Revisiting the Cubic UOV Signature Scheme

Dung H. Duong[1,2], Albrecht Petzoldt[1], Yacheng Wang[3], and Tsuyoshi Takagi[1,2]

[1] Institute of Mathematics for Industry, Kyushu University,
744 Motooka, Nishi-ku, Fukuoka 819-0395, Japan
{duong,petzoldt,takagi}@imi.kyushu-u.ac.jp
[2] JST, CREST, 4-1-8 Honcho, Kawaguchi, Saitama, 332-0012, Japan
[3] Graduate School of Mathematics, Kyushu University, Fukuoka, Japan
ma216004@math.kyushu-u.ac.jp

Abstract. As recently been emphasized by NSA and NIST, there is an increasing need for cryptographic schemes being secure against quantum computer attacks. Especially in the area of digital signature schemes, multivariate cryptography is one of the main candidates for this. At Inscrypt 2015, Nie et al. proposed a new multivariate signature scheme called CUOV [22], whose public key consists both of quadratic and cubic polynomials. However, the scheme was broken by an attack of Hashimoto [16]. In this paper we take a closer look on the CUOV scheme and its attack and propose two new multivariate signature schemes called CSSv and SVSv, which are secure against Hashimoto's attack and all other known attacks on multivariate schemes. Especially our second construction SVSv is very efficient and outperforms current multivariate signature schemes such as UOV and Rainbow in terms of key and signature size.

Keywords: Post-Quantum Cryptography, Multivariate Cryptography, Signature Schemes

1 Introduction

The currently most widely used public key cryptosystems are the number theory based schemes RSA [30], DSA [19] and ECC [18]. However, these schemes will become insecure as soon as large enough quantum computers arrive [31]. Therefore, one needs alternatives to those classical public key schemes, based on hard mathematical problems not affected by quantum computer attacks (so called post quantum cryptosystems). The increasing importance of research in this field has recently been emphasized by a number of authorities, including the American National Security Agency (NSA), who recommended governmental organizations to switch their security infrastructures from schemes such as RSA and ECC to post-quantum cryptosystems [14], and the National Institute of Standards and Technology (NIST), which is preparing to develop standards for these schemes [23].
According to [23], multivariate cryptography is one of the main candidates for this standardization. Multivariate schemes are in general very fast and require

only modest computational resources, which makes them attractive for the use on low cost devices like smart cards and RFID chips [3,5]. Since the late 1980's, many multivariate schemes both for encryption and signatures were proposed. One of the first was the Matsumoto-Imai cryptosystem [20], which was later extended to schemes such as Sflash [28] and HFE [25]. However, due to some flaws in the design (low rank of the private polynomials, low degree of regularity, ...), many of these schemes have been broken by direct, rank and differential attacks [24,12]. Another research direction led to the development of SingleField signature schemes such as UOV [17] and Rainbow [8]. These two schemes have withstood (for suitable parameters) cryptanalysis for nearly 20 years now and therefore are considered to provide high security. While the signature generation of UOV is very efficient, it has a very large public key. To deal with this, Ding and Schmidt [8] proposed the Rainbow signature scheme, which can be seen as a multi-layer version of UOV with smaller keys and shorter signatures. However, the multi-layer structure of Rainbow enables a number of new attacks [2,9] which makes the parameter choice of Rainbow to be a challenging task. Furthermore, this shows that one has to be very careful when designing new multivariate schemes on the basis of UOV and Rainbow.

At Inscrypt 2015, Nie et al. proposed a new idea of using cubic polynomials in the public key in a way that the key sizes are not too large and the signing process is efficient (CUOV) [22]. The use of cubic polynomials in the public key increases the degree of regularity of the system and hence increases the security against direct attacks. In addition, several attacks such as differential attacks are also not applicable against the scheme. Furthermore, the CUOV scheme has shorter signatures and a smaller private key than UOV and Rainbow. However, the scheme was broken by a newly developed attack of Hashimoto [16].

In this paper we revisit the CUOV scheme of Nie et al. [22] and analyze why it can be broken by Hashimoto's attack. Furthermore, we identify a number of components not relevant for the security of the scheme. By omitting these unnecessary components, we propose our first improved multivariate signature scheme, called CSSv (see Section 3). By our modifications, in addition to avoiding Hashimoto's attack, we make the signature generation much more stringent and reduce the number of cubic polynomials in the public key from 3 to 1, thus reducing the public key size by up to 40 %. We show that the resulting scheme resists not only Hashimoto's attack, but also all other known attacks on multivariate cryptosystems, including direct and rank attacks (Section 3.2). Based on our construction of CSSv, we then propose a second new multivariate signature scheme called SVSv (Section 4). While, as in the case of CUOV, the public key of CSSv consists of both cubic and quadratic polynomials, the public key of SVSv is completely quadratic, which decreases the key sizes further without weakening the security of the construction (Section 4.2). The scheme provides shorter signatures than Rainbow and reduces both public and private key size significantly (by 24% and 79% respectively compared to Rainbow).

2 The Cubic Unbalanced Oil and Vinegar Signature Scheme (CUOV)

In this section we recall the CUOV scheme of [22]. Before we come to the description of the scheme itself, we start with a short overview of the basic concepts of multivariate cryptography.

