Advance Access publication on 5 March 2019
doi:10.1093/comjnl/bxz013

Identity-Based Signature Schemes for Multivariate Public Key Cryptosystems

Jiahui Chen[1], Jie Ling[1], Jianting Ning[2] and Jintai Ding[3*]

[1]*Faculty of Computer, Guangdong University of Technology, Guangzhou, China*
[2]*School of Computing, National University of Singapore, Singapore, Singapore*
[3]*University of Cincinnati, Cincinnati, OH, USA*
[*]*Corresponding author: jintai.ding@gmail.com*

In this paper, we proposed an idea to construct a general multivariate public key cryptographic (MPKC) scheme based on a user's identity. In our construction, each user is distributed a unique identity by the key distribution center (KDC) and we use this key to generate user's private keys. Thereafter, we use these private keys to produce the corresponding public key. This method can make key generating process easier so that the public key will reduce from dozens of Kilobyte to several bits. We then use our general scheme to construct practical identity-based signature schemes named ID-UOV and ID-Rainbow based on two well-known and promising MPKC signature schemes, respectively. Finally, we present the security analysis and give experiments for all of our proposed schemes and the baseline schemes. Comparison shows that our schemes are both efficient and practical.

Keywords: ID-based signature; post-quantum cryptography; multivariate public key cryptosystems; key distribution center

Received 8 October 2018; revised 3 January 2019; editorial decision 27 January 2019
Handling editor: Joseph Liu

1. INTRODUCTION

To build a public key scheme based on user's identity was first brought in by Shamir in 1984 [1]. The original motivation to build an encryption system based on ID was to simplify certificate management in e-mail systems. There already exist some development in this area, several ID-based encryption schemes and signature schemes have been proposed since then. For example, Boneh and Franklin proposed an ID-based encryption scheme (BF-IBE) based on bilinear maps on an elliptic curve [2]. In [3], the authors proposed an identity (ID)-based signature scheme using gap Diffie–Hellman (GDH) groups. Li *et al.* proposed some identity-based encryption schemes with different properties such as leakage resilience [4, 5] and inputs leakage [6]. Also, identity-based cryptographic scheme has a lot of variants such as attribute-based encryption (ABE) schemes and ciphertext-policy attribute-based encryption (CP-ABE) schemes which can be used practically in the access control scene of cloud platforms [7–9].

On the other hand, multivariate public key cryptography is one of the most promising candidates for RSA algorithm. Since according to the Shor's algorithm [10], RSA and some other algorithms based on number theory will be broken in polynomial time after the emergence of quantum computers. Thereby, it is urgent to find an alternative of RSA. The security of MPKC is based on solving a set of random quadratic multivariate equations on a finite field is NP-hard. So far, evidence does not reveal that quantum computers could solve this kind of questions effectively. Plus, MPKC schemes are in general much more effective than RSA in computing.

Anyway, MPKC has shown its considerable potential in post-quantum era. Usually, a MPKC scheme on finite field $\mathbb{F}_q$ is built as

$$P = L_1 \circ F \circ L_2$$

in which F is a set of m quadratic multivariate equations in n variables and L_1 is an affine transformation from $\mathbb{F}_q^m$ to $\mathbb{F}_q^m$ and L_2 is an affine transformation from $\mathbb{F}_q^n$ to $\mathbb{F}_q^n$.

In addition, multivariate signature schemes with special properties, such as proxy signature and ring signature, are proposed. For example, Tang *et al.* [11] proposed the first MPKC proxy signature scheme based on the problem of Isomorphisms of Polynomials (IP). Petzoldt *et al.* [12] proposed the first provable MPKC threshold ring signature

scheme based on the result of [13]. Chen *et al*. [14] proposed the first online/offline signature based on UOV by utilizing the linear construction of the central map of UOV, so that the proposed scheme can be distributed in the wireless sensor networks. In addition, multivariate sequential aggregate signature scheme by Petzoldt *et al*. [15] and multivariate blind signature scheme by Bansarkhani *et al*. [16] are proposed to enrich this area.

In this paper, we focus on developing multivariate signature schemes with special properties and investigate how to build an MPKC identity-based scheme and then we will use this idea to build a series of MPKC identity-based schemes based on current promising MPKCs.

The structure of this paper is as follows: first, we will introduce how to build an ID-based MPKC scheme specifically. Then we propose two practical ID-based signature schemes: ID-based UOV and ID-based Rainbow. Next, we run some tests to verify the security and efficiency of our construction. Finally, we draw a conclusion.

2. PRELIMINARY

2.1. UOV and Rainbow

The UOV scheme is one of the earliest MPKC signature schemes. Even though its construction is very simple, it turns out to be one of the most secure MPKC scheme so far. On the other hand, Rainbow is one of the most popular schemes in MPKC schemes and rapid development in recent years. It could be regarded as an extension of UOV and has obvious advantages over efficiency and key size.

The central map F of UOV is composed of a set of so-called Oil–Vinegar polynomials which have the form

$$\sum_{i=1}^{o}\sum_{j=1}^{v} a_{ij}x_i x_j' + \sum_{i=1}^{v}\sum_{j=1}^{v} b_{ij}x_i' x_j' + \sum_{i=1}^{o} c_i x_i + \sum_{j=1}^{v} d_j x_j' + e. \quad (1)$$

In this polynomial, there are two kinds of variables: Oil variables (x_i) and Vinegar variables (x_j'). Once we assign a set of random values for Vinegar variables, the central map becomes a set of linear polynomials and can be easily inverted. When $v > o$, this scheme is called Unbalanced Oil and Vinegar scheme, the construction of a UOV scheme is as follows:

$$P = F \circ L_2, \quad (2)$$

where L_2 is an affine translation from $\mathbb{F}_q^n$ to $\mathbb{F}_q^n$. The construction does not have to compose an invertible affine transformation L_1 on the left.

On the other side, Rainbow is an extension of UOV scheme. It could be viewed as a multi-layer UOV scheme. Each layer is an independent UOV and each layer's variables (including Oil variables and Vinegar variables) are Vinegar variables of the next layer. Specifically, let us assume a Rainbow has u layers. We use v_i to represent the number of Vinegar variables of the ith layer and o_i to represent the number of Oil variables of the ith layer. Then we have $v_{i+1} = o_i + v_i$ and $v_{l+1} = n$. Each layer's Vinegar variables set and Oil variables set are represented as $\{x_1, ..., x_{v_i}\}$, $\{x_{v_i+1}, ..., x_{v_i+o_i}\}$ and the ith layer's polynomials have the form of

$$\sum_{i=1}^{v_i}\sum_{j=v_i+1}^{v_i+o_i} a_{ij}x_i x_j + \sum_{i=1}^{v_i}\sum_{j=1}^{v_i} b_{ij}x_i x_j + \sum_{i=1}^{v_i+o_i} c_i x_i + d \quad (3)$$

We can see that the above polynomial has the basic Oil–Vinegar polynomial form. Finally, the construction of a Rainbow scheme is as follows:

$$P = L_1 \circ F \circ L_2$$

Unlike UOV, to build the public key of Rainbow, a bijective linear transformation L_1 must be composited to cover the structure difference of different layers.

2.2. MPKC signature scheme

An MPKC signature scheme consists of the following algorithms: KeyGen, Sign and Verify.

Usually, an MPKC scheme over a finite field $\mathbb{F}_q$ is defined as

$$P = L_1 \circ F \circ L_2$$

in which F is a set of m quadratic multivariate polynomials in n variables, L_1 is an affine transformation from $\mathbb{F}_q^m$ to $\mathbb{F}_q^m$ and L_2 is an affine transformation from $\mathbb{F}_q^n$ to $\mathbb{F}_q^n$.

For an MPKC digital signature scheme, the setup algorithm Setup(1^λ), takes 1^λ as an input, and then outputs the system parameter *param* which mainly contains (n, m, q) and all the arithmetic operations hereafter are over this finite field.

The key generation algorithm KeyGen(*param*) takes *param* as an input, and then outputs $pk = \overline{F}$ and $sk = (L_1, F, L_2)$.

The signing algorithm Sign(M, L_1, F, L_2)) is described below.

Assume that the document needs to be signed is $M = (y_1, y_2, \ldots, y_m)$. First, the user who has access to private key calculates $M = L_1^{-1}(M)$, then it solves the equation: $M = F(X)$ and get X. Finally, it computes $S = L_2^{-1}(X)$.

Finally, the verification algorithm Verify(σ, M, P) returns one if $\overline{F}(\sigma) = M$, otherwise returns 0.

2.3. ID-based Signature Scheme

An ID-based signature scheme consists of the following algorithms: Setup, Extract, Sign and Verify.

- *Setup*: On an input of a security parameter k, it produces the master secret key msk and the common master public mpk, which include a description of a finite signature space and a description of a finite message space.
- *Extract*: On input of the signer's identity $ID \in \{0, 1\}^*$ and the master secret key msk, it outputs the signer's secret signing key usk_{ID}. (The corresponding public verification key upk_{ID} can be computed easily by everyone.)
- *Sign*: On input of a message m, a user's identities ID, and the secret keys of one members usk_{ID}, it outputs an ID-based signature σ on the message m.
- *Verify*: On input of a signature σ, a message m and the signers' identities ID, it outputs 1 for *true* or 0 for *false*, depending on whether σ is a valid signature signed by a certain member on a message m.

Security Concern for ID-based MPKC signature scheme: For MPKC signature schemes, since most of them cannot be provable secure, we need to define the security concern here. We say that as long as a ID-based MPKC scheme is secure from the current attacks on MPKC schemes, the ID-based scheme will satisfy the security for original signature. Then, besides the original security standards, it also has to resist ID attack that any user cannot impersonate as a key distributed center to extract other user's secret signing key. More precisely, any user cannot recover the master private key no matter how many secret signing key it has been extracted with different IDs.

3. THE GENERAL CONSTRUCTION OF ID-BASED MPKC SCHEMES

Assume that we have an MPKC signature scheme, we now describe our ID-based signature scheme as follows.

Suppose that the whole system consists of a trusted key distribution center (KDC), and a lot of users $U_1, U_2, \ldots, U_N$.

