ISO/IEC JTC 1/SC 27/WG 2

Cryptography and security mechanisms

Convenorship: JISC (Japan)

Document type:	Standing Document
Title:	WG 2 SD8 (Post-Quantum Cryptography) -- Part 5: Multivariate Cryptography
Status:	
Date of document:	2020-05-29
Source:	Editors (Bo-Yin Yang, Ward Beullens)
Expected action:	INFO
No. of pages:	1 + 18
Email of convenor:	t-chika@ipa.go.jp
Committee URL:	https://isotc.iso.org/ecom/livelink/open/jtc1sc27wg2

ISO/IEC JTC1 SC27 WG2
Standing Document 8 (SD8)
Part V - Multivariate Cryptography

Editors: Bo-Yin Yang and Ward Beullens

May 26, 2020

Contents

1 Introduction **2**

2 Solving Multivariate Systems **3**
- 2.1 XL 4
- 2.2 Gröbner Bases and $\mathbf{F_4}/\mathbf{F_5}$ 5
- 2.3 The Joux-Vitse Variation of XL 6

3 Trapdoor Multivariate Signature Schemes **6**
- 3.1 Big-Field Schemes 7
- 3.2 Small-Field Schemes 9

4 Fiat-Shamir Multivariate Signature schemes **14**

1 Introduction

Multivariate cryptography refers to public-key cryptography whose public keys represent a multivariate and nonlinear (usually quadratic) polynomial map. The main computational problem underlying multivariate cryptography is to find preimages for these multivariate polynomial maps. This problem is known to be NP-hard.

Problem $\mathcal{M}Q$: Given a system of polynomials $p_1, p_2, \cdots, p_m$, where each p_i is a quadratic polynomial in n variables, find a solution $\mathbf{x} = (x_1, \ldots, x_n)$ that satisfies $p_1(\mathbf{x}) = p_2(\mathbf{x}) = \cdots = p_m(\mathbf{x}) = 0$. All coefficients and variables are in $\mathbb{K} = \mathbb{F}_q$, the field with q elements.

It is conjectured that for a probabilistic Turing machine $\mathcal{A}$ with a subexponential cost function $\eta(n)$, the probability that $\mathcal{A}$ returns in time $\eta(n)$ a correct solution to a randomly drawn $\mathcal{MQ}(n, m, q)$ instance with $m \propto n$ approaches zero as $n \to \infty$. In other words, the $\mathcal{MQ}$ problem is hard on average. In Sect 2, we will briefly review the state of the art of algorithms for solving this problem and we will see that even for moderately sized polynomial systems (e.g. 48 equations $\mod 31$), this problem is intractable.

While it is possible to construct multivariate public-key cryptosystems (MPKCs) that depends solely on $\mathcal{MQ}$ (see Sect. 4), traditionally most MPKCs use a trapdoor construction that hides a quadratic map $\mathcal{Q} : \mathbb{K}^n \to \mathbb{K}^m$ for which preimages can be found easily by composing it with affine maps S, T. Concretely, the secret key consists of $\mathcal{Q}, S$, and T, and the public key is the map $\mathcal{P} = T \circ \mathcal{Q} \circ S$ that looks like a random map. These trapdoor schemes will we explained in Sect.3. For a digital signature scheme we will naturally have $n \geq m$ (surjective trapdoor function) and for an encryption scheme $n \leq m$ (injective trapdoor function). There are two main families of trapdoors, the first family uses a univariate polynomial over a large finite field and decomposes this univariate polynomial to a set of multivariate polynomials over a smaller base field. Schemes that use this strategy (e.g. HFE, HFEv-, QUARTZ, $\cdots$) are known as big-field schemes. In contrast, there are simpler schemes (called small-field schemes) that only use a single small field (e.g. UOV, Rainbow, LUOV). Since it is difficult to construct efficient and secure multivariate encryption schemes, we will only discuss digital signature schemes in this document.

To sign a message m the signer computes a preimage of $\mathbf{z} = \mathcal{H}(m)$ for the public key $\mathcal{P}$, where $\mathcal{H}$ is a hash function that outputs a vector in the codomain of $\mathcal{P}$. Given the secret key this can be done efficiently by first computing $\mathbf{y} = T^{-1}(\mathbf{z})$, then finding a preimage $\mathbf{x}$ such that $\mathcal{Q}(\mathbf{x}) = \mathbf{y}$ and then computing $\mathbf{w} = S^{-1}(\mathbf{x})$. Note that these preimages are usually not unique. To verify a signature one simply checks if $\mathcal{H}(m) = \mathcal{P}(\mathbf{w})$.

Note that an attacker can attempt to exploit the hidden structure of $\mathcal{P}$ to solve the system of equations. Therefore, the security of the system does not only rely on the $\mathcal{MQ}$ problem, but also on the problem of distinguishing the trapdoor functions from random quadratic systems. This is in contrast

Field size	2	4	16	31
Number of variables	144	88	56	48

Table 1: For several field sizes the table gives the number of variables required to make the $\mathcal{MQ}$ problem for random determined systems (i.e. $n = m$) achieve 128 bits of security.

to the Fiat-Shamir schemes of Sect. 4, which only use truly random systems of equations. In these schemes, the public key is the description of a set of quadratic multivariate equations $\mathcal{P}$, and the secret key is a solution $\mathbf{x}$ to the system $\mathcal{P}(\mathbf{x}) = 0$. A signature is then a Zero-Knowledge proof that the signer knows a solution to this public system $\mathcal{P}(\mathbf{x}) = 0$. The soundness property of the Zero-Knowledge proof implies that an adversary cannot forge signatures without knowing a solution, moreover, the Zero-Knowledge property implies that the signatures do not leak information about the secret key. Therefore, assuming it is hard to find a solution to the system $\mathcal{P}(\mathbf{x}) = 0$, one can prove that the signature scheme is unforgeable (in the ROM).

