Revisiting the Efficient Key Generation of ZHFE

Yasuhiko Ikematsu[1][2], Dung H. Duong[1][2], Albrecht Petzoldt[1], and Tsuyoshi Takagi[1][2]

[1] Institute of Mathematics for Industry, Kyushu University,
744 Motooka, Nishi-ku, Fukuoka 819-0395, Japan
[2] JST, CREST, 4-1-8 Honcho, Kawaguchi, Saitama, 332-0012, Japan
{y-ikematsu,duong,petzoldt,takagi}@imi.kyushu-u.ac.jp

Abstract. ZHFE, proposed by Porras at el. at PQCrypto'14, is one of the very few existing multivariate encryption schemes and a very promising candidate for post-quantum cryptosystems. The only one drawback is its slow key generation. At PQCrypto'16, Baena et al. proposed an algorithm to construct the private ZHFE keys, which is much faster than the original algorithm, but still inefficient for practical parameters. Recently, Zhang and Tan proposed another private key generation algorithm, which is very fast but not necessarily able to generate all the private ZHFE keys. In this paper we propose a new efficient algorithm for the private key generation of the ZHFE scheme. Our algorithm reduces the complexity from $O(n^{2\omega+1})$ by Baena et al. to $O(n^{\omega+3})$, where n is the number of variables and $2 < \omega < 3$ is a linear algebra constant. We also estimate the number of possible keys generated by all existing private key generation algorithms for ZHFE. Our algorithm generates as many private ZHFE keys as the original and Baena et al.'s ones. This makes our algorithm is the best appropriate for the ZHFE scheme.

Keywords: Post Quantum Cryptography, Multivariate Cryptography, Encryption Schemes, ZHFE

1 Introduction

In 1997, P. Shor [21] gave polynomial time quantum algorithms to factor large integers and to solve discrete logarithms. Thus, as soon as large-scale quantum computer are built, almost all public key cryptosystems currently used in practice such as RSA, DSA and ECC will become insecure. Post-Quantum Cryptography (PQC) stands for the study of cryptosystems that have the potential to resist such quantum computer attacks [1].

Recently, PQC has taken a lot of attention and become more and more important in the cryptographic research community, including also some authorities such as the American National Security Agency (NSA), who recommended governmental organizations to switch their security infrastructures from schemes such as RSA and ECC [9] to post quantum cryptosystems, and the National Institute of Standards and Technology (NIST), which is preparing to develop

Table 1. Complexity of key generation algorithms for ZHFE scheme. Here n is the number of variables, D is the degree chosen for efficient decryption.

Algorithm	Complexity	$q = 2$	$q = 3$	$q = 5$	$q = 7$
Original [20]	$\mathcal{O}(n^{3\omega})$	100%	100%	100%	100%
Baena et al. [2]	$\mathcal{O}(n^{2\omega+1})$	99.9%	99.9%	99.9%	99.9%
Ours	$\mathcal{O}(n^{\omega+3})$	99.5%	99.9%	99.9%	99.9%
Zhang-Tan [24]	$\mathcal{O}(\log_q D)$	28.9%	56.0%	76.0%	83.7%

standards for these schemes [14]. Among all possible candidates for PQC, multivariate public key cryptography (MPKC) [7] is one of the main candidates for the standardization. Multivariate schemes are in general very fast and require only modest computational resources, which makes them attractive for the use on low cost devices like smart cards and RFID chips [3,5]. In the area of digital signatures, there exists a large number of practical multivariate schemes [8,11,18]. The great difficulty for MPKC is encryption.

The C^* scheme introduced by Matsumoto and Imai [13], hence the name MI scheme, was considered to be the first encryption scheme. After MI was broken by Patarin [15], many encryption schemes have been proposed but then efficiently broken. Notably, Paratin invented the Hidden Field Equation cryptosystem (HFE) [16] which replaces the central map of the MI scheme by a low degree univariate polynomial. However, using low degree polynomials in the central map makes HFE be broken [12,6]. In order to thwart the attack, Porras et al. [19,20] cleverly proposed at PQCrypto'14 an interesting encryption scheme called ZHFE, which uses two high degree HFE polynomials in the central map, but a chosen low degree D polynomial for efficient decryption; see Section 2.2 for more details.

The ZHFE scheme [19,20] is one of the few existing multivariate encryption schemes at the moment, among ABC [22], SRP [25] and EFC [23]. However, what makes ZHFE important and attractive is its efficiency and thorough security analysis, see [20,17]. One drawback of ZHFE is its super slow key generation process, which involves solving large linear systems; the original method [20] for generating the private key needs to solve a linear system of about n^3 variables, resulting in a complexity of $\mathcal{O}(n^{3\omega})$, where $2 < \omega < 3$ is a linear algebra constant. At PQCrypto'16, Baena et al. [2] proposed an improved algorithm which reduces the complexity of this step to $\mathcal{O}(n^{2\omega+1})$. Their idea is to use a well representation of HFE polynomials. As a result, the matrix associated to the large linear system forms a shape close to a block diagonal matrix. For practical parameters, this algorithm is much faster than the original one but still inefficient. Recently, Zhang and Tan [24] proposed an algorithm which requires very little computation; their algorithm reduces the complexity to $\mathcal{O}(\log_q D)$ which makes their algorithm very fast; here D is the degree of the secret polynomial (see Section 2.2 for more details). However, their algorithm is based on the invertibility condition of some linear map, which is not necessarily fulfilled, and this prevents their algorithm from generating all the private ZHFE keys; see Section 2.3 for more details. Therefore, their structured key generation algorithm may possibly weaken the security of the scheme.

Our contribution. In this paper, we propose a new private key generation algorithm of the ZHFE scheme. The complexity of our algorithm is $\mathcal{O}(n^{\omega+3})$ which improves the one by Baena et al. [2]; for example, for 96-bit security parameters $(q = 7, D = 105, n = 55)$ and 111-bit security parameters $(q = 17, D = 595, n = 55)$, our algorithm is around 15 and 256 times faster than that of Banea et al. [2] respectively (see our implementation results in Table 3). Moreover, our algorithm generates as many private ZHFE keys as that of Baena et al. [2]. Our method is as follows: we first analyze again the algebraic structure of the central map in ZHFE scheme, following the route of Banea et al. [2]. At some stage, instead of working in the base field, we lift our problem to the extension field and use the properties of the extension field to construct an algorithm which is simpler and more efficient than that of Banea et al. [2]; see Section 3 for more details.

We also estimate the number of private ZHFE keys that all existing algorithms generate in Table 2. Zhang and Tan's algorithm [24] generates only those private ZHFE keys, for which the corank of a given linear map $\mathcal{L}$ is 0. As Table 2 shows, this condition is, in the case of $q = 2$, fulfilled by only 28.9 % of all possible keys, which means that the algorithm of [24] generates only a small part of the keys. In contrast to this, our algorithm generates nearly 100% of the keys, since it can deal with linear maps $\mathcal{L}$ of corank < 3. This, together with its efficiency, makes our algorithm to be the most appropriate private key generation algorithm of the ZHFE scheme.

Organization. Our paper is organized as follows: we briefly recall the ZHFE scheme and the various private key generation of Porras et al. [20], Baena et al. [2] and Zhang and Tan [24] in Section 2. Our algorithm is explicitly introduced and analyzed in Section 3. In Section 4 we present a MAGMA implementation of our algorithm and compare it with Baena's algorithm with respect to running time and memory consumption. Finally, we conclude our paper in Section 5.

2 The ZHFE Scheme and its Key Generation Algorithms

In this section, we briefly recall the basic concepts of multivariate encryption schemes and the ZHFE scheme [20]. We also recall the key generation process in the ZHFE scheme and the improved algorithms by Baena et al. [2] and Zhang and Tan [24].

