Small Public Keys and Fast Verification for $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic Public Key Systems

Albrecht Petzoldt[1], Enrico Thomae[2], Stanislav Bulygin[3], and Christopher Wolf[4]

[1,3]Technische Universität Darmstadt and
Center for Advanced Security Research Darmstadt (CASED)
[1]apetzoldt@cdc.informatik.tu-darmstadt.de, [3]Stanislav.Bulygin@cased.de
[2,4]Horst Görtz Institute for IT-security
Faculty of Mathematics
Ruhr-University of Bochum, 44780 Bochum, Germany
[2]Enrico.Thomae.rub.de, [4]Christopher.Wolf@rub.de, [4]chris@Christopher-Wolf.de

Abstract. Security of public key schemes in a post-quantum world is a challenging task—as both RSA and ECC will be broken then. In this paper, we show how post-quantum signature systems based on $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic ($\mathcal{MQ}$) polynomials can be improved up by about 9/10, and 3/4, respectively, in terms of public key size and verification time. The exact figures are 88% and 73%. This is particularly important for small-scale devices with restricted energy, memory, or computational power. In addition, we show that this reduction does not affect security and that it is also optimal in terms of possible attacks. We do so by combining the priory unrelated concepts of reduced and equivalent keys. Our new scheme is based on the so-called *Unbalanced Oil and Vinegar* class of $\mathcal{MQ}$-schemes. We have derived our results mathematically and verified the speed-ups through a C++ implementation.

Key words: Multivariate Quadratic Cryptography, Post-Quantum Cryptography, Implementation, Unbalanced Oil and Vinegar, UOV Signature Scheme

Current Version: 2011-06-03

1 Introduction

When finding an old sonnet of Shakespeare, we can usually determine its validity accurately by checking the wording, the ink, the paper, and so on. Similar techniques apply in disputes over last wills—or other documents of historical or financial interest. Even if they are several decades old, we can fairly certainly determine if they have been written by the person in question and sometimes even date them accurately.
For digital documents, this is a much more difficult task. They are electronically signed with the help of so-called *digital signature schemes*. The ones widely used today are Digital Signature Algorithms (DSAs) based on RSA and elliptic curve cryptography (ECDSA). Unfortunately, all these schemes are broken if large enough quantum computers will be built. The reason is the algorithm of Shor which breaks all cryptographic algorithms based on the difficulty of factoring and the discrete logarithm (DL) problem [Sho97]. This covers DL over numbers, RSA, and ECDSA. Even if unlikely now, quantum computers may be available in the medium future and are hence a concern for long-term-validity of authentication data. We must be sure that a document signed today is not repudiated 50 years later. Likewise, we do not want a signature that is generated today to be forged in the future. So to guard security even in the presence of quantum computers, post-quantum cryptography is needed and has hence become a vital research area [BBD09]. One possible solution in this context is the so-called *$\mathcal{M}$ultivariate $\mathcal{Q}$uadratic cryptography*. It is widely believed that it is secure against quantum computers.
In addition, $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic (or $\mathcal{MQ}$ for short) signature schemes have nice properties in terms of speed of signature generation and verification which make them superior to DL,

RSA and ECDSA. Note that ECDSA is the most efficient of the three. However, even when comparing to signature generation in $\mathcal{MQ}$ and ECC, the first are a factor of 2–50 faster on FPGA than the latter [BERW08]. Similar results have been demonstrated for comparison with RSA and ECC in software [YCC04, YCCC06, CCC$^+$08, CCC$^+$09]. One of the main reasons for this higher efficiency is the comparably small finite fields, *e.g.* $\mathbb{F}_{2^8}$ which allows for efficient hardware and software implementations. The other operations usually boil down to vector-matrix functions, which can be implemented efficiently, too. As an immediate consequence, we can use $\mathcal{MQ}$ schemes in restricted devices, *i.e.* with low energy or computational power.
Another point is the high flexibility of $\mathcal{MQ}$-schemes. This allows for the use of sparse polynomials in the *private key* as done in the TTS schemes of Yang, Chen, and Chen [YCC04]. This leads both to a significant reduction of the time needed for signature generation, as well as for the size of the private key. Another way to reduce the private key is by choosing the coefficients of the private maps from smaller fields (*e.g.* $\mathbb{F}_{16}$ instead of $\mathbb{F}_{256}$)), [CCC$^+$08]. In addition, we want to mention the so-called *similar keys* which exploit linear relations between public and private key [HWCL05]. However, they are not applicable to UOV and hence, do not achieve such a drastic reduction in size. Finally, one research direction deals with reducing the public key directly. In [PBB10, PBB11] it is shown how to reduce the public key size of the UOV scheme by choosing public coefficients in a structured way, cf. sect. 3.

1.1 Outline of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic Schemes

The main idea behind $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic cryptography is to choose a system $\mathcal{F}$ of m quadratic polynomials in n variables which can be easily inverted. Here $\mathcal{F}$ is called the *central map*. In addition, we need invertible affine maps S and T to hide the structure of the central map $\mathcal{F}$. The public key of the cryptosystem is now composed as $\mathcal{P} = T \circ \mathcal{F} \circ S$. For a secure $\mathcal{MQ}$-system, $\mathcal{P}$ must be difficult to invert. The private key consists of $(\mathcal{F}, T, S)$ and therefore allows efficient inversion of $\mathcal{P}$. More details on $\mathcal{MQ}$-systems are given in sect. 2. A more detailed overview on $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic schemes can be found in [Wol05, WP05b].

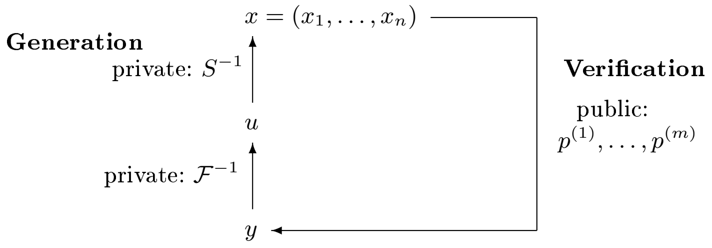

Fig. 1. Graphical representation of the UOV trapdoor $(\mathcal{F}, \mathcal{S})$ for signature generation and verification. Given is a hash value $y \in \mathbb{F}^m$ and a signature $x \in \mathbb{F}^n$.

Note that some $\mathcal{MQ}$-schemes, *e.g.* the Unbalanced Oil and Vinegar (UOV) signature scheme considered in this paper, omit T and construct their public key as $\mathcal{P} := \mathcal{F} \circ S$. See Fig. 1 for an overall picture of signature generation and verification for UOV.

1.2 Achievement

Combining two previously unrelated ideas, we deal with reducing the size of the public key. For $\mathcal{MQ}$ schemes like *Unbalanced Oil and Vinegar* (UOV—see below), typical choices of parameters lead to around 80 kByte for the public key. With our technique, we can bring this down to 9 kByte

(88.6% reduction). A similar approach for Unbalanced Oil and Vinegar was previously exploited in [PBB10, PBB11] but did not allow such drastic reductions. In addition, our modification also works for restricting the choice of the coefficients. By choosing them partially to be 0 or 1 only, verification time is reduced by up to 72%. This way, $\mathcal{MQ}$ verification can be performed in low-power, low-energy devices. For example for mobile devices, we can easily imagine a scenario where a server signs data which needs to be verified by a (comparably restricted) phone.
As further contribution, using Turán graphs we demonstrate that this reduction in size does *not* affect security. This is due to an observation regarding equivalent keys of [WP05a, WP11]. Moreover, we show that our size reduction is tight in the sense that any further reduction would reduce the security.