2 Solving Multivariate Systems

For the trapdoor schemes of Sect. 3, an adversary can forge a signature for a message m if he can if he can find a solution to the random-looking system $\mathcal{P}(\mathbf{x}) = \mathcal{H}(m)$. Even worse, in the case of the Fiat-Shamir schemes of Sect. 4 if an adversary can find a single solution to $\mathcal{P}(\mathbf{x}) = 0$, then he can use this solution as a secret key to sign any message. Therefore, it is critical for the security of all the multivariate signature schemes that the $\mathcal{MQ}$ problem is intractably hard. Luckily Table 1 shows that with current methods, the MQ problem becomes intractable already for reasonably small polynomial systems.

In this section, we describe the state of the art methods to solve the system of m equations $\mathcal{P}(\mathbf{w}) = \mathbf{z}$ in n variables $w_1, \dots w_n$. If $m \geq n$, the system is called (over-)determined, usually, the more overdetermined a system is, the easier it is to solve. If $m < n$, the system is called underdetermined. Unless the system is very underdetermined, the best approach is to guess $m - n$ variables randomly and proceed with the determined system with $m = n$ [CGMT02].

At the moment, the best known methods to solve equations are descendants of Buchberger's algorithm [Buc65] to compute a Gröbner basis, first investigated by Daniel Lazard's group [Laz83]. Macaulay generalized Sylvester's matrix to multivariate polynomials [Mac16]. The idea is to construct a matrix whose lines contain the multiples of the polynomials in the original system, the columns representing a basis of monomials up to a given degree. It was observed by D. Lazard [Laz83] that for a large enough degree, ordering the columns according to a monomial ordering and performing row reduction without column pivoting on the matrix is equivalent to Buchberger's algorithm. Reductions to 0 correspond to lines that are linearly dependent upon the previous ones and the leading term of a polynomial is given by the leftmost nonzero entry in the corresponding line.

Lazard's idea was rediscovered in 1999 by Courtois, Klimov, Patarin, and Shamir [CKPS00] as **XL**. Courtois *et al* proposed several adjuncts [Cou04, CP03, CP02] to XL. One tweak called XL2 merits a mention as an easy to understand precursor to $\mathbf{F_4}$. Another real improvement for $\mathbf{F_4}/\mathbf{F_5}$ as well as XL was FXL, where F means "fixing" (guessing) variables.

Some time *prior* to this, J. -C. Faugère had proposed a much improved Gröbner bases algorithm called $\mathbf{F_4}$ [Fau99]. A later version, $\mathbf{F_5}$ [Fau02], made headlines [FJ03] when it was used to solve HFE Challenge 1 in 2002.

2.1 XL

The eXtended linearized algorithm (XL) works by first extending a polynomial system by appending to the systems all the equations of the form $p_i\mathbf{x}^{\mathbf{b}}$, where p_i is one of the original equations and $\mathbf{x}^{\mathbf{b}}$ is a monomial of degree $D-2$, for some fixed value of D. The system is linearized by treating the monomials $\mathbf{x}^{\mathbf{b}} \in \mathcal{T}^{(D)}$ as independent variables. If the number of equations is larger than the number of monomials the linearized system can be solved with linear algebra methods. Otherwise, if the system is not solvable, repeat with a higher value of D until we have a solution, a contradiction, or until we reduce the system to a univariate equation in some variable. Note that increasing D increases the number of equations faster than it increases the number of monomials. The number of equations and independent equations are denoted $R^{(D)} = R = |\mathcal{R}|$ and $I^{(D)} = I = \dim(\mathrm{span}\mathcal{R})$.

If we accept solutions in arbitrary extensions of $K = \mathbb{F}_q$, then $T = \binom{n+D}{D}$ regardless of q. However, most crypto applications require solutions in $\mathbb{F}_q$ only. The above expression for T then only holds for large q, since we may identify x_i^q with x_i and cut substantially the number of monomials we need to manage. This "Reduced XL" (cf. C. Diem [Die04]) can lead to extreme savings compared to "Original XL," e.g., if $q = 2$, then $T = \sum_{j=0}^{D} \binom{n}{j}$.

Proposition 1 ([YC04, Theorem 7])**.** *If the equations p_i, with $\deg p_i := d_i$, and (*) $\underline{\text{relations } \mathcal{R}^{(D)} \text{ has no dependencies except the obvious ones generated}}$ $\underline{\text{by } p_ip_j = p_jp_i \text{ and } p_i^q = p_i}$, then*

$$T - I = [t^D]\ G(t) = [t^D]\ \frac{(1-t^q)^n}{(1-t)^{n+1}} \prod_{j=1}^{m} \left(\frac{1-t^{d_j}}{1-t^{q\,d_j}} \right). \tag{1}$$

There is always a certain degree D_{XL} above which Eq. 1 and hence the underlined condition (*) above cannot continue hold if the system has a solution, because the right hand side of Eq. 1 goes nonpositive. This is $D_{XL} := \min\{D : [t^D]\ G(t) \le 0\}$, called the $\underline{\text{degree of regularity for XL}}$. If (*) holds for as long as possible (which means for degrees up to D_{XL}), we say that the system is **K-semi-regular** or **q-semi-regular** (cf. [BFS04, YC04]).

Diem proves [Die04] for char 0 fields, and conjectures for all K that (i) a generic system (no algebraic relationship betweem the coefficients) is K-semi-regular and (ii) if $(p_i)_{i=1\cdots m}$ are *not* K-semi-regular, I can only decrease from the Eq. 1 prediction. Most experts seem to believe the conjecture [Die04] that a *random* system behaves like a generic system with probability close to 1.

Corollary 2. $T - I = [t^D]\left((1-t)^{-n-1}\prod_{j=1}^{m}\left(1-t^{d_i}\right)\right)$ *for generic equations if* $D \leq \min(q, D_{XL}^{\infty})$*, where* D_{XL}^{∞} *is the degree of the lowest term with a non-positive coefficient in* $G(t) = \left((1-t)^{-n-1}\prod_{j=1}^{m}\left(1-t^{d_i}\right)\right)$.

2.2 Gröbner Bases and $\mathbf{F_4}/\mathbf{F_5}$

In contrast to the XL algorithm, which aims to find solutions to a system of multivariate polynomial equations, the $\mathbf{F_4}$ and $\mathbf{F_5}$ algorithms aim to compute a Gröbner basis for the ideal generated by a set of multivariate polynomials. A Gröbner basis is a particularly nice generating set, for an Ideal. In particular, if a Gröbner basis is computed with respect to an elimination order, then it can be used to efficiently compute all the solutions of the system of equations. In practice, it is more efficient to compute a Gröbner bases with respect to the graded reverse lexicographic order, and then convert this Gröbner basis into a Gröbner basis with respect to an elimination order with an algorithm such as the FGLM algorithm [Fau99, Fau02].