First, each user U_i should register to the KDC, and then the KDC issues an unique identifier to U_i, which is known to the public can be denoted by

$$ID(U_i) = (z_1, z_2, \ldots, z_d).$$

Let

$$L_1(x_1, \ldots, x_m) = (L_{1,1}(x_1, \ldots, x_m), \ldots, L_{1,n}(x_1, \ldots, x_m)),$$

where

$$L_{1,i}(x_1, \ldots, x_m) = \sum L_{(1,i,j)}(z_1, \ldots, z_d)x_j + L_{(1,i,0)}(z_1, \ldots, z_d),$$

where each $L_{(1,i,j)}(z_1, \ldots, z_d)$ is a linear function of $z_1, \ldots, z_d$.

For example, we can choose the following parameters: $m = 2$, $d = 2$. We let

$$L_1(x_1, x_2) = ((z_1 + 2z_2)x_1 + (2z_1 - z_2)x_2, (z_1 - z_2)x_1 + (z_1 + z_2)x_2).$$

Also let

$$L_2(x_1, \ldots, x_n) = (L_{2,1}(x_1, \ldots, x_n), \ldots, L_{2,n}(x_1, \ldots, x_n)),$$

where

$$L_{2,i}(x_1, \ldots, x_n) = \sum L_{(2,i,j)}(z_1, \ldots, z_d)x_j + L_{(2,i,0)}(z_1, \ldots, z_d),$$

where each $L_{(2,i,j)}(z_1, \ldots, z_d)$ is a linear function of $z_1, \ldots, z_d$.

Then by this method, we have constructed two affine transformation L_1 from $\mathbb{F}_q^m$ to $\mathbb{F}_q^m$ and L_2 from $\mathbb{F}_q^n$ to $\mathbb{F}_q^n$.

REMARK 1. The key point here is to remember that x_i are variables but z_i are parameters which will be determined by the user's ID.

After the above construction, let

$$F_l = \sum_{i,j} \alpha_{l_{ij}} x_i x_j + \sum_i \beta_{l_i} x_i + \gamma_l.$$

where $\alpha_{l_{ij}} = A_{l_{ij}}(z_1, z_2, \ldots, z_d)$, $\beta_{l_i} = B_{l_i}(z_1, z_2, \ldots, z_d)$, $\gamma_l = C_l(z_1, z_2, \ldots, z_d)$, are all linear functions of $z_1, \ldots, z_d$. Here F is a set of m quadratic multivariate polynomials in n variables.

Then, we compute the public key as

$$\overline{F} = L_1 \circ F \circ L_2 = \begin{pmatrix} \overline{F}_1(x_1, \ldots, x_n) \\ \overline{F}_2(x_1, \ldots, x_n) \\ \cdots \cdots \\ \overline{F}_m(x_1, \ldots, x_n) \end{pmatrix},$$

where each public key polynomial $\overline{F}_l$ can be represented as

$$\overline{F}_l = \sum_{i,j} \bar{\alpha}_{l_{ij}} x_i x_j + \sum_i \bar{\beta}_{l_i} x_i + \bar{\gamma}_l.$$

where $\bar{\alpha}_{l_{ij}} = a_{l_{ij}}(z_1, z_2, \ldots, z_d)$, $\bar{\beta}_{l_i} = b_{l_i}(z_1, z_2, \ldots, z_d)$, $\bar{\gamma}_l = c_l(z_1, z_2, \ldots, z_d)$, are thus of degree 4,3,2 polynomials of $z_1, \ldots, z_d$.

If the value of $z_1, \ldots, z_d$ is given, we can compute the public key polynomials easily.

REMARK 2. For the efficiency consideration, a good idea may be that we should have no linear or constant terms since the missing point of this part will not reduce the security level in MPKC schemes. But we think it is also ok since the master keys are only need to generate once.

Then, KDC generates a MPKC private key for the user U_i corresponding to U_i's identifier $ID(U_i)$ in a different way as follows.

For each user, we need to choose two linear transformations L_1' and L_2' such that the map $L_1' \circ F \circ L_2'$ can still be easily inverted. This is the key point of the construction, otherwise the collusion attack can break the system easily (we will discuss in Section 5). Not every MPKC has such a property, but Rainbow, UOV scheme which we are going to talk about in this paper have.

Then we know that

$$\overline{F} = L_1 \circ F \circ L_2 = L_1 \circ L_1'^{-1} \circ L_1' \circ F \circ L_2' \circ L_2'^{-1} \circ L_2,$$

and $L_1 \circ L_1'^{-1}$, $L_1' \circ F \circ L_2'$ and $L_2'^{-1} \circ L_2$ are given to the user as its private key, represented by L_{1u}, F_u, L_{2u}.

In former section, we mentioned a MPKC scheme is generally built as

$$P = L_1 \circ F \circ L_2$$

in which F is a set of m quadratic multivariate equations in n variables and L_1 is an affine transformation from $\mathbb{F}_q^m$ to $\mathbb{F}_q^m$ and L_2 is an affine transformation from $\mathbb{F}_q^n$ to $\mathbb{F}_q^n$. The whole system is built on a finite field $\mathbb{F}_q = GF(q)$. The goal of our idea is to generate different parts of a user's private key through its ID. Ergo, its public key can also be computed via its ID.

So far, we can describe the setup process of our general ID-based MPKC scheme:

Setup The scheme takes polynomials which are used to compute master public key as the master public key, i.e. $mpk = (\overline{\alpha}_{lij}, \overline{\beta}_{li}, \overline{\gamma}_l)$. The according master secret key should be linear transformations which are used to compute coefficients of private keys $L_{(1,i,j)}$, $L_{(1,i,0)}$, $L_{(2,i,j)}$, $L_{(2,i,0)}$, α_{lij}, β_{li}, γ_l and also specific L_1' and L_2' corresponding to every user. More precisely, the master secret key $msk = (L_{(1,i,j)}, L_{(1,i,0)}, L_{(2,i,j)}, L_{(2,i,0)}, \alpha_{lij}, \beta_{li}, \gamma_l, L_1', L_2')$.

Thereafter, the extraction process of our general ID-based MPKC scheme is

Extract Given an arbitrary identify of a specific user $ID_u = z_1, \ldots, z_d$, KDC can compute the public polynomials P by using the master public key mpk. Also, by using master secret key msk, KDC can compute private key F, L_1, L_2 and L_1', L_2' via this ID. Then, in the extract process of ID-UOV, KDC will extract the private key of specific user as $L_{1u} = L_1 \circ L_1'^{-1}$, $F_u = L_1' \circ F \circ L_2'$, $L_{2u} = L_2'^{-1} \circ L_2$. Finally, this private key will be distributed to this specific user ID_u.

Accordingly, the signature generation process of our general ID-based MPKC scheme is

Sign Assume that the document needs to be signed is $Y = (y_1, y_2, \ldots, y_m)$. First, the user who has access to private key calculates $M = L_{1u}^{-1}(Y)$, then it solves the equation: $M = F_u(X)$ and get X. Finally, it computes $S = L_{2u}^{-1}(X)$.

Finally, the signature verification process of our general ID-based MPKC scheme is

Verify This process is very simple. Assume that the signature needs to be verified is X. The user only needs to substitute ID to the public master public key mpk and compute the public polynomials $\overline{F}$, and compute $Y' = \overline{F}(X)$ to see if Y' equals Y.

REMARK 3. If d is too big, the verification of the system is slow so sometimes we need to hash the user's ID into fixed one. And how to arrange the user's ID is out of scope of this paper.

4. OUR ID-BASED SIGNATURE SCHEME

In the previous section, we described how to build an ID-based MPKC scheme generally via a user's ID. In this section, we will propose two ID-based schemes based on two well-known MPKC schemes: UOV [17] and Rainbow [18].

4.1. ID-UOV: our proposed ID-based UOV schemes

1. Setup Given the system parameter, in the setup process of ID-UOV, the scheme takes the coefficients of polynomials P which are used to compute master public key as the master public key, i.e. $mpk = (\overline{\alpha}_{lij}, \overline{\beta}_{li}, \overline{\gamma}_l)$. The according master secret key should be linear transformations which are used to compute coefficients of private keys $L_{(2,i,j)}$, $L_{(2,i,0)}$; α_{lij}, β_{li}, γ_l; and also specific L_2' corresponding to every user. More precisely, the master secret key $msk = (L_{(2,i,j)}, L_{(2,i,0)}, \alpha_{lij}, \beta_{li}, \gamma_l, L_2')$. Finally, the master public key will be publicly known, while the master secret key will be known only to the KDC.

2. Extract Given an arbitrary identify of a specific user $ID_u = z_1, \ldots, z_d$, KDC can compute the public polynomials P by using the master public key mpk. Also, by using master secret key msk, KDC can compute private key F, L_2 and L_2' via this ID. Then, in the extract process of ID-UOV, KDC will extract the private key of specific user as $F_u = F \circ L_2'$, $L_{2u} = L_2'^{-1} \circ L_2$. Finally, this private key (F_u), L_{2u} will be distributed to this specific user ID_u.

3. Sign Given the private key F_u and L_{2u} and the document need to be signed:M, in the setup process of ID-UOV, the scheme chooses v random variables as Vinegar variables and returns $X = \{x_1, \ldots, x_{o+v}\}$ as the legitimate signature by running a regular UOV signature process.

4. Verify To verify the signature X of document M on input the public key $ID = z_1, \ldots, z_d$, all we need to do is substituting X to the public master public key mpk and compute the public polynomials P. Then, in the verify process of ID-UOV, the scheme check if $P(X) = M$, this signature is valid once it is true. Otherwise, it's not.

Next, also the extremely important one of our construction, we should choose an appropriate L_2' that could preserve the special structure of UOV scheme we base on. More precisely, after composing L_2', the polynomials in the new central map

should still stay in the form of Oil–Vinegar polynomials. Now we represent a UOV scheme's central polynomial by its corresponding matrix, and the Vinegar variables are denoted by its first $v = 56$ Variables. Then the matrices of the polynomials in central equation should be in the form of Fig. 1.

In Fig. 1, the gray areas represent the random entries while blank areas denote zero entries. The rest part of this paper follows the same rules.

Thereby, our problem is transformed to choose a L_2' that could keep the shape of the above matrix of central equation. To achieve that goal, we could pick a invertible affine transformation of the form as is shown in Fig. 2.

Once $L_{2'}$ is choosing in this form, we will get that $F_u = F \circ L_{2'}$, which means that the matrices form of central map F_u is

$$F_u = \begin{bmatrix} *_{v\times v} & *_{v\times o} \\ *_{o\times v} & 0_{o\times o} \end{bmatrix} \begin{bmatrix} *_{v\times v} & 0_{v\times o} \\ *_{o\times v} & *_{o\times o} \end{bmatrix} = \begin{bmatrix} *_{v\times v} & *_{v\times o} \\ *_{o\times v} & 0_{o\times o} \end{bmatrix}.$$

Thus, this will make sure that the map F_u can still be easily inverted.