2.1 Multivariate Cryptography

The basic objects of multivariate cryptography are systems of multivariate quadratic polynomials over a finite field K. The security of multivariate schemes is based on the *MQ-Problem* which asks for a solution of a given system of multivariate quadratic polynomials over the field K. The MQ-Problem is proven to be NP-hard even for quadratic polynomials over the field GF(2) [13].
To build a public key cryptosystem on the basis of the MQ-Problem, one starts with an easily invertible quadratic map $\mathcal{F} : K^n \to K^m$ (*central map*). To hide the structure of $\mathcal{F}$ in the public key, one composes it with two invertible affine (or linear) maps $\mathcal{T} : K^m \to K^m$ and $\mathcal{S} : K^n \to K^n$. The *public key* is therefore given by $\mathcal{P} = \mathcal{T} \circ \mathcal{F} \circ \mathcal{S} : K^n \to K^m$. The *private key* consists of $\mathcal{T}, \mathcal{F}$ and $\mathcal{S}$.

In this paper we consider multivariate signature schemes. For these schemes, we require $n \geq m$, which ensures that every message has a signature.

Signature Generation: To generate a signature for a message (or its hash value) $\mathbf{d} \in K^m$, one computes recursively $\mathbf{w} = \mathcal{T}^{-1}(\mathbf{d}) \in K^m$, $\mathbf{y} = \mathcal{F}^{-1}(\mathbf{w}) \in K^n$ and $\mathbf{z} = \mathcal{S}^{-1}(\mathbf{y})$. $\mathbf{z} \in K^n$ is the signature of the message $\mathbf{d}$. Here, $\mathcal{F}^{-1}(\mathbf{w})$ means finding one (of possibly many) pre-image of $\mathbf{w}$ under the central map $\mathcal{F}$.

Signature Verification: To check the authenticity of a signature $\mathbf{z} \in K^n$, the verifier simply computes $\mathbf{d}' = \mathcal{P}(\mathbf{z})$. If the result is equal to the message $\mathbf{d}$, the signature is accepted, otherwise rejected.

2.2 The CUOV Scheme

In [22], Nie et al. proposed a new multivariate signature scheme called Cubic Unbalanced Oil and Vinegar (CUOV). The scheme can be described as follows. Let K be a finite field with q elements and $o, v \in \mathbb{N}$. The number of variables in the scheme is given by $n = o + v$, the number of equations is o.

Key Generation: The *central map* $\mathcal{F}$ of the CUOV scheme has the form $\mathcal{F} = \bar{\mathcal{F}} \circ (\hat{\mathcal{F}} \times \mathrm{id}_v) : K^n \to K^o$. Here, $\hat{\mathcal{F}} : K^n \to K^o$ consists of one quadratic and $o-1$ affine polynomials of the form

$$\begin{cases} \hat{f}^{(1)} = \sum_{i=1}^{o} \sum_{j=o+1}^{n} a_{ij}^{(1)} \cdot y_i y_j + \sum_{i=o+1}^{n} \sum_{j=i}^{n} a_{ij}^{(1)} \cdot y_i y_j + \sum_{i=1}^{n} b_i^{(1)} \cdot y_i + c^{(1)}, \\ \hat{f}^{(2)} = \sum_{i=1}^{n} b_i^{(2)} \cdot y_i + c^{(2)}, \\ \quad \dots \\ \hat{f}^{(o)} = \sum_{i=1}^{n} b_i^{(o)} \cdot y_i + c^{(o)}, \end{cases} \tag{1}$$

where the coefficients $a_{ij}^{(k)}, b_j^{(k)}, c^{(k)}$ are random elements of K with $i \in \{1, \dots, v\}$, $j \in \{1, \dots, n\}$ and $k \in \{1, \dots, o\}$ and

$$\begin{aligned} \hat{\mathcal{F}} \times \mathrm{id}_v : K^n &\to K^n \\ (y_1, \dots, y_o, y_{o+1}, \dots, y_n) &\mapsto (\hat{f}^{(1)}, \dots, \hat{f}^{(o)}, \underbrace{y_{o+1}, \dots, y_n}_{\text{vinegar variables}}). \end{aligned}$$

Note that $\hat{f}^{(1)}$ has the form of an oil and vinegar polynomial with o oil and v vinegar variables (cf. [17]).

The map $\bar{\mathcal{F}}$ is a map from $K^o \times K^v$ to K^o, $(x_1, \dots, x_o, y_{o+1}, \dots, y_n) \mapsto (\bar{f}^{(1)}, \dots, \bar{f}^{(o)})$ of the form

$$\begin{cases} \bar{f}^{(1)} = r_1 \cdot (x_1 + x_1 \cdot x_2) + g_1(y_{o+1}, \dots, y_n), \\ \bar{f}^{(2)} = r_2 \cdot x_1 \cdot x_2 + g_2(y_{o+1}, \dots, y_n), \\ \bar{f}^{(3)} = r_3 \cdot (x_1 + x_2) \cdot x_3 + g_3(y_{o+1}, \dots, y_n), \\ \quad \dots \\ \bar{f}^{(o)} = r_o \cdot (x_{o-2} + x_{o-1}) \cdot x_o + g_o(y_{o+1}, \dots, y_n). \end{cases} \tag{2}$$

Here $r_1, \dots, r_o$ are random elements in $K \setminus \{0\}$, g_1, g_2, g_3 are random cubic polynomials in the v vinegar variables $y_{o+1}, \dots, y_n$, whereas $g_4, \dots, g_o$ are random quadratic maps.
Due to the structure of $\hat{\mathcal{F}}$ and $\bar{\mathcal{F}}$, the *central map* $\mathcal{F} = (f^{(1)}, \dots, f^{(o)})$ of the CUOV scheme consists of three cubic polynomials $f^{(1)}, f^{(2)}, f^{(3)}$ and $(o-3)$ quadratic polynomials $f^{(4)}, \dots, f^{(o)}$.