Buchberger's algorithm for computing a Gröbner bases for the ideal $\mathcal{I}$ generated by a set of polynomials $\mathcal{F}$ works by iteratively considering so-called S-polynomials, and trying to reduce them with respect to the current basis for $\mathcal{I}$. If the S-polynomial does not reduce to 0 it is added to the basis of I until a Gröbner basis is obtained. The $\mathbf{F_4}$ algorithm is an adaptation of the original Buchberger algorithm that rather than considering S-polynomials individually, reduces a large number of S-polynomials simultaneously by constructing a matrix and reducing the matrix.

A problem of both the original Buchberger's algorithm and $\mathbf{F_4}$ is that most of the S-polynomials reduce to zero, which means that a lot of the computational effort is wasted. The $\mathbf{F_5}$ is a further refinement of $\mathbf{F_4}$ designed to address this issue. In this algorithm, the sequence of S-polynomials is generated one by one (or the matrix row by row), while an algebraic criterion is used to determine, ahead of an elimination process, whether a row will be reduced to zero or not. Only the meaningful rows are retained. A complication resulting from this strategy is that the elimination must be done in a strictly ordered way.

An important quantity for estimating the runtime of $\mathbf{F_4}$ and $\mathbf{F_5}$ is "the operating degree", which for semi-regular systems (random systems are semi-regular with overwhelming probability) can be calculated with the following proposition.

Proposition 3 ([BFS04]). *If the eqs.* p_i *are* q*-semi-regular, at the operating degree*

$$D_{reg} := \min\left\{D : [t^D]\,\frac{(1-t^q)^n}{(1-t)^n}\prod_{i=1}^{m}\left(\frac{1-t^{d_i}}{1-t^{qd_i}}\right) < 0\right\}$$

both $\mathbf{F_4}$-$\mathbf{F_5}$ will terminate. Note that by specializing to a large field, we find

$$D_{reg}^{\infty} := \min\left\{ D : [t^D]\ (1-t)^{-n} \prod_{i=1}^{m} \left(1 - t^{d_i}\right) < 0 \right\}. \quad (2)$$

This operating degree allows us to estimate the running time of the $\mathbf{F_4}/\mathbf{F_5}$ algorithms:

Proposition 4. $\mathbf{F_4}/\mathbf{F_5}$ *runs in ($\omega :=$ the "order of matrix multiplications")*

$$C_{F_4/F_5} \propto c_\omega\, T^\omega \text{ multiplications.} \quad (3)$$

2.3 The Joux-Vitse Variation of XL

The "hybridized XL-related algorithm" [JV17] of Joux-Vitse is the currently known best practical solver of systems with $m = n$ over $\mathbb{F}_2$. This algorithm is a hybrid between solving the system by brute force (i.e. trying all the 2^n solutions) and the XL method. The idea is to first use an XL-like algorithm to obtain a new system of polynomials that can be brute-forced more efficiently.

Suppose we start with n variables and m quadratic equations over $\mathbb{F}_2$. Like in normal XL, Multiply each equation with every monomial up to and including degree $D-2$. This forms a set of equations up to degree D. The coefficient matrix of these equations forms the Macaulay matrix $Mac^{(D)}$ at degree D.

Now we fix k variables and we run an XL-like algorithm to find linear combinations of the equations that eliminate every term which is of degree 2 or higher in the chosen variables. Therefore, the resulting system is linear in the chosen variables. With k such equations, we may brute-force the remaining $n-k$ variables to obtain linear equations in the chosen k variables, then solve for them. *In practice, we want more than k equations in the chosen variables to ensure that most of these substituted systems are inconsistent.*

When the parameter D is larger, it will be possible to eliminate the non-linear monomials for a larger set of variables, (i.e. we can increase k), which will make enumerating all the 2^{n-k} options faster. However, increasing D will make the XL-like algorithm more expensive. Therefore, the parameter D should be chosen to minimize the total running time of the algorithm.

3 Trapdoor Multivariate Signature Schemes

In this section, we will describe the state of the art of multivariate trapdoor-based signature schemes. Roughly, the multivariate trapdoors can be divided into big-field schemes, that use the field structure of a large extension of a finite field, and small-field schemes, that only rely on relatively small finite fields. In this section, we will also review some of the attacks on these trapdoor schemes.

3.1 Big-Field Schemes

Let $\mathbb{K} \cong \mathbb{F}_q$ be a finite field of order q and $\mathbb{L}$ a degree-n extension of $\mathbb{K}$, with a "canonical" isomorphism ϕ identifying $\mathbb{L}$ with the vector space $\mathbb{K}^n$. That is, $\mathbb{K}^n \xrightarrow{\phi} \mathbb{L}$, $\mathbb{L} \xrightarrow{\phi^{-1}} \mathbb{K}^n$. Any function or map F from $\mathbb{L}$ to $\mathbb{L}$ can be expressed *uniquely* as a polynomial function with coefficients in $\mathbb{L}$ and degree less than q^n, namely

$$F(X) = \sum_{i=0}^{q^n-1} a_i X^i, \quad a_i \in \mathbb{L}.$$

Denote by $\deg_{\mathbb{L}}(F)$ the degree of $F(X)$ for any map F. Using ϕ, we can build a new map $F' : \mathbb{K}^n \to \mathbb{K}^n$

$$P(x_1, .., x_n) = (p_1(x_1, .., x_n), \dots, p_n(x_1, .., x_n)) = \phi^{-1} \circ F \circ \phi(x_1, .., x_n),$$

which is essentially F but viewed from the perspective of $\mathbb{K}^n$. We can identify F and F' unless there is a chance of confusion.

The main idea behind big-field trapdoors is to choose a map $F : \mathbb{L} \to \mathbb{L}$ that can be inverted easily (e.g. an exponentiation map, or a low degree polynomial), and to hide the structure of F by first descending to $\mathbb{K}^n$ to get the equivalent map $F' : \mathbb{K}^n \to \mathbb{K}^n$ and then composing F' with random affine maps.