4.2. ID-Rainbow: our proposed ID-based Rainbow schemes

1. Setup Given the system parameter, in the setup process of ID-Rainbow, the scheme takes polynomials which are used to compute master public key as the master public key, i.e. $mpk = (\overline{\alpha}_{lij}, \overline{\beta}_{li}, \overline{\gamma}_l)$. The according master secret key should be linear transformations which are used to compute coefficients of private keys $L_{(1,i,j)}$, $L_{(1,i,0)}$, $L_{(2,i,j)}$, $L_{(2,i,0)}$, α_{lij}, β_{li}, γ_l and also specific L_1' and L_2' corresponding to every user. More precisely, the master secret key $msk = (L_{(1,i,j)}, L_{(1,i,0)}, L_{(2,i,j)}, L_{(2,i,0)}, \alpha_{lij}, \beta_{li}, \gamma_l, L_1', L_2')$. Finally, the master public key will be publicly known, while the master secret key will be known only to the KDC.

2. Extract Given an arbitrary identify of a specific user $ID_u = z_1, \ldots, z_d$, KDC can compute the public polynomials P by using the master public key mpk. Also, by using master secret key msk, KDC can compute private key F, L_1, L_2 and L_1', L_2' via this ID. Then, in the extract process of ID-UOV, KDC will extract the private key of specific user as $L_{1u} = L_1 \circ L_1'^{-1}$, $F_u = L_{1'} \circ F \circ L_2'$, $L_{2u} = L_2'^{-1} \circ L_2$. Finally, this private key will be distributed to this specific user ID_u.

3. Sign Given the private key L_{1u}, F_u and L_{2u} and the document needs to be signed: M, in the setup process of ID-Rainbow, the scheme chooses random variables as Vinegar variables and returns $X = \{x_1, \ldots, x_{o+v}\}$ as the legitimate signature by running a regular Rainbow signature process.

4. Verify To verify the signature X of document M on input the public key $ID = z_1, \ldots, z_d$, all we need to do is substitute X to the public master public key mpk and compute the public polynomials P. Then, in the verify process of ID-Rainbow, the scheme check if $P(X) = M$, this signature is valid once it is true. Otherwise, it is not.

Similarly, the Rainbow's corresponding matrices have the following forms of Fig. 3.

Moreover, to keep the multi-layer structure of central equation, the structure of L_1' and L_2' should have the shapes as shown in Figs. 4 and 5.

Similarly, once L_1' and L_2' are chosen in these forms, we will get that $F_u = L_1' \circ F \circ L_2'$, which means that the matrices form of central map F_u is also the same as F. Thus, this will make sure that the map F_u can still be easily inverted.

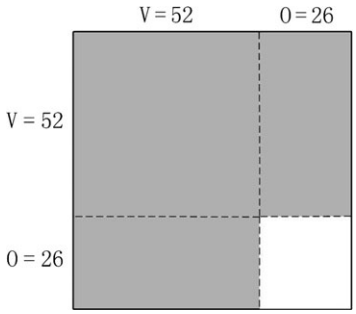

FIGURE 1. Oil–Vinegar scheme corresponding matrix.

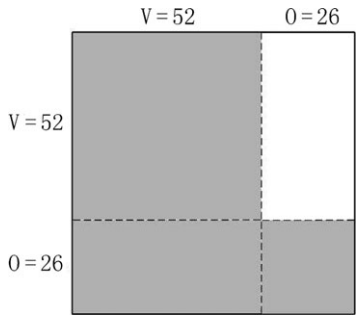

FIGURE 2. L_2' for ID-based Oil–Vinegar scheme.

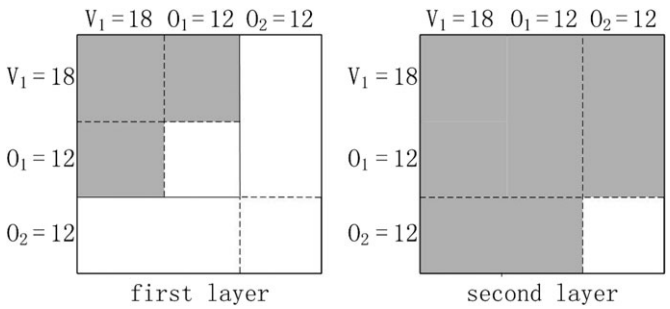

FIGURE 3. Rainbow scheme corresponding matrix.

4.3. A toy example of ID-based MPKC scheme

In this section, we present a toy example of ID-UOV with $o = 2$, $v = 2$, $q = 4$. The general UOV does not have to composite affine transformation L_1 on the left side of central equation. However, to illustrate our ID-based idea better, we still bring in L_1 in this example.

Let $GF(2^2)$ be the finite field with four elements. The multiplicative group for the non-zero elements of this field can be generated by the field element α which satisfies $\alpha^2 + \alpha + 1 = 0$. The field elements of can be presented as $\{0, 1, \alpha, \alpha^2\}$.

In this case, L_1' can be any invertible linear transformation and L_2' must be chosen in the way that $F \circ L_2'$ is still in the form of unbalanced Oil–Vinegar map.

For example,

$$L_1(x_1, x_2) = \begin{pmatrix} \alpha z_2 & \alpha z_1 + z_2 \\ z_1 & z_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$$L_2(x_1, x_2, x_3, x_4) = \begin{pmatrix} z_1 & 0 & 0 & 0 \\ 0 & z_2 & \alpha z_1 & 0 \\ z_1 + z_2 & 0 & \alpha z_2 & 0 \\ z_2 & 0 & 0 & z_1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}$$

$$F = \begin{pmatrix} F_1 \\ F_2 \end{pmatrix}$$

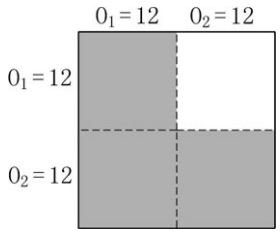

FIGURE 4. L_1' for ID-based Rainbow scheme.

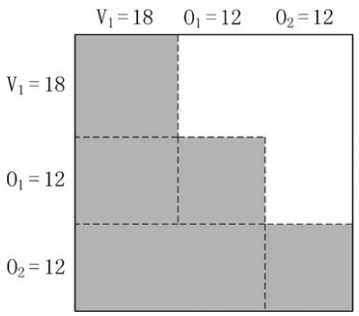

FIGURE 5. L_2' for ID-based Rainbow scheme.

$$\begin{aligned} F_1 = {} & z_2x_3^2 + (\alpha z_1 + z_2)x_4^2 + (\alpha^2 z_1 + \alpha z_2)x_3x_4 \\ & + (z_1 + z_2)x_1x_3 + (\alpha z_1 + \alpha^2 z_2)x_1x_4 \\ & + \alpha^2 z_1x_2x_3 + (z_1 + \alpha^2 z_2)x_2x_4 \\ & + (\alpha z_1 + \alpha z_2)x_1 + (z_1 + \alpha z_2)x_2 + z_1x_3 \\ & + (\alpha^2 z_1 + \alpha^2 z_2)x_4 + (\alpha^2 z_1 + z_2) \end{aligned}$$

$$\begin{aligned} F_2 = {} & (z_1 + z_2)x_3^2 + \alpha z_1x_4^2 + \alpha^2 z_2x_3x_4 + z_1x_1x_3 \\ & + z_2x_1x_4 + (\alpha z_1 + \alpha^2 z_2)x_2x_3 \\ & + \alpha^2 z_1x_2x_4 + (\alpha^2 z_1 + z_2)x_1 + \alpha z_1x_2 \\ & + (\alpha z_1 + \alpha z_2)x_3 + \alpha z_2x_4 + (z_1 + \alpha z_2) \end{aligned}$$

Then

$$\bar{F} = L_1 \circ F \circ L_2 = \begin{pmatrix} \bar{F}_1 \\ \bar{F}_2 \end{pmatrix}$$

$$\begin{aligned} \bar{F}_1 = {} & (\alpha z_1^3 z_2 + \alpha z_1^2 z_2^2 + z_2^4)x_1^2 + (\alpha^2 z_1^3 z_2 \\ & + \alpha z_2^4)x_1x_2 + (z_1^4 + a^2 z_1^3 z_2 + z_1^2 z_2^2)x_1x_3 + z_1^3 z_2 x_1x_4 \\ & + (z_1^2 z_2^2 + a^2 z_1 z_2^3 + z_2^4)x_2x_3 \\ & + (z_1^3 z_2 + z_1^2 z_2^2 + z_1 z_2^3)x_2x_4 \\ & + (\alpha z_1^3 z_2 + \alpha z_2^4)x_3^2 \\ & + (az_1^4 + \alpha z_1^3 z_2 + \alpha z_1^2 z_2^2)x_3x_4 \\ & + (\alpha^2 z_1^4 + z_1^3 z_2 + \alpha z_1^2 z_2^2)x_4^2 \\ & + (\alpha z_1^3 + \alpha z_1^2 z_2 + z_1 z_2^2 + z_2^3)x_1 \\ & + (\alpha^2 z_1^2 z_2 + \alpha^2 z_2^3)x_2 \\ & + (z_1^3 + z_1^2 z_2 + \alpha^2 z_2^3)x_3 \\ & + (\alpha z_1^2 z_2 + \alpha^2 z_1 z_2^2)x_4 \\ & + (\alpha z_1^2 + \alpha^2 z_1 z_2) \end{aligned}$$

$$\begin{aligned} \bar{F}_2 = {} & (z_1^4 + z_1^2 z_2^2 + \alpha^2 z_1 z_2^3 + \alpha z_2^4)x_1^2 \\ & + (\alpha^2 z_1^3 z_2 + z_1 z_2^3 + \alpha^2 z_2^4)x_1x_2 \\ & + (z_1^4 + \alpha z_1^3 z_2 + \alpha^2 z_1^2 z_2^2 + \alpha z_1 z_2^3 + z_2^4)x_1x_3 \\ & + (z_1^4 + \alpha z_1^3 z_2 + \alpha^2 z_1 z_2^3)x_1x_4 \\ & + (z_1^2 z_2^2 + \alpha^2 z_1 z_2^3 + z_2^4)x_2x_3 \\ & + z_1^3 z_2 x_2x_4 + (\alpha z_1^3 z_2 + \alpha z_1 z_2^3 \\ & + z_1^2 z_2^2 + \alpha^2 z_2^4)x_3^2 + (\alpha z_1^4 + z_1^3 z_2 \\ & + \alpha^2 z_1^2 z_2^2 + z_1 z_2^3)x_3x_4 \\ & + (\alpha z_1^4 + \alpha^2 z_1^3 z_2)x_4^2 + (\alpha^2 z_1^3 + z_1^2 z_2 \\ & + \alpha z_1 z_2^2)x_1 + z_1^2 z_2 x_2 \\ & + (\alpha z_1^3 + \alpha z_1^2 z_2 + \alpha^2 z_1 z_2^2 + \alpha^2 z_2^3)x_3 \\ & + (\alpha^2 z_1^3 + \alpha^2 z_1^2 z_2 + \alpha z_1 z_2^2)x_4 \\ & + (\alpha^2 z_1^2 + \alpha z_2^2) \end{aligned}$$