1.3 Organization

The structure of this paper is as follows: After giving some introduction in sect. 1, we continue with the background on $\mathcal{MQ}$-schemes and in particular UOV in sect. 2. In sect. 3, we review the cyclic construction from [PBB10]. This is followed by a security proof regarding cyclic keys in UOV in sect. 4. Using these results, we outline our new constructions, its implementation, efficiency, and security implications in sect. 5. The paper concludes with sect. 6. In app. A, we give some background on Turán graphs, and in app. B, we show how this relates to our monomial ordering in sect. 5.3.

2 $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic Cryptography

2.1 Notation

Solving non-linear systems of m equations in n variables over a finite field is a difficult problem in general. Restricting to the seemingly *easy* case of quadratic equations is still difficult. Actually this problem is also known as $\mathcal{MQ}$-problem which is proven to be NP-hard in the worst-case [GJ79], even over $\mathbb{F}_2$.

Let $\mathcal{P}$ be a $\mathcal{MQ}$ system of the form

$$\begin{aligned} p^{(1)}(x_1,\ldots,x_n) &= 0\\ p^{(2)}(x_1,\ldots,x_n) &= 0\\ &\vdots\\ p^{(m)}(x_1,\ldots,x_n) &= 0, \end{aligned} \tag{1}$$

with

$$p^{(k)}(x_1,\ldots,x_n) := \sum_{1\le i\le j\le n} \gamma_{ij}^{(k)} x_i x_j + \sum_{1\le i\le n} \beta_i^{(k)} x_i + \alpha^{(k)}. \tag{2}$$

Let $\pi^{(k)}$ be the coefficient vector of $p^{(k)}(x_1,\ldots,x_n)$ w.r.t. lexicographic order of monomials, *i.e.*

$$\pi^{(k)} = (\gamma_{11}^{(k)}, \gamma_{12}^{(k)}, \ldots, \gamma_{1n}^{(k)}, \gamma_{22}^{(k)}, \gamma_{23}^{(k)}, \ldots, \gamma_{nn}^{(k)}, \beta_1^{(k)}, \ldots, \beta_n^{(k)}, \alpha^{(k)}).$$

Let M_P be the corresponding coefficient matrix

$$M_P := \begin{pmatrix} \pi^{(1)} \\ \vdots \\ \pi^{(m)} \end{pmatrix}.$$

Denote with $\mathcal{MQ}(\mathbb{F}^n, \mathbb{F}^m)$ the class of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic polynomial systems in n input variables and m polynomials each, as defined in (2). We now have the public key of multivariate schemes as a vector $\mathcal{P} \in \mathcal{MQ}(\mathbb{F}^n, \mathbb{F}^m)$. Each coefficient is a quadratic polynomial $p^{(i)}$ from $\mathbb{F}[x_1, \ldots, x_n]$ and $1 \leq i \leq m$

$$\mathcal{P} := \begin{pmatrix} p^{(1)}(x_1, \ldots, x_n) \\ \vdots \\ p^{(m)}(x_1, \ldots, x_n) \end{pmatrix}.$$

In the following we omit the constant and linear part of $p^{(k)}$ as it was shown as early as 1998 that it does not contribute to the security of UOV [KS98, KPG99].
Note that the ordering of monomials (and thus coefficients) in the matrix M_P does not necessarily have to be lexicographic ordering. We may want to order monomials of the public key in a certain way. Therewith, the ordering of coefficients (columns of M_P) is then changed accordingly.

2.2 Unbalanced Oil and Vinegar

In this subsection we introduce the Oil and Vinegar Signature Scheme, which was proposed by J. Patarin in [Pat97].

Let $\mathbb{F}_q$ be a finite field. Denote the number of oil variables by $o \in \mathbb{N}$, the number of vinegar variables by $v \in \mathbb{N}$ and set $n := o + v$. Let $V := \{1, \ldots, v\}$ and $O := \{v + 1, \ldots, n\}$ denote the sets of indices of vinegar and oil variables. The private key $\mathcal{F} := (f^{(1)}(u), \ldots, f^{(m)}(u))$ is defined by

$$f^{(k)}(u_1, \ldots, u_n) := \sum_{i \in V, j \in O} \widetilde{\gamma}_{ij}^{(k)} u_i u_j + \sum_{i,j \in V, i \leq j} \widetilde{\gamma}_{ij}^{(k)} u_i u_j\,. \tag{3}$$

It is important for finding a pre-image that the variables in $f^{(k)}$ are not completely mixed, *i.e.* oil variables are only multiplied by vinegar variables and never by oil variables. This construction leads to an efficient way to invert $\mathcal{F}$. If we assign arbitrary values to the vinegar variables and if we set $m = o$ we obtain a system of o linear equations in o variables. With high probability this system has a solution. If not we try again with a different choice for the vinegar variables $x_1, \ldots, x_v$. In the public key $\mathcal{P}$, the central map $\mathcal{F}$ is hidden by composing it with a linear map $S : \mathbb{F}_q^n \to \mathbb{F}_q^n$, *i.e.* $\mathcal{P} := \mathcal{F} \circ S$.

$$\begin{tikzcd} \mathbb{F}_q^n \arrow[r, "\mathcal{P}"] \arrow[d, "S"'] & \mathbb{F}_q^m \\ \mathbb{F}_q^n \arrow[ur, "\mathcal{F}"'] & \end{tikzcd}$$

Signature generation: To sign a document d, one uses a hash function $\mathcal{H} : \mathbb{F}_q^\star \to \mathbb{F}_q^m$ to compute the hash value $\mathbf{h} = \mathcal{H}(d) \in \mathbb{F}_q^m$. After that one computes first $\mathbf{y} := \mathcal{F}^{-1}(\mathbf{h})$ and then $\mathbf{z} := S^{-1}(\mathbf{y})$. The signature of the document d is $\mathbf{z} \in \mathbb{F}_q^n$. In a slight abuse of notation we write $\mathcal{F}^{-1}(\mathbf{h})$ for finding one (of possibly many) pre-image of $\mathbf{h}$ under $\mathcal{F}$.
Signature verification: To verify the authenticity of a signature, one computes the hash value $\mathbf{h}$ of the corresponding document and the value $\mathbf{h}' = \mathcal{P}(\mathbf{z})$. If $\mathbf{h} = \mathbf{h}'$ holds, the signature is accepted, otherwise rejected.
In his original paper [Pat97], Patarin suggested to use $o = v$ (Balanced Oil and Vinegar—OV). After this scheme was broken by Kipnis and Shamir in [KS98], it was proposed in [KPG99] to use $2o \leq v$ (Unbalanced Oil and Vinegar (UOV)). UOV parameters $q = 256$, $(o, v) = (26, 52)$ give 80-bit security against the most efficient attacks currently known [BFP09].

3 Reviewing Cyclic Keys

In this section we review the approach of [PBB10] to create a UOV-based scheme with a partially cyclic public key.
Remember that, in the case of the Unbalanced Oil and Vinegar signature scheme [KPG99], the public key $\mathcal{P}$ is given as the concatenation of the central UOV-map $\mathcal{F}$ and a linear invertible map S, *i.e.*

$$\mathcal{P} = \mathcal{F} \circ S. \tag{4}$$

In [PBB10] it is observed, that this equation (after fixing the affine map S), leads to a linear relation between the coefficients of the quadratic monomials of $\mathcal{P}$ and $\mathcal{F}$ of the form

$$p_{ij}^{(k)} = \sum_{r=1}^{n} \sum_{s=r}^{n} \alpha_{ij}^{rs} \cdot f_{rs}^{(k)}, \tag{5}$$

where $p_{ij}^{(k)}$ and $f_{ij}^{(k)}$ are the coefficients of $x_i x_j$ in the k-th component of $\mathcal{P}$ and $\mathcal{F}$ respectively and the α_{ij}^{rs} are given as

$$\alpha_{ij}^{rs} = \begin{cases} s_{ri} \cdot s_{si} & (i = j) \\ s_{ri} \cdot s_{sj} + s_{rj} \cdot s_{si} & \text{otherwise} \end{cases}. \tag{6}$$

Here $s_{ij} \in \mathbb{F}$ denote the coefficients of the linear map S. Let $D := \frac{v \cdot (v+1)}{2} + ov$ be the number of non-zero quadratic terms in any component of $\mathcal{F}$ and $D' := \frac{n \cdot (n+1)}{2}$ be the number of quadratic terms in the public polynomials. Let M_P and M_F be the coefficient matrices of $\mathcal{P}$ and $\mathcal{F}$ respectively (w.r.t. graded lexicographical ordering of monomials). The matrices M_P and M_F are divided into submatrices as shown in Figure 2. Note that, due to the absence of oil $\times$ oil terms in the central polynomials, we have a block of zeros in the middle of M_F.