To keep the size of the public key growing too large, we want the multivariate polynomial map F' to be quadratic. This happens if the univariate polynomial F is of the form

$$F(X) = \sum_{i,j=0}^{n-1} a_{ij} X^{q^i+q^j} + \sum_{i=0}^{n-1} b_i X^{q^i} + c.$$

HFE and variants

Such a $F(X)$ with a fixed low $\mathbb{L}$-degree is used to build the HFE multivariate public key cryptosystems, as in the following

$$F(X) \quad = \quad \sum_{i,j=0}^{q^i+q^j \leq D, j \leq i} a_{ij} X^{q^i+q^j} + \sum_{i=0}^{q^i \leq D} b_i X^{q^i} + c;$$

Note that the coefficients are values in $\mathbb{L}$, and all coefficients $a_{ii} = 0$ if $q = 2$, since those are covered by the b-part of the coefficients.

HFE itself is not secure if D is not large enough, and slow if D is large. Thus usually it is used in a variant form. HFEv, which uses the vinegar modification, is built from this polynomial:

$$\mathcal{F}(X, \bar{X}) := \sum_{i,j} a_{ij} X^{q^i+q^j} + \sum_{i,j} b_{ij} X^{q^i} \bar{X}^{q^j} + \sum_{i,j} \alpha_{ij} \bar{X}^{q^i+q^j} + \sum_i b_i X^{q^i} + \sum_i \beta_i' \bar{X}^{q^i} + c \quad (4)$$

where the auxiliary variable $\bar{X}$ occupies only a subspace of small rank v in $\mathbb{L} \cong \mathbb{K}^n$. The function $\mathcal{F}$ is quadratic in the components of X and $\bar{X}$, and so is

$\mathcal{P} = T \circ \mathcal{F} \circ S$ for affine bijections T and S in $\mathbb{K}^n$ and $\mathbb{K}^{n+v}$. We hope that $\mathcal{P}$ is hard to invert to the adversary, while the legitimate user, with the knowledge of (S, T) can compute X by substituting a random $\bar{X}$, then solving for X via root-finding algorithms such as Berlekamp (or Cantor-Zassenhaus, if $q \neq 2$).

In addition to the v (Vinegar) variation, typically we use a "minus" variation in which a of the equations are not released to the public.

The GeMMS signature scheme is one of the Round2 candidates of the NIST Post-Quantum Cryptography standardization project. GeMMS is an instantiation of HFEv-. Table 2 shows the key and signature sizes of current GeMSS instances. The security of HFEv- type signature schemes is well-studied. The main drawbacks are the large key sizes and the slow signing times. GeMSS128 takes 736 million cycles on an Intel Haswell processor (about 0.2 seconds on a 3GHz core) to produce a signature.

	Security Level	parameters (n, D, a, v, k)	$\vert$pk$\vert$ (kB)	$\vert$sk$\vert$ (kB)	$\vert$sig$\vert$ (bit)[1]
GeMSS128	128	$(174, 513, 12, 12, 4)$	352.2	13.4	258
GeMSS192	192	$(265, 513, 22, 20, 4)$	1238.0	34.1	411
GeMSS256	256	$(354, 513, 30, 33, 4)$	3040.7	75.9	576

[1] not including 128 bit salt

Table 2: Key and Signature sizes of GeMSS instances

	parameters (n, D, a, v, k)	public map time (kC)[1]	private map time (MC)[1]	keypair gen time (MC)[1]
GeMSS128	$(174, 513, 12, 12, 4)$	80.8	736	38.3
GeMSS192	$(265, 513, 22, 20, 4)$	237	2520	176
GeMSS256	$(354, 513, 30, 33, 4)$	568	3600	485

[1] Intel Haswell, `gcc-7.3`

Table 3: Speed of GeMSS with AVX2 and PCLMULQDQ (Haswell)

Attacks on the HFEv- scheme

The following attacks have to be taken into account when choosing parameters for the HFEv- scheme:

Direct attacks Like any Multivariate scheme, HFEv- can be attacked with the system solving algorithms of Sect. 2. The complexity of these attacks depends on the degree of regularity of the system that we try to solve. However, for the specific case of HFEv-, this degree is lower than that of random systems, which makes solving them considerably easier. An upper bound on the degree of regularity of HFEv- systems is derived by Ding and Yang [DY13]. This bound is reasonably tight according to experiments and is used to derive the parameters for GUI-184 given above.

Minrank-Based Attacks Kipnis and Shamir proposed a rank attack against the HFE cryptosystem. The key idea of this attack is to consider the public and private maps of HFE as univariate polynomial maps over the extension field. Due to the special structure of the HFE central map, the rank of the corresponding matrix is limited by $r = \lceil \log_2(D-1) \rceil + 1$. It is therefore possible to recover the affine transformation S by solving an instance of the MinRank problem.

Bouillaget et al. improved this attack by showing that the map S can be found by computing a Gröbner Basis over the base field $\mathbb{F}_2$ (Minors Modelling). By doing so, they could speed up the attack of Kipnis and Shamir significantly. The complexity of the MinRank attack against HFE using the Minors Modelling approach can be estimated as

$$C_{\text{MinRank/HFE}} = a_\omega \binom{n+r}{r}^\omega$$

where $r = \lceil \log_2(D-1) \rceil + 1$ is the rank of the matrix corresponding to the central map and ω and a_ω are as above. The attack generalizes to the case of HFEv-, where the rank of the matrix is given by $r + a + v$. Therefore, we can estimate the complexity of our attack as

$$C_{\text{MinRank/HFEv-}} = \begin{cases} a_\omega \binom{n+r+a+v}{r+a+v}^\omega, & r+a+v \text{ even,} \\ a_\omega \binom{n+r+a+v-1}{r+a+v}^\omega, & r+a+v \text{ odd.} \end{cases}$$

Since the quadratic systems to be solved during this attack are highly overdetermined, the attack can not be sped up with the help of Grover's algorithm.

3.2 Small-Field Schemes

The original small-field scheme is the (Unbalanced) Oil and Vinegar scheme [Pat97, KPG99]. The scheme is simple and uses only simple arithmetic (Gaussian elimination) over a small finite field.

