Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2013, Article ID 872962, 8 pages
http://dx.doi.org/10.1155/2013/872962

Research Article

Sieve Method for Polynomial Linear Equivalence

Baocang Wang[1,2] and Yupu Hu[1,3]

[1] *State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an 710071, China*
[2] *Guangxi Experiment Center of Information Science, Guilin University of Electronic Technology, Guilin, Guangxi 541004, China*
[3] *Guangxi Key Lab of Wireless Wide Band Communication and Signal Processing, Guilin University of Electronic Technology, Guilin 541004, China*

Correspondence should be addressed to Baocang Wang; bcwang79@aliyun.com

Received 5 August 2013; Accepted 2 November 2013

Academic Editor: Jacek Rokicki

We consider the polynomial linear equivalence (PLE) problem arising from the multivariate public key cryptography, which is defined as to find an invertible linear transformation $\mathscr{L}$ satisfying $\mathscr{P} = \mathscr{S} \circ \mathscr{L}$ for given nonlinear polynomial maps $\mathscr{P}$ and $\mathscr{S}$ over a finite field $\mathbb{F}_q$. Some cryptographic and algebraic properties of PLE are discussed, and from the properties we derive three sieves called multiplicative, differential, and additive sieves. By combining the three sieves, we propose a sieve method for the PLE problem. As an application of our sieve method, we show that it is infeasible to construct public key encryption schemes from the PLE problem.

1. Introduction

With the rapid development of information technology, privacy and authentication have become two important issues that we must resolve in communication networks. Public key cryptography is undoubtedly one of the most important tools to resolve both problems in the area of information and network security engineering. Tremendous efforts had been made to achieve more practical and efficient public key ciphers in the cryptographic literature [1]. However, we should note that only a small number of them survived the serious security scrutiny, amongst which are the two widelyused cryptosystems RSA based on the integer factorization problem [1] and ECC based on the discrete logarithm problem on elliptic curves over finite fields [1, 2]. However, there exist polynomial-time algorithms for factoring large integers and solving the discrete logarithm problems on any finite cyclic group [3, 4]. Therefore, RSA and ECC are at the risk of being totally broken by quantum algorithms if practical quantum computing devices are available. Based on the considerations, cryptographers began to construct some alternative postquantum (i.e., quantum-resistant) public key cryptosystems from other mathematically intractable problems, especially those proven NP-complete or NP-hard problems.

Multivariate public key cryptography (MPKC) is an important kind of postquantum public key ciphers [5]. The security of MPKC resides in the proven fact that it is NP-hard to solve a random system of nonlinear equations over finite fields [6]. MPKC was once considered very attractive and interesting also due to its high speed in key generation, encryption, and decryption, easy implementation on both hardware and software, and its simple mathematical description [5]. In a multivariate public key cryptosystem, we first define a nonlinear easy-to-invert map $\mathscr{S}$ called central map, then we randomly choose two invertible affine transformations $\mathscr{L}_1$ and $\mathscr{L}_2$, and finally we publish the nonlinear map $\mathscr{P} = \mathscr{L}_2 \circ \mathscr{S} \circ \mathscr{L}_1$ as the public key and keep $\mathscr{L}_1$, $\mathscr{L}_2$, and $\mathscr{S}$ as the secret key. Sometimes the central map $\mathscr{S}$ may have a very special structure, which makes it useless to keep the central map $\mathscr{S}$ secret. One important problem in MPKC is the problem of isomorphism of polynomials (IP) [7–15]; namely, given nonlinear maps $\mathscr{S}$ and $\mathscr{P}$, find two invertible linear transformations $\mathscr{L}_1$ and $\mathscr{L}_2$ such that $\mathscr{P} = \mathscr{L}_2 \circ \mathscr{S} \circ \mathscr{L}_1$. The IP problem lies at the core of MPKC in which many multivariate cryptosystems were constructed based on

the assumed intractability of IP problem [16–22]. The IP problem is widely believed as an intractable problem, and known algorithms for the IP problem achieve exponential complexity [7–15].

In the IP problem, if $\mathscr{L}_2$ is known or equal to the identity transformation, the IP problem turns out to be the isomorphism of polynomials with one secret (IP1s) problem [7, 8, 12, 13, 15, 23–25], which had been used in MPKC [7, 17, 26–28]. The IP1s problem was shown to be at least as difficult as the graph isomorphism problem [8, 29]. The graph isomorphism problem had been extensively studied for about half century, and no efficient algorithm was known for it, so the IP1s problem was also widely believed to be an intractability problem. When we restrict the invertible affine transformation $\mathscr{L}_1$ to be a linear one, the special IP1s problem was renamed as polynomial linear equivalence (PLE) problem in [24]. In [24], it was shown that the PLE problem is not a restriction on IP1s, and in fact PLE and IP1s are polynomial-time equivalent. In [23], an algorithm was developed to solve the IP1s problem used in the construction of the identification scheme in [7], and the algorithm breaks some challenges of the scheme in [7]. In [24], the differential properties of PLE were fully explored to derive an algorithm for PLE, which transforms the PLE problem into a linear algebraic problem. Some other algorithms were also developed to solve the IP1s problem [12, 13, 15, 25]. These algorithms perform efficiently in some special cases of the IP1s problem.

Previous results about the IP and IP1s problems were established by considering the underlying problems as mathematical problems. However, some cryptographic properties of the cryptographic IP1s problem are maybe overlooked. For example, the central map used in MPKC is required to be easy-to-invertible, and the cryptographic property may help us establish some other algorithms for solving the IP1s problem. In this paper, we utilize the cryptographic property to develop an algorithm for solving the PLE problem and hence the IP1s problem. We fully explore the multiplicative, differential, and additive cryptographic properties existing in the PLE problem. Based on the three properties, we provide a sieve algorithm for the PLE problem. Assume that the central map only has polynomially bounded pre-images; the proposed sieve algorithm is a polynomial-time algorithm. Apart from previously known algorithms based on differential analysis, Gröbner basis, exhaustive search, and linear algebraic methods, we provide a new type of algorithm. The sieve method may be of independent interests and may provide some new insights into the IP-like problems.

The rest of the paper is organized as follows. In Section 2, we formalize the notations, review MPKC in a conceptual level, and define IP-like problems. In Section 3, we elaborate on the proposed sieve method for the PLE problem. Section 4 provides some concluding remarks.