To hide the structure of $\mathcal{F}$ in the public key, we choose randomly an invertible affine map $\mathcal{S} : K^n \to K^n$. The *public key* is given by $\mathcal{P} = \mathcal{F} \circ \mathcal{S} : K^n \to K^o$ and consists of three cubic polynomials $p^{(1)}, p^{(2)}, p^{(3)}$ and $(o-3)$ quadratic polynomials $p^{(4)}, \dots, p^{(o)}$. The *private key* consists of the polynomials $\hat{f}^{(1)}, \dots, \hat{f}^{(o)}$ and $g_1, \dots, g_o$, the invertible affine map $\mathcal{S}$ and the field elements $r_1, \dots, r_o$. [4]
The key generation process is illustrated in Algorithm 1.

[4] In contrast to the standard construction of multivariate cryptography (see above), Nie et al. did not use a second affine map $\mathcal{T}$. The reason for this is that $\mathcal{T}$ would turn the public key into a completely cubic map and therefore increase the key size drastically.

Algorithm 1 Key Generation of CUOV [22]

Input: Finite field K with q elements and integers o, v
Output: CUOV key pair $((\mathcal{F}, \mathcal{S}), \mathcal{P})$

1: Choose randomly o polynomials $\hat{f}^{(i)}$ in $n = o + v$ variables as shown in (1).
2: Choose 3 random cubic polynomials g_1, g_2, g_3 in v variables.
3: Choose $o - 3$ random quadratic polynomials $g_4, \ldots, g_o$ in v variables.
4: Choose random elements $r_1, \ldots, r_o \in K \setminus \{0\}$.
5: Define $\bar{f}^{(1)}(x_1, \ldots, x_o, y_{o+1}, \ldots, y_n), \ldots, \bar{f}^{(o)}(x_1, \ldots, x_o, y_{v+1}, \cdots, y_n)$ as shown in (2)
6: The central map is $\mathcal{F} = (f^{(1)}, \ldots, f^{(o)}) : K^n \to K^o$. where for each $i = 1, \ldots, o$ we have $f^{(i)} = \bar{f}^{(i)}(\hat{f}^{(1)}, \ldots, \hat{f}^{(o)}, y_{v+1}, \ldots, y_n)$
7: Choose randomly an invertible affine map $\mathcal{S} : K^n \to K^n$.
8: $\mathcal{P} = \mathcal{F} \circ \mathcal{S} : K^n \to K^o$
9: **return** $((\mathcal{F}, \mathcal{S}), \mathcal{P})$

Signature Generation: To generate a signature $\mathbf{z} \in K^n$ for a message (hash value) $\mathbf{d} = (d_1, \ldots, d_o) \in K^o$, the signer performs the following steps.

(1) Choose random values for the vinegar variables $y_{o+1}, \ldots, y_n$ and substitute them into the polynomials $\hat{f}^{(1)}, \ldots, \hat{f}^{(o)}$ and $g_1, \ldots, g_o$.
(2) Compute x_1 by $x_1 = \frac{1}{r_1} \cdot (d_1 - g_1) - \frac{1}{r_2} \cdot (d_2 - g_2)$, $x_2 = \frac{1}{r_2 \cdot x_1} \cdot (d_2 - g_2)$ and recursively $x_i = \frac{1}{r_i \cdot (x_{i-2} + x_{i-1})} \cdot (d_i - g_i)$ $(i = 3, \ldots, o)$. If any of the denominators in these equations happens to be zero, choose other values for the vinegar variables $y_{o+1}, \ldots, y_n$.
(3) Solve the linear system given by the last $o - 1$ equations of (1) to obtain for $y_2, \ldots, y_o$ univariate linear representations in y_1. If this fails, choose other values for the vinegar variables $y_{o+1}, \ldots, y_n$.
(4) Substitute the linear relations found in the previous step into $\hat{f}^{(1)}$ and solve the resulting linear equation for y_1.
(5) Compute a signature $\mathbf{z} \in K^n$ for $\mathbf{d}$ by $\mathbf{z} = \mathcal{S}^{-1}(y_1, \ldots, y_n)$.

Signature Verification: To check the authenticity of a signature $\mathbf{z} \in K^n$, the verifier simply computes $\mathbf{d}' = \mathcal{P}(\mathbf{z}) \in K^o$. If the result is equal to the message $\mathbf{d}$, the signature is accepted, otherwise rejected.

2.3 The attack of Hashimoto [16]

In the case of the CUOV scheme we have

$$\bar{f}^{(1)} - \frac{r_1}{r_2} \cdot \bar{f}^{(2)} = r_1 \cdot x_1 + \underbrace{\left(g_1(y_{o+1}, \ldots, y_n) - \frac{r_1}{r_2} \cdot g_2(y_{o+1}, \ldots, y_n)\right)}_{\text{cubic map in } y_{o+1}, \ldots, y_n}. \tag{3}$$

By denoting

$$D_c p^{(i)}(\mathbf{z}) = p^{(i)}(\mathbf{z} + \mathbf{c}) - p^{(i)}(\mathbf{z}) \tag{4}$$

for $p^{(i)}$ being the i-th component of the CUOV public key, some fixed vector $\mathbf{c} \in K^n$ and Q_i being the coefficient matrix of the corresponding quadratic form ($i = 1, 2$), Hashimoto showed that, due to equation (3), there exists an (easy to find) linear combination $Q_1 + \beta \cdot Q_2$ of rank at most v. By using this fact, Hashimoto could identify (the linear representations of) the vinegar variables $y_{o+1}, \ldots, y_n$, compute an equivalent central map and therefore forge signatures.