Suppose $v < n$ is an integer and $m = o = n - v$. The variables $x_1, \ldots, x_v$ are called *vinegar* variables and $x_{v+1}, \ldots, x_n$ are called *oil* variables.

Take a map $\mathcal{F} : \mathbb{K}^n \to \mathbb{K}^m$ of the form $\mathbf{y} = \mathcal{F}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_o(\mathbf{x}))$, where

$$f_l(\mathbf{x}) = \sum_{i=1}^{v} \sum_{j=i}^{n} \alpha_{ij}^{(l)} x_i x_j,$$

for all l in $\{1, \cdots, o\}$, and where all the coefficients are randomly chosen from the base field $\mathbb{K}$. Here we notice that there are no quadratic terms between oil variables, which means that the oil variables and vinegar variables are not fully mixed (like oil and vinegar in a salad dressing) and this explains the name of the scheme.

The public map $\mathcal{P}$ is constructed as $\mathcal{P} = \mathcal{Q} \circ S$, where S is an invertible linear map. Here the change of basis is a process to fully "mix" the oil and vinegar.

The original Oil and Vinegar signature scheme has $m = o = v = n/2$. When $o < v$, it becomes the unbalanced Oil and Vinegar signature scheme. The public key is $\mathcal{P} = (p_1, \dots, p_o)$, the polynomial components of $\mathcal{P}$. The secret key consists of the linear map S and the map $\mathcal{F}$.

Given a message $\mathbf{y} = (y_1, \dots, y_o)$, in order to sign it, one needs to try to find a vector $\mathbf{w} = (w_1, \dots, w_n)$ such that $\mathcal{P}(\mathbf{w}) = \mathbf{y}$. With the secret key, this can be done easily. First, one guesses values for each vinegar variable $x_1, \dots, x_v$, and obtains a set of o linear equations with the o oil variables $x_{v+1}, \dots, x_n$. With high probability, it has a solution. If it does not have a solution, one guesses at the vinegar variables again until one finds a pre-image of a given element in $\mathbb{K}^o$. Then one applies S^{-1}. To check if $\mathbf{w}$ is indeed a legitimate signature for $\mathbf{y}$, one only needs to get the public map $\mathcal{P}$ and check if indeed $\mathcal{P}(\mathbf{w}) = \mathbf{y}$.

What algebraic property is most significant in an unbalanced Oil-and-Vinegar system? No doubt the lack of pure oil cross-terms. Equivalently, if we have an UOV structure, then the quadratic part of each component q_i in the central map from $\mathbf{x}$ to $\mathbf{y}$, when expressed as a symmetric matrix looks like

$$M_i := \left[\begin{array}{ccc|ccc} \alpha^{(i)}_{11} & \cdots & \alpha^{(i)}_{1v} & \alpha^{(i)}_{1,v+1,} & \cdots & \alpha^{(i)}_{1n} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \alpha^{(i)}_{v1} & \cdots & \alpha^{(i)}_{vv} & \alpha^{(i)}_{v,v+1,} & \cdots & \alpha^{(i)}_{vn} \\ \hline \alpha^{(i)}_{v+1,1,} & \cdots & \alpha^{(i)}_{v+1,v,} & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \alpha^{(i)}_{n1} & \cdots & \alpha^{(i)}_{nv} & 0 & \cdots & 0 \end{array}\right] \quad \left(\text{or for short, } \left[\begin{array}{c|c} * & * \\ \hline * & 0 \end{array}\right]\right). \tag{5}$$

We should mention the fact that there are many *equivalent keys* [WP05]. Computing the essential part of secret keys is part of the attack of Sec. 3.2.

There are two variants of UOV in the second round of the NIST PQC Standardization Project. These are Rainbow and LUOV, which are discussed below.

The Rainbow Signature Scheme

The rainbow signature scheme is essentially multiple layers of the UOV scheme on top of each other. We describe Rainbow [DS05] with u layers:

- The segment structure is given by a sequence $0 < v_1 < v_2 < \cdots < v_{u+1} = n$.

- For $l = 1, \dots, u+1$, set $S_l := \{1, 2, \dots, v_l\}$ so that $|S_l| = v_l$ and $S_0 \subset S_1 \subset \cdots \subset S_{u+1} = S$. Denote by $o_l := v_{l+1} - v_l$ and $O_l := S_{l+1} \setminus S_l$ for $l = 1 \cdots u$.

- The central map $\mathcal{F}$ has component polynomials $y_{v_1+1} = f_{v_1+1}(\mathbf{x})$, $y_{v_1+2} = f_{v_1+2}(\mathbf{x}), \dots,$ $y_n = f_n(\mathbf{x})$ — *notice unusual indexing* — of the following form

$$y_k = f_k(\mathbf{x}) = \sum_{i=1}^{v_l}\sum_{j=i}^{n} \alpha_{ij}^{(k)} x_i x_j + \sum_{i<v_{l+1}} \beta_i^{(k)} x_i, \text{ if } k \in O_l := \{v_l + 1 \cdots v_{l+1}\}.$$

In every f_k, where $k \in O_l$, there is no cross-term $x_i x_j$ where both i and j are in O_l at all. So given all the y_i with $v_l < i \le v_{l+1}$, and all the x_j with $j \le v_l$, we can compute $x_{v_l+1}, \ldots, x_{v_{l+1}}$.

- To expedite computations, some coefficients $(\alpha_{ij}^{(k)})$ may be fixed (e.g., set to zero), chosen at random (and included in the private key), or be interrelated in a predetermined manner.
- To invert $\mathcal{F}$, fix randomly $x_1, \ldots x_{v_1}$, i.e., all x_k, $k \in S_1$. From the components of $\mathbf{y}$ that corresponds to the polynomials $f_{v_1+1}, \ldots f_{v_2}$, we obtain a set of o_1 equations in the variables x_k, $(k \in O_1)$. We can solve these linear equations to obtain the values of x_k, $(k \in O_1)$. Then we substitute these values into the next o_2 equations to get a new linear system of equations. We repeat this process to find all remaining variables.

The following table shows the key and signature sizes of Rainbow instances.