2. Preliminaries

2.1. Notations. Throughout this paper, the following notations will be used. We use $\mathbb{F}_q$ to denote a finite field with order q being a prime power. In this paper, we only consider the PLE problem over $\mathbb{F}_q$ with $q > 2$. We use bold lowercase letters for vectors and bold capital letters for matrices. The generalized linear group over $\mathbb{F}_q$ is denoted as $\mathrm{GL}(\mathbb{F}_q, n)$ which consists of all n-dimensional invertible matrices over $\mathbb{F}_q$. For two sets A and B, we define $A + B = \{a + b : a \in A, b \in B\}$ and $A - B = \{a - b : a \in A, b \in B\}$. For a set $S \subset \mathbb{F}_q^n$ and a nonzero element $a \in \mathbb{F}_q$, we define $aS = \{a\mathbf{s} : \mathbf{s} \in S\}$ and $S/a = \{a^{-1}\mathbf{s} : \mathbf{s} \in S\}$, where a^{-1} stands for the inverse of a in $\mathbb{F}_q$. For a map $\mathcal{S} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ and a vector $\mathbf{s} \in \mathbb{F}_q^m$, we use the symbol $\mathcal{S}^{-1}(\mathbf{s})$ to denote the preimages set of $\mathbf{s}$ under the map $\mathcal{S}$; namely, $\mathcal{S}^{-1}(\mathbf{s}) = \{\mathbf{x} \in \mathbb{F}_q^n : \mathcal{S}(\mathbf{x}) = \mathbf{s}\}$.

2.2. Multivariate Public Key Cryptosystems. The multivariate public key cryptosystems almost always follow the following designs [5]; namely, first define an easy-to-invert central map and then disguise the central map as a seeminglyhard nonlinear map via two invertible affine transformations.

Key Generation. Let $\mathbb{F}_q$ be a finite field with order q being a prime power. Firstly, define a nonlinear central map $\mathcal{S} : \mathbb{F}_q^n \to \mathbb{F}_q^m$; namely, for $\mathbf{x} \in \mathbb{F}_q^n$, $\mathcal{S}(\mathbf{x}) = (s_1(\mathbf{x}), \ldots, s_m(\mathbf{x}))$. In case of a public key encryption scheme, we require that for any $\mathbf{b} \in \mathbb{F}_q^m$, all the solutions $\mathbf{x} \in \mathbb{F}_q^n$ (if the solutions exist) to the system of nonlinear equations $\mathcal{S}(\mathbf{x}) = \mathbf{b}$ can be efficiently determined; namely, $\mathcal{S}^{-1}(\mathbf{b}) = \{\mathbf{x} \in \mathbb{F}_q^n : \mathcal{S}(\mathbf{x}) = \mathbf{b}\}$. In case of a digital signature scheme, we require that for any $\mathbf{b} \in \mathbb{F}_q^m$, we can efficiently find one solution $\mathbf{x} \in \mathcal{S}^{-1}(\mathbf{b})$ (if the solutions exist) to the system of nonlinear equations $\mathcal{S}(\mathbf{x}) = \mathbf{b}$. Secondly, randomly choose two invertible affine transformations $\mathscr{L}_1 : \mathbb{F}_q^n \to \mathbb{F}_q^n$ and $\mathscr{L}_2 : \mathbb{F}_q^m \to \mathbb{F}_q^m$; namely, choose invertible matrices $\mathbf{M}_1$ ($\mathbf{M}_2$, resp.) uniformly and at random from $\mathrm{GL}(\mathbb{F}_q, n)$ ($\mathrm{GL}(\mathbb{F}_q, m)$, resp.) and two vectors $\mathbf{v}_1$ ($\mathbf{v}_2$, resp.) uniformly and at random from $\mathbb{F}_q^n$ ($\mathbb{F}_q^m$, resp.), and define the two affine transformations $\mathscr{L}_1 : \mathbb{F}_q^n \to \mathbb{F}_q^n$ and $\mathscr{L}_2 : \mathbb{F}_q^m \to \mathbb{F}_q^m$ as for $\mathbf{x} \in \mathbb{F}_q^n$, $\mathscr{L}_1(\mathbf{x}) = \mathbf{x}\mathbf{M}_1 + \mathbf{v}_1$, and for $\mathbf{y} \in \mathbb{F}_q^m$, $\mathscr{L}_2(\mathbf{y}) = \mathbf{y}\mathbf{M}_2 + \mathbf{v}_2$. Thirdly, compute the inverses $\mathscr{L}_1^{-1}$ and $\mathscr{L}_2^{-1}$ of $\mathscr{L}_1$ and $\mathscr{L}_2$; namely, for $\mathbf{x} \in \mathbb{F}_q^n$, $\mathscr{L}_1^{-1}(\mathbf{x}) = (\mathbf{x} - \mathbf{v}_1)\mathbf{M}_1^{-1}$, and for $\mathbf{y} \in \mathbb{F}_q^m$, $\mathscr{L}_2^{-1}(\mathbf{y}) = (\mathbf{y} - \mathbf{v}_2)\mathbf{M}_2^{-1}$. Finally, compute $\mathscr{P} = \mathscr{L}_2 \circ \mathcal{S} \circ \mathscr{L}_1 : \mathbb{F}_q^n \to \mathbb{F}_q^m$; namely, for $\mathbf{x} \in \mathbb{F}_q^n$, $\mathscr{P}(\mathbf{x}) = \mathscr{L}_2 \circ \mathcal{S} \circ \mathscr{L}_1(\mathbf{x}) = \mathcal{S}(\mathbf{x}\mathbf{M}_1 + \mathbf{v}_1)\mathbf{M}_2 + \mathbf{v}_2$. The public key is the nonlinear map $\mathscr{P}$, and the secret key consists of $\mathcal{S}$, $\mathscr{L}_1^{-1}$, and $\mathscr{L}_2^{-1}$.

Encryption. For a plaintext $\mathbf{m} \in \mathbb{F}_q^n$, the corresponding ciphertext is computed via $\mathbf{c} = \mathscr{P}(\mathbf{m}) \in \mathbb{F}_q^m$.

Decryption. Given a ciphertext $\mathbf{c} \in \mathbb{F}_q^m$, we firstly compute $\mathbf{y} = \mathscr{L}_2^{-1}(\mathbf{c}) = (\mathbf{c} - \mathbf{v}_2)\mathbf{M}_2^{-1}$. Secondly, compute all the preimages of $\mathbf{y} \in \mathbb{F}_q^m$ under the nonlinear map $\mathcal{S}$; namely, $\mathcal{S}^{-1}(\mathbf{y}) = \{\mathbf{x} \in \mathbb{F}_q^n : \mathcal{S}(\mathbf{x}) = \mathbf{y}\}$. Thirdly, for all the vectors $\mathbf{x} \in \mathcal{S}^{-1}(\mathbf{y})$, we compute $\mathbf{m} = \mathscr{L}_1^{-1}(\mathbf{x}) = (\mathbf{x} - \mathbf{v}_1)\mathbf{M}_1^{-1}$ to obtain a set of candidate plaintexts. Finally, we use some redundant information to exactly pick out the plaintext $\mathbf{m}$.

The design also applies to digital signature schemes.

Signature. To sign a message $\mathbf{m} \in \mathbb{F}_q^m$, we firstly compute $\mathbf{z} = \mathscr{L}_2^{-1}(\mathbf{m})$ then invert $\mathcal{S}$ to get a pre-image $\mathbf{y} \in \mathcal{S}^{-1}(\mathbf{z})$ and finally compute $\mathbf{x} = \mathscr{L}_1^{-1}(\mathbf{y})$. The vector $\mathbf{x} \in \mathbb{F}_q^n$ is the signature on the message $\mathbf{m}$.