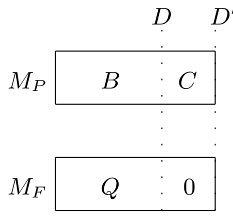

Fig. 2. Layout of the matrices M_P and M_F

Furthermore, the authors of [PBB10] defined the so called transformation matrix $A_{UOV} \in \mathbb{F}_q^{D \times D}$ containing the coefficients α_{ij}^{rs} of equation (5)

$$A_{UOV} = \left(\alpha_{ij}^{rs}\right) \; (1 \leq r \leq v, r \leq s \leq n \text{ for the rows}, \; 1 \leq i \leq v, i \leq j \leq n \; \text{ for the columns}), \textit{i.e.}$$

$$A_{UOV} = \begin{pmatrix} \alpha_{11}^{11} & \alpha_{12}^{11} & \dots & \alpha_{vn}^{11} \\ \alpha_{11}^{12} & \alpha_{12}^{12} & \dots & \alpha_{vn}^{12} \\ \vdots & & & \vdots \\ \alpha_{11}^{vn} & \alpha_{12}^{vn} & \dots & \alpha_{vn}^{vn} \end{pmatrix}. \tag{7}$$

With this notation, equation (5) yields

$$B = Q \cdot A_{UOV} \tag{8}$$

If matrix A_{UOV} is invertible, this equation has a solution for Q. Experiments indicate that this condition is fulfilled with high probability. By solving equation (8) for Q, the authors of [PBB10] were able to insert a partially circulant matrix B into the UOV public key. By doing so, they reduced the public key size of the scheme by a factor of 6. After choosing matrix B, we can use alg. 1 to compute the corresponding key.
Note that by $T := S^{-1}$ and

$$\alpha_{ij}^{rs} = \begin{cases} t_{ri} \cdot t_{si} & (i = j) \\ t_{ri} \cdot t_{sj} + t_{rj} \cdot t_{si} & \text{otherwise} \end{cases} \tag{9}$$

we derive a map $\widehat{A}_{UOV} = (\alpha_{ij}^{rs})$ with $1 \leq r, s \leq n$ and $1 \leq i, j \leq v$ that maps the public to the private coefficients, *i.e.*

$$Q = M_P \cdot \widehat{A}_{UOV}. \tag{10}$$

Algorithm 1 summarizes key generation:

Algorithm 1 Key Generation for UOV schemes

```
1: Choose an $o \times D$ matrix $B$ (e.g. partially circulant or generated by an LRS).
2: Choose randomly a linear map $\mathcal{S}$ (represented by an $n \times n$-matrix $S$). If $S$ is not invertible, choose again.
3: Compute for $S$ the corresponding transformation matrix $A_{\text{UOV}}$ (using equations (6) and (7)). If $A_{\text{UOV}}$ is not invertible, go back to step 2.
4: Solve the linear system given by equation (8) to get the matrix $Q$ and there with the coefficients of the central polynomials.
5: Compute the public key as $\mathcal{P} = \mathcal{F} \circ \mathcal{S}$.
```

4 Security of UOV

Due to equivalent keys [WP05a, WP11] UOV contains a lot of redundancy. We show which part of the public key is important for security and which part can be chosen such that the public key gets as small as possible.
It is rather intuitive that the linear and constant part of the public key do not provide extra security because we can easily separate them from the quadratic part. This was previously exploited by Kipnis and Shamir in their cryptanalysis of (balanced) Oil and Vinegar [KS98]. But it is quite surprising that also a fraction of the quadratic part is not essential for security. This is implied by the observation of equivalent keys by Wolf and Preneel [WP11]. Remember $p^{(k)}(x)$ for $1 \leq k \leq o$ are the public polynomials and $f^{(k)}(u)$ are the private polynomials. The private linear transformation was denoted by S, *i.e.* $Sx = u$. We call a polynomial f' equivalent to the private polynomial f if it has the same structure, *i.e.* no quadratic terms in oil variables. The following transformation Ω on the variables u preserves the structure of f.

$$\Omega = \begin{pmatrix} \Omega_{v \times v}^{(1)} & 0 \\ \Omega_{o \times v}^{(2)} & \Omega_{o \times o}^{(3)} \end{pmatrix}$$

Let $u' := \Omega u$ and $u'^{\intercal} C' u'$ a private polynomial denoted in quadric form. This means that C' is a matrix in $\mathbb{F}^{n \times n}$. The vinegar variables $u'_1, \ldots, u'_v$ are expressed as sums of vinegar variables $u_1, \ldots, u_v$ by transformation Ω and thus no quadratic term in oil variables occurs in $f(u) = u^{\intercal} \Omega^{\intercal} C' \Omega u$. By

$$x^{\intercal} \widetilde{C} x = x^{\intercal} S^{\intercal} \hat{C} S x = x^{\intercal} S^{\intercal} \Omega^{\intercal} C' \Omega S x$$

$\hat{C}$ and C' describe equivalent private maps of the public map $\widetilde{C}$ and thus S and ΩS are equivalent private keys. Let

$$S = \begin{pmatrix} S^{(1)} & S^{(2)} \\ S^{(3)} & S^{(4)} \end{pmatrix}$$

By choosing Ω for which $\Omega^{(1)}S^{(1)} = I$, $\Omega^{(2)}S^{(2)} + \Omega^{(3)}S^{(4)} = I$ and
$\Omega^{(2)}S^{(1)} + \Omega^{(3)}S^{(3)} = 0$ hold, we get a private key $\widetilde{S}$ of following form

$$\widetilde{S} = \begin{pmatrix} I & S' \\ 0 & I \end{pmatrix}. \tag{11}$$

For a key recovery attack it is sufficient to find any of the equivalent keys. Thus an attacker can search for a private key of form (11).

Remark 1. If the private linear map S is of form (11), the following condition on the coefficients of the public and private map hold

$$\gamma_{ij}^{(k)} = \widetilde{\gamma}_{ij}^{(k)} \text{ for all } k \text{ and } i, j \in V, i \leq j$$

This means we can assume that the attacker actually knows or even chooses all coefficients of squared vinegar variables in the private map. This means that the $\gamma_{ij}^{(k)}$ with $i, j \in V$ in the public map do not hide any secret and thus we can choose them in an arbitrary way, *e.g.* matrix B cyclic or even fixed.

Proposition 1. *The first $\frac{v(v+1)}{2}$ coefficients in the public key $\mathcal{P}$ do not provide any security in the sense of key recovery attacks. Arbitrarily fixing these coefficients does not leak information about the private key.*

By equation (8) we are even able to choose the first $\frac{v(v+1)}{2} + ov$ coefficients of $\mathcal{P}$ of a special shape and thus save memory. Proving the security of this construction is not as obvious as in the latter construction. We have to show that additionally fixing ov coefficients does not leak any information about the private maps S or $\mathcal{F}$, *i.e.* we have to show that the remaining non-zero coefficients $\widetilde{\gamma}_{ij}^{(k)}$ for $i \in V$ and $j \in O$ are not biased by the fixed choice of $\mathcal{P}$, *i.e.* uniformly distributed over the random choice of elements in S.
Usually, we fix values on the central polynomials $\mathcal{F}$ and then compute the public polynomials $\mathcal{P}$ for a given linear transformation S. However, for this construction we turn things around by *first* fixing the public polynomials and *then* computing the missing parts of the central polynomials. Intuitively, this should be equally secure. We capture this in the following proposition and also give a necessary condition on B for the security of $\mathcal{F}, \mathcal{P}, S$.