Now we choose the two linear transformations L_1' and L_2',

$$L_1' = \begin{pmatrix} \alpha & \alpha \\ 1 & \alpha^2 \end{pmatrix}$$

$$L_2' = \begin{pmatrix} 1 & 0 & \alpha & \alpha^2 \\ 0 & 1 & 0 & \alpha^2 \\ \alpha^2 & \alpha & 0 & 0 \\ 0 & \alpha^2 & 0 & 0 \end{pmatrix}$$

then compute

$$L_1'^{-1}$$

and

$$L_2'^{-1}$$

$$L_1'^{-1} = \begin{pmatrix} 1 & \alpha^2 \\ \alpha & \alpha^2 \end{pmatrix}$$

$$L_2'^{-1} = \begin{pmatrix} 0 & 0 & \alpha & 1 \\ 0 & 0 & 0 & \alpha \\ \alpha^2 & \alpha^2 & 1 & \alpha \\ 0 & \alpha & 0 & \alpha^2 \end{pmatrix}$$

then compute the private key $L_1 \circ L_1'^{-1}$, $L_1' \circ F \circ L_2'$ and $L_2'^{-1} \circ L_2$.

$$L_{u1} = L_1 \circ L_1'^{-1} = \begin{pmatrix} \alpha^2 z_1 & z_1 + \alpha z_2 \\ z_1 + \alpha z_2 & \alpha^2 z_1 + \alpha^2 z_2 \end{pmatrix}$$

$$L_{u2} = L_2'^{-1} \circ L_2 = \begin{pmatrix} \alpha z_1 + \alpha^2 z_2 & 0 & \alpha^2 z_2 & z_1 \\ \alpha z_2 & 0 & 0 & \alpha z_1 \\ \alpha z_1 + \alpha^2 z_2 & \alpha^2 z_2 & z_1 + \alpha z_2 & \alpha z_1 \\ \alpha^2 z_2 & \alpha z_2 & \alpha^2 z_1 & \alpha^2 z_1 \end{pmatrix}$$

$$F_u = L_1' \circ F \circ L_2' = \begin{pmatrix} F_{u1} \\ F_{u2} \end{pmatrix}$$

$$\begin{aligned} F_{u1} = {} & (\alpha^2 z_1 + z_2)x_1^2 + (z_1 + z_2)x_1x_2 + \alpha z_2 x_1 x_3 \\ & + (\alpha^2 z_1 + z_2)x_1x_4 + (z_1 + \alpha^2 z_2)x_1 \\ & + z_1 x_2^2 + (\alpha^2 z_1 + \alpha z_2)x_2x_3 + \alpha z_1 x_2 x_4 + \alpha^2 z_2 x_2 \\ & + (\alpha^2 z_1 + \alpha z_2)x_3 + (\alpha z_1 + z_2)x_4 + (\alpha^2 z_1 + z_2) \end{aligned}$$

$$\begin{aligned} F_{u2} = {} & (\alpha z_1 + \alpha^2 z_2)x_1x_2 \\ & + (\alpha z_1 + z_2)x_1x_3 + z_2 x_1 x_4 + \alpha z_2 x_1 \\ & + (z_1 + \alpha z_2)x_2^2 + (\alpha^2 z_1 + \alpha^2 z_2)x_2x_3 \\ & + (\alpha z_1 + \alpha z_2)x_2x_4 + (\alpha z_1 + z_2)x_2 \\ & + \alpha z_2 x_3 + \alpha z_2 x_4 \end{aligned}$$

and the public keys are

$$\begin{aligned} \bar{F} &= L_{u1} \circ F_u \circ L_{u2} \\ &= L_1 \circ L_1'^{-1} \circ L_1' \circ F \circ L_2' \circ L_2'^{-1} \circ L_2 = \begin{pmatrix} \bar{F}_1 \\ \bar{F}_2 \end{pmatrix} \end{aligned}$$

$$\begin{aligned} \bar{F}_1 = {} & (\alpha z_1^3 z_2 + \alpha z_1^2 z_2^2 + z_2^4)x_1^2 \\ & + (\alpha^2 z_1^3 z_2 + \alpha z_2^4)x_1x_2 \\ & + (z_1^4 + a^2 z_1^3 z_2 + z_1^2 z_2^2)x_1x_3 + z_1^3 z_2 x_1 x_4 \\ & + (z_1^2 z_2^2 + a^2 z_1 z_2^3 + z_2^4)x_2x_3 \\ & + (z_1^3 z_2 + z_1^2 z_2^2 + z_1 z_2^3)x_2x_4 \\ & + (\alpha z_1^3 z_2 + \alpha z_2^4)x_3^2 \\ & + (a z_1^4 + \alpha z_1^3 z_2 + \alpha z_1^2 z_2^2)x_3x_4 \\ & + (\alpha^2 z_1^4 + z_1^3 z_2 + \alpha z_1^2 z_2^2)x_4^2 \\ & + (\alpha z_1^3 + \alpha z_1^2 z_2 + z_1 z_2^2 + z_2^3)x_1 \\ & + (\alpha^2 z_1^2 z_2 + \alpha^2 z_2^3)x_2 \\ & + (z_1^3 + z_1^2 z_2 + \alpha^2 z_2^3)x_3 \\ & + (\alpha z_1^2 z_2 + \alpha^2 z_1 z_2^2)x_4 \\ & + (\alpha z_1^2 + \alpha^2 z_1 z_2) \end{aligned}$$

$$\begin{aligned} \bar{F}_2 = {} & (z_1^4 + z_1^2 z_2^2 + \alpha^2 z_1 z_2^3 + \alpha z_2^4)x_1^2 \\ & + (\alpha^2 z_1^3 z_2 + z_1 z_2^3 + \alpha^2 z_2^4)x_1x_2 \\ & + (z_1^4 + \alpha z_1^3 z_2 + \alpha^2 z_1^2 z_2^2 + \alpha z_1 z_2^3 + z_2^4) \\ & x_1x_3 + (z_1^4 + \alpha z_1^3 z_2 + \alpha^2 z_1 z_2^3)x_1x_4 \\ & + (z_1^2 z_2^2 + \alpha^2 z_1 z_2^3 + z_2^4)x_2x_3 \\ & + z_1^3 z_2 x_2 x_4 + (\alpha z_1^3 z_2 + \alpha z_1 z_2^3 + z_1^2 z_2^2 \\ & + \alpha^2 z_2^4)x_3^2 + (\alpha z_1^4 + z_1^3 z_2 \\ & + \alpha^2 z_1^2 z_2^2 + z_1 z_2^3)x_3x_4 \\ & + (\alpha z_1^4 + \alpha^2 z_1^3 z_2)x_4^2 \\ & + (\alpha^2 z_1^3 + z_1^2 z_2 + \alpha z_1 z_2^2)x_1 \\ & + z_1^2 z_2 x_2 + (\alpha z_1^3 + \alpha z_1^2 z_2 + \alpha^2 z_1 z_2^2 + \alpha^2 z_2^3)x_3 \\ & + (\alpha^2 z_1^3 + \alpha^2 z_1^2 z_2 + \alpha z_1 z_2^2)x_4 \\ & + (\alpha^2 z_1^2 + \alpha z_2^2) \end{aligned}$$

they are the same as the original public keys. Then we chose a user u,

$$ID(U_u) = (1, \alpha).$$

then his private keys are

$$L_{u1} = \begin{pmatrix} \alpha^2 & \alpha \\ \alpha & \alpha \end{pmatrix}$$

$$L_{u2} = \begin{pmatrix} \alpha^2 & 0 & 1 & 1 \\ \alpha^2 & 0 & 0 & \alpha \\ \alpha^2 & 1 & \alpha & \alpha \\ 1 & \alpha^2 & \alpha^2 & \alpha^2 \end{pmatrix}$$

$$F_{u1} = x_1^2 + \alpha^2 x_1 x_2 + \alpha^2 x_1 x_3 + x_1 x_4 + x_2^2 + \alpha x_2 x_4 + x_2 + 1$$

$$F_{u2} = \alpha^2 x_1 x_2 + \alpha x_1 x_4 + \alpha^2 x_1 + \alpha x_2^2 + \alpha x_2 x_3 + x_2 x_4 + \alpha^2 x_3 + \alpha^2 x_4$$

and public keys are

$$\bar{F}_1 = \alpha x_1 x_2 + \alpha^2 x_1 x_3 + \alpha x_1 x_4 + \alpha x_2 x_3 + \alpha^2 x_1 + \alpha x_2 + x_4 + \alpha^2$$

$$\bar{F}_2 = \alpha x_1^2 + x_1 x_2 + x_1 x_4 + \alpha x_2 x_3 + \alpha x_2 x_4 + \alpha^2 x_3^2 + \alpha^2 x_3 x_4 + \alpha^2 x_4^2 + \alpha x_2 + \alpha^2 x_4 + \alpha$$

5. SECURITY ANALYSIS

5.1. Resistance to ID attack

The added two linear transformations L_1' and L_2' can help our ID-based scheme to be secure from ID attacks. Without L_1' and L_2', each coefficient of the master secret key is just linear combinations of the ID's d elements: $z_1, \ldots, z_d$. Thereby, if $\mathcal{A}$ choose d IDs to query and get d secret signing keys, $\mathcal{A}$ can solve these linear transformations. Therefore, given any ID, $\mathcal{A}$ can generate an identical private key and legitimate signature. After bringing in L_1' and L_2'. The secret signing key given to user will be totally different from before. We will analysis it for the two proposed schemes respectively.