Verification. The verifier decides whether $\mathbf{m} = \mathscr{P}(\mathbf{x})$ or not. If the equations are satisfied, the verifier accepts $\mathbf{x}$ as the valid signature of $\mathbf{m}$. Otherwise, the verifier refuses to accept $\mathbf{x}$ as the valid signature of $\mathbf{m}$.

Remarks. The central map $\mathcal{S}$ always has a special structure in that it allows us to efficiently find the pre-images. So in some cases, it is useless to keep the central map secret. For example, the MI [16] central map is $\mathcal{S}(X) = X^{q^\theta+1}$, which makes it meaningless to keep $\mathcal{S}$ secret. Several paddings were suggested on the basic construction of MPKC in order to obtain a higher level of security [30], for example, the plus method [30], the minus method [30], and so on.

2.3. Definitions. The following definitions are closely related to the key recovery attacks on multivariate public key cryptosystems.

Definition 1 (IP [7]). Given two nonlinear polynomial maps $\mathcal{S} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ and $\mathscr{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m$, find two invertible affine transformations $\mathscr{L}_1 : \mathbb{F}_q^n \to \mathbb{F}_q^n$ and $\mathscr{L}_2 : \mathbb{F}_q^m \to \mathbb{F}_q^m$ such that $\mathscr{P} = \mathscr{L}_2 \circ \mathcal{S} \circ \mathscr{L}_1$. Equivalently, find two invertible matrices $\mathbf{M}_1 \in \mathrm{GL}(\mathbb{F}_q, n)$ and $\mathbf{M}_2 \in \mathrm{GL}(\mathbb{F}_q, m)$ and two vectors $\mathbf{v}_1 \in \mathbb{F}_q^n$ and $\mathbf{v}_2 \in \mathbb{F}_q^m$ such that $\mathscr{P}(\mathbf{x}) = \mathcal{S}(\mathbf{x}\mathbf{M}_1 + \mathbf{v}_1)\mathbf{M}_2 + \mathbf{v}_2$.

When $\mathscr{L}_2$ is known or equal to the identity transformation, we get the definition of the IP1s problem [8, 23].

Definition 2 (IP1s). Given two nonlinear polynomial maps $\mathcal{S} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ and $\mathscr{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m$, find an invertible affine transformation $\mathscr{L} : \mathbb{F}_q^n \to \mathbb{F}_q^n$ such that $\mathscr{P} = \mathcal{S} \circ \mathscr{L}$. Equivalently, find an invertible matrix $\mathbf{M} \in \mathrm{GL}(\mathbb{F}_q, n)$ and a vector $\mathbf{v} \in \mathbb{F}_q^n$ such that $\mathscr{P}(\mathbf{x}) = \mathcal{S}(\mathbf{x}\mathbf{M} + \mathbf{v})$.

It was shown in [24] that the IP1s problem and the PLE problem are polynomial-time equivalent. So we only need to discuss the following PLE problem in order to discuss the IP1s problem.

Definition 3 (PLE). Given two nonlinear polynomial maps $\mathcal{S} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ and $\mathscr{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m$, find an invertible linear transformation $\mathscr{L} : \mathbb{F}_q^n \to \mathbb{F}_q^n$ such that $\mathscr{P} = \mathcal{S} \circ \mathscr{L}$. Equivalently, find an invertible matrix $\mathbf{M} \in \mathrm{GL}(\mathbb{F}_q, n)$ such that $\mathscr{P}(\mathbf{x}) = \mathcal{S}(\mathbf{x}\mathbf{M})$.

3. The Proposed Sieve Method for PLE

We pay our attention to a special case of the PLE problem: the preimages of the central map $\mathcal{S}$ are easy to determine. Namely, we are given two nonlinear polynomial maps $\mathcal{S} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ and $\mathscr{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ and an efficient algorithm $\mathscr{A}$ to solve the preimages of the central map $\mathcal{S}$. We want to find an invertible matrix $\mathbf{M} \in \mathrm{GL}(\mathbb{F}_q, n)$ such that

$$\begin{aligned}\left(p_1(\mathbf{x}), \ldots, p_m(\mathbf{x})\right) &= \mathscr{P}(\mathbf{x}) = \mathcal{S}(\mathbf{x}\mathbf{M}) \\ &= \left(s_1(\mathbf{x}\mathbf{M}), \ldots, s_m(\mathbf{x}\mathbf{M})\right).\end{aligned} \tag{1}$$

3.1. Case of $\mathcal{S}$ Being Injective. If $\mathcal{S}$ is an easy-to-invert injective polynomial map, the PLE problem turns out to be very easy. We randomly choose n linearly dependent row vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{F}_q^n$ and denote the matrix consisting of the n vectors as

$$\mathbf{X} = \begin{pmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{pmatrix}. \tag{2}$$

For $i = 1, \ldots, n$, we compute $\mathscr{P}(\mathbf{x}_i) = (p_1(\mathbf{x}_i), \ldots, p_m(\mathbf{x}_i))$. Note that $\mathscr{P}(\mathbf{x}_i) = \mathcal{S}(\mathbf{x}_i\mathbf{M}) = (s_1(\mathbf{x}_i\mathbf{M}), \ldots, s_m(\mathbf{x}_i\mathbf{M}))$, so $\mathbf{y}_i = \mathbf{x}_i\mathbf{M}$ is a solution to the system of equations $\mathcal{S}(\mathbf{y}) = \mathscr{P}(\mathbf{x}_i)$. Further noting that $\mathcal{S}$ is an easy-to-invert injective polynomial map, we conclude that $\mathbf{y}_i = \mathbf{x}_i\mathbf{M}$ is the unique solution to the system of equations $\mathcal{S}(\mathbf{y}) = \mathscr{P}(\mathbf{x}_i)$. So we can apply the algorithm $\mathscr{A}$ to determine the unique solution $\mathbf{y}_i$ to the system of equations $\mathcal{S}(\mathbf{y}) = \mathscr{P}(\mathbf{x}_i)$. We denote the matrix consisting of the n row vectors $\mathbf{y}_1, \ldots, \mathbf{y}_n \in \mathbb{F}_q^n$ as

$$\mathbf{Y} = \begin{pmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_n \end{pmatrix}. \tag{3}$$

We rewrite the equations $\mathbf{y}_i = \mathbf{x}_i\mathbf{M}$ for $i = 1, \ldots, n$ in terms of matrix, so we have $\mathbf{Y} = \mathbf{X}\mathbf{M}$, from which we immediately get $\mathbf{M} = \mathbf{X}^{-1}\mathbf{Y}$.

3.2. General Case. We consider a more general case; namely, $\mathcal{S}$ is an easy-to-invertible noninjective polynomial map.

