(m) = \tilde{O}(m)$. However, it seems hard to obtain an algorithm with $T(m) = \tilde{O}(m)$ for a positive integer m.

It would be also worthwhile to determine whether the lower bound for DLPwAI in the generic group model coincides with the complexity of the proposed algorithm. It may give another evidence of hardness to solve DLPwAI other than $p \pm 1$ cases.

References

1. Boneh, D., Boyen, X.: Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223-238. Springer, Heidelberg (2004)
2. Boneh, D., Boyen, X.: Short Signatures Without Random Oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56-73. Springer, Heidelberg (2004)
3. 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)
4. Boneh, D., Gentry, C., Waters, B.: Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 258–275. Springer, Heidelberg (2005)
5. Brown, D., Gallant, R.: The Static Diffie-Hellman Problem, Available in http://eprint.iacr.org/2004/306. (2004)
6. Chou., W.-S., Gomez-Calderon, J., Mullen, G. L.: Value Sets of Dickson Polynomials over Finite Fields. Journal of Number Theory 30, pp. 334–344. (1988)
7. Cheon, J. H.: Security Analysis of the Strong Diffie-Hellman Problem. In: Vaude- nay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 1-11. Springer, Heidelberg (2006)
8. Cheon, J. H.: Discrete Logarithm Problems with Auxiliary Inputs. Journal of Cryptology 23(3), pp. 457-476. (2010)
9. Carlitz, L., Lewis, D. J., Mills, W. H., Strauss, E. G.: Polynomials over Finite Fields with Minimal Value Sets. Mathematika 8, pp. 121–130. (1961)
10. den Boer, B.: Diffie-Hellman is as strong as discrete log for certain primes. Crypto 1988. LNCS, vol. 403, pp. 530–539. (1989)
11. Galbraith, S. D., Holmes, M.: A non-uniform birthday problem with applications to discrete logarithms. Discrete Applied Mathematics 160, pp. 1547–1560. (2012)
12. Gomez-Calderon, J., Madden, D. J.: Polynomials with Small Value Set over Finite Fields. Journal of Number Theory 28, pp. 167–188. (1988)
13. Kim, M.: Integer Factorization and Discrete Logarithm with Additional Information. Ph.D dissertation. Seoul National University. (2011)
14. Kozaki, S., Kutsuma, T., Matsuo, K.: Remarks on Cheons algorithms for pairing-related problems. Pairing 2007. LNCS, vol. 4575, pp. 302-316. (2007)
15. Maurer, U.: Towards the equivalence of breaking the Diffie-Hellman protocol and computing discrete logarithms. Crypto 1994. LNCS, vol. 839, pp. 271–281. (1994)
16. Mit'kin, D. A.: Polynomials with minimal set of values and the equation $f(x) = f(y)$ in a finite prime field. Matematicheskie Zametki. vol. 38(1), pp.3–14. (1985)

17. Mohassel, P.: Fast Computation on Encrypted Polynomials and Applications. In: Lin, D., Tsudik, G., Wang, X. (eds.) CANS 2011. LNCS, vol. 7092, pp.234–254. (2011)
18. Maurer, U., Wolf, S.: The Relationship Between Breaking the Diffie-Hellman Protocol and Computing Discrete Logarithms. SIAM Journal on Computing 28(5), pp. 1689–1721. (1999)
19. Satoh, T.: On Generalization of Cheon's Algorithm. http://eprint.iacr.org/2009/058.pdf (2009)
20. Sakemi, Y., Hanaoka, G., Izu, T., Takenaka, M., Yasuda, M.: Solving a discrete logarithm problem with auxiliary input on a 160-bit elliptic curve. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol 7293, pp. 595–608. (2012)
21. Shoup, V.: Lower Bounds for Discrete Logarithms and Related Problems. Eurocrypt97. LNCS, vol. 1233, pp. 256-266.(1997)
22. Sun, D.: Elliptic curves with the minimized security loss of the strong DiffieHellman problem. Ph.D. Dissertation. Seoul National University. (2007)
23. Uchiyama, S.: Note on the Mean Value of $V(f)$. Proc. Japan Acad 31, pp. 199–201 (1955).
24. Wan, D., Shiue, P.J.-S., Chen, C. S.: Value sets of polynomials over finite fileds. Proc. AMS 119(3), pp. 711–717 (1993)
25. von zur Gathen, J., Gehgard, J.: Modern Computer Algebra. Cambridge University Press. (1999)