For UOV, if $\mathcal{A}$ can figure out how to compute L_{2u} which is the easiest part, then the whole system will be broken. However, after bringing into L_2', the computation of each private key's coefficients will be quadratic polynomials of $n_v = n^2 \times d + n^2 - o^2$ variables in L_{2u}. Since we can extract unlimited number of equations, so finally we will have to solve an overdetermined polynomial system. In this case, we could use XL algorithm [19] to solve this system (the complexity will discuss in Section 5.4). Normally, if the Adversary can extract more equations, this system will also be easier to solve. According to the conclusion in [19], if $\mathcal{A}$ can extract $n_v^2/2$ linearly independent equations, the attack will be most efficient. However, in real application, the number of equations extracted is dependent on the others and extract so many linear independent equations will be at extremely high cost. Even in this case, we give $\mathcal{A}$ the capacity to extract linear independent equations. The whole complexity includes extracting equations and solving this system of equations which is around $(n_v \times (n_v - 1)/2)^3$ if $\mathcal{A}$ uses general Gauss Elimination. Thus, the whole complexity of cracking this ID-based signature system is around $n_v^8/4$. Thereby, it is secure from ID-attack.

For Rainbow, the easiest part is to figure out L_{1u}. After bringing into L_2' and L_1', the computation of L_{1u}'s each coefficients will be quadratic polynomials in $n_v = m^2 \times d + m^2 - o_2^2$ variables. The adversary also uses XL algorithm and the whole complexity of cracking this ID-based signature system is around $n_v^8/4$. So this scheme is also secure from ID attack.

REMARK 4. Upon this analysis, we can choose a proper small d for our schemes.

On the other side, the resultant public key and secret key are both on the same form of the basic schemes, which will face with the current attack technique on MPKC, we will do this security analysis of our signature schemes in the following subsections. According to the following analysis, our ID-based schemes are secure from the current attacks on the MPKC signature.

Next, we briefly describe the current attacks of MPKC on our schemes as follows.

5.2. The Kipnis and Shamir attack

The Kipnis and Shamir (KS) attack [20] is first proposed by Kipnis and Shamir to attack the balanced Oil and Vinegar (OV) scheme. The goal of this attack is to find the pre-image of the Oil subspace $O = \{x \in K_n : x_1 = \ldots = x_v = 0\}$ under the affine invertible transformation T. To achieve this, it forms a random linear combination $P = \sum_{j=1}^{o} \beta_j H_j$, multiplies it with the inverse of one of the H_i and figures out the invariant subspaces of this matrix.

Now, we take a look at this attack to ID-UOV scheme, as described in the recent technique [21], the Kipnis and Shamir attack takes time about $O(q^{v-o-1}o^4)$ to break a (q, v, o)-UOV scheme. Also, while applying such attack to Rainbow, it treats all the polynomials of Rainbow as the polynomials of the last layer which have v_u Vinegar variables and o_u Oil variables. On the other side, this attack cannot cause any security threat to ID-Rainbow and the complexity of this attack is $q^{n-2o_u-1}o_u^4$ [22].

5.3. MinRank attack

MinRank attack is based on the so-called MinRank problem [23] and an effective algorithm [24] which could solve this

problem. The essence of the MinRank attack is finding a linear combination of the public multivariate polynomials' corresponding symmetric matrices which has minimum rank (precisely, it does not have to be the minimum rank but it should be less than the largest possible minimum rank). Let H_i be the symmetric matrix representing the homogenous quadratic part of the ith public polynomial. In the MinRank attack, one tries to find linear combinations $H = \sum_{i=1}^{m} \alpha_i H_i$ of the matrices representing the homogeneous quadratic parts of the public polynomials such that $\text{rank}(H) = r \leq n$. In the case of ID-UOV, one can find that all the matrices Q_i representing the homogeneous quadratic parts of the central equations have full rank n. And this prevents the MinRank attack. More precisely, the full rank rate in the associated central symmetric matrix of ID-UOV is close to $\frac{q^{n(n-1)/2}\prod_{i=1}^{n}(q^i-1)}{q^{n^2}}$.

However, ID-Rainbow has the possible minimum rank $v_1 + o_1$. So, the attack can find a linear combination which has rank of at most $v_1 + o_1$. Such a polynomial would be a linear combination of the o_1 polynomials from the first layer of Oil and Vinegar scheme.

Below we take typical parameters of Rainbow with $u = 4$ layers $(v_1, o_1, o_2, o_3, o_4)$ for example, all the elements are in $\mathbb{F}_q$.

The first layer of Rainbow is a balanced Oil and Vinegar scheme with o_1 Oil and v_1 Vinegar variables in each polynomial. The corresponding matrix M of the central polynomial of the first layer has the following structure:

$$\begin{pmatrix} *_{v_1\times v_1} & *_{v_1\times o_1} & 0_{v_1\times(o_2+o_3+o_4)} \\ *_{o_1\times v_1} & 0_{o_1\times o_1} & 0_{o_1\times(o_2+o_3+o_4)} \\ 0_{(o_2+o_3+o_4)\times v_1} & 0_{(o_2+o_3+o_4)\times o_1} & 0_{(o_2+o_3+o_4)\times(o_2+o_3+o_4)} \end{pmatrix}$$

where $*$ represents random value in $\mathbb{F}_q$ and 0 represents 0 in $\mathbb{F}_q$ and this is a matrix has rank of at most $v_1 + o_1$.

Now, in the MinRank attack, we represent a quadratic multivariate polynomial by its associated symmetric matrix.

Assuming that the matrices corresponding to the public key are $Q_1, \ldots, Q_m$, we need to find the linear combination

$$M = \sum_{k=1}^{m} \lambda_k Q_k \tag{5}$$

has minimum rank r.

To show how to find an above linear combination, the key idea is to find a vector in the kernel space of the desired linear combination. The probability that a random chosen vector w lies in the kernel of M is $1/q^r$. Also if a vector w lies in the kernel of M, it will satisfy the equation

$$\left(\sum_{k=1}^{m} \lambda_k Q_k\right) w = Mw = 0 \tag{6}$$

Which is linear in the m unknowns $\lambda_1, \ldots, \lambda_m$. As $m < n$, we could solve $\lambda_1, \ldots, \lambda_m$ using this equation (this process needs complexity m^3) and the solution to test if M has rank r. If so, we find the desired linear combination, if not, we need to generate another vector w and do the process again. The probability that the matrix M has rank r is equal to

$$\left(1 - \frac{1}{q^6}\right) \times \left(1 - \frac{1}{q^5}\right) \times \cdots \times \left(1 - \frac{1}{q}\right) < 1 - \frac{1}{q}. \tag{7}$$

So finally the total complexity of MinRank attack on ID-Rainbow is estimated by $q^{r+1} \times m^3$.

5.4. Direct attack

There are many direct attack algorithms working on MPKCs, such as XL [19] and Gröbner Basis algorithms such as F_4 [25] and F_5 [26]. The idea of direct attack on our scheme is to add $n - m$ linear equations. In this way, the number of variables can be reduced to m so as to create a determined system. On the other hand, a system with n variables and m equations is expected to have $q^{(n-m)}$ solutions on average. Therefore, adding a total of $n - m$ linear equations will lead to one solution on average. Repeating this experiment a few times, we will find at least one solution.

For the complexity of these algorithms, in [27], Bettale *et al.* asserted that, for a semi-regular system, the computational complexity of F_4 is bounded by $\mathcal{O}\left(\left(t\binom{n + d_{reg} - 1}{d_{reg}}\right)^{\omega}\right)$, where n is the number of variables, t is the number of equations, ω is a linear algebra constant and $2 \leq \omega \leq 3$, in general we set $\omega = 2$ for lower bound complexity and $\omega = 3$ for upper bound complexity. d_{reg} is the degree of regularity of the system, which is the index of the first non-positive coefficient in the Hilbert series $S_{m,n}$ with

$$S_{m,n} = \frac{\prod_{i=1}^{m}(1 - z^{d_i})}{(1 - z)^n},$$

where d_i is the degree of the ith equation.

Among all the direct attacks algorithm, the Hybrid F_5 (HF_5) algorithm [27] which is currently the fastest algorithm to solve the problem. The main idea is to guess some of the variables to create overdetermined systems before applying F_5 algorithm. Thus, one has to run the F_5 algorithm several times (depending on how many variables he/she guesses) to find a solution of the original system. When guessing u variables over $\mathbb{F}_q$, this number is given by q^u. The complexity of solving a semi-regular system of t multivariate equations in n variables over $\mathbb{F}_q$ by the HF_5 algorithm can be estimated as $q^u\mathcal{O}\left(\left(t\binom{n - u + d_{reg} - 1}{d_{reg}}\right)^{\omega}\right)$.

In fact, the best direct attack algorithm is hybrid algorithm HF_5 for medium fields, and Gröbner Basis algorithms F_4(or F_5) for large fields to solve multivariate polynomial equations.

When the underground field is small and assume that $m = \varepsilon n^2$, $\varepsilon > 0$, as is discussed in [19], the best algorithm is XL algorithm, and the complexity is $\mathcal{O}((n)^{\omega D}/D!)$, where $D \approx \lceil 1/\sqrt{\varepsilon} \rceil$.

5.5. UOV reconciliation attack and Rainbow band separation attack

UOV reconciliation attack [28] could viewed as an improved version of direct attack. It tries to find a sequence of basis that could transform the public key of UOV into the central Oil–Vinegar form. However, the main part of this attack is still direct attack. Its complexity could be transformed into directly solving a quadratic system of $m = o$ equations in v variables. For a regular UOV, since $v > o$, directly solving public key of UOV or using reconciliation attack could all be transferred to directly solving an under-defined system (the number of variables is greater than the number of equations). Before applying direct attack to an under-defined system, one should assign random values to variables to make the whole system a generic one or overdefined one [27]. Consequently, reconciliation attack against UOV is as difficult as a direct attack against it since both of them end up with solving a generic or over-defined system of quadratic equations with the same number of equations.

Below we briefly describe how UOV reconciliation attack works on attacking ID-UOV.

In a reconciliation attack, it tries to find a sequence of basis which would help to invert the public map.