3.2.1. Basic Idea. The basic idea for the sieve method to solve the PLE problem is to firstly randomly choose n linearly dependent row vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{F}_q^n$, and then for $i = 1, \ldots, n$, compute $\mathscr{P}(\mathbf{x}_i)$. Note that the system of nonlinear equations $\mathcal{S}(\mathbf{y}) = \mathscr{P}(\mathbf{x}_i)$ must have at least a solution $\mathbf{y}_i = \mathbf{x}_i\mathbf{M}$ in that $\mathcal{S}(\mathbf{y}_i) = \mathcal{S}(\mathbf{x}_i\mathbf{M}) = \mathscr{P}(\mathbf{x}_i)$. Secondly, we apply the algorithm $\mathscr{A}$ to get the nonempty set of the solutions to $\mathcal{S}(\mathbf{y}) = \mathscr{P}(\mathbf{x}_i)$; namely,

$$\mathcal{S}^{-1}\left(\mathscr{P}(\mathbf{x}_i)\right) = \left\{\mathbf{y} \in \mathbb{F}_q^n : \mathcal{S}(\mathbf{y}) = \mathscr{P}(\mathbf{x}_i)\right\}. \tag{4}$$

However, the noninjectivity of the central polynomial map $\mathcal{S}$ says that the solutions set $\mathcal{S}^{-1}(\mathscr{P}(\mathbf{x}_i))$ may include some other solutions except $\mathbf{y}_i = \mathbf{x}_i\mathbf{M}$. We are only interested in the targeted solution $\mathbf{y}_i = \mathbf{x}_i\mathbf{M}$ and want to develop a method to pick out the vector $\mathbf{y}_i = \mathbf{x}_i\mathbf{M}$ from all the solutions in $\mathcal{S}^{-1}(\mathscr{P}(\mathbf{x}_i))$. If for $i = 1, \ldots, n$ we can determine the corresponding $\mathbf{y}_i = \mathbf{x}_i\mathbf{M}$, we just denote the matrices

consisting of the row vectors $\mathbf{x}_i$ and $\mathbf{y}_i$ as $\mathbf{X}$ and $\mathbf{Y}$, respectively. Similarly to the discussions in Section 3.1, we can solve the PLE problem just by computing $\mathbf{M} = \mathbf{X}^{-1}\mathbf{Y}$.

In what follows, we will design three types of sieves called multiplicative sieve, differential sieve, and additive sieve, respectively. When we apply the three sieves to $\mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i))$, we hope that the targeted solution $\mathbf{y}_i$=$\mathbf{x}_i\mathbf{M}$ can pass the sieves, and other useless solutions in $\mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i))$ are distilled out as many as possible. Now we discuss some properties of the PLE problem.

3.2.2. Sieve Strategies. Let $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{F}_q^n$ be n linearly dependent row vectors, the set of solutions to the equations $\mathcal{S}(\mathbf{y}) = \mathcal{P}(\mathbf{x}_i)$ be $\mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i))$, and $\mathbf{y}_i = \mathbf{x}_i\mathbf{M} \in \mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i))$. We have the following results.

Theorem 4 (multiplicative strategy). *For any $a \neq 0, 1$ in $\mathbb{F}_q$, $a\mathbf{y}_i$ is also a solution to the system of nonlinear equations $\mathcal{S}(\mathbf{y}) = \mathcal{P}(a\mathbf{x}_i)$; namely, $a\mathbf{y}_i \in \mathcal{S}^{-1}(\mathcal{P}(a\mathbf{x}_i))$, or equivalently, $\mathbf{y}_i \in \mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i)) \cap (\mathcal{S}^{-1}(\mathcal{P}(a\mathbf{x}_i))/a)$.*

Proof. We first note that $\mathbf{y}_i = \mathbf{x}_i\mathbf{M}$, so $a\mathbf{y}_i = a\mathbf{x}_i\mathbf{M}$. Therefore, we have $\mathcal{S}(a\mathbf{y}_i) = \mathcal{S}(a\mathbf{x}_i\mathbf{M}) = \mathcal{P}(a\mathbf{x}_i)$, namely, $a\mathbf{y}_i \in \mathcal{S}^{-1}(\mathcal{P}(a\mathbf{x}_i))$; or equivalently, $\mathbf{y}_i \in \mathcal{S}^{-1}(\mathcal{P}(a\mathbf{x}_i))/a$. Note that $\mathbf{y}_i \in \mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i))$, so we have $\mathbf{y}_i \in \mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i)) \cap (\mathcal{S}^{-1}(\mathcal{P}(a\mathbf{x}_i))/a)$. □

The theorem of *Multiplicative Strategy* implies a method to sieve out some useless vectors from a set of vectors containing the targeted vector $\mathbf{y}_i = \mathbf{x}_i\mathbf{M}$. More precisely, we let $S_i \subset \mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i))$ such that the targeted vector $\mathbf{y}_i = \mathbf{x}_i\mathbf{M} \in S_i$. The multiplicative sieve algorithm `MulSieve` given in Algorithm 1 takes input as $(\mathbb{F}_q, \mathcal{S}, \mathcal{P}, \mathbf{x}_i, S_i)$ and outputs a subset $S_i^{\times}$ of S_i; namely, $S_i^{\times} = \texttt{MulSieve}(\mathbb{F}_q, \mathcal{S}, \mathcal{P}, \mathbf{x}_i, S_i)$. From the proof of *Multiplicative Strategy* theorem, we know that the targeted vector $\mathbf{y}_i = \mathbf{x}_i\mathbf{M}$ can pass the multiplicative sieve. So the set $S_i^{\times}$ output by `MulSieve` is not empty.

We note that if $\mathcal{S}$ and hence $\mathcal{P}$ are homogeneous polynomials, all the preimages in S_i can pass the multiplicative sieve. So in this case, we must have $S_i^{\times} = S_i$. In general cases, $\mathcal{S}$ is not homogeneous, and the multiplicative sieve method can sieve out some preimages of S_i.