2.1 Multivariate Public Key Cryptography

The basic objects of multivariate public key cryptography are systems of multivariate quadratic polynomials over a finite field $\mathbb{F}$. The security of multivariate schemes is based on the *MQ-Problem* which asks for a solution of a given system of multivariate quadratic polynomials over the field $\mathbb{F}$. The MQ-Problem is proven to be NP-Hard even for quadratic polynomials over the field GF(2) [10]. To build a public key cryptosystem on the basis of the MQ-Problem, one starts with an easily invertible quadratic map $\mathcal{F} : \mathbb{F}^n \rightarrow \mathbb{F}^m$ (*central map*). To hide the structure of $\mathcal{F}$ in the public key, one composes it with two invertible affine (or linear) maps $\mathcal{T} : \mathbb{F}^m \rightarrow \mathbb{F}^m$ and $\mathcal{S} : \mathbb{F}^n \rightarrow \mathbb{F}^n$. The *public key* is therefore given by $\mathcal{P} = \mathcal{T} \circ \mathcal{F} \circ \mathcal{S}$. The *private key* consists of $\mathcal{T}, \mathcal{F}$ and $\mathcal{S}$. In this paper we consider multivariate encryption schemes. For these schemes, we require $n \leq m$.

Encryption: to encrypt a message $\mathbf{x} \in \mathbb{F}^{\mathbf{n}}$, one simply computes $\mathbf{w} = \mathcal{P}(\mathbf{x})$ from the public key.

Decryption: to decrypt a given ciphertext $\mathbf{w} \in \mathbb{F}^{\mathbf{m}}$, one computes recursively $\mathbf{z} = \mathcal{T}^{\mathbf{-1}}(\mathbf{w}), \mathbf{y} = \mathcal{F}^{\mathbf{-1}}(\mathbf{z})$ and $\mathbf{x} = \mathcal{S}^{\mathbf{-1}}(\mathbf{y})$. Here $\mathbf{y}$ is the preimage of $\mathbf{z}$ under the easy to invert central map $\mathcal{F}$. The condition $n \leq m$ guarantees that this pre image and therefore the recovered plaintext will be unique.

Figure 1 shows a graphical illustration of the encryption and decryption process of multivariate schemes.

Decryption

$$\begin{tikzcd}
\mathbf{w} \in K^m \arrow[r, "\mathcal{T}^{-1}"] & \mathbf{z} \in \mathbb{F}^m \arrow[r, "\mathcal{F}^{-1}"] & \mathbf{y} \in \mathbb{F}^n \arrow[r, "\mathcal{S}^{-1}"] & \mathbf{x} \in \mathbb{F}^n \arrow[lll, bend left, "\mathcal{P}"]
\end{tikzcd}$$

Encryption

Fig. 1. General workflow of multivariate encryption schemes

2.2 The ZHFE Encryption Scheme

Let $\mathbb{F}$ be a finite field with q elements and $\mathbb{K}$ a degree n extension of $\mathbb{F}$. Let $\phi : \mathbb{K} \rightarrow \mathbb{F}^n$ be the canonical isomorphism between $\mathbb{K}$ and the vector space $\mathbb{F}^n$. Consider two HFE polynomials F_1 and F_2:

$$F_1 = \sum a_{i,j} X^{q^i+q^j} + \sum a'_i X^{q^i} + a'', \ F_2 = \sum b_{i,j} X^{q^i+q^j} + \sum b'_i X^{q^i} + b'', \quad (1)$$

whose the coefficients are undetermined. Next randomly choose $4n$ scalars $\alpha_1, ..., \alpha_{2n}$, $\beta_1, ..., \beta_{2n}$ of $\mathbb{K}$. Define four linear polynomials:

$$\begin{aligned} L_{00}(X) = \sum_{i=1}^{n} \alpha_i X^{q^{i-1}}, \quad L_{01}(X) = \sum_{i=1}^{n} \alpha_{n+i} X^{q^{i-1}}, \\ L_{10}(X) = \sum_{i=1}^{n} \beta_i X^{q^{i-1}}, \quad L_{11}(X) = \sum_{i=1}^{n} \beta_{n+i} X^{q^{i-1}}. \end{aligned} \tag{2}$$

We construct the following polynomial with q-Hamming weight three:

$$\Psi(X) := X\left(L_{00}(F_1) + L_{01}(F_2)\right) + X^q\left(L_{10}(F_1) + L_{11}(F_2)\right). \tag{3}$$

Fix a positive integer D. This D must be chosen such that each univariate polynomial equation over $\mathbb{K}$ of degree less than or equal to D can be solved efficiently by Berlekamp's algorithm. In order to generate a ZHFE key, we have to determine the coefficients of F_1, F_2 such that

$$\deg \Psi(X) \leq D.$$

In this paper, we propose an efficient algorithm to choose such coefficients of F_1, F_2; cf. Section 3. Once such coefficients are given, the ZHFE scheme [19,20] is constructed as follows. Randomly choose invertible affine transformations $\mathcal{S}$ and $\mathcal{T}$ on $\mathbb{F}^n$ (resp. $\mathbb{F}^{2n}$). Then the public key $\mathcal{P} : \mathbb{F}^n \to \mathbb{F}^{2n}$ is given by

$$\mathcal{P} = \mathcal{T} \circ (\phi \times \phi) \circ (F_1, F_2) \circ \phi^{-1} \circ \mathcal{S}.$$

This is a $2n$-tuple of quadratic polynomials over $\mathbb{F}$ in n variables.

Public Key: The field $\mathbb{F}$ and the map $\mathcal{P}$.

Private Key: $\alpha_1, ..., \alpha_{2n}, \beta_1, ..., \beta_{2n}$, F_1, F_2, Ψ, $\mathcal{S}$ and $\mathcal{T}$.

Encryption: For a plaintext message $x \in \mathbb{F}^n$ with redundant information, the ciphertext is $w = \mathcal{P}(x) \in \mathbb{F}^{2n}$.

Decryption: For a given ciphertext $w \in \mathbb{F}^{2n}$, we first compute $(W_0, W_1) = (\phi^{-1} \times \phi^{-1})(\mathcal{T}^{-1}(w)) \in \mathbb{K} \times \mathbb{K}$. Next we consider the equation of degree $\max\{D, q\}$:

$$\Psi(X) - X(L_{00}(W_0) + L_{01}(W_1)) - X^q(L_{10}(W_0) + L_{11}(W_1)) = 0.$$

We can solve this equation efficiently by our choice of D. For each solution X_0 of this equation, we compute $x_0 = S^{-1} \circ \phi(X_0)$. Then we can find the plaintext among the resulting x_0 thanks to the added redundant information.

2.3 Algorithms for the Private Key Generation of ZHFE scheme

As seen above, the central part of the private key generation of ZHFE scheme is the computation of suitable coefficients of F_1 and F_2 and of Ψ for given $\alpha_1, ..., \alpha_{2n}, \beta_1, ..., \beta_{2n}$. In this section we introduce the known algorithms for this step.

Table 2. Ratio of linear map $\mathcal{L}$ in equation (4) over all possible linear maps $\mathrm{M}_{2n}(\mathbb{F})$ with respect to the corank of $\mathcal{L}$

q	n	corank $\mathcal{L}=0$	corank $\mathcal{L}\leq 1$	corank $\mathcal{L}\leq 2$	corank $\mathcal{L}\leq 3$
$q=2$	$n\geq 35$	28.9%	86.6%	99.5%	99.9%
$q=3$	$n\geq 35$	56.0%	98.0%	99.9%	99.9%
$q=5$	$n\geq 35$	76.0%	99.8%	99.9%	99.9%
$q=7$	$n\geq 35$	83.7%	99.9%	99.9%	99.9%

The original algorithm In the original papers [19],[20], F_1 and F_2 were computed by solving a large linear system over the small field $\mathbb{F}$ obtained from vanishing coefficients of Ψ. The size of this linear system is about n^3. Thus the complexity of this private key generation is $\mathcal{O}(n^{3\omega})$, where $2 < \omega < 3$ is a linear algebra constant. In fact, this algorithm is very inefficient for practical parameters (See [2, Table 3, Old method]).