Claim. The first $\frac{v(v+1)}{2} + ov$ coefficients in the public key $\mathcal{P}$ do not provide any security in the sense of key recovery attacks. Fixing these coefficients at random does not leak information about the private key.

We have to show that despite of fixing B we do not get systematic dependencies among the coefficients $\widetilde{\gamma}_{ij}^{(k)}$, *i.e.* for every choice of S we get another private map $\mathcal{F}$. In other words, we could reach q^{n^2} choices of private coefficients $\widetilde{\gamma}_{ij}^{(k)}$ (with $i \in V, j \in O, 1 \leq k \leq o$) by choosing S. Due to equivalent keys the space of private maps $\mathcal{F}$ decomposes into equivalence classes. As every attacker can choose a private key S according to (11) it is sufficient to show that we could produce at least one element out of every class, *i.e.* that we can reach exactly q^{ov} choices of private coefficients. Restricting (10) to the ov columns which produce the coefficients $\widetilde{\gamma}_{ij}^{(k)}$ with $i \in O$, $j \in V$. Denote M_F' the coefficients of vinegar $\times$ oil terms for *all* polynomials in $\mathcal{F}$.

We now use S according to (10) and now obtain the system $BA' = M'_F$ linear in the s_{ij} for $1 \leq i \leq v$, $v+1 \leq j \leq n$, an $(D \times ov)$-matrix A', a fixed $(o \times D)$-matrix B and an $(ov \times o)$-matrix M'_F. The question is, whether the map BA' is injective, *i.e.* whether all possible choices for the private coefficients $\widetilde{\gamma}_{ij}^{(k)}$ can occur. Without loss of generality we restrict our argumentation to the first column. There are exactly v non-zero entries (that are the variables s_{ij} for $1 \leq i \leq v$ and $j = v+1$) in the first column of matrix A'. The restricted map is injective, *i.e.* we can produce all possible choices of coefficients $\widetilde{\gamma}_{ij}^{(k)}$ in the first column of M'_F, iff the corresponding v columns of B have full rank. This is the constraint we fulfill by construction of B in section 5.2. Using more refined methods and in particular a heavier machinery, it should be possible to formally prove the above claim. Because of the above claim the following scheme is as secure as the standard UOV scheme. In comparison to the case of Algorithm 2 the $(o \times D)$ matrix B is a fixed part of the algorithm. In the remainder of this paper, we refer to this scheme as Compressed UOV.

Algorithm 2 Key Generation for Compressed UOV

```
1: Choose randomly a linear map $S$ (represented by an $n \times n$-matrix $S$). If $S$ is not invertible, choose again.
2: Compute for $S$ the corresponding transformation matrix $A_{\mathrm{UOV}}$ (using equations (6) and (7)). If $A_{\mathrm{UOV}}$ is not
   invertible, go back to step 1.
3: Solve the linear system given by equation (8) to get the matrix $Q$ and therewith the quadratic coefficients of
   the central polynomials.
4: Compute the public key as $\mathcal{P} = \mathcal{F} \circ S$.
```

5 The New Construction

We are now investigating, how much additional structure we can hide in B in order to speed up the verification process. We choose elements of the matrix B uniformly at random from the ground field $\mathbb{F}_2$. In order to make sure that message recovery attacks are still difficult, we have to choose the ordering of monomials appropriately, as explained in sect. 5.3.

5.1 Message Recovery Attacks

Let $M_P = (B|C)$ with B a $(o \times D)$ matrix. After we claimed that choosing B fixed does not leak to an attacker information about the private key, we have to clarify how structured B can be chosen without decreasing security against message recovery attacks. Obviously $B = 0$ is a bad trail, as this would imply $C = 0$. We also have to assure that B has full rank, as otherwise C would also not have full rank. In general our goal is that solving a system with the coefficient matrix M_P over $\mathbb{F}_q$ using Gröbner bases should be as difficult as solving a random instance over $\mathbb{F}_q$.
We now first introduce our choice of B. Afterwards we explain, why message recovery remains hard and also why this is the smartest way to choose B, *i.e.* this is the shortest public key one can hope for.

5.2 Choice of B

The first $(o \times o)$ block in B can be chosen to be the identity matrix I as every attacker is able to reach this situation by Gaussian Elimination. Furthermore this ensures B to have full rank. The remaining part B_1 of B has to be chosen in such a way that there are no systematic dependencies, *i.e.* every m columns with $m \geq o$ have a big chance to have full rank. Otherwise we could produce large zero-blocks which would decrease the complexity of Gröbner Bases algorithms.

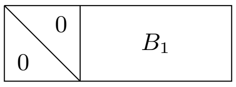

Fig. 3. Structure of matrix B.

We suggest to choose every element of the matrix B_1 uniformly at random from $\mathbb{F}_2$. Note that for such B_1 the rank property above is fulfilled with overwhelming probability. Note that B is no longer part of the public key. Once B is constructed, it is fixed, and thus is a part of the key generation algorithm.

5.3 Ordering of monomials

As opposed to the method described in [PBB10, PBB11], we need to choose a special monomial ordering for our construction. In order to understand why and how this monomial ordering is constructed, let us recall how direct (Gröbner) attacks on multivariate signature schemes work. In the message recovery attack, the attacker is facing the problem of solving the public UOV system $\mathcal{P}(\mathbf{x}) = \mathbf{h}$ directly. This system is defined over $\mathbb{F}_q$ and has o equations and $n = o + v$ variables. Such a system has on average $q^{n-o} = q^v$ solutions. Consider the values of v usually used (*e.g.* $v = 52$), such a system has a huge amount of solutions (for $q = 2^8$ and $v = 52$ it is 2^{416}). Gröbner bases methods have a great difficulty in solving such a system in this case, since they have to describe a huge variety. Since an attacker is interested in only *one* solution for the signature forgery, recovering all solutions is unnecessary. By fixing values of any v variables in the public system, an attacker obtains a quadratic system in o variables and o equations. On average such a system has a unique solution. Solving the new system with Gröbner bases is much easier.
Going back to the matrix M_P we see that C contains coefficients of monomials x_ix_j with $i, j \in \{x_{v+1}, \ldots, x_n\}$, since the ordering we chose in sect. 3 is the graded lexicographic ordering. Now if the attacker fixes values for variables $x_{v+1}, \ldots, x_n$, the monomials represented by C will become constants. Therewith, the resulting quadratic system will have only quadratic terms over $\mathbb{F}_2$ coming from the matrix B_1. Clearly, Gröbner bases computations will be much easier then, since the attacker does not have to deal with $\mathbb{F}_{2^8}$ arithmetics that much. Thus we have to ensure that an attacker is not able to remove many monomials with coefficients in $\mathbb{F}_{2^8}$ by assigning v variables to some values.
Note that we do not consider monomials of the form x_i^2. If such monomials remain after fixing v variables they do not force us to calculate in $\mathbb{F}_{2^8}$ as they are linear due to the Frobenius homomorphism. Note that for UOV over fields with odd characteristic, it makes sense to consider such monomials.