Theorem 5 (additive strategy). *For $1 \leq i < j \leq n$, one must have that $\mathbf{y}_i + \mathbf{y}_j$ is a solution to the system of equations $\mathcal{S}(\mathbf{y}) = \mathcal{P}(\mathbf{x}_i + \mathbf{x}_j)$. That is, if one denotes the solutions set to the system of nonlinear equations $\mathcal{S}(\mathbf{y}) = \mathcal{P}(\mathbf{x}_i + \mathbf{x}_j)$ as $\mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i + \mathbf{x}_j))$, one must have*

$$\mathbf{y}_i + \mathbf{y}_j \in \mathcal{S}^{-1}\left(\mathcal{P}\left(\mathbf{x}_i + \mathbf{x}_j\right)\right) \cap \left(\mathcal{S}^{-1}\left(\mathcal{P}\left(\mathbf{x}_i\right)\right) + \mathcal{S}^{-1}\left(\mathcal{P}\left(\mathbf{x}_j\right)\right)\right). \tag{5}$$

Proof. It is obvious that $\mathbf{y}_i + \mathbf{y}_j \in \mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i)) + \mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_j))$. So we just need to verify $\mathbf{y}_i + \mathbf{y}_j \in \mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i + \mathbf{x}_j))$; namely, $\mathcal{P}(\mathbf{x}_i + \mathbf{x}_j) = \mathcal{S}(\mathbf{y}_i + \mathbf{y}_j)$. Recalling $\mathbf{y}_i = \mathbf{x}_i\mathbf{M}$, $\mathbf{y}_j = \mathbf{x}_j\mathbf{M}$, and $\mathcal{P}(\mathbf{x}) = \mathcal{S}(\mathbf{x}\mathbf{M})$, we immediately have $\mathcal{S}(\mathbf{y}_i + \mathbf{y}_j) = \mathcal{S}((\mathbf{x}_i + \mathbf{x}_j)\mathbf{M}) = \mathcal{P}(\mathbf{x}_i + \mathbf{x}_j)$. □

The theorem of *Additive Strategy* implies another sieve method called additive sieve method `AddSieve` in Algorithm 2. The input of the additive sieve algorithm `AddSieve` consists of $(\mathbb{F}_q, \mathcal{S}, \mathcal{P}, \mathbf{x}_i, \mathbf{x}_j, S_i, S_j)$, where $1 \leq i < j \leq n$, S_i (S_j, resp.) is a subset of $\mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i))$ ($\mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_j))$, resp.), and $\mathbf{y}_i \in S_i$ ($\mathbf{y}_j \in S_j$, resp.). The output of `AddSieve` is two nonempty sets $S_i^+ \subset S_i$ and $S_j^+ \subset S_j$; namely,

$$\left(S_i^+, S_j^+\right) = \texttt{AddSieve}\left(\mathbb{F}_q, \mathcal{S}, \mathcal{P}, \mathbf{x}_i, \mathbf{x}_j, S_i, S_j\right). \tag{6}$$

From the proof of *Multiplicative Strategy* theorem, we know that the targeted vectors $\mathbf{y}_i = \mathbf{x}_i\mathbf{M}$ and $\mathbf{y}_j = \mathbf{x}_j\mathbf{M}$ can pass the additive sieve. So the sets S_i^+ and S_j^+ output by `AddSieve` are not empty.

In lines 6–8 of Algorithm 2, if $\mathbf{y}^{(i)}$ or $\mathbf{y}^{(j)}$ had been put into S_i^+ or S_j^+, the algorithm does nothing.

Theorem 6 (differential strategy). *For any nonzero vector $\mathbf{z} \in \mathbb{F}_q^n$ and any element $a \neq 0, 1$ in $\mathbb{F}_q$, let the set of the solutions to the system of nonlinear equations $\mathcal{S}(\mathbf{y}) = \mathcal{P}(\mathbf{x}_i + \mathbf{z})$ and $\mathcal{S}(\mathbf{y}) = \mathcal{P}(\mathbf{x}_i + a\mathbf{z})$ be $\mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i + \mathbf{z}))$ and $\mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i + a\mathbf{z}))$, respectively. One must have*

$$(a-1)\,\mathbf{y}_i \in a\mathcal{S}^{-1}\left(\mathcal{P}\left(\mathbf{x}_i + \mathbf{z}\right)\right) - \mathcal{S}^{-1}\left(\mathcal{P}\left(\mathbf{x}_i + a\mathbf{z}\right)\right). \tag{7}$$

Proof. Note that $\mathbf{y}_i = \mathbf{x}_i\mathbf{M}$, so we have $(\mathbf{x}_i + \mathbf{z})\mathbf{M} = \mathbf{y}_i + \mathbf{z}\mathbf{M}$. From $\mathcal{P}(\mathbf{x}) = \mathcal{S}(\mathbf{x}\mathbf{M})$, we get $\mathcal{P}(\mathbf{x}_i + \mathbf{z}) = \mathcal{S}((\mathbf{x}_i + \mathbf{z})\mathbf{M}) = \mathcal{S}(\mathbf{y}_i + \mathbf{z}\mathbf{M})$. So we have $\mathbf{y}_i + \mathbf{z}\mathbf{M} \in \mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i + \mathbf{z}))$; namely,

$$a\left(\mathbf{y}_i + \mathbf{z}\mathbf{M}\right) \in a\mathcal{S}^{-1}\left(\mathcal{P}\left(\mathbf{x}_i + \mathbf{z}\right)\right). \tag{8}$$

We further notice that $\mathcal{P}(\mathbf{x}_i + a\mathbf{z}) = \mathcal{S}((\mathbf{x}_i + a\mathbf{z})\mathbf{M}) = \mathcal{S}(\mathbf{y}_i + a\mathbf{z}\mathbf{M})$, so we conclude that

$$\mathbf{y}_i + a\mathbf{z}\mathbf{M} \in \mathcal{S}^{-1}\left(\mathcal{P}\left(\mathbf{x}_i + a\mathbf{z}\right)\right). \tag{9}$$

Combining (8) and (9) implies

$$\begin{aligned}(a-1)\,\mathbf{y}_i &= a\left(\mathbf{y}_i + \mathbf{z}\mathbf{M}\right) - \left(\mathbf{y}_i + a\mathbf{z}\mathbf{M}\right) \\ &\in a\mathcal{S}^{-1}\left(\mathcal{P}\left(\mathbf{x}_i + \mathbf{z}\right)\right) - \mathcal{S}^{-1}\left(\mathcal{P}\left(\mathbf{x}_i + a\mathbf{z}\right)\right),\end{aligned} \tag{10}$$

from which we complete the proof. □

The theorem of *Differential Strategy* implies another sieve method called differential sieve method `DifSieve` in Algorithm 3. The input of the differential sieve algorithm `DifSieve` consists of $(\mathbb{F}_q, \mathcal{S}, \mathcal{P}, \mathbf{x}_i, S_i)$, where S_i is a subset of $\mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i))$ and $\mathbf{y}_i \in S_i$. The output of `DifSieve` is a nonempty set $S_i^- \subset S_i$; namely, $S_i^- = \texttt{DifSieve}(\mathbb{F}_q, \mathcal{S}, \mathcal{P}, \mathbf{x}_i, S_i)$. From the proof of *Differential Strategy* theorem, we know that the targeted vectors $\mathbf{y}_i = \mathbf{x}_i\mathbf{M}$ can pass the differential sieve. So the set S_i^- output by the `DifSieve` algorithm is nonempty.