Recall that the central polynomials of UOV can be represented in a symmetric matrix form as follows:

$$M_i = \begin{bmatrix} *_{v\times v} & *_{v\times o} \\ *_{o\times v} & 0_{o\times o} \end{bmatrix}$$

In addition, the invertible affine transformation L's corresponding matrix could be decomposed to

$$\begin{aligned} M_L &= \begin{bmatrix} *_{v\times v} & *_{v\times o} \\ *_{o\times v} & *_{o\times o} \end{bmatrix} \\ &= \begin{bmatrix} I_{v\times v} & *_{v\times o} \\ 0_{o\times v} & I_{o\times o} \end{bmatrix} \begin{bmatrix} *_{v\times v} & 0_{v\times o} \\ *_{o\times v} & *_{o\times o} \end{bmatrix} \end{aligned}$$

where I means identity sub-matrix. Let $M_{L'} = \begin{bmatrix} I_{v\times v} & *_{v\times o} \\ 0_{o\times v} & I_{o\times o} \end{bmatrix}$.

Instead of finding the original M_L, the attacker using this attack try to solve an equivalent $M_{L'}$ which will transform the public map into Oil–Vinegar form since $\begin{bmatrix} *_{v\times v} & 0_{v\times o} \\ *_{o\times v} & *_{o\times o} \end{bmatrix}$ will not affect its structural form.

Then $M_{L'}$ can be further decomposed as $M_{L'} = P_{v+1}P_{v+2}\cdots P_n$ in which

$$P_n = I_n + \left[\begin{array}{ccc|c} 0 & \cdots & 0 & a_1 \\ 0 & \cdots & 0 & a_2 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & a_v \\ \hline 0 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 \end{array}\right];$$

Which means that $M_{L'}$'s factor P_n will make the lower right 1×1 sub-matrix be zero, P_{n-1} will make the lower right 2×2 be zero and so on. We thus can solve $M_{L'}$ by solving P_n, $P_{n-1}\cdots P_{v+1}$ one by one. The process of UOV reconciliation attack becomes

(1) Perform basis change $x_i = x_i' - a_i x_n'$ for $i = 1\cdots v$, and $x_i = x_i'$, $i = v + 1\cdots n$. Evaluate the public polynomial by substituting this new basis x'.
(2) Let all coefficient $(x_n')^2$ be zero and use direct attack algorithm like F_4, F_5 to solve a_i. There will be $m = o$ equations in v variables (as the beginning of this section says).
(3) Repeat the process to find P_{n-1}. That is, we then set $x_i' = x_i'' - a_i' x_n''$ for $i = 1\cdots v$. Under this new basis, every ${x_{n-1}''}^2$ and $x_{n-1}'' x_n''$ will be zero. This will yield $2m$ equations in v variables.
(4) Repeat this process to find $P_{n-2}, \ldots, P_{v+1}$. Then we get $M_{L'}$.

The complexity of this attack is determined by the first two steps. The complexity of substituting a new basis x' is v^3, and the complexity of using direct attack algorithm to solve a system with m equations in v variables can be find in Section 5.4.

Rainbow band separation (RBS) attack is an enhancement of reconciliation attack against UOV to attack Rainbow. Below we also discuss it.

UOV reconciliation attack could be used directly to a Rainbow scheme since it could be viewed as a UOV of the last layer. However, Rainbow's multiple-layer structure brings into more disadvantages. The Reconciliation attack can be improved to RBS attack. More precisely, The matrix representation of Rainbow's layer structure is as follows:

$$M_i = \begin{bmatrix} *_{v\times v} & 0_{v\times o} \\ 0_{o\times v} & 0_{o\times o} \end{bmatrix} \quad \text{for } i < m - o;$$

$$M_i = \begin{bmatrix} *_{v\times v} & *_{v\times o} \\ *_{o\times v} & 0_{o\times o} \end{bmatrix} \quad \text{for others;}$$

Which means only the last o equations has the normal UOV structure. The first $m - o$ equations have more zeroes to explore. Thus, the RBS attack will take the above linear transformation L into consideration. Since P_n recover the last variable x_n in central equations. Recover a row of L and P_n would make a zero column at one equation. Since the number of non-zero last column is o, if we pick any $o + 1$ columns, we expect to find a linear combination of o columns that could cancel the non-zero entries of the other column.

So the process of RBS attack on our ID-Rainbow can be described as

(1) Perform basis change $x_i = x_i' - a_i x_n'$ for $i = 1 \cdots v$, and $x_i = x_i', i = v + 1 \cdots n$. Evaluate the public polynomial by substituting this new basis x'. Note that in this processes the number of o in Rainbow is much smaller than that in UOV.
(2) Let all coefficient $(x_n')^2$ be zero and get $m = o$ equations in v variables (these two process are the same as UOV reconciliation attack but we do not solve it right now since we can have more equations as follows).
(3) Make a linear combination of public polynomials $z_1' = z_1 - \alpha_1^{(1)} z_{v+1} - \alpha_2^{(1)} z_{v+2} - \cdots - \alpha_o^{(1)} z_m$ by using the new o variables $\alpha_i^{(1)}$, there will be $n - 1$ more equations since cross-terms involving x_n' are set to zeros.
(4) Together there are $m + n - 1$ equations and $n = o + v$ variables in total and can be solved by direct attack algorithm like F_4 and the last v part of the solution is a_i. Thus, we find P_n (direct attack in this situation is more efficient since it is an over-defined system).
(5) Repeat the process to find P_{n-1}. That is, set $x_i' = x_i'' - a_i' x_n''$ for $i = 1 \cdots v$. In this new basis change, every ${x_{n-1}''}^2$ and $x_{n-1}'' x_n''$ will be zero. Also, set the linear combination $z_2' = z_2 - \alpha_1^{(2)} z_{v+1} - \alpha_2^{(2)} z_{v+2} - \cdots - \alpha_o^{(2)} z_m$ with a zero-column. This process produces $2m + n - 2$ equations in n variables.
(6) Repeat this process to find $P_{n-2}, \cdots P_{v+1}$. Then we get $M_{L'}$.

Finally, the complexity of this attack is determined by the first four steps.

5.6. Other attacks

There exist other attacks besides the above attacks in MPKC, such as Thomae's attack [29], linearization equation attack [30] and differential attack.

Thomae's attack is an efficient algebraic key recovery attack to break Enhanced STS, Enhanced TTS and their variants. This attack mainly makes use of 'good keys' and 'missing cross-terms' to attack systems. The good keys are a generalization of equivalent keys in a MPKC scheme, and its process is similar to UOV reconciliation attack and Rainbow Band Separation attack. In fact, it is just generalization of Rainbow Band Separation attack, since in our schemes, we do not refer to TTS scheme, we can just take UOV reconciliation attack and Rainbow Band Separation attack into consideration. For more detail Thomae's attack, we recommend to see [29]. Currently, a practical cryptanalysis using this type of attack to find equivalent public key on a new encryption scheme can be found in [31].

The linearization equation attack is first discussed in [30] to break C∗. Later, the high order linearization equation attack [32] was proposed to attack the MFE cryptosystem. The core essence of linearization equation attack is to construct a potential bijection between the ciphertext and the plaintext. However, the central map of all our identity-based signature scheme is not a bijection, so the attack cannot work on them.

The differential attack is successfully applied to break C^*, PMI and Sflash. In this attack, one uses the fact that the differential of the public key of any MPKC is an affine map, and the dimension of the kernel of the differential is invariant. According to these facts, one can gain some information about the secret key to attack the corresponding cryptosystem. However, in the case of these two schemes, the dimension of the expected kernel has no invariant in the central map. Thus, the attacker cannot find some linearly independent vectors to build the kernel. So the differential attack is unpractical to attack them.

5.7. Experiments of attacks on our proposed ID-based schemes

Furthermore, based on the above algorithms, we provide some experiments support with assistance to the above known attacks on our schemes. The following attacks are programmed in Magma [33] and run on a workstation, with a Dual XEON Quad Core 2.27 GHz processor, 24 GB of main random access memory and the operation system is Scientific Linux 5.11 (Boron). We run direct algebraic attack F_4, the Reconciliation attack for ID-UOV (Rainbow Band Separation attack for ID-Rainbow, respectively), and the MinRank and KS attacks against our constructions and random quadratic equations. Results are given by the average attacking time for 50 tests of each scheme and is shown in Table 1.

In Table 1, we use the symbol 'NIL' to denote attack failure. We can see that there are attack failures in the Reconciliation attack and KS attack on random polynomial equations, this is because such attacks are mainly structural attacks which is of no use to the totally random situation. Finally, as is shown in Table 1, the time of all the attacks increases as the parameters grow, which indicates that the schemes can resist all these attacks under proper parameters.

6. PERFORMANCE AND COMPARISONS OF OUR ID-BASED SIGNATURE SCHEMES

6.1. Performance of ID-UOV and ID-Rainbow

Suppose that the length of the prime p in binary expression is L bits. Table 2 shows the performance requirements of ID-Rainbow and ID-UOV:

Mater Public Key Size: For ID-Rainbow, KDC needs to store the total coefficients of all the public polynomials, each coefficient contains a random degree four polynomial of $z_1,\ldots,z_d$, so each form of the coefficient contains $(d+1)(d+2)(d+3)(d+4)/24$ elements, and the total number of coefficients is $n(n+1)/2$ for one polynomial in both schemes, also the number of polynomials in these two schemes is o_1+o_2. For ID-UOV, KDC needs to store the total coefficients of all the public polynomials, each coefficient contains a random degree three polynomial of $z_1,\ldots,z_d$, so each form of the coefficient contains $(d+1)(d+2)(d+3)/6$ elements, and the total number of coefficients is $n(n+1)/2$ for one polynomial in both schemes, also the number of polynomials in these two schemes are o. So the master public key size of these two schemes is shown as in Table 2.

Master Private Key Size: For ID-UOV, each user needs to store the coefficients of all the central mapping polynomials and the affine invertible map, each polynomial contains $((o\cdot v+v(v+1)/2+n+1)+n(n+1)))$ coefficients, and since each coefficient is linear of $z_1,\ldots,z_d$, so it contains d elements. For ID-Rainbow, each user needs to store the coefficients of all the central mapping polynomials and the affine invertible maps. The four invertible maps contain $(2(o_1+o_2)^2+2n^2)d$ elements, and for the elements in the central mapping polynomials, it consists of o_2 polynomials with o_1+v_1 Vinegar variables and o_2 Oil variables, o_1 polynomials with v_1 Vinegar and o_1 Oil variables, so each polynomial contains $o_1(o_1v_1+v_1(v_1+1)/2+o_1(o_1+v_1+1)+o_2(o_2(v_1+o_1)+(v_1+o_1)(v_1+o_1+1)/2)+o_2(n+1)$ coefficients, and since each coefficient is linear of $z_1,\ldots,z_d$, so it contains d elements. Thus, the master secret key size of these two schemes shown as in Table 2.