Denote by $\overline{C}$ the set of monomials whose coefficients are contained in the matrix C. We can represent this set as a graph $G(V, E)$ with $V := \{x_1, \ldots, x_n\}$ being the vertices and $E := \{e(x_i, x_j) \,|\, x_ix_j \in \overline{C}\}$ being the edges. By construction we have $|E| = \frac{o(o+1)}{2}$. In the following our goal is to construct the graph G in the way, such that the induced monomial ordering precludes an attacker from removing too many $\mathbb{F}_{2^8}$-terms independent on the choice of variables

to be fixed. Note also that by "monomial ordering" we do not mean a monomial well-ordering as in the theory of Gröbner bases, but just some ordering of monomials w.r.t. which the columns of the coefficient matrix M_P are ordered.
For the following we need two definitions.

Definition 1. *Let $G(V, E)$ be a graph. A subset $V' \subseteq V$ is called a k-*independent set*, if $|V'| = k$ and $\{e(v_i, v_j) : v_i, v_j \in V'\} \cap E = \emptyset$.*

Definition 2. *For a graph $G(V, E)$ the set $V' \subseteq V$ is called a k-*clique*, if $|V'| = k$ and all the vertices $v_i \in V'$ are pairwise connected,* i.e. *$\{e(v_i, v_j) : v_i, v_j \in V'\} \subseteq E$.*

We observe the following. If G contains an o-independent set, an attacker is able to fix v variables in such a way that all the monomials in $\overline{C}$ are removed.
So our task is to choose the edges of G in such a way, that G does not contain a k-independent set (for minimal k). For fixed k, the problem of finding a graph without k-independent set and minimal number of edges is solved by the complementary Turán graph. Details about the definition and basic properties of the Turán graph can be found in app. A.
So we start with $k = 1$ and construct the complementary Turán graph $\mathrm{CT}(n, 1)$. We then increase k until the number of edges in $\mathrm{CT}(n, k)$ will be less or equal to $\frac{o\cdot(o+1)}{2}$. If the number of edges in $\mathrm{CT}(n, k)$ is less than $\frac{o\cdot(o+1)}{2}$, we add arbitrarily edges until we reach the number of monomials in $\overline{C}$. By doing so we get a graph G with $\mathrm{CT}(n, k) \leq G$.

Example 1. In our case $(o = 26, v = 52)$ we find a solution for $k = 8$ and thus it is assured that at least 30 monomials over $\mathbb{F}_{2^8}$ remain after fixing v variables. The best attack on this parameter set is called HybridF$_5$ [BFP09] and uses fixing v and then guessing two variables before applying Faugères F_5 algorithm [Fau02] to compute Gröbner Bases. But even if we fix/guess $v+2$ variables, there will remain at least 24 monomials over $\mathbb{F}_{2^8}$. So an attacker can not hope to transfer the system into a smaller field. More details to these experiments can be found in app. B.

Once we have constructed our graph G as above, it defines which monomials should be in $\overline{C}$. Therefore, we can now define an induced ordering on quadratic monomials, such that monomials from $\overline{C}$ are bigger than those that are not from this set. For the monomials not being in $\overline{C}$ we define real squares (*i.e.* x_i^2 for $i = 1, \ldots, n$) to be smaller than other monomials. Once we defined an ordering of monomials, it is fixed and is a system parameter.
Let us investigate the effect of the new ordering on the construction of matrix A_{UOV}. In sect. 3 the columns of the matrix A_{UOV} corresponded to the first D monomials w.r.t. graded lexicographical ordering. Now we have to choose the columns of $\widetilde{A_{UOV}}$ in such a way that they correspond to the first D monomials in the monomial ordering defined above. With respect to the graph G, if the i-th edge of complementary graph $\overline{G}$ (which is actually a subgraph of the Turán graph $\mathrm{T}(n, k)$) connects the vertices v_{i_1} and v_{i_2}, we have

$$\widetilde{A_{UOV}} = \begin{pmatrix} \alpha_{11}^{11} & \alpha_{22}^{12} & \ldots & \alpha_{nn}^{11} & \widetilde{\alpha}_1^{11} & \widetilde{\alpha}_2^{11} & \ldots & \widetilde{\alpha}_{D-n}^{11} \\ \alpha_{11}^{12} & \alpha_{22}^{12} & \ldots & \alpha_{nn}^{11} & \widetilde{\alpha}_1^{12} & \widetilde{\alpha}_2^{12} & \ldots & \widetilde{\alpha}_{D-n}^{12} \\ \vdots & & & & & & & \vdots \\ \alpha_{11}^{vn} & \alpha_{22}^{vn} & \ldots & \alpha_{nn}^{vn} & \widetilde{\alpha}_1^{vn} & \widetilde{\alpha}_2^{vn} & \ldots & \widetilde{\alpha}_{D-n}^{vn} \end{pmatrix}. \tag{12}$$

Here, the coefficients α_{ii}^{rs} are given by (6) and the $\widetilde{\alpha}_i^{rs}$ are given by

$$\widetilde{\alpha}_i^{rs} = \begin{cases} s_{rv_{i1}} \cdot s_{sv_{i2}} & (v_{i1} = v_{i2}) \\ s_{rv_{i1}} \cdot s_{sv_{i2}} + s_{rv_{i2}} \cdot s_{sv_{i1}} & \text{otherwise} \end{cases}. \tag{13}$$

With this notation we have (as in sect. 3)

$$B = Q \cdot \widetilde{A_{UOV}}, \tag{14}$$

where Q contains the coefficients of the non-zero terms of $\mathcal{F}$ (in lexicographical order).
In Algorithm 3 the matrix B is chosen as shown in Figure 3 with a fixed matrix $B_1 \in_R \mathbb{F}_2^{o \times \mu}$ ($\mu = D - o$).

Algorithm 3 Key Generation for reduced UOV schemes

```
1: Choose randomly a linear map $S$ (represented by an $n \times n$-matrix $S$). If $S$ is not invertible, choose again.
2: Compute for $S$ the corresponding transformation matrix $\widetilde{A_{\mathrm{UOV}}}$ (using equations (6), (13) and ((12)). If $\widetilde{A_{\mathrm{UOV}}}$
   is not invertible, go back to step 1.
3: Solve the linear system given by equation (14) to get the matrix $Q$ and therewith the quadratic coefficients
   of the central polynomials.
4: Compute the public key as $\mathcal{P} = \mathcal{F} \circ S$.
```

5.4 Efficiency of the Verification Process

During the verification process one has to evaluate for each public polynomial the equations

$$P_i(\mathbf{z}) = (z_1, \ldots, z_n) \cdot P_i \cdot (z_1, \ldots, z_n)^T, 1 \leq i \leq o,$$

with $\mathbf{z} = (z_1, \ldots, z_n)$ being the signature of the message and P_i being the (upper triangular) matrix representing the i-th public polynomial.
To evaluate this equation for a randomly chosen P_i one needs $\frac{n \cdot (n+1)}{2} + n$ $\mathbb{F}_{256}$-multiplications for each of the o polynomials, or $o \cdot \frac{n \cdot (n+3)}{2}$ $\mathbb{F}_{256}$-multiplications and the same number of additions for the whole key.
For our reduced version we can do better. When evaluating $P_i(\mathbf{z})$ we test for each element of P_i if it is an element of $\mathbb{F}_2$. If this is the case, we further test if the element is 0 or 1. If it is 0, then we do not have to do anything. If it is 1, then we have to carry out one addition. Only in the case of the element being in $\mathbb{F}_{256} \setminus \mathbb{F}_2$, we have to carry out one multiplication.
By this strategy we are able to reduce the number of $\mathbb{F}_{256}$-multiplications needed during the verification process to $o \cdot \left(\frac{o \cdot (o+1)}{2} + n\right)$. Additionally we need $o \cdot n \cdot (n+1)$ if clauses and on average $o \cdot \left(\frac{n \cdot (n+1)}{4} + \frac{o \cdot (o+1)}{2} + n\right)$ additions.