3 Our first improved scheme

In this section we take a closer look at the CUOV signature scheme and Hashimoto's attack. We analyze which properties make the scheme insecure and develop a strategy to avoid these weaknesses. Furthermore, we identify some components of CUOV which are not relevant for the security of the scheme. By removing them from the scheme, we can make the signature generation process much more stringent and reduce the public key size of the scheme. We denote our improved scheme by CSSv (Cubic Signature Scheme with Vinegar).

By studying Hashimoto's attack closely, we find that it works mainly due to the fact that, in the case of CUOV, we have a linear combination of the central polynomials $\bar{f}^{(i)}$ which is the sum of a quadratic form $\mathcal{X}$ in $y_1, \ldots, y_n$ and a cubic polynomial $\mathcal{G}$ in $y_{o+1}, \ldots, y_n$ (c.f. equation (3)). By taking the differential (equation (4)), the quadratic terms of $\mathcal{X}$ vanish, and there remain only quadratic terms in the variables $y_{o+1}, \ldots, y_n$. For the attacker this means that Hashimoto's attack works if and only if there exists an (easy to find) relation of the public polynomials of the form

$$\mathcal{Y} = \sum_{i=1}^{o} a_i \cdot p^{(i)} = \mathcal{X} + \mathcal{G},$$

with $\mathcal{X}$ being a quadratic map of rank n and $\mathcal{G}$ being a cubic map of rank v.
To prevent Hashimoto's attack, we therefore have to design our scheme in a way that such a relation does not exist. In the CSSv scheme, this is achieved by reducing the number of cubic polynomials from 3 to 1 and introducing an additional affine map $\mathcal{T}$ (see Section 3.2).
Furthermore, we identified the following components of CUOV not relevant for the security of the scheme. By omitting them, we can make the signature generation process much more straightforward and reduce the key sizes significantly.

1. The use of the coefficients r_i in equation (2) is unnecessary, since these factors can easily be included into the maps $\hat{f}^{(1)}, \ldots, \hat{f}^{(o)}$.
2. Instead of using an oil and vinegar polynomial for $\hat{f}^{(1)}$, we can easily switch to a random quadratic one. In this case we have to solve in step (4) of the signing process a univariate quadratic polynomial.
3. Taking the sum $x_{i-2} + x_{i-1}$ in equations $3, \ldots, o$ of (2) does not bring extra security into the scheme since the result is still a linear combination of $y_1, \ldots, y_n$.

4. The summation $(x_1 + x_1 \cdot x_2)$ in the first component of (2) is unnecessary, too, since $\hat{f}^{(1)}$ was chosen as a random polynomial.

3.1 The CSSv Signature Scheme

In this subsection we propose our first improved scheme CSSv, which is obtained by applying our strategy to prevent Hashimoto's attack and removing the above identified unnecessary components from the CUOV scheme of Nie et al. [22]. Our scheme can be described as follows.

Key Generation: Let K be a finite field with q elements and $o, v \in \mathbb{N}$. We set $n = o + v$. As in the case of the CUOV scheme (see previous section), the *central map* $\mathcal{F}$ of the CSSv scheme has the form $\mathcal{F} = \bar{\mathcal{F}} \circ (\hat{\mathcal{F}} \times \mathrm{id}_v) : K^n \to K^o$, with id_v being the identity map in K^v. The map $\hat{\mathcal{F}} = (\hat{f}^{(1)}, \ldots, \hat{f}^{(o)})$ has the form

$$\begin{cases} \hat{f}^{(1)} = \sum_{i=1}^{n} \sum_{j=i}^{n} a_{ij}^{(1)} \cdot y_i y_j + \sum_{i=1}^{n} b_i^{(1)} \cdot y_i + c^{(1)} \\ \hat{f}^{(2)} = \sum_{i=1}^{n} b_i^{(2)} \cdot y_i + c^{(2)} \\ \quad \cdots \\ \hat{f}^{(o)} = \sum_{i=1}^{n} b_i^{(o)} \cdot y_i + c^{(o)} \end{cases} \tag{5}$$

with a random quadratic polynomial $\hat{f}^{(1)}$ and affine maps $\hat{f}^{(2)}, \ldots, \hat{f}^{(o)}$ in the variables $y_1, \ldots, y_n$.
The map $\bar{\mathcal{F}} : K^o \times K^v \to K^o, (x_1, \ldots, x_o, y_{o+1}, \ldots, y_n) \mapsto (\bar{f}^{(1)}, \ldots, \bar{f}^{(o)})$ is given by

$$\begin{cases} \bar{f}^{(1)} = x_1 + g_1(y_{o+1}, \ldots, y_n) \\ \bar{f}^{(2)} = x_1 \cdot x_2 + g_2(y_{o+1}, \ldots, y_n) \\ \quad \cdots \\ \bar{f}^{(o)} = x_{o-1} \cdot x_o + g_o(y_{o+1}, \ldots, y_n). \end{cases} \tag{6}$$

Here we choose randomly a cubic polynomial g_2 and $(o-1)$ quadratic polynomials $g_1, g_3, \ldots, g_o$ in the v variables $y_{o+1}, \ldots, y_n$.