Baena et al.'s algorithm At PQCrypto'16, Baena et al. [2] proposed a new improved algorithm for the private key generation of ZHFE scheme. Their idea is to use a well representation of HFE polynomials. As a result, the matrix associated to the large linear system forms a shape close to a block diagonal matrix. Then the complexity of this algorithm is $\mathcal{O}(n^{2\omega+1})$. This algorithm is much faster than the original one, but still inefficient for practical parameters. We obtain our algorithm by improving this one. Thus we will explain this algorithm in our language in Section 3.1.

Zhang and Tan's algorithm Recently, Zhang and Tan proposed [24] the algorithm that constructs the central map (F_1, F_2) and Ψ so that $\Psi := XF_1 + X^qF_2$ has degree D at most. Thus the algorithm requires only very little computation. In fact, the complexity is $\mathcal{O}(\log_q D)$. But this algorithm do not necessarily give all private ZHFE keys. Strictly speaking, if we define a linear map $\mathcal{L}$ over $\mathbb{F}$ on $\mathbb{K}^2$ by

$$\mathcal{L} : \mathbb{K}^2 \ni (X, Y) \mapsto (L_{00}(X) + L_{01}(Y), L_{10}(X) + L_{11}(Y)) \in \mathbb{K}^2, \qquad (4)$$

then this algorithm can generate all private keys with $\mathcal{L}$ nonsingular. $\mathcal{L}$ can be represented as a matrix in $M_{2n}(\mathbb{F})$ due to $\mathbb{K} = \mathbb{F}^n$. We stress that the corank of $\mathcal{L}$ is crucial for the efficient construction of private ZHFE keys, where the corank of $\mathcal{L}$ is $2n - \mathrm{Rank}\ \mathcal{L}$. In particular, if $\mathcal{L}$ is singular (corank of $\mathcal{L} \geq 1$), the private keys can not necessarily generate by this algorithm. To be more precise, assume that we have found polynomials F_1, F_2 such that $\Psi = XF_1 + X^qF_2$ is of degree less than or equal to D. In order to find another $\alpha'_1, \cdots, \alpha'_{2n}, \beta'_1, \cdots, \beta'_{2n} \in \mathbb{K}$ such that the corresponding polynomial

$$\Psi' = X\left(L'_{00}(F_1) + L'_{01}(F_2)\right) + X^q\left(L'_{10}(F_1) + L'_{11}(F_2)\right)$$

is of degree less than or equal to given D, then one needs to solve about n^2 equations in $4n$ variables $\alpha'_1, \cdots, \alpha'_{2n}, \beta'_1, \cdots, \beta'_{2n}$. For recommended parameters $(q = 7, D = 105, n = 55)$ one has a system of 3016 equations in 220 variables, which has at most one solution. Hence the linear map $\mathcal{L}'$ corresponding to the later private keys has corank 0. Hence, Zhang and Tan's algorithm does not work for corank $\mathcal{L} \geq 1$. The ratio of $\mathcal{L}$ with respect to the corank of $\mathcal{L}$ is given by Table 2 if the linear map $\mathcal{L}$ is randomly distributed in $M_{2n}(\mathbb{F})$ and n is enough large, for example $n \geq 35$.

3 Our new Key Generation Algorithm for ZHFE

In this section, we propose our new private key generation algorithm of ZHFE scheme. Here, we assume that n is odd, say $n = 2l + 1$, and $q > 2$. The reason why we assume n odd will be explained in Remark 1.

3.1 Baena et al.'s Algorithm

Since our new algorithm is obtained by improving Baena et al.'s one [2], we explain it here in our language.

Let F be an HFE polynomial. If F is a linear combination of $X^{q^{i-1}+q^{j-1}}$, $(1 \leq i, j \leq n)$ over $\mathbb{K}$, then it is called a quadratic HFE polynomial. For $1 \leq d \leq l+1$ and $1 \leq i \leq n$, set $X_{d,i} := X^{q^{i-1}+q^{i-1+d-1}}$.

Proposition 1 ([2, Section 3.1]). *Every quadratic HFE polynomial F can be uniquely written as*

$$F = \sum_{1 \leq d \leq l+1} \sum_{1 \leq i \leq n} a_{d,i} X_{d,i}, \quad (a_{d,i} \in \mathbb{K}).$$

In Proposition 1, we call $a_{d,i}$ *the (d,i)-coefficient* of F, and write $F_{d,i} = a_{d,i}$.

We represent the two quadratic HFE polynomials F_1, F_2 of equation (1) according to Proposition 1 as follows:

$$F_1 = \sum_{1 \leq d \leq l+1} \sum_{1 \leq i \leq n} a_{d,i} X_{d,i}, \quad F_2 = \sum_{1 \leq d \leq l+1} \sum_{1 \leq i \leq n} b_{d,i} X_{d,i}. \tag{5}$$

Here the coefficients are to be determined. Randomly choose $4n$ scalars $\alpha_1, ..., \alpha_{2n}$, $\beta_1, ..., \beta_{2n}$ of $\mathbb{K}$ and set

$$\bar{F}_1 := L_{00}(F_1) + L_{01}(F_2), \quad \bar{F}_2 := L_{10}(F_1) + L_{11}(F_2), \tag{6}$$

where the L_{ij} is defined as in equation (2). Thus

$$\Psi = X\bar{F}_1 + X^q \bar{F}_2. \tag{7}$$

Our goal is to determine the coefficients $a_{d,i}, b_{d,i}$ of equation (5) such that $\deg \Psi \leq D$.

First we compute the (d, i)-coefficients $\bar{F}_{1,d,i}, \bar{F}_{2,d,i}$ of the two quadratic HFE polynomials $\bar{F}_1, \bar{F}_2$. For n scalars $z_1, z_2, ..., z_n \in \mathbb{K}$, we define an $n \times n$ matrix by

$$L_1(z_1, z_2, ..., z_n) := (z_{j-i+1}^{q^{i-1}})_{i,j} = \begin{pmatrix} z_1 & z_2 & z_3 & \cdots & z_n \\ z_n^q & z_1^q & z_2^q & \cdots & z_{n-1}^q \\ z_{n-1}^{q^2} & z_n^{q^2} & z_1^{q^2} & \cdots & z_{n-2}^{q^2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ z_2^{q^{n-1}} & z_3^{q^{n-1}} & z_4^{q^{n-1}} & \cdots & z_1^{q^{n-1}} \end{pmatrix}. \tag{8}$$

Here $j - i + 1$ is calculated modulo n. By using this notation, we can represent the n-tuple $(\bar{F}_{i,d,1}, \bar{F}_{i,d,2}, ..., \bar{F}_{i,d,n})$ as follows:

Lemma 1 ([2, Corollary 1]). (i) *For any d, we have*

$$(\bar{F}_{1,d,1}, \bar{F}_{1,d,2}, ..., \bar{F}_{1,d,n}) = (\alpha_1, ..., \alpha_n) \cdot L_1(a_{d,1}, ..., a_{d,n}) + (\alpha_{n+1}, ..., \alpha_{2n}) \cdot L_1(b_{d,1}, ..., b_{d,n}). \tag{9}$$

(ii) *For any d, we have*

$$(\bar{F}_{2,d,1}, \bar{F}_{2,d,2}, ..., \bar{F}_{2,d,n}) = (\beta_1, ..., \beta_n) \cdot L_1(a_{d,1}, ..., a_{d,n}) + (\beta_{n+1}, ..., \beta_{2n}) \cdot L_1(b_{d,1}, ..., b_{d,n}). \tag{10}$$

Lemma 2 ([2, Lemma 1]).

$$X \cdot X_{d,i} = X^q \cdot X_{d',i'} \iff \begin{cases} d = d' = l + 1, i = 2, i' = l + 2, \\ d' = d - 1, i = i' = n + 3 - d, (2 \leq d \leq l + 1), \\ d' = d + 1, i = 2, i' = 1, (1 \leq d \leq l). \end{cases} \tag{11}$$

If $X \cdot X_{d,i} = X^q \cdot X_{d',i'}$ then we write $(d, i) \rightsquigarrow (d', i')$. By this lemma, we can describe the conditions for F_1, F_2 so that $\deg \Psi \leq D$.