3.3. The Sieve Method. We elaborate on the novel sieve algorithm `SieAlg` for solving the PLE problem as in Algorithm 4. The input for the sieve algorithm contains the description of the finite field $\mathbb{F}_q$ and two multivariate nonlinear polynomial

(1) Set the set $S_i^{\times} = \Phi$. //Φ denotes an empty set.
(2) Choose an element $a_i \neq 0, 1$ uniformly and randomly from $\mathbb{F}_q$ and compute $\mathcal{P}(a_i\mathbf{x}_i)$.
(3) **for** each vector $\mathbf{y} \in S_i$ **do**
(4) Compute $\mathcal{S}(a_i\mathbf{y})$.
(5) **if** $\mathcal{S}(a_i\mathbf{y}) = \mathcal{P}(a_i\mathbf{x}_i)$ **then**
(6) Add $\mathbf{y}$ to the set $S_i^{\times}$.
(7) **end if**
(8) **end for**
(9) **return** $S_i^{\times}$.

Algorithm 1: Multiplicative sieve: $\texttt{MulSieve}\left(\mathbb{F}_q, \mathcal{S}, \mathcal{P}, \mathbf{x}_i, S_i\right)$.

(1) Set the two sets $S_i^{+} = \Phi$ and $S_j^{+} = \Phi$.
(2) Compute $\mathcal{P}\left(\mathbf{x}_i + \mathbf{x}_j\right)$.
(3) **for** each $\mathbf{y}^{(i)} \in S_i$ **do**
(4) **for** each $\mathbf{y}^{(j)} \in S_j$ **do**
(5) Compute $\mathcal{S}\left(\mathbf{y}^{(i)} + \mathbf{y}^{(j)}\right)$.
(6) **if** $\mathcal{P}\left(\mathbf{x}_i + \mathbf{x}_j\right) = S\left(\mathbf{y}^{(i)} + \mathbf{y}^{(j)}\right)$ **then**
(7) Add $\mathbf{y}^{(i)}$ to S_i^{+}, and $\mathbf{y}^{(j)}$ to S_j^{+}, respectively.
(8) **end if**
(9) **end for**
(10) **end for**
(11) **return** $\left(S_i^{+}, S_j^{+}\right)$.

Algorithm 2: Additive sieve: $\texttt{AddSieve}\left(\mathbb{F}_q, \mathcal{S}, \mathcal{P}, \mathbf{x}_i, \mathbf{x}_j, S_i, S_j\right)$.

maps $\mathcal{S} : \mathbb{F}_q^n \to \mathbb{F}_q^m$ and $\mathcal{P} : \mathbb{F}_q^n \to \mathbb{F}_q^m$. The output of the sieve algorithm is either an invertible matrix $\mathbf{M} \in \mathrm{GL}(\mathbb{F}_q, n)$ such that $\mathcal{P}(\mathbf{x}) = \mathcal{S}(\mathbf{x}\mathbf{M})$ or a failure symbol $\perp$. We assume that the central map $\mathcal{S}$ is easy to invert; namely, there exists an efficient algorithm $\mathcal{A}$ to compute all the preimages of $\mathcal{S}$. The proposed sieve algorithm runs as in Algorithm 4.

3.4. Analysis. We analyze the computational complexity of the proposed sieve algorithm for the PLE problem. We denote the computational costs for determining the preimages set of the polynomial central map $\mathcal{S}$ as Δ. Note that we assume that the central map $\mathcal{S}$ is easy to invert, so Δ is upper-bounded by a polynomial about the number n of the variables and the number m of the involved equations. We let the computational costs for computing $\mathcal{S}(\mathbf{x})$ for any $\mathbf{x} \in \mathbb{F}_q^n$ as δ. Note that $\mathcal{P}(\mathbf{x}) = \mathcal{S}(\mathbf{x}\mathbf{M})$, so for any $\mathbf{x} \in \mathbb{F}_q^n$, it requires about the same computational costs δ to compute $\mathcal{P}(\mathbf{x})$. We denote the upper bound for the number of preimages of a vector $\mathbf{b} \in \mathbb{F}_q^m$ under the central map $\mathcal{S}$ as l; namely, the preimages set $\mathcal{S}^{-1}(\mathbf{b})$ has at most l vectors. Note that we assume that $\mathcal{A}$ is a polynomial-time algorithm to find all the preimages for the central map $\mathcal{S}$, so l is also polynomially bounded.

In the algorithm initialization phase of the sieve algorithm SieAlg for the PLE problem, we need to compute $\mathcal{P}(\mathbf{x}_i)$ and the pre-image set $\mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i))$ for each $i = 1, \ldots, n$. So in the algorithm initialization phase, the computational costs are measured as $\mathcal{O}(n(\Delta + \delta))$.

In the multiplicative sieve phase, for each $i = 1, \ldots, n$, we need to compute $\mathcal{P}(a_i\mathbf{x}_i)$, which costs $\mathcal{O}(n\delta)$. Then for each $\mathbf{y} \in \mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i))$, we need to compute $\mathcal{S}(a_i\mathbf{y})$, so the computational costs are $\mathcal{O}(nl\delta)$. So the total costs in the multiplicative sieve phase are $\mathcal{O}(n\delta + nl\delta) = \mathcal{O}(nl\delta)$.

In the additive sieve phase, for $1 \leq i < j \leq n$, we need to compute $\mathcal{P}(\mathbf{x}_i + \mathbf{x}_j)$, which costs $\mathcal{O}(n^2\delta)$. For $1 \leq i < j \leq n$, for each $\mathbf{y}^{(i)} \in S_i^{\times}$ and for each $\mathbf{y}^{(j)} \in S_j^{\times}$, we need to compute $\mathcal{S}(\mathbf{y}^{(i)} + \mathbf{y}^{(j)})$, which cost $\mathcal{O}(n^2l^2\delta)$. So the computational costs in the additive sieve phase are $\mathcal{O}(n^2l^2\delta)$.

In the differential sieve phase, for each $i = 1, \ldots, n$, we need to compute $\mathcal{P}(\mathbf{x}_i + \mathbf{z}_i)$, $\mathcal{P}(\mathbf{x}_i + a_i\mathbf{z}_i)$, and the preimages sets $\mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i + \mathbf{z}_i))$ and $\mathcal{S}^{-1}(\mathcal{P}(\mathbf{x}_i + a_i\mathbf{z}_i))$. In this step, the computational costs are $\mathcal{O}(n\delta + n\Delta)$. We also need to do exhaustive search for $\mathbf{y} \in S_i^{+}$, which costs $\mathcal{O}(l)$. So during the differential sieve phase, we need to do $\mathcal{O}(n\delta + n\Delta + l)$ computations.

Finally, the sieve method needs to compute $\mathbf{M} = \mathbf{X}^{-1}\mathbf{Y}$, which costs $\mathcal{O}(n^3)$.

To summarize, the computational costs for the sieve algorithm are the sum of the aforementioned computational costs. So the computational complexity of the sieve algorithm is $\mathcal{O}(n^2l^2\delta + n\Delta + l + n^3)$. Therefore, the proposed sieve algorithm is polynomial-time if there exists a polynomial-time algorithm $\mathcal{A}$ to determine the preimages set of the central map $\mathcal{S}$.