	Security Level	parameters $(\mathbb{F}, v_1, o_1, o_2)$	$\lvert\mathsf{pk}\rvert$ (KB)	$\lvert\mathsf{sk}\rvert$ (KB)	$\lvert\mathsf{sig}\rvert$ (bit)[1]
Rainbow Ia	128	$(\mathbb{F}_{16}, 32, 32, 32)$	58.1	93.0	512
Rainbow IIIc	192	$(\mathbb{F}_{256}, 68, 36, 36)$	206.7	511.4	1,248
Rainbow Vc	256	$(\mathbb{F}_{256}, 92, 48, 48)$	491.9	1,227.1	1,632

[1] including 128 bit salt

Table 4: Key and Signature sizes of Rainbow instances

	parameters $(\mathbb{F}, v_1, o_1, o_2)$	public map time (kC)[1]	private map time (kC)[1]	keypair gen time (MC)[1]
Rainbow Ia	$(\mathbb{F}_{16}, 32, 32, 32)$	26.9	75.4	8.9
Rainbow IIIc	$(\mathbb{F}_{256}, 68, 36, 36)$	141	654	88.3
Rainbow Vc	$(\mathbb{F}_{256}, 92, 48, 48)$	276	836	121

[1] Intel Haswell 2.2GHz, `gcc-7.3`, Linux

Table 5: Speed of Rainbow instances using AVX2 (Haswell)

The Lifted UOV Scheme

The second small-field scheme in the second round of the NIST PQC project is, "Lifted" UOV [BP17]. A public key of the LUOV is the same as a standard UOV public key defined over $\mathbb{F}_2$. However, this public key is lifted to an extension field $\mathbb{F}_2^n \mapsto \mathbb{F}_2^m$ and used as a UOV key over $\mathbb{F}_{2^k}$. This technique is used to reduce the key sizes, because the coefficients of the public and private key now live in $\mathbb{F}_2$, but comes at the cost of larger signatures. The following tables show the key and signature sizes of LUOV instances and the signing and verification speeds.

	Security Level	parameters $(\mathbb{F}, m, v)$	$\vert\mathsf{pk}\vert$ (KB)	$\vert\mathsf{sk}\vert$ (Bytes)	$\vert\mathsf{sig}\vert$ (Bytes)[1]
LUOV-7-57-197	128	$(\mathbb{F}_{2^7}, 57, 197)$	11.5	32	239
LUOV-7-83-283	192	$(\mathbb{F}_{2^7}, 83, 283)$	35.4	32	337
LUOV-7-110-374	256	$(\mathbb{F}_{2^7}, 110, 374)$	82.0	32	440

[1] including salt

Table 6: Key and Signature sizes of LUOV instances

	parameters $(\mathbb{F}, m, v)$	KeyGen time (MC)	Signing time (kC)[1]	Verification time (kC)[1]
LUOV-7-57-197	$(\mathbb{F}_{2^7}, 57, 197)$	1.1	224	49
LUOV-7-83-283	$(\mathbb{F}_{2^7}, 83, 283)$	4.6	643	152
LUOV-7-110-374	$(\mathbb{F}_{2^7}, 110, 374)$	9.7	1100	331

[1] Assumes some preprocessing on the secret key/private key.

Table 7: Speed of Rainbow instances using AVX2 (Kaby Lake)

Attacks on UOV

UOV attack of Kipnis and Shamir The original Oil and Vinegar proposal was made with $o = v$, i.e. where the number of oil and vinegar variables is the same. This turned out to be insecure due to the attack of Kipnis and Shamir [KS98].

The basic observation here is that if one considers the symmetric matrices M_i associated to the polynomials p_i in the public key, then it holds that all the products $M_i^{-1}M_j$, for $i, j \in \{1, \cdots, m\}$ share a common eigenspace, namely the image of the oil variables after applying the secret map T. It turns out that from the products $M_i^{-1}M_j$ this common eigenspace can be recovered in polynomial time, and that this eigenspace reveals enough information about the secret key T to be able to forge signatures on arbitrary messages.

Later it was shown by Kipnis *et al* [KPG99] that the same attack works if $v < o$; even if $v > o$ it can be done in time directly proportional to q^{v-o}, and hence parameters should be chosen such that $v - o$ is not too small.

reconciliation attack The reconciliation attack of Ding *et al.* [DYC$^+$08] attempts to find a sequence of changes of basis that put the public key back into the form of the secret key, which lets us invert the public map and forge signatures. Equivalently, the goal is to compute matrices $T_1, \cdots, T_m$, such that their product T is an equivalent secret key.

The key observation is that each T_i can be computed separately by solving a system of quadratic equations. The complexity of the attack is dominated by computing T_1, which amounts to solving a system of o equations in $n - o$ variables. However, this only works when $o > n - o$, because otherwise the initial system is underdetermined, and there is not enough information to recover T_1 (which is unique, so randomly guessing variables does not help).

Attacks on Rainbow

MinRank and HighRank attacks In a Rainbow public key, there are different kinds of equations. The equations in the first layer contain only terms containing the first v_2 variables, the next layer consists of equations with the first v_3 variables, while the last layer has terms containing all the variables. These equations are mixed with the affine map S. However, by considering the rank of the quadratic forms corresponding to each of the central equations. Suppose M is the matrix that corresponds to (the quadratic part) of a polynomial in the first layer of the secret key. Since this polynomial only involves v_2 variables, the rank of this matrix will be at most v_2. Applying the affine transformation T maps this matrix to T^tMT, which does not affect the rank. Therefore, to find a polynomial in the first layer it suffices to find a linear combination of the public polynomials that has low rank. This problem is called the MinRank problem and was already studied in the context of attacking the HFEv- system.

Rather than searching for polynomials that involve few variables, it is also possible to look for variables that only appears in very few polynomials. This attack is called the Dual Rank attack or HighRank attack. This attack works by guessing a linear combination of the secret polynomial (in the hope that this lies in the span of the secret polynomials without the last layer). If the guess is correct, then the matrix that corresponds to the guessed polynomial will have a nontrivial kernel (i.e. because the last $n - v_u$ variables do not appear in the equation). Moreover, this kernel is shared by all the polynomials in the span of the polynomials in the first $u - 1$ layers of the secret key. The complexity of this attack is approximately $\left(sn^2 + n^3/6\right) q^{n-v_u}$ field multiplications.