Corollary 1. *If the coefficients $a_{d,i}, b_{d,i}$ of F_1, F_2 satisfy the following three conditions, then we have $\deg \Psi \leq D$.*

(i) $\bar{F}_{1,d,i} = -\bar{F}_{2,d',i'}$ *for any $(d, i) \rightsquigarrow (d', i')$ such that* $\deg X \cdot X_{d,i} > D$.
(ii) $\bar{F}_{1,d,i} = 0$ *if (d, i) is not in Lemma 2 and satisfies* $\deg X \cdot X_{d,i} > D$.
(iii) $\bar{F}_{2,d',i'} = 0$ *if (d', i') is not in Lemma 2 and satisfies* $\deg X^q \cdot X_{d',i'} > D$.

Proof. By Lemmas 1 and 2, it is easy to compute the coefficients of degree $> D$ in Ψ. Then the conditions of F_1, F_2 so that $\deg \Psi \leq D$ are equivalent to (i),(ii) and (iii).

Note that $\deg(X \cdot X_{d,i}) = 1 + q^{i-1} + q^{(i-1+d-1 \mod n)}$. Also $\deg(X^q \cdot X_{d',i'}) = q + q^{i'-1} + q^{(i'-1+d'-1 \mod n)}$.

Finally, it follows from Lemma 1 and Corollary 1 that:

Theorem 1. *Randomly choose $4n$ scalars $\alpha_1, ..., \beta_{2n}$ of $\mathbb{K}$. Also we take any scalars $c_{j,d,i} \in \mathbb{K}$, $(1 \leq j \leq 2, 1 \leq d \leq l+1, 1 \leq i \leq n)$ with the assumptions (i),(ii),(iii) in Corollary 1. If $a_{d,i}$ and $b_{d,i}$ are solutions of equations*

$$\begin{aligned}(c_{1,d,1}, c_{1,d,2}, ..., c_{1,d,n}) &= (\alpha_1, ..., \alpha_n) \cdot L_1(a_{d,1}, ..., a_{d,n}) \\ &\quad + (\alpha_{n+1}, ..., \alpha_{2n}) \cdot L_1(b_{d,1}, ..., b_{d,n}) \qquad (A_d), \\ (c_{2,d,1}, c_{2,d,2}, ..., c_{2,d,n}) &= (\beta_1, ..., \beta_n) \cdot L_1(a_{d,1}, ..., a_{d,n}) \\ &\quad + (\beta_{n+1}, ..., \beta_{2n}) \cdot L_1(b_{d,1}, ..., b_{d,n}) \qquad (B_d),\end{aligned}$$

for any $1 \leq d \leq l+1$, then F_1, F_2 satisfy that $\deg \Psi \leq D$. Also we have

$$\Psi = \sum_{1 \leq d \leq l+1} \left(\sum_{1 \leq i \leq n} c_{1,d,i} X \cdot X_{d,i} + \sum_{1 \leq i \leq n} c_{2,d,i} X^q \cdot X_{d,i} \right).$$

The equations $(A_d), (B_d)$ in Theorem 1 are not linear systems in $a_{d,i}, b_{d,i}$. They can be reduced to linear systems over the small field $\mathbb{F}$. Baena et al. [2] obtained F_1, F_2 and Ψ by solving such linear systems over the small field $\mathbb{F}$. Our strategy in obtaining F_1, F_2 and Ψ is to lift equations $(A_d), (B_d)$ to linear systems over the big field $\mathbb{K}$; cf. Section 3.2.

3.2 Main Idea

Here we explain the main idea of the proposed algorithm for efficient private key generation of ZHFE scheme.

For any $1 \leq d \leq l+1$, set

$$x_{d,i} := a_{d,n+2-i}^{q^{i-1}}, \quad y_{d,i} := b_{d,n+2-i}^{q^{i-1}}.$$

Then

$$a_{d,i} = x_{d,n+2-i}^{q^{i-1}}, \quad b_{d,i} = y_{d,n+2-i}^{q^{i-1}}.$$

Also we have

$$L_1(a_{d,1}, a_{d,2}, ..., a_{d,n}) = \begin{pmatrix} x_{d,1} & x_{d,n}^{q} & x_{d,n-1}^{q^2} & \cdots & x_{d,2}^{q^{n-1}} \\ x_{d,2} & x_{d,1}^{q} & x_{d,n}^{q^2} & \cdots & x_{d,3}^{q^{n-1}} \\ x_{d,3} & x_{d,2}^{q} & x_{d,1}^{q^2} & \cdots & x_{d,4}^{q^{n-1}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{d,n} & x_{d,n-1}^{q} & x_{d,n-2}^{q^2} & \cdots & x_{d,1}^{q^{n-1}} \end{pmatrix}.$$

By using these, the equation (A_d) is equivalent to the following (A'_d):

$$(c_{1,d,1}, c_{1,d,2}, ..., c_{1,d,n}) = (\alpha_1, ..., \alpha_n, \alpha_{n+1}, ..., \alpha_{2n}) \begin{pmatrix} x_{d,1} & x_{d,n}^q & \cdots & x_{d,2}^{q^{n-1}} \\ x_{d,2} & x_{d,1}^q & \cdots & x_{d,3}^{q^{n-1}} \\ \vdots & \vdots & \ddots & \vdots \\ x_{d,n} & x_{d,n-1}^q & \cdots & x_{d,1}^{q^{n-1}} \\ y_{d,1} & y_{d,n}^q & \cdots & y_{d,2}^{q^{n-1}} \\ y_{d,2} & y_{d,1}^q & \cdots & y_{d,3}^{q^{n-1}} \\ \vdots & \vdots & \ddots & \vdots \\ y_{d,n} & y_{d,n-1}^q & \cdots & y_{d,1}^{q^{n-1}} \end{pmatrix},$$

which is equivalent to the following equation (A''_d):

$$(c_{1,d,1}, c_{1,d,2}^{q^{n-1}}, ..., c_{1,d,n}^q) = (x_{d,1}, ..., x_{d,n}, y_{d,1}, ..., y_{d,n}) \begin{pmatrix} \alpha_1 & \alpha_2^{q^{n-1}} & \cdots & \alpha_n^q \\ \alpha_2 & \alpha_3^{q^{n-1}} & \cdots & \alpha_1^q \\ \alpha_3 & \alpha_4^{q^{n-1}} & \cdots & \alpha_2^q \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_n & \alpha_1^{q^{n-1}} & \cdots & \alpha_{n-1}^q \\ \alpha_{n+1} & \alpha_{n+2}^{q^{n-1}} & \cdots & \alpha_{2n}^q \\ \alpha_{n+2} & \alpha_{n+3}^{q^{n-1}} & \cdots & \alpha_{n+1}^q \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{2n} & \alpha_{n+1}^{q^{n-1}} & \cdots & \alpha_{2n-1}^q \end{pmatrix}.$$

Remark 1. If we assume that n is even, then we can not obtain a linear system as above. In fact, if n is even, then in the case $d = n/2 + 1$, $x_{d,i}^{q^{j-1}}$ and $x_{d,i}^{q^{j-1+n/2}}$ appear on each j-column in the matrix in (A'_d). Thus we can not have a linear system as the linear system $(A''_{n/2+1})$. That is the reason why we consider n odd in this paper.