5.5 Further Security Considerations

In this section we discuss the security of schemes generated by alg. 2 under known attacks against UOV-like schemes [KS98, KPG99, WBP04, YC04b, YC04a, YCC04, BWP05, YC05, BFP09]. These include

(1) Direct attacks
(2) Rank attacks
(3) UOV-Reconciliation attack
(4) UOV attack

This seems redundant as we have shown the security against direct attacks already in the above claim. However, our aim is to verify that theory is supported by practical evidence. In the following, we therefore take a closer look at these attacks.

Direct attacks. The most straightforward way for an attacker to forge signatures is the so called "direct" attack, *i.e.* the attacker tries to solve the public system $\mathcal{P}(\mathbf{x}) = \mathbf{h}$ by a dedicated solving algorithm like *e.g.* Gröbner Basis algorithms (Buchberger, F_4, F_5) or one of the XL-family (Mutant-XL, FXL).
As before to reduce the complexity of the attack one fixes v variables before applying the algorithm. For our experiments below we first created a determined system by fixing v of the variables and then solved the resulting system (in degree reversed lexicographical ordering) by the F_4 [Fau99] algorithm. For doing so, we used MAGMA ver. 2-13 and the command `GroebnerBasis`. Table 1 shows the results.

Scheme (q,o,v)	(2^8,9,18)	(2^8,10,20)	(2^8,11,22)	(2^8,12,24)	(2^8,13,26)
Compressed UOV	7.5	54	387	3019	23667
0/1 UOV	7.4	54	385	3016	23652
UOV	7.5	54	387	3023	23672
random system	7.5	54	386	3025	23674

Table 1. Running time of direct attacks against UOV schemes (in sec)

As the table shows, the running time of direct attacks against our schemes is nearly the same as for the standard UOV scheme and for random systems. So, for $o \geq 26$ equations [BFP09] our scheme appears to be secure against direct attacks.

UOV-Reconciliation.

Definition 3. *Let $p(\mathbf{x}) = p(x_1, \ldots, x_n)$ be a quadratic multivariate polynomial and*

$$dp(\mathbf{x}, \mathbf{c}) = p(\mathbf{x} + \mathbf{c}) - p(\mathbf{x}) - p(\mathbf{c}) + p(\mathbf{0})$$

its discrete differential. We define H_p to be the symmetric matrix such that

$$dp = \mathbf{x}^T \cdot H_p \cdot \mathbf{c}$$

For the matrix H_{p_i} representing the quadratic part of the i-th public polynomial we write in short H_i. Analogous, we denote the symmetric matrix representing the homogeneous quadratic part of the i-th central polynomial by Q_i $(i = 1, \ldots, o)$.

The goal of the UOV-Reconciliation attack is to find a change of variables which brings the matrices H_i into UOV-form, which means that the lower right $o \times o$ submatrix is the zero matrix. By doing so, the attacker creates an equivalent private key and therefore is able to forge signatures for arbitrary messages.
To achieve this goal, the attacker has to solve several multivariate quadratic systems. The complexity of the attack is mainly determined by the complexity of the first step which is the solving of a quadratic system of o equations in v variables. Table 2 shows the time MAGMA needs for solving this initial system for our schemes and the standard UOV scheme.

Since, for the parameters proposed in Subsection 2.2, the UOV scheme is believed to be secure against the UOV-Reconciliation attack, we can assume the same for our schemes.

Rank attacks. In this paragraph we look at the behavior of rank attacks against the standard UOV and our scheme. To do this, we carried out experiments with 10000 instances of our scheme for different parameters $(2^8, o, v)$. We observed that, just as in the case of the standard

Scheme (q,o,v)	(2^8,9,18)	(2^8,10,20)	(2^8,11,22)	(2^8,12,24)	(2^8,13,26)
Compressed UOV	7.4	54	386	3013	23658
0/1 UOV	7.3	55	384	3018	23654
UOV	7.5	54	388	3019	23665

Table 2. Running time of the UOV-Reconciliation attack (in sec)

UOV scheme, all the matrices Q_i representing the homogeneous quadratic parts of the central equations have full rank n. This prevents the MinRank attack. Furthermore, all the variables $x_1, \ldots, x_n$ appear in every of the o central equations, which prevents HighRank attacks.
All in all, this is not very surprising as rank attacks usually only work against *private* keys of the STS structure [WBP04]. Still, to be on the safe side, we also verified that our new construction does not fall to this previously known attack.

Original UOV Attack. The goal of the original UOV attack [KPG99] is to find the pre-image of the oil subspace $\mathcal{O} = \{x \in \mathbb{F}_q^n : x_1 = \cdots = x_v = 0\}$ under the affine invertible transformation S. To achieve this, one 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 looks for invariant subspaces of this matrix. For each parameter set $(2^8, o, v)$ listed in the table we created 100 instances of our two schemes and the standard UOV. Then we attacked these instances by the UOV-attack to find out the number of trials we need to find a basis of $S^{-1}(\mathcal{O})$.

Scheme (q,o,v)	(256,5,7)	(256,8,11)	(256,12, 15)	(256,15, 18)
Compressed UOV	1728	532614	847362	1146382
0/1 UOV	1726	532682	847394	1157638
UOV	1734	531768	852738	1182621

Table 3. Average number of trials in the UOV-attack

Summary of Further Security Considerations. As the previous four subsections showed, known attacks against the UOV signature scheme do not work significantly better against our schemes, which means that they can not use the special structure of our public key. So, in this sense our scheme appears to be secure and we do not have to adapt our parameter sets.
It might also be possible that dedicated attacks against our scheme exist. Still, the security considerations we presented above suggest that such attacks would be very difficult to mount.

5.6 Parameters and Implementation

In this section, we give our choice of parameters and show how they transfer to a practical C++ implementation. More concrete and based on the security considerations of sect. 5.5, we propose for our scheme the same parameters as for the standard UOV scheme, namely field size $q = 256$, $(o, v) = (26, 52)$. Additionally, Table 4 gives one more conservative parameter set, namely $(q, o, v) = (256, 28, 56)$.
We implemented the verification process of our scheme and the standard UOV in C++. Our straightforward implementation shows that we get quite a significant reduction in terms of the time of the verification process. We carried out 1,000,000 verification processes for each parameter set on an Intel Dual Core 2 with 2.53 GHz and an AMD Athlon XP 2400+ with 2.00 GHz. Table 5 shows the average running time of the verification process.

6 Conclusion

In this paper, we have shown that $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic public key schemes can benefit from much smaller public key sizes (cf. Table 4) without any degeneration of security. The overall idea requires some flexibility in the private key. To our knowledge, only the two $\mathcal{MQ}$-schemes UOV and Rainbow have these. UOV was covered in this article. However, Rainbow has a more difficult internal structure, so we have to leave a concrete application of our improvement to Rainbow as an open question, which we plan to address.

Scheme(q, o, v)	public key size (kB)	private key size (kB)	hash size (bit)	signature size (bit)	reduction of public key size (%)
UOV(256,26,52)	78.2	75.3	208	624	-
0/1 UOV(256,26,52)	8.9	75.3	208	624	88.6
UOV(256,28,56)	97.6	93.4	224	672	-
0/1 UOV(256,28,56)	11.1	93.4	224	672	88.6

Table 4. Proposed parameters for UOV schemes

The security proof made use of the idea of equivalent keys. Hereby, each public key can be assigned many private keys. We have turned this idea around by considering transformations of the *public* key $\mathcal{P}$ instead and showed that an attacker does not gain from this specific structure as he can generate it himself and we will derive an equally useful private key $(\mathcal{F}, \mathcal{S})$.