Rainbow band separation attack The reconciliation attack against UOV can be improved in the Rainbow setting [DYC$^+$08]. Recall that in the Rainbow scheme the last column of the matrices corresponding to the central equations in all but the last layer is zero. This can be expressed by adding $n-1$ additional equations and o additional variables to the first reconciliation system. Therefore the complexity of the improved attack (called Rainbow separation attack) is that of solving a system of $n + m - 1$ equations in n variables.

Attacks on LUOV

The LUOV scheme uses a public key defined over $\mathbb{F}_2$ as a map over an extension field $\mathbb{F}_{2^k}$. Therefore, it is important that solving a system over $\mathbb{F}_{2^k}$ where all the coefficients except for the linear terms are in $\mathbb{F}_2$ is hard. Initially, the designers of LUOV assumed that solving such a system is equally hard as solving a system where all the coefficients are in $\mathbb{F}_{2^k}$. It was shown by Ding et al. that this is not true in the case that the system is underdetermined and when k is a composite number (such that there exist non-trivial subfields of $\mathbb{F}_{2^k}$) [DZD+19]. They show that the problem of solving a system of m polynomials in n variables of this special form over $\mathbb{F}_{2^k}$, can be reduced to solving a system of m polynomials over a subfield field $\mathbb{F}_{2^{k'}}$, where $k'|k$ and $k'n > km$. Therefore, the LUOV designers proposed updated parameters where k is chosen to be prime.

4 Fiat-Shamir Multivariate Signature schemes

Rather than constructing trapdoors that might be susceptible to structural attacks, it is also possible to build signatures that only depend on the hardness of uniformly random instances of the $\mathcal{M}Q$ problem. One such scheme is MQDSS, a digital signature scheme which is a second-round candidate for the NIST PQC project. MQDSS is derived from a zero-knowledge proof-based Identification scheme by Sakumoto, Shirai, and Hiwatari [SSH11]. It has a security reduction to $\mathcal{MQ}$, but unfortunately, it has large signature sizes compared to the $\mathcal{M}Q$-based trapdoor signature schemes. We describe the 5-pass public-key Identification scheme on which MQDSS is based below.

- $\mathcal{P}$ be a random MQ instance
- Its "polar" form $D\mathcal{P}(\mathbf{x}, \mathbf{y}) := \mathcal{P}(\mathbf{x}+\mathbf{y}) - \mathcal{P}(\mathbf{x}) - \mathcal{P}(\mathbf{y}) - \mathcal{P}(\mathbf{0})$
- $\mathcal{P}(\mathbf{s}) = \mathbf{p}$ is the public key, $\mathbf{s}$ is the secret.
- Peter picks and commits random $(\mathbf{r_0}, \mathbf{t_0}, \mathbf{e_0})$, sets $\mathbf{r_1} = \mathbf{s} - \mathbf{r_0}$ and commits $(\mathbf{r_1}, D\mathcal{P}(\mathbf{t_0}, \mathbf{r_1}) + \mathbf{e_0})$.
- Vera sends random α,
- Peter sets and sends $\mathbf{t_1} := \alpha\mathbf{r_0} - \mathbf{t_0}$, $\mathbf{e_1} := \alpha\mathcal{P}(\mathbf{r_0}) - \mathbf{e_0}$.
- Vera sends challenge Ch, Peter sends $\mathbf{r}_{Ch}$.
- Vera checks the commit of either $(\mathbf{r_0}, \alpha\mathbf{r_0} - \mathbf{t_1}, \alpha\mathcal{P}(\mathbf{r_0}) - \mathbf{e_1})$ or $(\mathbf{r_1}, \alpha(\mathbf{p} - \mathcal{P}(\mathbf{r_1})) - D\mathcal{P}(\mathbf{t_1}, \mathbf{r_1}) - \mathbf{e_1})$.

With r repetitions for large enough r Vera can be satisfied that Peter knows $\mathbf{s}$. The Fiat-Shamir transform of this ID scheme is the MQDSS scheme.

The following tables show the key and signature sizes and of MQDSS instances and the performance measurements of its implementation. Recently, an improved zero-knowledge proof for MQ has been proposed by Beullens [Beu], which results in a signature scheme (called MUDFISH) that is provably secure in the QROM (in contrast to the ROM), and which results in a signature scheme that is roughly a factor 2 faster and with roughly a factor 2 smaller signatures.

	Security Level	parameters (k, q, n, r)	public key size (B)	private key size (B)	signature size (kB)[1]
MQDSS I/II	128	$(128, 31, 48, 184)$	46	16	27.8
MQDSS III/IV	192	$(192, 31, 64, 277)$	64	24	58.5

[1] We have updated the parameters to take the recent attack of Kales and Zaverucha into account [KZ19].

Table 8: Key and Signature sizes of MQDSS instances

	parameters (k, q, n, r)	public map time (MC)	private map time (MC)	keypair gen time (MC)
MQDSS I/II	$(128, 31, 48, 184)$	5.2	3.5	1.1
MQDSS III/IV	$(192, 31, 64, 277)$	12.3	8.4	2.5

Table 9: Key and Signature sizes of MQDSS instances

References

[Beu] Ward Beullens. On sigma protocols with helper for mq and pkp, fishy signature schemes and more. Technical report, Cryptology ePrint Archive, Report 2019/490, 2019. https://eprint. iacr. org .

[BFS04] M. Bardet, J.-C. Faugère, and B. Salvy. On the complexity of Gröbner basis computation of semi-regular overdetermined algebraic equations. In *Proceedings of the International Conference on Polynomial System Solving*, pages 71–74, 2004. Previously INRIA report RR-5049.

[BP17] Ward Beullens and Bart Preneel. Field lifting for smaller UOV public keys. In Arpita Patra and Nigel P. Smart, editors, *Progress in Cryptology - INDOCRYPT 2017 - 18th International Conference on Cryptology in India, Chennai, India, December 10-13, 2017, Proceedings*, volume 10698 of *Lecture Notes in Computer Science*, pages 227–246. Springer, 2017.