Similarly, the equation (B_d) is equivalent to the following equation (B''_d):

$$(c_{2,d,1}, c_{2,d,2}^{q^{n-1}}, ..., c_{2,d,n}^q) = (x_{d,1}, ..., x_{d,n}, y_{d,1}, ..., y_{d,n}) \begin{pmatrix} \beta_1 & \beta_2^{q^{n-1}} & \cdots & \beta_n^q \\ \beta_2 & \beta_3^{q^{n-1}} & \cdots & \beta_1^q \\ \beta_3 & \beta_4^{q^{n-1}} & \cdots & \beta_2^q \\ \vdots & \vdots & \ddots & \vdots \\ \beta_n & \beta_{n+1}^{q^{n-1}} & \cdots & \beta_{n-1}^q \\ \beta_{n+1} & \beta_{n+2}^{q^{n-1}} & \cdots & \beta_{2n}^q \\ \beta_{n+2} & \beta_{n+3}^{q^{n-1}} & \cdots & \beta_{n+1}^q \\ \vdots & \vdots & \ddots & \vdots \\ \beta_{2n} & \beta_{n+1}^{q^{n-1}} & \cdots & \beta_{2n-1}^q \end{pmatrix}.$$

For n scalars $z_1, z_2, ..., z_n$ of $\mathbb{K}$, define an $n \times n$ matrix by

$$L_2 \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{pmatrix} := (z_i^{q^{n-j+1}})_{i,j} = \begin{pmatrix} z_1 & z_2^{q^{n-1}} & z_3^{q^{n-2}} & \cdots & z_n^q \\ z_2 & z_3^{q^{n-1}} & z_4^{q^{n-2}} & \cdots & z_1^q \\ z_3 & z_4^{q^{n-1}} & z_5^{q^{n-2}} & \cdots & z_2^q \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ z_n & z_1^{q^{n-1}} & z_2^{q^{n-2}} & \cdots & z_{n-1}^q \end{pmatrix}.$$

By using this notation, set

$$L := \begin{pmatrix} L_2 \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{pmatrix} & L_2 \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} \\ L_2 \begin{pmatrix} \alpha_{n+1} \\ \alpha_{n+2} \\ \vdots \\ \alpha_{2n} \end{pmatrix} & L_2 \begin{pmatrix} \beta_{n+1} \\ \beta_{n+2} \\ \vdots \\ \beta_{2n} \end{pmatrix} \end{pmatrix} \in M_{2n}(K). \tag{12}$$

Remark 2. It is easy to prove that $\operatorname{Rank} L = \operatorname{Rank} \mathcal{L}$, where $\mathcal{L}$ is defined in Section 2.3.

Now we can restate Theorem 1 by using this L as follows:

Theorem 2. *Randomly choose $4n$ scalars $\alpha_1, ..., \beta_{2n}$ of $\mathbb{K}$. Also we take any scalars $c_{j,d,i} \in \mathbb{K}$ $(1 \leq j \leq 2, 1 \leq d \leq l+1, 1 \leq i \leq n)$ with the assumptions (i),(ii),(iii) in Corollary 1. Let $x_{d,i}$ and $y_{d,i}$ be solutions of the linear system*

$$(c_{1,d,1}, c_{1,d,2}^{q^{n-1}}, ..., c_{1,d,n}^q, c_{2,d,1}, c_{2,d,2}^{q^{n-1}}, ..., c_{2,d,n}^q) = (x_{d,1}, ..., x_{d,n}, y_{d,1}, ..., y_{d,n}) \cdot L \tag{$\star$}$$

for any $1 \leq d \leq l+1$. If we set

$$F_1 = \sum_{1 \leq d \leq l+1} \sum_{1 \leq i \leq n} x_{d,n+2-i}^{q^{i-1}} X_{d,i}, \quad F_2 = \sum_{1 \leq d \leq l+1} \sum_{1 \leq i \leq n} y_{d,n+2-i}^{q^{i-1}} X_{d,i},$$

then F_1, F_2 satisfy $\deg \Psi \leq D$. Also we have

$$\Psi = \sum_{1 \leq d \leq l+1} \left(\sum_{1 \leq i \leq n} c_{1,d,i} X \cdot X_{d,i} + \sum_{1 \leq i \leq n} c_{2,d,i} X^q \cdot X_{d,i} \right).$$

Proof. Solving (A_d) is equivalent to solving (A''_d). Similarly, solving (B_d) is equivalent to solving (B''_d). Also solving (A''_d) and (B''_d) is equivalent to solving $(\star)$. Thus we have the theorem.

Thus we can reduce the equations $(A_d), (B_d)$ in Theorem 1 to the linear system $(\star)$ over the big field $\mathbb{K}$.

3.3 Our Proposed Algorithm

Here, we explain an algorithm to solve the linear systems in Theorem 2. This is the our new algorithm to generate F_1, F_2 and Ψ; see Algorithm 2 in Appendix A for overview of our algorithm in this section.

Set

$$c_{d,i} := c_{1,d,i}, \quad c_{d,n+i} := c_{2,d,i} \quad \text{for } d, i.$$

Take a sequence $1 \le i_1 < i_2 < \cdots < i_{m-1} < i_m \le 2n$, where $1 \le m \le 2n$. We denote by $L[i_1, i_2, ..., i_m]$ the $2n \times m$ matrix that is obtained by leaving each i_j-column of L. Similarly, we define

$$(c_1, c_2, ..., c_n, c_{n+1}, c_{n+2}, ..., c_{2n})[i_1, i_2, ..., i_m] := (c_{i_1}, c_{i_2}, , ..., c_{i_m}).$$

Now, we explain our algorithm that gives solutions of the linear systems $(\star)$ for well chosen scalars $c_{d,i}$.

$\underline{d = l + 1}$

$$S'_{l+1} := \{i \mid 1 \le i \le n,\ \deg X \cdot X_{l+1,i} \le D\} \cup \{n+i' \mid 1 \le i' \le n,\ \deg X^q \cdot X_{l+1,i'} \le D\},$$

$$S_{l+1} := \{1, ..., 2n\} \smallsetminus (S'_{l+1} \cup \{l+3, n+1\}).$$

Randomly choose a scalar z in $\mathbb{K}$. For any $i \in S_{l+1}$, set

$$c_{l+1,i} := \begin{cases} z & \text{if } i = 2 \text{ and } 2 \in S_{l+1}, \\ -z & \text{if } i = n+l+2 \text{ and } 2 \in S_{l+1}, \\ 0 & \text{otherwise.} \end{cases}$$

Then we consider the following linear system:

$$\begin{aligned} &(c_{l+1,1}, c_{l+1,2}^{q^{n-1}}, ..., c_{l+1,n}^{q}, c_{l+1,n+1}, c_{l+1,n+2}^{q^{n-1}}, ..., c_{l+1,2n}^{q})[S_{l+1}] \\ &= (x_{l+1,1}, ..., x_{l+1,n}, y_{l+1,1}, ..., y_{l+1,n}) \cdot L[S_{l+1}]. \end{aligned}$$