We remark on the sieve algorithm as follows. In the sieve algorithm, we assume that the order of the underlying finite field $\mathbb{F}_q$ is $q > 2$. If the PLE problem is defined over $\mathbb{F}_2$, we can consider the PLE problem over some extension field of $\mathbb{F}_2$ in order to make the proposed sieve algorithm applicable. The proposed sieve algorithm does not sieve out the $\mathbf{y}_i$ satisfying $\mathbf{y}_i = \mathbf{x}_i\mathbf{M}$. In other words, the proposed sieve algorithm does not sieve out the right answer for the PLE problem. When the algorithm quits and fails to output the right answer, we can reimplement the sieve algorithm one more time in order to increase the probability that we can solve the PLE problem.

4. Conclusions

In this paper, we developed a new method for solving the IP1s problem. It is shown that if there exists a polynomial-time algorithm for determining all the preimages of the central map $\mathcal{S}$, the proposed sieve algorithm for the PLE problem is

(1) Set the set $S_i^- = \Phi$.
(2) Randomly choose an element $a_i \neq 0, 1$ from $\mathbb{F}_q$ and a nonzero vector $\mathbf{z}_i \in \mathbb{F}_q^n$.
(3) Compute $\mathscr{P}(\mathbf{x}_i + \mathbf{z}_i)$ and $\mathscr{P}(\mathbf{x}_i + a_i\mathbf{z}_i)$.
(4) Compute the pre-images sets $\mathscr{S}^{-1}(\mathscr{P}(\mathbf{x}_i + \mathbf{z}_i))$ and $\mathscr{S}^{-1}(\mathscr{P}(\mathbf{x}_i + a_i\mathbf{z}_i))$.
(5) **for** each $\mathbf{y} \in S_i$ **do**
(6) **if** $(a_i - 1)\mathbf{y} \in a_i\mathscr{S}^{-1}(\mathscr{P}(\mathbf{x}_i + \mathbf{z}_i)) - \mathscr{S}^{-1}(\mathscr{P}(\mathbf{x}_i + a_i\mathbf{z}_i))$ **then**
(7) Add $\mathbf{y}$ to S_i^-.
(8) **end if**
(9) **end for**
(10) **return** S_i^-

Algorithm 3: Differential sieve: `DifSieve` $\left(\mathbb{F}_q, \mathscr{S}, \mathscr{P}, \mathbf{x}_i, S_i\right)$.

(1) Randomly choose n linearly independent vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{F}_q^n$. Store the n row vectors $\mathbf{x}_i$ as an invertible n-dimensional matrix $\mathbf{X}$.
(2) **for** $i = 1, \ldots, n$ **do**
(3) Compute $\mathscr{P}(\mathbf{x}_i)$ and the pre-image set $\mathscr{S}^{-1}(\mathscr{P}(\mathbf{x}_i))$.
(4) Store the set $S_i = \mathscr{S}^{-1}(\mathscr{P}(\mathbf{x}_i))$.
(5) **end for** `//Algorithm Initialization`.
(6) **for** $i = 1, \ldots, n$ **do**
(7) Run `MulSieve`$\left(\mathbb{F}_q, \mathscr{S}, \mathscr{P}, \mathbf{x}_i, S_i\right)$ and assign the output to S_i.
(8) **end for** `//Multiplicative Sieve`.
(9) **for** $i = 1, \ldots, n-1$ **do**
(10) **for** $j = i + 1, \ldots, n$ **do**
(11) Run `AddSieve`$\left(\mathbb{F}_q, \mathscr{S}, \mathscr{P}, \mathbf{x}_i, \mathbf{x}_j, S_i, S_j\right)$ and assign the output to S_i and S_j.
(12) **end for**
(13) **end for** `//Additive Sieve`.
(14) **for** $i = 1, \ldots, n$ **do**
(15) Run `DifSieve`$\left(\mathbb{F}_q, \mathscr{S}, \mathscr{P}, \mathbf{x}_i, S_i\right)$ and assign the output to S_i.
(16) **end for** `//Differential Sieve`.
(17) **for** $i = 1, \ldots, n$ **do**
(18) Randomly choose a vector from S_i and assign the vector to $\mathbf{y}_i$.
(19) **end for**
(20) Denote the set of vectors $\mathbf{y}_1, \ldots, \mathbf{y}_n$ as a matrix $\mathbf{Y}$.
(21) Compute $\mathbf{M} = \mathbf{X}^{-1}\mathbf{Y}$.
(22) **if** $\mathscr{P}(\mathbf{x}) = \mathscr{S}(\mathbf{x}\mathbf{M})$ **then**
(23) **return** $\mathbf{M}$.
(24) **else**
(25) **return** $\perp$.
(26) **end if**

Algorithm 4: The sieve algorithm for PLE: `SieAlg` $\left(\mathbb{F}_q, \mathscr{S}, \mathscr{P}\right)$.

efficient. As an application of the proposed sieve algorithm for the IP1s problem, we can show that it is infeasible to construct multivariate public key encryption schemes from the IP1s problem due to the following reasons.

(i) Multivariate public key encryption schemes require that the central map must be easy-to-invert.

(ii) Multivariate public key encryption schemes require that ciphertext must be decipherable. So for any vector $\mathbf{b} \in \mathbb{F}_q^n$, it must be computationally feasible to determine all the preimages of $\mathbf{b}$ under the central map $\mathscr{S}$. This means that the number of the preimages of the central map $\mathscr{S}$ must be polynomially bounded.

The above both things demonstrate that the proposed sieve algorithm is applicable to the IP1s problem used in multivariate public key encryption schemes.

As a new method for solving the IP-like problems, the proposed sieve method is far from perfect. So further discussions on the new method belong to our future work.

Conflict of Interests

The authors declares that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (nos. 61173152 and 61173151), the 111 Project (no. B08038), the ISN Foundation (no. ISN1103007), the Fundamental Research Funds for the Central Universities (no. JY10000901009), and the Natural Science Basic Research Plan in Shaanxi Province of China (Program no. 2012JM8005).

References

[1] N. Koblitz and A. J. Menezes, "A survey of public-key cryptosystems," *SIAM Review*, vol. 46, no. 4, pp. 599–634, 2004.

[2] P. Wang and F. Zhang, "An efficient collision detection method for computing discrete logarithms with pollard's rho," *Journal of Applied Mathematics*, vol. 2012, Article ID 635909, 15 pages, 2012.

[3] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, 1997.

[4] D. Cheung, D. Maslov, J. Mathew, and D. K. Pradhan, "On the design and optimization of a quantum polynomial-time attack on elliptic curve cryptography," in *Proceedings of the 3rd Workshop on Theory of Quantum Computation, Communication, and Cryptography (TQC '08)*, vol. 5016 of *LNCS*, pp. 96–104, Springer, Tokyo, Japan, 2008.

[5] J. Ding, J. E. Gower, and D. S. J. Schmidt, *Multivariate Public Key Cryptosystems*, vol. 25 of *Advances in Information Security*, Springer, Berlin, Germany, 2006.