[Buc65] B. Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal.* PhD thesis, Innsbruck, 1965.

[CGMT02] Nicolas Courtois, Louis Goubin, Willi Meier, and Jean-Daniel Tacier. Solving underdefined systems of multivariate quadratic equations. In *Public Key Cryptography — PKC 2002*, volume 2274 of *Lecture Notes in Computer Science*, pages 211–227. David Naccache and Pascal Paillier, editors, Springer, 2002.

[CKPS00] Nicolas T. Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *Advances in Cryptology — EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 392–407. Bart Preneel, ed., Springer, 2000. Extended Version: `http://www.minrank.org/xlfull.pdf`.

[Cou04] Nicolas Courtois. Algebraic attacks over $GF(2^{\mathrm{k}})$, application to HFE challenge 2 and SFLASH-v2. In *Public Key Cryptography — PKC 2004*, volume 2947 of *Lecture Notes in Computer Science*, pages 201–217. Feng Bao, Robert H. Deng, and Jianying Zhou (editors), Springer, 2004. ISBN 3-540-21018-0.

[CP02] Nicolas T. Courtois and Josef Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In *Advances in Cryptology — ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 267–287. Yuliang Zheng, ed., Springer, 2002.

[CP03] Nicolas T. Courtois and Jacques Patarin. About the XL algorithm over gf(2). In *The Cryptographer's Track at RSA Conference 2003*, volume 2612 of *Lecture Notes in Computer Science*, pages 141–157. Springer, 2003.

[Die04] Claus Diem. The XL-algorithm and a conjecture from commutative algebra. In *Advances in Cryptology — ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 323–337. Pil Joong Lee, ed., Springer, 2004. ISBN 3-540-23975-8.

[DS05] Jintai Ding and Dieter Schmidt. Rainbow, a new multivariable polynomial signature scheme. In *Conference on Applied Cryptography and Network Security — ACNS 2005*, volume 3531 of *Lecture Notes in Computer Science*, pages 164–175. Springer, 2005.

[DY13] Jintai Ding and Bo-Yin Yang. Degree of Regularity for HFEv and HFEv-. In Philippe Gaborit, editor, *PQCrypto*, volume 7932 of *Lecture Notes in Computer Science*, pages 52–66. Springer, 2013.

[DYC$^+$08] Jintai Ding, Bo-Yin Yang, Chia-Hsin Owen Chen, Ming-Shing Chen, and Chen-Mou Cheng. New differential-algebraic attacks and reparametrization of rainbow. In *Applied Cryptography and Network Security*, volume 5037 of *Lecture Notes in Computer Science*, pages 242–257. Springer, 2008. cf. `http://eprint.iacr.org/2008/108`.

[DZD$^+$19] Jintai Ding, Zheng Zhang, Joshua Deaton, Kurt Schmidt, and FNU Vishakha. New attacks on lifted unbalanced oil vinegar, 2019. Published at the Second PQC Standardization Conference, `https://csrc.nist.gov/CSRC/media/Events/Second-PQC-Standardization-Conference/documents/accepted-papers/ding-new-attacks-luov.pdf`.

[Fau99] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases (F_4). *Journal of Pure and Applied Algebra*, 139:61–88, June 1999.

[Fau02] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). In *International Symposium on Symbolic and Algebraic Computation — ISSAC 2002*, pages 75–83. ACM Press, July 2002.

[FJ03] Jean-Charles Faugère and Antoine Joux. Algebraic cryptanalysis of Hidden Field Equations (HFE) using Gröbner bases. In *Advances in Cryptology — CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 44–60. Dan Boneh, ed., Springer, 2003.

[JV17] Antoine Joux and Vanessa Vitse. A crossbred algorithm for solving boolean polynomial systems. In Jerzy Kaczorowski, Josef Pieprzyk, and Jacek Pomykala, editors, *Number-Theoretic Methods in Cryptology - First International Conference, NuTMiC 2017, Warsaw, Poland, September 11-13, 2017, Revised Selected Papers*, volume

10737 of *Lecture Notes in Computer Science*, pages 3–21. Springer, 2017.

[KPG99] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced Oil and Vinegar signature schemes. In *Advances in Cryptology — EUROCRYPT 1999*, volume 1592 of *Lecture Notes in Computer Science*, pages 206–222. Jacques Stern, ed., Springer, 1999.

[KS98] Aviad Kipnis and Adi Shamir. Cryptanalysis of the oil and vinegar signature scheme. In *Advances in Cryptology — CRYPTO 1998*, volume 1462 of *Lecture Notes in Computer Science*, pages 257–266. Hugo Krawczyk, ed., Springer, 1998.

[KZ19] Daniel Kales and Greg Zaverucha. Forgery attacks on mqdssv2. 0. 2019.

[Laz83] Daniel Lazard. Gröbner-bases, Gaussian elimination and resolution of systems of algebraic equations. In *EUROCAL 83*, volume 162 of *Lecture Notes in Computer Science*, pages 146–156. Springer, March 1983.

[Mac16] F. S. Macaulay. *The algebraic theory of modular systems*, volume xxxi of *Cambridge Mathematical Library.* Cambridge University Press, 1916.

[Pat97] Jacques Patarin. The oil and vinegar signature scheme. *Dagstuhl Workshop on Cryptography, September 1997*, 1997.

[SSH11] Koichi Sakumoto, Taizo Shirai, and Harunaga Hiwatari. Public-key identification schemes based on multivariate quadratic polynomials. In Phillip Rogaway, editor, *CRYPTO*, volume 6841 of *Lecture Notes in Computer Science*, pages 706–723. Springer, 2011.

[WP05] Christopher Wolf and Bart Preneel. Superfluous keys in $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic asymmetric systems. In *Public Key Cryptography — PKC 2005*, volume 3386 of *Lecture Notes in Computer Science*, pages 275–287. Serge Vaudenay, ed., Springer, 2005. Extended version `http://eprint.iacr.org/2004/361/`.

[YC04] Bo-Yin Yang and Jiun-Ming Chen. Theoretical analysis of XL over small fields. In *ACISP 2004*, volume 3108 of *Lecture Notes in Computer Science*, pages 277–288. Springer, 2004.