Note that since the scalars $c_{l+1,i}$ $(i \notin S_{l+1})$ do not occur in this system, this system is well-defined. After we find a solution $(x_{l+1,1}, ..., x_{l+1,n}, y_{l+1,1}, ..., y_{l+1,n})$ of this system, the other scalars $c_{l+1,i}$, $(i \notin S_{l+1})$ are given by the formula

$$\begin{aligned} &(c_{l+1,1}, c_{l+1,2}^{q^{n-1}}, ..., c_{l+1,n}^{q}, c_{l+1,n+1}, c_{l+1,n+2}^{q^{n-1}}, ..., c_{l+1,2n}^{q}) \\ &= (x_{l+1,1}, ..., x_{l+1,n}, y_{l+1,1}, ..., y_{l+1,n}) \cdot L. \end{aligned}$$

$\underline{1 < d < l + 1}$

$$S'_d := \{i \mid 1 \le i \le n,\ \deg X \cdot X_{d,i} \le D\} \cup \{n+i' \mid 1 \le i' \le n,\ \deg X^q \cdot X_{d,i'} \le D\},$$

$$S_d := \{1, ..., 2n\} \smallsetminus (S'_d \cup \{(n + 2 - d \mod n) + 1, n + 1\}).$$

For any $i \in S_d$, we set

$$c_{d,i} := \begin{cases} -c_{d+1,n+1} & \text{if } i = 2 \text{ and } 2 \in S_d, \\ -c_{d+1,n+2-d} & \text{if } i = 2n+2-d \text{ and } 2n+2-d \in S_d, \\ 0 & \text{otherwise.} \end{cases}$$

Then we consider the following linear system:

$$\begin{aligned} &(c_{d,1}, c_{d,2}^{q^{n-1}}, ..., c_{d,n}^{q}, c_{d,n+1}, c_{d,n+2}^{q^{n-1}}, ..., c_{d,2n}^{q})[S_d] \\ &= (x_{d,1}, ..., x_{d,n}, y_{d,1}, ..., y_{d,n}) \cdot L[S_d]. \end{aligned}$$

After we find a solution $(x_{d,1}, ..., x_{d,n}, y_{d,1}, ..., y_{d,n})$ of this system, the other scalars $c_{d,i}$, $(i \notin S_d)$ are given by the formula

$$\begin{aligned} &(c_{d,1}, c_{d,2}^{q^{n-1}}, ..., c_{d,n}^{q}, c_{d,n+1}, c_{d,n+2}^{q^{n-1}}, ..., c_{d,2n}^{q}) \\ &= (x_{d,1}, ..., x_{d,n}, y_{d,1}, ..., y_{d,n}) \cdot L. \end{aligned}$$

$\underline{d = 1}$

$$S'_1 := \{i \mid 1 \leq i \leq n,\ \deg X \cdot X_{1,i} \leq D\} \cup \{n+i' \mid 1 \leq i' \leq n,\ \deg X^q \cdot X_{1,i'} \leq D\},$$

$$S_1 := \{1, ..., 2n\} \smallsetminus S'_1.$$

For any $i \in S_1$, we set

$$c_{1,i} := \begin{cases} -c_{2,n+1} & \text{if } i = 2 \text{ and } 2 \in S_1, \\ -c_{2,1} & \text{if } i = n+1 \text{ and } n+1 \in S_1, \\ 0 & \text{otherwise.} \end{cases}$$

Then we consider the following linear system:

$$\begin{aligned} &(c_{1,1}, c_{1,2}^{q^{n-1}}, ..., c_{1,n}^{q}, c_{1,n+1}, c_{1,n+2}^{q^{n-1}}, ..., c_{1,2n}^{q})[S_1] \\ &= (x_{1,1}, ..., x_{1,n}, y_{1,1}, ..., y_{1,n}) \cdot L[S_1]. \end{aligned}$$

After we find a solution $(x_{1,1}, ..., x_{1,n}, y_{1,1}, ..., y_{1,n})$ of this system, the other scalars $c_{1,i}$, $(i \notin S_1)$ are given by the formula

$$\begin{aligned} &(c_{1,1}, c_{1,2}^{q^{n-1}}, ..., c_{1,n}^{q}, c_{1,n+1}, c_{1,n+2}^{q^{n-1}}, ..., c_{1,2n}^{q}) \\ &= (x_{1,1}, ..., x_{1,n}, y_{1,1}, ..., y_{1,n}) \cdot L. \end{aligned}$$

Finally, we have two quadratic HFE polynomials F_1, F_2 and Ψ such that $\deg \Psi \leq D$:

$$F_1 = \sum_{1 \leq d \leq l+1} \sum_{1 \leq i \leq n} x_{d,n+2-i}^{q^{i-1}} X_{d,i}, \quad F_2 = \sum_{1 \leq d \leq l+1} \sum_{1 \leq i \leq n} y_{d,n+2-i}^{q^{i-1}} X_{d,i},$$

$$\Psi = \sum_{1 \leq d \leq l+1} \left(\sum_{i \in S'_d, i \leq n} c_{d,i} X \cdot X_{d,i} + \sum_{i \in S'_d, i > n} c_{d,i} X^q \cdot X_{d,i-n} \right).$$

```
Algorithm 1: Generating Matrix $L$ of Corank $r$ (Section 4.2)
    input  : a fielf $\mathbb{F}$ with $q$ elements, integers $n$ and $r$, $\mathbb{K}$ the extension field of
             degree $n$ over $\mathbb{F}$, an $\mathbb{F}$-basis $(\theta_1, ..., \theta_n)$ of $\mathbb{K}$
    output: $\alpha_1, ..., \alpha_{2n}, \beta_1, ..., \beta_{2n}$, $L$ with corank $r$ (See (12) for $L$)
    $M \leftarrow (\theta_i^{q^{j-1}})_{1 \le i,j \le n}$;
    $A, B \leftarrow Random(GL_{2n}(\mathbb{F}))$;
    $L' \leftarrow \begin{pmatrix} M^{-1} & \\ & M^{-1} \end{pmatrix} \cdot A \cdot \begin{pmatrix} 1_{2n-r} & \\ & 0_r \end{pmatrix} \cdot B \cdot \begin{pmatrix} M & \\ & M \end{pmatrix}$,  $2n \times 2n$ matrix;
    $\alpha_i \leftarrow L'_{i,1}$  : the $(i,1)$-entry of $L'$, $(1 \le i \le 2n)$;
    $\beta_i \leftarrow L'_{i,n+1}$  $(1 \le i \le 2n)$;
    $L \leftarrow (L'_1, L'_n, L'_{n-1}, ..., L'_2, L'_{n+1}, L'_{2n}, L'_{2n-1}, ..., L'_{n+2})$;
    where $L'_i$ is the $i$-th column of $L'$
```

Remark 3. If corank $L \le 2$, then each $L[S_d]$ has the full rank. Thus all the above linear systems have solutions. Therefore, if corank $L \le 2$, then our algorithm terminates. Also if corank $L \ge 3$, then our algorithm failed in our experiments. But Table 2 implies that the class of L with corank ≥ 3 is very small in total. Thus we may take L to be corank $L \le 2$. Notice that Baena et al.'s algorithm [2] succeeds for corank $L \le 5$. For L with higher corank, their algorithm also works, but produces $\Psi = 0$ making the corresponding ZHFE scheme insure under linearization attack. However, it suffices to only consider linear maps L of corank less than 3 for their majority, cf. Table 2.

4 Complexity and Implementation Results

In this section, we give the complexity and implementation results for our private key generation algorithm of ZHFE scheme.

4.1 The Complexity of the Proposed Algorithm

We can easily prove the complexity of our proposed algorithm (Algorithm 2) discussed in Section 3.3 in the following theorem.