The *central map* $\mathcal{F} = (f^{(1)}, \ldots, f^{(o)})$ therefore consists of one cubic polynomial $f^{(2)}$ and $(o-1)$ quadratic polynomials $f^{(1)}, f^{(3)}, \ldots, f^{(o)}$ in the variables $y_1, \ldots, y_n$. In order to hide the structure of $\mathcal{F}$ in the public key, we choose two invertible affine maps $\mathcal{S} : K^n \to K^n$ and $\mathcal{T} : K^o \to K^o$. While the map $\mathcal{S}$ is chosen completely at random, the matrix T representing the map $\mathcal{T}$ has the form

$$T = \begin{pmatrix} \star_{1\times 1} & \star_{1\times 1} & \star_{1\times(o-2)} \\ \star_{(o-1)\times 1} & 0_{(o-1)\times 1} & \star_{(o-1)\times(o-2)} \end{pmatrix} \in K^{o\times o}. \tag{7}$$

The *public key* has the form $\mathcal{P} = (p^{(1)}, \ldots, p^{(o)}) = \mathcal{T} \circ \mathcal{F} \circ \mathcal{S} : K^n \to K^o$, the *private key* consists of $\hat{\mathcal{F}}, g_1, \ldots, g_o, \mathcal{S}$ and $\mathcal{T}$. Due to the special form of the map $\mathcal{T}$, the public key consists of one cubic polynomial $p^{(1)}$ and $(o-1)$ quadratic polynomials $p^{(2)}, \ldots, p^{(o)}$ in n variables. The key generation process is illustrated in Algorithm 2.

Algorithm 2 Key Generation of CSSv

Input: Finite field K with q elements and integers o, v
Output: CSSv key pair $((\mathcal{F}, \mathcal{S}, \mathcal{T}), \mathcal{P})$

1: Choose randomly 1 quadratic polynomial $\hat{f}^{(1)}$ and $(o-1)$ affine maps $\hat{f}^{(2)}, \ldots, \hat{f}^{(o)}$ in the $n = o + v$ variables $y_1, \ldots, y_n$.
2: Choose 1 random cubic polynomial g_2 in the v variables $y_{o+1}, \ldots, y_n$
3: Choose $o-1$ random quadratic polynomials $g_1, g_3, g_4, \ldots, g_o$ in the v variables $y_{o+1}, \ldots, y_n$
4: Define $\bar{f}^{(1)}(x_1, \ldots, x_o, y_{o+1}, \ldots, y_n), \ldots, \bar{f}^{(o)}(x_1, \cdots, x_o, y_{o+1}, \ldots, y_n)$ as in (6)
5: The central map is $\mathcal{F} = (f^{(1)}, \cdots, f^{(o)}) : K^n \to K^o$ where for each $i = 1, \ldots, o$ we have $f^{(i)} = \bar{f}^{(i)}(\hat{f}^{(1)}, \ldots, \hat{f}^{(o)}, y_{o+1}, \ldots, y_n)$
6: Choose a randomly invertible affine map $\mathcal{S} : K^n \to K^n$
7: Choose a randomly invertible affine map $\mathcal{T} : K^o \to K^o$ as in (7)
8: $\mathcal{P} = \mathcal{T} \circ \mathcal{F} \circ \mathcal{S} : K^n \to K^o$
9: **return** $((\mathcal{F}, \mathcal{S}, \mathcal{T}), \mathcal{P})$

Signature Generation: In order to generate a signature for a message (or hash value) $\mathbf{d} \in K^o$, the signer performs the following steps.

1. Compute $\mathbf{w} = \mathcal{T}^{-1}(\mathbf{d}) \in K^o$.
2. Choose random values for the vinegar variables $y_{o+1}, \ldots, y_n$ and substitute them into the polynomials $\hat{f}^{(1)}, \ldots, \hat{f}^{(o)}$ and $g_1, \ldots, g_o$.
3. Compute $x_1 = w_1 - g_1$ and recursively $x_i = \frac{1}{x_{i-1}} \cdot (w_i - g_i)$ $(i = 2, \ldots, o)$. If one of the x_i $(i = 1, \ldots, o-1)$ occurs to be 0, choose other values for the vinegar variables $y_{o+1}, \ldots, y_n$.
4. Solve the linear system given by the last $o-1$ equations of (5) to obtain univariate linear representations of $y_2, \ldots, y_o$ in the single variable y_1. If this fails, choose other values for the vinegar variables $y_{o+1}, \ldots, y_n$.
5. Substitute these relations into the first equation of (5) to get a univariate quadratic equation in the variable y_1, and solve it. If the equation has no solution, choose other values for the vinegar variables $y_{o+1}, \ldots, y_n$.
6. Compute a signature $\mathbf{z} \in K^n$ of the message $\mathbf{d}$ by $\mathbf{z} = \mathcal{S}^{-1}(y_1, \ldots, y_n)$.

Signature Verification: To check if $\mathbf{z} \in K^n$ is indeed a valid signature for a message $\mathbf{d} \in K^o$, the verifier simply computes $\mathbf{d}' = \mathcal{P}(\mathbf{z})$. If $\mathbf{d}' = \mathbf{d}$ holds, the signature is accepted, otherwise it is rejected.