Private Key Size: For ID-UOV, each signature needs to store the resultant coefficients of all the central mapping polynomials and the affine invertible map, each polynomial contains $(ov+v(v+1)/2+n+1)+n(n+1))$ elements, the invertible map contains $n(n+1)$ elements and the number of polynomials is o. Also for ID-Rainbow, the private key size is similar to the above except they don't need to store the coefficient constructed by $z_1,\ldots,z_d$. The final results are shown as in Table 2.

Public Key Size:For both schemes, the public key size is only the user's identity, so the size is only dL bits.

Computation on Setup: The main computation of setup process in UOV is to randomly construct the map F and $\overline{F}$. It will need $O(d^4)$ to construct a random polynomial, so the computation complexity of this process is calculated as in Table 2.

Computation on Key Extraction: The main computation of key extraction is to evaluate the polynomials with the random d variables. Since it is linear, the complexity is d, plus there are n^2 coefficients, for both schemes.

Computation on Signature Generation: The main computation of signature generation in UOV is to evaluate the polynomials with the random v vinegar variables and solve o linear equations in the o variables. So the computation complexity of key generation is $O(o\cdot v)+S$. In ID-Rainbow, we need first to evaluate the polynomials with the random v_1+o_1 vinegar variables with o_1+o_2 polynomials, and solve o_1+o_2 linear equations in the o_1+o_2 variables. Then, do it again for the first layer, so the complexity is $O(o_2(v_1+o_1)+o_1v_1)+2S$.

Computation on Signature Verification: For ID-Rainbow, the main computation of signature generation in UOV is to first construct the public polynomials, this needs computation complexity of $(o_1+o_2)d^4$ and then evaluate the public polynomials with the signature, this needs computation complexity of $(o_1+o_2)n$. The same situation comes to ID-UOV.

6.2. Practical parameters for these two schemes

According to the above security analysis in Sections 5 and 6.1, we suggest three practical parameter sets.

Then, Table 3 shows the parameters and key size of ID-Rainbow and ID-UOV.

As Table 3 shows, ID-Rainbow can produce shorter public key and private key than ID-UOV. The reason for this is that we can adjust the parameters more concisely to get the same security level in ID-Rainbow. There is a drawback that the master public key sizes of our ID-based schemes are a little bit large, but consider that the master key in a KDC needs just to construct once, we think it is OK. While on the other side, the public key of the current signing user is extremely small (only a few bits), and is a most important advantage in a cryptosystem.

6.3. Running time of our schemes

To further show the efficiency of ID-Rainbow and ID-UOV, we compare it with other signature schemes (including multivariate signature schemes and non-multivariate signature schemes) from the length of the message, length of the signature, size of the public key, size of the secret key, signing time and verification time.

We compare our schemes with Gui [34], QUARTZ [35], UOV and Rainbow [18], HS-Sign [36] which are current secure and promising multivariate signature schemes. All the schemes are running in MAGMA V2.19, with the hardware

TABLE 1. Result of experiments with some attacks using MAGMA (v2.19).

Schemes	Direct	Reconciliation	MinRank	KS
ID-UOV (3, 6, 8)	0.020 s	0.003 s	1.025 s	0.876 s
ID-Rainbow (2, 1, 2, 8)	0.020 s	0.003 s	0.020 s	1.030 s
Random (11, 11, 2)	1.122 s	NIL	0.953 s	NIL
ID-UOV (4, 8, 8)	0.250 s	0.018 s	2.693 s	0.876 s
ID-Rainbow (4, 2, 2, 8)	0.455 s	0.016 s	0.189 s	3.325 s
Random (12, 12, 8)	2.832 s	NIL	2.735 s	NIL
ID-UOV (5, 10, 8)	3.672 s	0.242 s	12.416 s	14.892 s
ID-Rainbow (5, 2, 3, 8)	4.715 s	0.313 s	3.159 s	32.451 s
Random (13, 13, 8)	13.125 s	NIL	13.628 s	NIL
ID-UOV (6, 12, 8)	344.235 s	24.020 s	311.451 s	50.154 s
ID-Rainbow (5, 3, 3, 8)	402.653 s	31.107 s	4.628 s	160.420 s
Random (14, 14, 8)	53.202 s	NIL	55.252 s	NIL

TABLE 2. Performance requirements by our proposed ID-based signature schemes.

	ID-Rainbow(q, o_1, o_2, v_1)	ID-UOV(q, o, v)
Master public key size(bit)	$\begin{pmatrix}(o_1+o_2)(d+1)\\ \cdot(d+2)(d+3)(d+4)\\ \cdot(n+2)(n+1)/48\end{pmatrix}\cdot L$	$\begin{pmatrix}o(d+1)(d+2)(d+3)\\ \cdot(n+2)(n+1)/12\end{pmatrix}\cdot L$
Master secret key size(bit)	$\begin{pmatrix}2(o_1+o_2)^2+2n^2\\ +o_1(o_1v_1+v_1(v_1+1)/2)\\ +o_1(o_1+v_1+1)\\ +o_2(o_2(v_1+o_1)\\ +(v_1+o_1)(v_1+o_1+1)/2)\\ +o_2(n+1)\end{pmatrix}\cdot dL$	$\begin{pmatrix}2n^2\\ +o(ov+v(v+1)/2)\\ +o(n+1)\end{pmatrix}\cdot dL$
Private key size(bit)	$\begin{pmatrix}(o_1+o_2)^2+n^2\\ +o_1(o_1v_1+v_1(v_1+1)/2)\\ +o_1(o_1+v_1+1)\\ +o_2(o_2(v_1+o_1)\\ +(v_1+o_1)(v_1+o_1+1)/2)\\ +o_2(n+1)\end{pmatrix}\cdot L$	$\begin{pmatrix}n^2\\ +o(ov+v(v+1)/2)\\ +o(n+1)\end{pmatrix}\cdot L$
Public key size(bit)	$d\cdot L$	$d\cdot L$
Setup	$O((o_1+o_2)d^4)$	$O(od^4)$
Key extraction	$O(dn^2)$	$O(dn^2)$
Signature generation	$O(o_2(v_1+o_1)+o_1v_1)+2S$	$O(ov)+S$
signature verification	$O((o_1+o_2)(d^4+n))$	$O(od^4+on)$

Notation for Table 2: o_1, v_1, v_2, o, v: the number of Oil and Vinegar variables, respectively; d:the size of the user's ID; n: $n=v_1+o_1+o_2$ for ID-Rainbow, $n=v+o$ for ID-UOV; S: average time required by a Gaussian Elimination function.

and software below: a workstation with a Dual XEON Quad Core 2.27 GHz processor, 24 GB of main random access memory and the operation system is Scientific Linux 5.11 (Boron). Here, we let all the scheme with security level 2^{80} and the comparison results are summarized in Table 4 in terms of efficiency and storage.

From Table 4, we can see that the signing time of ID-Rainbow is faster than that of UOV and QUARTZ, but a little slower than that of Gui and Rainbow scheme in the same security level. Also, the private key size of our schemes is smaller than most of the other schemes, except the private key size of Gui and QUARTZ, the reason is that these two schemes are

TABLE 3. Parameter and key size of our proposed schemes.

Security	Scheme	ID-Rainbow(q, o_1, o_2, v_1, d)	ID-UOV(q, o, v, d)
	Parameters	(256, 14, 14, 20, 5)	(256, 26, 52, 5)
2^{80}	Master Public Key	4220 KB	4493.2 KB
	Master Private Key	142.6 KB	131.2 KB
	Public Key	40 bits	40 bits
	Private Key	25.5 KB	77.9 KB
	Parameters	(256, 18, 18, 24, 8)	(256, 32, 64, 8)
2^{96}	Master Public Key	32.1 MB	15 MB
	Master Private Key	427.4 KB	1432.5 KB
	Public Key	64 bits	64 bits
	Private Key	48.6 KB	179.4 KB
	Parameters	(256, 20, 20, 28, 8)	(256, 48, 96, 8)
2^{128}	Master Public Key	45.6 MB	50 MB
	Master Private Key	551.8 KB	4770.1 KB
	Public Key	64 bits	64 bits
	Private Key	70 KB	596.3 KB

TABLE 4. Comparison between ID-UOV, ID-Rainbow and other multivariate signature schemes.

Schemes and Parameters	Message (bits)	Signature (bits)	PK (KB)	SK (KB)	Sign (ms)	Verify (ms)
Gui (96,5,6,6)	160	128	61.6	3.1	89	12
QUARTZ (103,129,3,4)	160	128	71.9	3.1	387	36
UOV (256,26,52)	208	624	16	15.5	185	40
Rainbow (256,14,14,20)	224	368	33.5	25.5	57	20
HS-Sign (253,26,52,24)	192	624	14.8	11.2	187	31
ID-UOV (256,26,52,5)	192	624	0.003	15.5	183	301
ID-Rainbow (256,14,14,20,5)	224	368	0.003	25.5	54	366

mixed field constructions. We can see that the verification time of our schemes is a little slower than that of other schemes, this is because we need to do an added operation to construct the public polynomials. However, due to this sacrifice, our schemes enjoy extremely small public key, which is a most considerable parameters in a cryptosystem. The result shows that our schemes is competitive with all the current promising MPKC schemes, so we think they are promising MPKC schemes.

7. CONCLUSION

In this paper, we introduced an idea of constructing an ID-based MPKC scheme. First, we briefly introduced the conception of MPKC schemes and how to build an MPKC scheme with a user's ID. Next, we propose two practical signature schemes with this idea: ID-UOV and ID-Rainbow. While we illustrating our ideas of constructing these schemes, we also write programs to verify its efficiency and security. This paper mainly focus on the theory part. In the future work, we would like to give more attention on the practical part of the fundamental MPKC scheme, we will do real deployment of our identity-based MPKC schemes into popular application such as blockchain application, application of TinyOS system in the sensor nodes, etc.

ACKNOWLEDGEMENTS

This work was supported in part by the National Key Research and Development Program (grant# 2016YFB0200602), the National Nature Science Foundation of China (618002072) and the project of Guangzhou Science and Technology (grant# 201802010043, 201807010058).J. Ding is partially supported by NSF.