Theorem 3. *The complexity of our algorithm in Section 3.3 is given by* $\mathcal{O}(n^{\omega+3})$.

Proof. In our algorithm proposed in Section 3.3, we obtain a private ZHFE key by solving $l+1$ linear systems over the big field $\mathbb{K}$. Here each linear system has at most $2n$ varibles and at most $2n$ equations. Thus the complexity is

$$(l+1) \times (2n)^\omega \times (\log q^n)^2 = \mathcal{O}(n^{\omega+3}). \qquad \square$$

Thus our algorithm improves the original algorithm of $\mathcal{O}(n^{3\omega})$ and Baena et al.'s algorithm of $\mathcal{O}(n^{2\omega+1})$ (See Table 1).

Table 3. The comparison of timings between Baena et al.'s algorithm [2] and our algorithm.

			Our algorithm		Baena et al.'s	
q	D	n	CPU time [s]	Max Memory [MB]	CPU time [s]	Max Memory [MB]
7	105	15	0.09	10	0.59	11
7	105	31	3.47	11	22.27	43
7	**105**	**55**	**39.19**	**18**	**607.06**	**338**
17	105	15	0.13	9	3.06	14
17	105	31	3.91	11	348.57	81
17	**595**	**55**	**62.91**	**22**	**15350.79**	**683**

4.2 Our Experiments

In order to perform the experiments of generating the private ZHFE keys, we need to decide the matrix $L \in M_{2n}(\mathbb{K})$ in equation (12), where L is generated by $\alpha_1, ..., \alpha_{2n}, \beta_1, ..., \beta_{2n}$ in equation (2). Note that our proposed algorithm works only for matrices L with corank $0, 1$ and 2 (cf. Remark 3), and thus we have to investigate how to generate such a matrix. In the following we describe an algorithm for generating the matrix L of any corank $0 \leq r \leq 2n$. For the matrix $\begin{pmatrix} 1_{2n-r} & \\ & 0_r \end{pmatrix}$ in $M_{2n}(\mathbb{K})$ of corank r, we multiply random invertible matrices $A, B \in GL_{2n}(\mathbb{F})$ and matrices $\begin{pmatrix} M^{-1} & \\ & M^{-1} \end{pmatrix}, \begin{pmatrix} M & \\ & M \end{pmatrix}$ from both sides. The resulting matrix of corank r implies $\alpha_1, ..., \alpha_{2n}, \beta_1, ..., \beta_{2n}$ used for the private ZHFE keys. The explicit algorithm is described in Algorithm 1.

On the other hand, as can be seen in Table 2, in most cases the corank L is 0 or 1 for randomly chosen $\alpha_1, ..., \alpha_{2n}, \beta_1, ..., \beta_{2n}$. Note that the corank of L is equal to the corank $\mathcal{L}$ in equation (4) (cf. Remark 2). Therefore if we generate the matrix L by Algorithm 1 with $r \leq 2$, then we can generate almost all instances L for the key generation algorithms of ZHFE scheme.

4.3 Comparison of Timings

The implementation results and the comparison between our algorithm and Baena et al.'s algorithm [2] are presented in Table 3. All the experiments in this section were performed using Magma V2.20-10 [4] with a processor Intel(R) Core(TM) i5-4300U CPU @ 1.90GHz, running Windows 7 Professional SP1.

Notice that according to the estimation of Zhang and Tan [24], the parameters ($q = 7, n = 55, D = 105$), which is recommended in the original paper [20], and ($q = 17, n = 55, D = 595$) are for 96-bit and 111-bit security level respectively. In Table 3 we present the timings of our experiments using these parameters and in addition we run experiments under other parameters $n = 15, 31$.

Our algorithm in Table 3 presents timing to generate a private ZHFE key, that is, $\alpha_1, ..., \alpha_{2n}$, $\beta_1, ..., \beta_{2n}$, L, F_1, F_2 and Ψ. Here, we used Algorithm 1 to generate $\alpha_1, ..., \alpha_{2n}, \beta_1, ..., \beta_{2n}$ and L with corank $L \leq 2$. For example, for the recommended parameters $(q = 7, D = 105, n = 55)$ at 96-bit security, our algorithm takes 39.19 seconds and the max memory is 18 mega bytes.[3] For comparison, we also present timing to generate such a private ZHFE key by Baena et al.'s algorithm. For the recommended parameters $(q = 7, D = 105, n = 55)$, our algorithm is around 15 times faster than that of Baena et al. [2].

5 Conclusion

In this paper, we proposed a new efficient algorithm for generating private keys of the ZHFE scheme [20]. Our algorithm has complexity $O(n^{\omega+3})$ which improves the original [19] and Baena's [2] algorithm whose complexities are $O(n^3)$ and $O(n^{2\omega+1})$ respectively. Here n is the number of variables and $2 < \omega < 3$ is a linear algebra constant. Our algorithm is in practice very fast compared to that of Baena et al.: for recommended parameter $(q = 7, n = 55, D = 105)$ at 96-bit security, our algorithm is around 15 times faster than that of Baena et al.; cf. Table 3. Moreover, in contrast to Zhang and Tan's algorithm [24], our algorithm generates as many private ZHFE keys as the previous ones [20,2], as estimated in Table 2. Although our algorithm works for linear maps L with corank $L \leq 2$ (cf. Remark 3), it already generates around 99% private keys in total (cf. Table 2). This makes our algorithm to be the most appropriate for generating private ZHFE keys.

Acknowledgments

This work was supported by CREST, JST. The second author also acknowledges the Japanese Society for the Promotion of Science (JSPS) for financial support under grant KAKENHI 16K17644.

References

1. D.J. Bernstein, J. Buchmann, E. Dahmen: Post-Quantum Cryptography. Springer, 2009.
2. J. B. Baena, D. Cabarcas, D. E. Escudero, J. Porras-Barrera, J. A. Verbel: Efficient ZHFE Key Generation. PQCrypto 2016, LNCS, vol. 9606, pp. 213–232. Springer 2016.
3. A. Bogdanov, T. Eisenbarth, A. Rupp, C. Wolf. Time-area optimized public-key engines: MQ-cryptosystems as replacement for elliptic curves. CHES 2008, LNCS, vol. 5154, pp. 45–61. Springer, 2008.
4. W. Bosma, J. Cannon, C. Playoust: The Magma algebra system. I. The user language. J. Symbolic Comput. 24, 3-4 (1997), pp. 235–265.

[3] Here we used the Magma's command `GetMaximumMemoryUsage` to measure max memory. Note also that we used the Magma's command `Solution` to solve linear systems in the algorithm.