	Intel Dual Core 2 2.53 GHz			AMD Athlon XP 2400+ 2.00 GHz		
(o, v)	standard UOV	0/1 UOV	reduction factor	standard UOV	0/1 UOV	reduction factor
(26,52)	0.49 ms	0.14 ms	71 %	0.68 ms	0.19 ms	72 %
(28,56)	0.54 ms	0.14 ms	72 %	0.74 ms	0.20 ms	73 %
(32,64)	0.75 ms	0.20 ms	72 %	1.03 ms	0.28 ms	73 %

Table 5. Running time of the verification process

As we can enforce a specific form on the public key $\mathcal{P}$, we can also use it to speed up public key operations, namely verification of signatures. As we see in Table 5, this reduces the overall time by about 73% or a markable factor of 3.7.
As the construction is very general, it can be used on other platforms (*e.g.* GPU, FPGA) as well. We actually expect similar gains in area reduction or speed there, too.
From a theoretical perspective, forcing a specific structure on the central polynomials $\mathcal{F}$ or the public polynomials $\mathcal{P}$ are equivalent: We can do either. Hence, for specific application domains it might be useful to find a certain *trade-off*. For example, we could reduce the computational workload on a server based on the maximal available memory on a smart card.
All in all, the usability gap between RSA and ECC on the one side and post-quantum cryptography on the other side is shrinking again. For specific applications, the latter could actually be of use in practice.

Acknowledgements

We thank Ishtiaq Shah for doing the implementation of our scheme. Furthermore we want to thank our financial supporters. The first author is supported by the Horst Görtz Foundation

(HGS) within the project where the third author is the principal investigator. The third author is supported by the DFG grant BU 630/22-1. The second author was supported by the German Science Foundation (DFG) through an Emmy Noether grant where the forth author is principal investigor. All authors were in part supported by the European Commission through the IST Programme under contract ICT-2007-216676 Ecrypt II.

Bibliography

[BBD09] Daniel J. Bernstein, Johannes Buchmann, and Erik Dahmen. *Post-Quantum Cryptography.* Springer, 2009. ISBN 978-3-540-88701-0.

[BERW08] Andrey Bogdanov, Thomas Eisenbarth, Andy Rupp, and Christopher Wolf. Time-area optimized public-key engines: -cryptosystems as replacement for elliptic curves? In Elisabeth Oswald and Pankaj Rohatgi, editors, *CHES*, volume 5154 of *Lecture Notes in Computer Science*, pages 45–61. Springer, 2008.

[BFP09] Luk Bettale, Jean-Charles Faugère, and Ludovic Perret. Hybrid approach for solving multivariate systems over finite fields. *In Journal of Mathematical Cryptology*, 3:177–197, 2009.

[BWP05] An Braeken, Christopher Wolf, and Bart Preneel. A study of the security of Unbalanced Oil and Vinegar signature schemes. In *The Cryptographer's Track at RSA Conference 2005*, volume 3376 of *Lecture Notes in Computer Science.* Alfred J. Menezes, editor, Springer, 2005. 13 pages, cf `http://eprint.iacr.org/2004/222/`.

[CCC$^+$08] Anna Inn-Tung Chen, Chia-Hsin Owen Chen, Ming-Shing Chen, Chen-Mou Cheng, and Bo-Yin Yang. Practical-sized instances of multivariate pkcs: Rainbow, tts, and lic-derivatives. In Johannes Buchmann and Jintai Ding, editors, *PQCrypto*, volume 5299 of *Lecture Notes in Computer Science*, pages 95–108. Springer, 2008.

[CCC$^+$09] Anna Inn-Tung Chen, Ming-Shing Chen, Tien-Ren Chen, Chen-Mou Cheng, Jintai Ding, Eric Li-Hsiang Kuo, Frost Yu-Shuang Lee, and Bo-Yin Yang. Sse implementation of multivariate pkcs on modern x86 cpus. In Christophe Clavier and Kris Gaj, editors, *CHES*, volume 5747 of *Lecture Notes in Computer Science*, pages 33–48. Springer, 2009.

[Fau99] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases (F_4). *Journal of Pure anNational Institute of Standards and Technologyd 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.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability — A Guide to the Theory of NP-Completeness.* W.H. Freeman and Company, 1979. ISBN 0-7167-1044-7 or 0-7167-1045-5.

[HWCL05] Yuh-Hua Hu, Lih-Chung Wang, Chun-yen Chou, and Feipei Lai. Similar keys of multivariate quadratic public key cryptosystems. In Yvo Desmedt, Huaxiong Wang, Yi Mu, and Yongqing Li, editors, *Cryptology and Network Security, 4th International Conference, CANS 2005, Xiamen, China, December 14-16, 2005, Proceedings*, volume 3810 of *Lecture Notes in Computer Science*, pages 211–222. Springer, 2005.

[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, editor, 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, editor, Springer, 1998.

[Pat97] Jacques Patarin. The oil and vinegar signature scheme. Presented at the Dagstuhl Workshop on Cryptography, September 1997. transparencies.

[PBB10] Albrecht Petzoldt, Stanislav Bulygin, and Johannes Buchmann. A multivariate signature scheme with a partially cyclic public key. In *Proceedings of SCC'10*, pages 229–235, 2010.

[PBB11] Albrecht Petzoldt, Stanislav Bulygin, and Johannes Buchmann. Linear recurring sequences for the UOV key generation. In D. Catalano, editor, *Proceedings of PKC'11*, volume 6571 of *LNCS*, pages 335–350. Springer, 2011.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, October 1997.

[WBP04] Christopher Wolf, An Braeken, and Bart Preneel. Efficient cryptanalysis of RSE(2)PKC and RSSE(2)PKC. In *Conference on Security in Communication Networks — SCN 2004*, volume 3352 of *Lecture Notes in Computer Science*, pages 294–309. Springer, September 8–10 2004. Extended version: `http://eprint.iacr.org/2004/237`.

[Wol05] Christopher Wolf. *$\mathcal{M}$ultivariate $\mathcal{Q}$uadratic Polynomials in Public Key Cryptography.* Ph.D. thesis, Katholieke Universiteit Leuven, Belgium, November 2005. `http://hdl.handle.net/1979/148`, 156+xxiv pages.

[WP05a] 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, editor, Springer, 2005. Extended version `http://eprint.iacr.org/2004/361/`.

[WP05b] Christopher Wolf and Bart Preneel. Taxonomy of public key schemes based on the problem of multivariate quadratic equations. Cryptology ePrint Archive, Report 2005/077, 12$^{\text{th}}$ of May 2005. `http://eprint.iacr.org/2005/077/`, 64 pages.

[WP11] Christopher Wolf and Bart Preneel. Equivalent keys in multivariate quadratic public key systems. *Journal of Mathematical Cryptology*, 2011. 45 pages, *to appear*.

[YC04a] Bo-Yin Yang and Jiun-Ming Chen. All in the XL family: Theory and practice. In *ICISC 2004*, pages 67–86. Springer, 2004.

[YC04b] Bo-Yin Yang and Jiun-Ming Chen. Theoretical analysis of XL over small fields. In *ACISP 2004*, volume 3108 of *LNCS*, pages 277–288. Springer, 2004.

[YC05] Bo-Yin Yang and Jiun-Ming Chen. Building secure tame-like multivariate public-key cryptosystems: The new TTS. In *ACISP 2005*, volume 3574 of *LNCS*, pages 518–531. Springer, July 2005.

[YCC04] Bo-Yin Yang, Jiun-Ming Chen, and Yen-Hung Chen. TTS: High-speed signatures on a low-cost smart card. In *CHES 2004*, volume 3156 of *LNCS*, pages 371–385. Springer, 2004.

[YCCC06] Bo-Yin Yang, Doug Chen-Mou Cheng, Bor-Rong Chen, and Jiun-Ming Chen. Implementing minimized multivariate public-key cryptosystems on low-resource embedded systems. In *SPC 2006*, volume 3934 of *LNCS*, pages 73–88. Springer, 2006.