[6] M. R. Garey and D. S. Johnson, *Computers and Intractability, A Guide to the Theory of NP-Completeness*, Freeman, San Francisco, Calif, USA, 1979.

[7] J. Patarin, "Hidden fields equations (HFE) and isomorphism of polynomials (IP): two new families of asymmetric algorithms," in *Proceedings of Advances in Cryptology-Eurocrypt 1996*, vol. 1070 of *LNCS*, pp. 33–48, Springer, Saragossa, Spain, 1996.

[8] J. Patarin, L. Goubin, and N. Courtois, "Improved algorithms for isomorphisms of polynomials," in *Proceedings of Advances in Cryptology- Eurocrypt 1998*, vol. 1403 of *LNCS*, pp. 184–200, Springer, Espoo, Finland, 1998.

[9] F. Levy-dit-Vehel and L. Perret, "Polynomial equivalence problems and applications to multivariate cryptosystems," in *Proceedings of the Conference on Progress in Cryptology (INDOCRYPT '03)*, vol. 2904 of *LNCS*, pp. 235–251, Thomas Johansson, Subhamoy Maitra, New Delhi, India, 2003.

[10] L. Perret and A. Bayad, "A differential approach to a polynomial equivalence problem," in *Proceedings of International Symposium on Information Theory (ISIT '04)*, p. 140, IEEE Press, Chicago, Ill, USA, July 2004.

[11] J. Faugere and L. Perret, "Polynomial equivalence problems: algorithmic and theoretical aspects," in *Proceedings of Advances in Cryptology (Eurocrypt '06)*, vol. 4004 of *LNCS*, pp. 30–47, Springer, St. Petersburg, Russia, 2006.

[12] C. Bouillaguet, J. Faugere, P. Fouque, and L. Perret, "Isomorphism of polynomials: new results," http://citeseerx.ist.psu.edu.

[13] C. Bouillaguet, J. Faugere, P. Fouque, and L. Perret, "Differentialalgebraic algorithms for the isomorphism of polynomials problem," IACR Cryptology ePrint Archive 2009, http://eprint.iacr.org/2009/583.

[14] C. Bouillaguet, P. Fouque, and A. Veber, "Graph-theoretic algorithms for the isomorphism of polynomials 'problem'," in *Proceedings of Ad- vances in Cryptology (Eurocrypt '13)*, vol. 7881 of *LNCS*, pp. 211–227, Springer, Athens, Greece, 2013.

[15] J. Berthomieu, J. Faugere, and L. Perret, "Polynomial-time algorithms for quadratic isomorphism of polynomials," 2013, http://arxiv.org/abs/1307.4974.

[16] T. Matsumoto and H. Imai, "Public quadratic polynomial-tuples for efficient signature-verification and message-encryption," in *Proceedings of the Advances in Cryptology (Eurocrypt '88)*, vol. 330 of *LNCS*, pp. 419–453, Springer, Davos, Switzerland, 1988.

[17] A. Kipnis, J. Patarin, and L. Goubin, "Unbalanced oil and vinegar signature schemes," in *Proceedings of the Advances in Cryptology (Eurocrypt '99)*, vol. 1592 of *LNCS*, pp. 206–222, Springer, Prague, Czech Republic, 1999.

[18] J. Patarin, N. Courtois, and L. Goubin, "Flash, a fast multivariate signature algorithm," in *Proceedings of the Cryptographers Track at RSA Conference (CT-RSA '01)*, vol. 2020 of *LNCS*, pp. 298–307, Springer, San Francisco, Calif, USA, 2001.

[19] O. Billet and H. Gilbert, "A traceable block cipher," in *Proceedings of Advances in Cryptology (Asiacrypt '00)*, vol. 2894 of *LNCS*, pp. 331–346, Springer, Taipei, Taiwan, 2003.

[20] J. Ding, C. Wolf, and B. Y. Yang, "l-invertible cycles for multivariate quadratic public key cryptography," in *Proceedings of the 10th IACR International Conference on Practice and Theory of Public Key Cryptography (PKC '07)*, vol. 4450 of *LNCS*, pp. 266–281, Springer, Beijing, China, 2007.

[21] J. Baena, C. Clough, and J. Ding, "Square-vinegar signature scheme," in *Proceedings of the 2nd International Workshop on Post-Quantum Cryptography (PQCrypto '08)*, vol. 5299 of *LNCS*, pp. 17–30, Springer, Cincinnati, Ohio, USA, 2008.

[22] C. Clough, J. Baena, J. Ding, B. Y. Yang, and M. S. Chen, "Square, a new multivariate encryption scheme," in *Proceedings of the Cryptogra- phers Track at RSA Conference (CT-RSA '09)*, vol. 5473 of *LNCS*, pp. 252–264, Springer, San Francisco, Calif, USA, 2009.

[23] W. Geiselmann, W. Meier, and S. Rainer, "An attack on the isomorphisms of polynomials problem with one secret," *International Journal of Information Security*, vol. 2, no. 1, pp. 59–64, 2003.

[24] L. Perret, "A fast cryptanalysis of the isomorphism of polynomials with one secret problem," in *Proceedings of Advances in Cryptology (Eurocrypt '05)*, vol. 3439 of *LNCS*, pp. 354–370, Springer, Aarhus, Denmark, 2005.

[25] C. Bouillaguet, J. Faugere, P. Fouque, and L. Perret, "Practical cryptanalysis of the identification scheme based on the isomorphism of polynomial with one secret problem," in *Proceedings of the 14th IACR International Conference on Practice and Theory of Public Key Cryptography (PKC '11)*, vol. 6571 of *LNCS*, pp. 473–493, Springer, Taormina, Italy, 2011.

[26] K. Sakumoto, T. Shirai, and H. Hiwatari, "Public-key identification schemes based on multivariate quadratic polynomials," in *Proceedings of Advances in Cryptology (Crypto '11)*, vol. 6841 of *LNCS*, pp. 706–723, Springer, Santa Barbara, Calif, USA, 2011.

[27] S. Tang and L. Xu, "Proxy signature scheme based on isomorphisms of polynomials," in *Proceedings of the 6th International Conference on Network and System Security (NSS '12)*, vol. 7645 of *LNCS*, pp. 113–125, Springer, Fujian, China, 2012.

[28] S. Tang and L. Xu, "Towards provably secure proxy signature scheme based on isomorphisms of polynomials," *Future Generation Computer Systems*, 2013.

[29] M. Agrawal and N. Saxena, "Equivalence of f-algebras and cubic forms," in *Proceedings of the 23rd Annual Symposium on Theoretical Aspects of Computer Science (STACS '06)*, vol. 3884 of *LNCS*, pp. 115–126, Springer, Marseille, France, 2006.

[30] C. Wolf and B. Preneel, "Taxonomy of public key schemes based on the problem of multivariate quadratic equations," IACR Cryptology ePrint Archive 2005, https://eprint.iacr.org/2005/077.