3.2 Security

Rank Attacks There are two main types of rank attacks: The MinRank attack [2,6] and the HighRank attack [15]. The goal of the MinRank attack is to find a linear combination of the matrices associated to the homogeneous quadratic parts of the public polynomials of low rank. The idea is that such a linear combination corresponds to a central polynomial.
In the case of the CSSv scheme, the matrices associated to the central polynomials have rank $\geq v + 2$ ($v + 1$ if q even and v odd). Recovering such a central

polynomial by solving a MinRank Problem has a complexity of at least q^{v+2}. By choosing the parameter v in an appropriate way, it is therefore easy to prevent attacks of the MinRank type.
The HighRank attack tries to find (the linear representations of) the variables which appear the fewest times in the central polynomials. However, since all the variables $y_1, \ldots, y_n$ appear in every component of the central map, the HighRank attack is not applicable against CSSv.

Direct Attacks The most straightforward method to attack a multivariate cryptosystem is the direct attack. For this type of attack, one tries to solve the equation $\mathcal{P}(\mathbf{z}) = \mathbf{d}$ directly as an instance of the MQ-Problem. The most efficient and popular tool for this are Gröbner bases methods such as the F_4 algorithm [11]. The complexity of this algorithm can be estimated by

$$O\left(m \cdot \binom{n + \mathrm{d}_{reg} - 1}{\mathrm{d}_{reg}}^{\omega}\right),$$

where d_{reg} is the so called degree of regularity of the system and $2 < \omega \leq 3$ is the linear algebra constant.
In order to estimate the security of our scheme against direct attacks, we have to study the degree of regularity of the public systems. To do this, we carried out a number of experiments with MAGMA [4] (see Table 2 in the appendix of this paper). As our experiments showed, the public systems of CSSv behave, for $v = \frac{o}{2}$, very similar to random systems. On the other hand we found that, for smaller values of v, the public systems are significantly easier to solve. [5] In our parameter selection (see Section 5), we therefore choose $o = 2 \cdot v$ and the value of o in such a way, that the complexity of a direct attack against our scheme is beyond the proposed levels of security. As we found, this choice also prevents the MinRank attack against our scheme.

Linearization Equations Attack The Linearization Equations attack was first successfully used by Patarin [24] to break the Matsumoto-Imai cryptosystem [20]. The idea of this attack is to look for equations of the form

$$\sum_{i=1}^{n}\sum_{j=1}^{m} \alpha_{ij} \cdot z_i \cdot d_j + \sum_{i=1}^{n} \beta_i \cdot z_i + \sum_{j=1}^{m} \gamma_j \cdot d_j + \delta \tag{8}$$

fulfilled by the message / signature pairs $(\mathbf{d}, \mathbf{z})$ of a cryptosystem. By substituting a given message $\mathbf{d}^\star$ into (8), one obtains a linear equation in the components z_i of the signature which helps to forge a signature $\mathbf{z}^\star$ for the message $\mathbf{d}^\star$.
However since, in the case of the CSSv scheme, the maps $\hat{f}^{(1)}, \ldots, \hat{f}^{(o)}$ and $g_1, \ldots, g_o$ are chosen completely at random, there should not exist any linearization equations for our scheme.

[5] Our experiments showed that the same holds for the original CUOV scheme. In our comparison (see Table 1) we therefore changed the parameters compared to [22] to cover this fact.

Differential Attacks In a differential attack one looks for symmetries or invariants of the differential

$$\mathcal{G}(\mathbf{x}, \mathbf{y}) = \mathcal{P}(\mathbf{x} + \mathbf{y}) - \mathcal{P}(\mathbf{x}) - \mathcal{P}(\mathbf{y}) + \mathcal{P}(\mathbf{0})$$

of the public key of a multivariate cryptosystem. Differential attacks were successfully applied to attack multivariate BigField Schemes such as Sflash [10] and PMI [12]. However, differential properties have also been found for SingleField Schemes such as SimpleMatrix [32]. However, while the structure of the map $\bar{\mathcal{F}}$ looks similar to the central map of the SimpleMatrix scheme [32], the differential properties are efficiently destroyed by the use of the random quadratic maps $g_1, \ldots, g_o$.

Hashimoto's attack To simplify the description, let us assume here that the affine map $\mathcal{S}$ is the identity map, i.e. we have $\mathcal{P} = \mathcal{T} \circ \mathcal{F}$. [6] As shown above, Hashimoto's attack relies on the fact that there exists an (easy to find) relation of the public polynomials $p^{(1)}, \ldots, p^{(o)}$ of the form

$$\mathcal{Y} = \sum_{i=1}^{o} a_i \cdot p^{(i)} = \mathcal{X} + \mathcal{G}, \tag{9}$$

with $\mathcal{X}$ being a quadratic form in the variables $z_1, \ldots, z_n$ and $\mathcal{G}$ being a cubic polynomial in $z_{o+1}, \ldots, z_n$. Since the only quadratic terms in the public key of CSSv are contained in $p^{(1)}$, we have $a_1 \neq 0$. But this implies that $\mathcal{Y}$ also contains cubic terms in the variables $z_1, \ldots, z_o$. Furthermore, since $p^{(1)}$ is the only cubic polynomial in $\mathcal{P}$ and the structure of the central polynomials is efficiently hidden by the use of the affine map $\mathcal{T}$, we can not remove these terms from $\mathcal{Y}$ without recovering $\mathcal{T}$ (i.e. solving a MinRank problem). Therefore, finding a relation of the form (9) is infeasible, which means that Hashimoto's attack is not applicable to our scheme.