REFERENCES

[1] Shamir, A. (1985) Identity-Based Cryptosystems and Signature Schemes. In Blakley, G.R. and Chaum, D. (eds.) *Advances in Cryptology—CRYPTO 1984.* Lecture Notes in Computer Science, Vol. 196, pp. 47–53. Springer, Berlin, Heidelberg.

[2] Boneh, D. and Franklin, M. (2001) Identity-Based Encryption from the Weil Pairing. In Kilian, J. (ed.) *Advances in Cryptology—CRYPTO 2001,* Lecture Notes in Computer Science, Vol. 2139, pp. 213–229. Springer, Berlin, Heidelberg.

[3] Choon, J.C. and Hee Cheon, J. (2003) An Identity-Based Signature from Gap Diffie–Hellman Groups. In Desmedt, Y.G. (ed.) *Public Key Cryptography—PKC 2003,* Lecture Notes in Computer Science, Vol. 2567, pp. 18–30. Springer, Berlin, Heidelberg.

[4] Li, J., Teng, M., Zhang, Y. and Yu, Q. (2016) A leakage-resilient CCA-secure identity-based encryption scheme. *Comput. J.*, **59**, 1066–1075.

[5] Li, J., Yu, Q. and Zhang, Y. (2018) Identity-based broadcast encryption with continuous leakage resilience. *Inf. Sci.*, **429**, 177–193.

[6] Li, J., Guo, Y., Yu, Q., Lu, Y. and Zhang, Y. (2016) Provably secure identity-based encryption resilient to post-challenge continuous auxiliary inputs leakage. *Secur. Commun. Netw.*, **9**, 1016–1024.

[7] Ning, J., Dong, X., Cao, Z., Wei, L. and Lin, X. (2015) White-box traceable ciphertext-policy attribute-based encryption supporting flexible attributes. *IEEE Trans. Inf. Forensics Secur.*, **10**, 1274–1288.

[8] Ning, J., Cao, Z., Dong, X., Ma, H., Wei, L. and Liang, K. (2018) Auditable-times outsourced attribute-based encryption for access control in cloud computing. *IEEE Trans. Inf. Forensics Secur.*, **13**, 94–105.

[9] Ning, J., Cao, Z., Dong, X. and Wei, L. (2018) White-box traceable CP-ABE for cloud storage service: how to catch people leaking their access credentials effectively. *IEEE Trans. Dependable Secure Comput.*, **15**, 883–897.

[10] Shor, P. (1996) Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, **26**, 1484–1509.

[11] Tang, S. and Xu, L. (2014) Towards provably secure proxy signature scheme based on isomorphisms of polynomials. *Future Generation Comput. Syst.*, **30**, 91–97.

[12] Petzoldt, A., Bulygin, S. and Buchmann, J. (2013) A multivariate based threshold ring signature scheme. *Appl. Algebra Eng. Commun. Comput.*, **24**, 255–275.

[13] Sakumoto, K., Shirai, T. and Hiwatari, H. (2011) On Provable Security of UOV and HFE Signature Schemes against Chosen-Message Attack. In Yang, B.Y. (ed.) *Post-Quantum Cryptography -PQCrypto 2011,* Lecture Notes in Computer Science, Vol. 7071, pp. 68–82. Springer, Berlin, Heidelberg.

[14] Chen, J., Tang, S., He, D. and Tan, Y. (2017) Online/offline signature based on UOV in wireless sensor networks. *Wirel. Netw.*, **23**, 1719–1730.

[15] Petzoldt, A., Szepieniec, A. and Mohamed, M.S.E. (2017) A Practical Multivariate Blind Signature Scheme. In Kiayias, A. (ed.) *Financial Cryptography and Data Security—FC 2017,* Lecture Notes in Computer Science, Vol. 10322, pp. 437–454. Springer, Cham.

[16] El Bansarkhani, R., Mohamed, M.S.E. and Petzoldt, A. (2016) MQSAS—A Multivariate Sequential Aggregate Signature Scheme. In Bishop, M. and Nascimento, A. (eds.) *Information Security—ISC 2016,* Lecture Notes in Computer Science, Vol. 9866, pp. 426–439. Springer, Cham.

[17] Kipnis, A., Patarin, J. and Goubin, L. (1999) Unbalanced Oil and Vinegar Signature Schemes. In Stern, J. (ed.) *Advances in Cryptology—EUROCRYPT 1999,* Lecture Notes in Computer Science, Vol. 1592, pp. 206–222. Springer, Berlin, Heidelberg.

[18] Ding, J. and Schmidt, D. (2005) Rainbow, A New Multivariable Polynomial Signature Scheme. In Ioannidis, J., Keromytis, A. and Yung, M. (eds.) *Applied Cryptography and Network Security -ACNS 2005,* Lecture Notes in Computer Science, Vol. 3531, pp. 164–175. Springer, Berlin, Heidelberg.

[19] Courtois, N., Klimov, A., Patarin, J. and Shamir, A. (2000) Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations. In Preneel, B. (ed.) *Advances in Cryptology -EUROCRYPT 2000,* Lecture Notes in Computer Science, Vol. 1807, pp. 392–407. Springer, Berlin, Heidelberg.

[20] Kipnis, A. and Shamir, A. (1998) Cryptanalysis of the Oil and Vinegar Signature Scheme. In Krawczyk, H. (ed.) *Advances in Cryptology—CRYPTO 1998,* Lecture Notes in Computer Science, Vol. 1462, pp. 257–266. Springer, Berlin, Heidelberg.

[21] Cao, W., Hu, L., Ding, J. and Yin, Z. (2011) Kipnis–Shamir Attack on Unbalanced Oil-Vinegar Scheme. In Bao, F. and Weng, J. (eds.) *Information Security Practice and Experience —ISPEC 2011,* Lecture Notes in Computer Science, Vol. 6672, pp. 168–180. Springer, Berlin, Heidelberg.

[22] Petzoldt, A., Bulygin, S. and Buchmann, J. (2010) Selecting Parameters for the Rainbow Signature Scheme. In Sendrier, N. (ed.) *Post-Quantum Cryptography—PQCrypto 2010,* Lecture Notes in Computer Science, Vol 6061, pp. 218–240. Springer, Berlin, Heidelberg.

[23] Courtois, N.T. (2001) Efficient Zero-Knowledge Authentication Based on a Linear Algebra Problem MinRank. In Boyd, C. (ed.) *Advances in Cryptology—ASIACRYPT 2001,* Lecture Notes in Computer Science, Vol. 2248, pp. 402–421. Springer, Berlin, Heidelberg.

[24] Goubin, L. and Courtois, N.T. (2000) Cryptanalysis of the TTM Cryptosystem. In Okamoto, T. (ed.) *Advances in Cryptology—ASIACRYPT 2000,* Lecture Notes in Computer Science, Vol. 1976, pp. 44–57. Springer, Berlin, Heidelberg.

[25] Faugère, J.-C. (1999)) A new efficient algorithm for computing Gröbner bases (F_4). *J. Pure Appl. Algebra*, **139**, 61–88.

[26] Faugère, J.-C. (2002) A New Efficient Algorithm for Computing Gröbner Bases without Reduction to Zero (F5). In Mora, T. (ed.) *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation—ACM ISSAC 2002*, pp.75–83. ACM, New York.

[27] Bettale, L., Faugère, J. and Perret, L. (2009) Hybrid approach for solving multivariate systems over finite fields. *J. Math. Cryptol.*, **3**, 177–197.

[28] Ding, J., Yang, B.Y., Chen, C.H.O., Chen, M.S. and Cheng, C.M. (2008) New Differential-Algebraic Attacks and Reparametrization of Rainbow. In Bellovin, S.M., Gennaro, R., Keromytis, A. and Yung, M. (eds.) *Applied Cryptography and Network Security—ACNS 2008,* Lecture Notes in Computer Science, Vol. 5037, pp. 242–257. Springer, Berlin, Heidelberg.

[29] Thomae, E. and Wolf, C. (2012) Cryptanalysis of Enhanced TTS, STS and All Its Variants, or: Why Cross-Terms Are Important. In Mitrokotsa, A. and Vaudenay, S. (eds.) *Progress in Cryptology—AFRICACRYPT 2012,* Lecture Notes in Computer Science, Vol. 7374, pp. 188–202. Springer, Berlin, Heidelberg.

[30] Patarin, J. (1995) Cryptanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt'88. In Coppersmith, D. (ed.) *Advances in Cryptology—CRYPTO 1995,* Lecture Notes in Computer Science, Vol. 963, pp. 248–261. Springer, Berlin, Heidelberg.

[31] Chen, J., Tan, C.H. and Li, X. (2018) Practical cryptanalysis of a public key cryptosystem based on the morphism of polynomials problem. *Tsinghua Sci. Technol.*, **23**, 671–679.

[32] Ding, J., Hu, L., Nie, X., Li, J. and Wagner, J. (2007) High Order Linearization Equation (HOLE) Attack on Multivariate Public Key Cryptosystems. In Okamoto, T. and Wang, X. (eds.) *Public Key Cryptography—PKC 2007,* Lecture Notes in Computer Science, Vol. 4450, pp. 233–248. Springer, Berlin, Heidelberg.

[33] Bosma, W., Cannon, J. and Playoust, C. (1997) The Magma algebra system. I: the user language. *J. Symbolic Comput.*, **24**, 235–265.

[34] Petzoldt, A., Chen, M.S., Yang, B.Y., Tao, C. and Ding, J. (2015) Design Principles for HFEv- Based Multivariate Signature Schemes. In Iwata, T. and Cheon, J. (eds.) *Advances in Cryptology—ASIACRYPT 2015,* Lecture Notes in Computer Science, Vol. 9452, pp. 311–334. Springer, Berlin, Heidelberg.

[35] Patarin, J., Courtois, N. and Goubin, L. (2001) QUARTZ, 128-Bit Long Digital Signatures. In Naccache, D. (ed.) *Topics in Cryptology—CT-RSA 2001,* Lecture Notes in Computer Science, Vol. 2020, pp. 282–297. Springer, Berlin, Heidelberg.

[36] Chen, J., Tang, S. and Zhang, X. (2017) HS-Sign: a security enhanced UOV signature scheme based on hyper-sphere. *KSII Trans. Internet Inf. Syst.*, **11**, 3166–3187.