5. A.I.T. Chen, M.-S. Chen, T.-R. Chen, C.-M. Cheng, J. Ding, E. L.-H. Kuo, F. Y.-S. Lee, B.-Y. Yang. SSE implementation of multivariate PKCs on modern x86 cpus. CHES 2009, LNCS, vol. 5747, pp. 33 - 48. Springer, 2009. The user language. J. Symbolic Comput. 24, 3-4 (1997), pp. 235–265.
6. N. Courtois: The security of hidden field equations (HFE). In Naccache, C., editor, Progress in cryptology, CT-RSA, LNCS, vol. 2020, pp. 266–281. Springer 2001.
7. J. Ding, J. E. Gower, D. S. Schmidt: Multivariate Public Key Cryptosystems. Springer, 2006.
8. J. Ding, D. S. Schmidt: Rainbow, a new multivariate polynomial signature scheme. ACNS 2005, LNCS, vol. 3531, pp. 164-175. Springer 2005.
9. D. Goodin: NSA preps quantum-resistant algorithms to head off crypto-apocalypse. `http://arstechnica.com/security/2015/08/nsa-preps-quantum-resistant-algorithms-to-head-off-crypto-apocolypse/`.
10. M. R. Garey and D. S. Johnson: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman and Company 1979.
11. A. Kipnis, L. Patarin, L. Goubin: Unbalanced Oil and Vinegar Schemes. EUROCRYPT 1999, LNCS, vol. 1592, pp. 206–222. Springer 1999.
12. A. Kipnis, A. Shamir: Cryptanalysis of the HFE public key cryptosystem by relinearization. Advances in Cryptology – CRYPTO'99, LNCS, vol. 1666, pp. 19–30, Springer 1999.
13. T. Matsumoto, H. Imai: Public quadratic polynomial-tuples for efficient signature-verification and message-encryption. EUROCRYPT 1988. LNCS, vol. 330, pp. 419–453. Springer 1988.
14. National Institute of Standards and Technology: Report on Post Quantum Cryptography, NISTIR draft 8105. `http://csrc.nist.gov/publications/drafts/nistir-8105/nistir_8105_draft.pdf`.
15. J. Patarin: Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocrypt 88. CRYPTO 1995, LNCS, vol. 963, pp. 248–261. Springer, 1995.
16. J. Patarin: Hidden Field Equations (HFE) and Isomorphisms of Polynomials (IP): Two New Families of Asymmetric Algorithms. EUROCRYPT, LNCS, vol. 1070, pp. 33–48. Springer, 1996.
17. R. Perlner, D. Smith-Tone: Security Analysis and Key Modification for ZHFE. PQCrypto 2016, LNCS, vol. 9606, pp. 197 - 212. Springer 2016.
18. A. Petzoldt, M.S. Chen, B.Y. Yang, C. Tao, J. Ding: Design Principles for HFEv-based Signature Schemes. ASIACRYPT 2015 - Part I, LNCS, vol. 9452, pp. 311–334. Springer 2015.
19. J. Porras, J. Baena, J. Ding: New candidates for multivariate trapdoor functions. Cryptology ePrint Archive, Report 2014/387, 2014.
20. J. Porras, J. Baena, J. Ding: ZHFE, a New Multivariate Public Key Encryption Scheme. PQCrypto 2014, LNCS, vol. 8772, pp.229–245. Springer 2014.
21. P. Shor: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer, SIAM J. Comput. 26 (5), pp. 1484–1509 (1997).
22. C. Tao, A. Diene, S. Tang, J. Ding: Simple matrix scheme for encryption. PQCrypto 2013, LNCS, vol. 7932, pp. 231–242, Springer 2013.
23. A. Szepieniec, J. Ding, B. Preneel: Extension Field Cancellation: a New Central Trapdoor for Multivariate Quadratic Systems. PQCrypto 2016, LNCS, vol. 9606, pp. 182–196, Springer 2016.
24. W. Zhang and C. H. Tan: On the Security and Key Generation of the ZHFE Encryption Scheme. IWSEC 2016, LNCS, vol. 9836, pp. 289–304. Springer 2016.
25. T. Yasuda, K. Sakurai: A multivariate encryption scheme with Rainbow. ICICS 2015, LNCS, vol. 9543, pp. 222–236, Springer 2016.

A Our algorithm in Section 3.3

The expression $f \xleftarrow{R} W$ denotes that f is an element chosen uniformly at random from the set W.

Algorithm 2: Our Proposed Algorithm (Section 3.3)

```
input  : $\mathbb{F}$: field with $q$ elements, $n = 2l + 1$: odd integer, $\mathbb{K}$: extension field of degree $n$,
         $L$: the $2n \times 2n$ matrix chosen by Algorithm 1, $D$: interger
output: $F_1, F_2, \Psi$: private key
$c_d \leftarrow (0, 0, ..., 0)$, length $2n$, $1 \leq d \leq l + 1$;
if $2 \in S_{l+1}$ then
    $c_{l+1,2} \leftarrow Random(\mathbb{K})$;
    $c_{l+1,n+l+2} \leftarrow -c_{l+1,2}$;
$c'_{l+1} \leftarrow (c_{l+1,1}, c_{l+1,2}^{q^{n-1}}, c_{l+1,3}^{q^{n-2}}, ..., c_{l+1,n}^{q}, c_{l+1,n+1}, c_{l+1,n+2}^{q^{n-1}}, ..., c_{l+1,2n}^{q})$;
$f \xleftarrow{R} W := \{f \in \mathbb{K}^{2n} \mid f \cdot L[S_{l+1}] = c'_{l+1}[S_{l+1}]\}$;
$g \leftarrow f \cdot L$;
$c_{l+1} \leftarrow (g_1, g_2^q, ..., g_n^{q^{n-1}}, g_{n+1}, g_{n+2}^q, ..., g_{2n}^{q^{n-1}})$;
$x_{l+1} \leftarrow (f_1, ..., f_n)$,   $y_{l+1} \leftarrow (f_{n+1}, ..., f_{2n})$;
$d \leftarrow l$;
while $d > 1$ do
    if $2 \in S_d$ then
        $c_{d,2} \leftarrow -c_{d+1,n+1}$;
    if $2n + 2 - d \in S_d$ then
        $c_{d,2n+2-d} \leftarrow -c_{d+1,n+2-d}$;
    $c'_d \leftarrow (c_{d,1}, c_{d,2}^{q^{n-1}}, c_{d,3}^{q^{n-2}}, ..., c_{d,n}^{q}, c_{d,n+1}, c_{d,n+2}^{q^{n-1}} ..., c_{d,2n}^{q})$;
    $f \xleftarrow{R} W := \{f \in \mathbb{K}^{2n} \mid f \cdot L[S_d] = c'_d[S_d]\}$;
    $g \leftarrow f \cdot L$;
    $c_d \leftarrow (g_1, g_2^q, ..., g_n^{q^{n-1}}, g_{n+1}, g_{n+2}^q, ..., g_{2n}^{q^{n-1}})$;
    $x_d \leftarrow (f_1, ..., f_n)$,   $y_d \leftarrow (f_{n+1}, ..., f_{2n})$;
    $d \leftarrow d - 1$;
if $2 \in S_1$ then
    $c_{1,2} \leftarrow -c_{2,n+1}$;
if $n + 1 \in S_1$ then
    $c_{1,n+1} \leftarrow -c_{2,1}$;
$c'_1 \leftarrow (c_{1,1}, c_{1,2}^{q^{n-1}}, c_{1,3}^{q^{n-2}}, ..., c_{1,n}^{q}, c_{1,n+1}, c_{1,n+2}^{q^{n-1}} ..., c_{1,2n}^{q})$;
$f \xleftarrow{R} W := \{f \in \mathbb{K}^{2n} \mid f \cdot L[S_1] = c'_1[S_1]\}$;
$g \leftarrow f \cdot L$;
$c_1 \leftarrow (g_1, g_2^q, ..., g_n^{q^{n-1}}, g_{n+1}, g_{n+2}^q, ..., g_{2n}^{q^{n-1}})$;
$x_1 \leftarrow (f_1, ..., f_n)$,   $y_1 \leftarrow (f_{n+1}, ..., f_{2n})$;
$F_1 \leftarrow \sum_{1 \leq d \leq l+1} \sum_{1 \leq i \leq n} x_{d,n+2-i}^{q^{i-1}} X_{d,i}$;
$F_2 \leftarrow \sum_{1 \leq d \leq l+1} \sum_{1 \leq i \leq n} y_{d,n+2-i}^{q^{i-1}} X_{d,i}$;
$\Psi \leftarrow \sum_{1 \leq d \leq l+1} \left( \sum_{i \in S'_d, i \leq n} c_{d,i} X \cdot X_{d,i} + \sum_{i \in S'_d, i > n} c_{d,i} X^q \cdot X_{d,i-n} \right)$;
```