4 Our second improved scheme

In this section we propose, based on the idea of the CSSv scheme, a second signature scheme, which we call the Simple Vector Signature Scheme with Vinegar (SVSv) [7]. Our goal here is to get rid off the cubic equations in the private and public polynomials and therefore to reduce the size of the public key further.

[6] By doing so, we do not have to distinguish between a quadratic form of rank v and a quadratic form in v variables.

[7] The design of our scheme is inspired by the SimpleMatrix scheme [32]. Hence the name.

4.1 Construction

Key generation: Let K be a finite field with q elements, $o, v, r \in \mathbb{N}$ and set $n = o + v + r$. [8] As in the case of the CUOV and the CSSv scheme, the *central map* of the SVSv scheme has the form $\mathcal{F} = \bar{\mathcal{F}} \circ (\hat{\mathcal{F}} \times \mathrm{id}_v)$, where id_v is the identity map in K^v. The map $\hat{\mathcal{F}} = (\hat{f}^{(1)}, \ldots, \hat{f}^{(o)}) : K^n \to K^o$ consists of o randomly chosen affine polynomials in the n variables $y_1, \ldots, y_n$. The map $\bar{\mathcal{F}} : K^o \times K^{v+r} \to K^o$ is given by

$$\begin{cases} \bar{f}^{(1)} &= x_1^2 + g_1(y_{o+1}, \ldots, y_n) \\ \bar{f}^{(2)} &= x_1 \cdot x_2 + g_2(y_{o+1}, \ldots, y_{o+v}) \\ & \cdots \\ \bar{f}^{(o)} &= x_{o-1} \cdot x_o + g_o(y_{o+1}, \ldots, y_{o+v}) \end{cases} \tag{10}$$

where $g_1, \ldots, g_o$ are randomly chosen quadratic polynomials in the vinegar variables $y_{o+1}, \ldots, y_n$. Therefore, in contrast to the CUOV and CSSv scheme, all the components of the central map of the SVSv scheme are quadratic polynomials. To hide the structure of $\mathcal{F}$ in the public key, we combine it with two randomly chosen invertible affine maps $\mathcal{T} : K^o \to K^o$ and $\mathcal{S} : K^n \to K^n$. The *public key* is given by $\mathcal{P} = \mathcal{T} \circ \mathcal{F} \circ \mathcal{S} : K^n \to K^o$ and consists of o quadratic polynomials in n variables. The *private key* consists of the o affine polynomials $\hat{f}^{(1)}, \ldots, \hat{f}^{(o)}$ in n variables, the o quadratic polynomials $g_1, \ldots, g_o$ in $v + r$ variables and the two affine maps $\mathcal{S}$ and $\mathcal{T}$. The key generation is illustrated in Algorithm 3.

Algorithm 3 Key Generation of SVSv

Input: Finite field K with q elements and integers o, v, r
Output: SVSv key pair $((\mathcal{F}, \mathcal{S}, \mathcal{T}), \mathcal{P})$

1: Choose randomly o affine polynomials $\hat{f}^{(i)}$ in the $n = o + v$ variables $y_1, \ldots, y_n$
2: Choose a random quadratic polynomial g_1 in the $v + r$ variables $y_{o+1}, \ldots, y_n$
3: Choose $o - 1$ random quadratic polynomials $g_2, \ldots, g_o$ in the v variables $y_{o+1}, \ldots, y_{o+v}$
4: Define polynomials $\bar{f}^{(1)}(x_1, \ldots, x_o, y_{o+1}, \ldots, y_n), \ldots, \bar{f}^{(o)}(x_1, \ldots, x_o, y_{o+1}, \ldots, y_n)$ as shown in (10)
5: The central map is $\mathcal{F} = (f^{(1)}, \ldots, f^{(o)}) : K^n \to K^o$ where, for each $i = 1, \ldots, o$, we have $f^{(i)} = \bar{f}^{(i)}(\hat{f}^{(1)}, \ldots, \hat{f}^{(o)}, y_{o+1}, \ldots, y_n)$
6: Choose randomly invertible affine maps $\mathcal{S} : K^n \to K^n$ and $\mathcal{T} : K^o \to K^o$
7: $\mathcal{P} = \mathcal{T} \circ \mathcal{F} \circ \mathcal{S} : K^n \to K^o$
8: **return** $((\mathcal{F}, \mathcal{S}, \mathcal{T}), \mathcal{P})$

[8] The reason for using the parameter r is to ensure that all components of the central map have the same rank (see Section 4.2). For the case of $(q \bmod 2) = (v \bmod 2) = 0$, we use $r = 2$, otherwise $r = 1$.

Signature Generation: To generate a signature for a message $\mathbf{d} = (d_1, \ldots, d_o) \in K^o$, the signer performs the following steps.