A The Turán graph

The Turán graph $\mathrm{T}(n,k)$ (named after the Hungarian mathematician Pal Turán) is defined as follows: The set V of n vertices is partitioned into k subsets $A_1,\ldots,A_k$, whose sizes are as equal as possible, *i.e.* $\bigcup_{i=1}^{k} A_i = V,\ A_i \cap A_j = \emptyset,\ ||A_i| - |A_j|| \leq 1$ for $i \neq j$.
Two vertices are connected by an edge whenever they belong to different subsets, *i.e.* $e(v_i,v_j) \in E \Leftrightarrow v_i \in A_r,\ v_j \in A_s$ with $r \neq s$.
The number of edges in $\mathrm{T}(n,k)$ is given by $\frac{1}{2} \cdot \sum_{i=1}^{k} |A_i| \cdot (n - |A_i|)$ and is upper bounded by $\left(1 - \frac{1}{k}\right) \cdot \frac{n^2}{2}$.
Since every set of $(k+1)$ vertices contains at least two vertices in the same subset A_i, the Turán graph does not contain a $(k+1)$ *clique*. According to Turán's theorem it is the graph with the maximum possible number of edges with this property.
The complementary graph of the Turán graph $\mathrm{T}(n,k)$ is denoted by $\mathrm{CT}(n,k)$. Here, two vertices are connected by an edge if they belong to the same subset, *i.e.* $e(v_i,v_j) \in E \Leftrightarrow \exists r$ s.t. $v_i, v_j \in A_r$.
The number of edges is given by $|E| = \sum_{i=1}^{k} \binom{|A_i|}{2}$, which is bounded from below by $\frac{n}{2} \cdot \left(\frac{n}{k} - 1\right)$.
Since every set of $(k+1)$ vertices contains at least two vertices from the same subset A_i, the graph $\mathrm{CT}(n,k)$ does not contain a $(k+1)$ *independent set*. From Turán's theorem it follows that $\mathrm{CT}(n,k)$ is the graph with the minimal number of edges with this property. We have used this property in sect. 5.3 to find an optimal monomial ordering.
Figure 4 shows for $n = 8$ and $k = 3$ the Turán and the complementary Turán graph.

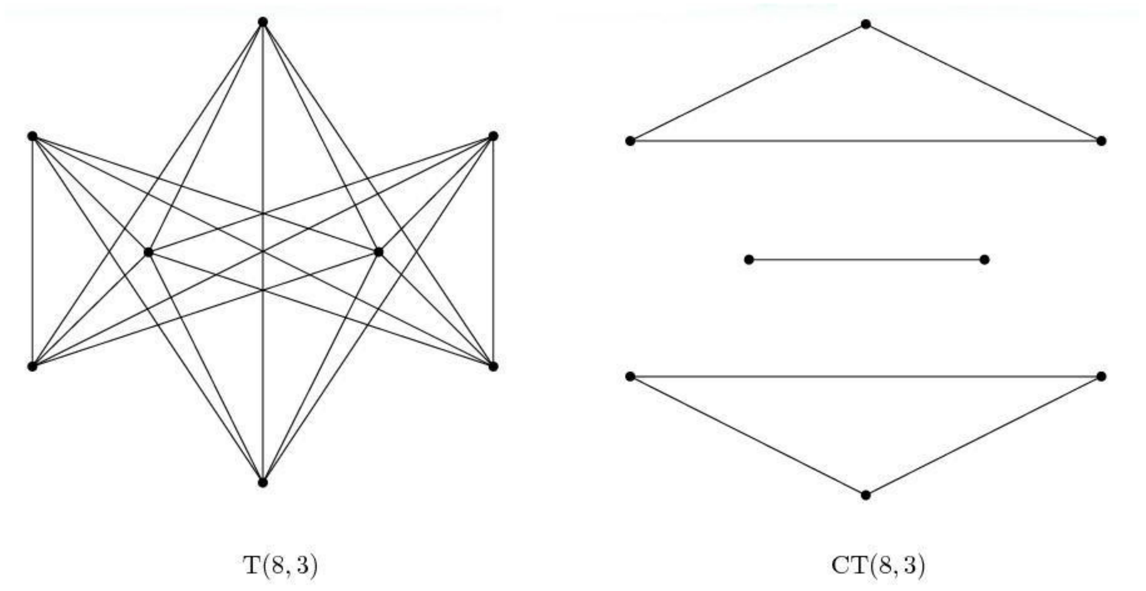

Fig. 4. Turán graph $\mathrm{T}(8,3)$ and complementery graph $\mathrm{CT}(8,3)$

B Details of the Experiments from Example 1

The parameters we are working with are
$(o, v) = (26, 52)$
$n = o + v = 78$
$D = \frac{v\cdot(v+1)}{2} + o \cdot v = 2730$
$D_2 = \frac{o\cdot(o+1)}{2} = 351$

Finding the optimal k
In this step we want to find the minimal $k \in \mathbb{N}$ such that the number of edges in the complementary Turá graph $\mathrm{CT}(n, k)$ (see app. A) is less or equal to D_2.

k	1	2	3	4	5	6	7	8
# edges in $\mathrm{CT}(78, k)$	3003	1482	975	722	570	468	396	342
	$> D_2$	$> D_2$	$> D_2$	$> D_2$	$> D_2$	$> D_2$	$> D_2$	$\leq D_2$

To obtain the total number of 351 monomials in $\overline{C}$, we add 9 arbitrarily chosen edges to $\mathrm{CT}(78, 8)$. This yields graph G (see sect. 5.3).

Fixing of variables
In $\mathrm{CT}(78, 8)$, the 78 vertices are divided into 8 groups $A_1, \ldots, A_8$ as follows

i	1	2	3	4	5	6	7	8
$\lvert A_i\rvert$	10	10	10	10	10	10	9	9

When attacking the scheme directly, an attacker fixes a certain number of variables before applying a Gröbner basis method. Since the number of edges in the graph is given by

$$\sum_{i=1}^{k} \binom{|A_i|}{2},$$

it is obvious, that the best strategy for the attacker is to remove from each group A_i approximately the same number of vertices. Note that by removing r vertices using the above strategy the attacker creates the graph $\mathrm{CT}(n - r, k)$.

When fixing v out of the n variables, the attacker creates the graph $\mathrm{CT}(o, k)$. In our example, we get the graph $\mathrm{CT}(26, 8)$ with 26 vertices divided into 8 groups A'_i as follows:

i	1	2	3	4	5	6	7	8
$\lvert A'_i\rvert$	4	4	3	3	3	3	3	3

The number of edges in this graph is $\sum_{i=1}^{8} \binom{|A'_i|}{2} = 30$.

When fixing/guessing $v + 2 = 28$ variables, the attacker implicitly creates the graph $\mathrm{CT}(24, 8)$, whose vertices are divided into 8 groups A''_i as follows:

i	1	2	3	4	5	6	7	8
$\lvert A''_i\rvert$	3	3	3	3	3	3	3	3

The number of edges in this graph is $\sum_{i=1}^{8} \binom{|A''_i|}{2} = 24$.

The following table shows for some values of o the optimal number k, as well as the number of monomials in $\overline{C}$ and the minimal number of monomials remaining in $\overline{C}$ after guessing v (denoted by $|\overline{C}|_v$) and $v+2$ variables ($|\overline{C}|_{v+2}$).

o	10	18	26	28	30	32	36	40	44	48
k	7	8	8	8	8	8	9	9	9	9
$\lvert\overline{C}\rvert$	55	171	351	406	465	528	666	820	990	1080
$\lvert\overline{C}\rvert_v$	3	12	30	36	42	48	54	70	86	105
$\lvert\overline{C}\rvert_{v+2}$	1	8	24	30	36	42	48	62	78	95