(1) Compute the pre-image $\mathbf{w} = \mathcal{T}^{-1}(\mathbf{d})$.
(2) Choose random values for the vinegar variables $y_{o+1}, \ldots, y_n$ and substitute them into the polynomials $\hat{f}^{(1)}, \ldots, \hat{f}^{(o)}$ and $g_1, \ldots, g_o$. We obtain the values of $x_1, \ldots, x_o$ as follows:
   (a) Compute $x_1 = \sqrt{w_1 - g_1} = \begin{cases} (w_1 - g_1)^{1/2} & q = 1 \bmod 2 \\ (w_1 - g_1)^{q/2} & q = 0 \bmod 2 \end{cases}$. If $x_1 = 0$ holds, we choose other values for the vinegar variables $y_{o+1}, \ldots, y_n$.
   (b) Inductively, for $i = 2, \ldots, o$, x_i can be obtained by $x_i = (w_i - g_i)/x_{i-1}$. If x_i occurs to be 0, we choose other values for the vinegar variables $y_{o+1}, \ldots, y_n$.
(3) Having found $(x_1, \ldots, x_o)$, we solve the linear system given by $\hat{f}^{(1)}, \ldots, \hat{f}^{(o)}$ for $(y_1, \ldots, y_o)$. If there is no solution, we go back to Step (2).
(4) From a solution $(y_1, \ldots, y_n)$, a signature $\mathbf{z} \in K^n$ for $\mathbf{d}$ is easily obtained by computing $\mathbf{z} = \mathcal{S}^{-1}(y_1, \ldots, y_n)$.

Signature Verification: To check the authenticity of a signature $\mathbf{z} \in K^n$, one simply computes $\mathbf{d}' = \mathcal{P}(\mathbf{z})$. If the result is equal to the message $\mathbf{d}$, the signature is accepted, otherwise rejected.

4.2 Security

Rank attacks Similar to our analysis in Section 3.2, we study here the security of our scheme against the MinRank and the HighRank attack.
In the case of the SVSv scheme, the rank of all matrices $G_1, \ldots, G_o$ associated to the homogeneous quadratic parts of the central map components is $v + 2$ ($v + 1$ in the case of even q and odd v).
In order to ensure that all the matrices G_i have the same rank, we use the parameter r of our scheme. For odd q and $r = 0$, the rank of G_1 would be 1 less than the rank of the other matrices G_i ($i = 2, \ldots, o$). In order to avoid this, we increase the number of variables in g_1 by 1. In the case of even q, the situation is a bit more complicated, since the rank of the matrices G_i is always even. In this case, we choose $r = 1$ if v is odd and $r = 2$ otherwise.
The complexity of a MinRank attack against our scheme is therefore greater or equal to q^{v+2}. By choosing the parameter v in an appropriate way, we therefore can easily defend our scheme against the MinRank attack.
Since, similar to the case of CSSv, every component of the central map of SVSv contains all the variables $y_1, \ldots, y_n$, the HighRank is not applicable against our scheme.

Direct attacks In order to estimate the security of our scheme against direct attacks, we carried out a number of experiments with MAGMA [4] (see Table 3 in the appendix of this paper).
As our experiments showed, the public systems of SVSv behave, for $o = 2 \cdot v$, very

similar to random systems, whereas, for smaller values of v, the SVSv systems are significantly easier to solve. In our parameter selection (see next section), we therefore choose $o = 2 \cdot v$ and the value of o in such a way that the complexity of a direct attack against the scheme is beyond the proposed levels of security. As we find, this parameter choice also prevents the MinRank attack.

Hashimoto's attack [16] Again, let us assume that the affine map $\mathcal{S}$ is the identity map, i.e. $\mathcal{P} = \mathcal{T} \circ \mathcal{F}$. In order to make Hashimoto's attack work, we have to find a relation of the public polynomials of the form

$$\mathcal{Y} = \sum_{i=1}^{o} a_i \cdot p^{(i)} = \mathcal{X} + \mathcal{G}$$

with a quadratic map $\mathcal{X}$ in $z_1, \ldots, z_n$ and a cubic map $\mathcal{G}$ in $z_{o+1}, \ldots, z_n$. In order to get cubic terms in $\mathcal{Y}$, the coefficients a_i have to be polynomials itself. However, this implies that $\mathcal{Y}$ also contains cubic terms in the variables $z_1, \ldots, z_o$. Removing them requires to reconstruct the map $\mathcal{T}$ (i.e. solving a MinRank problem) which, as shown above, is infeasible.

Other attacks Similar to the CSSv scheme (see previous section), Linearization Equations Attacks are not applicable to SVSv due to the random choice of the maps $\hat{f}^{(1)}, \ldots, \hat{f}^{(o)}$ and $g_1, \ldots, g_o$. Furthermore, the use of the vinegar maps $g_1, \ldots, g_o$ efficiently destroys the differential properties of the central map $\mathcal{F}$ and therefore prevents differential attacks.

5 Parameters and Efficiency

In Table 1, we compare our CSSv and SVSv with the original CUOV [22], UOV [17] and Rainbow [8] signature schemes in terms of key and signature size. As can be seen from the table, our schemes provide, for the same security level, shorter signatures and smaller public keys than CUOV, UOV and Rainbow. In particular, SVSv achieves a reduction of the public key size of up to 55%, 79% and 24% compared to CUOV, UOV and Rainbow respectively. Regarding the private key size, the reduction factors are 13%, 93% and 79% respectively.

The signature generation process of both the CSSv and the SVSv scheme can be implemented very efficiently. Besides solving systems of linear equations, the signature generation of CSSv requires only the solution of a univariate quadratic equation; see Step 5 of the Signature Generation in Section 3.1. In the case of the SVSv scheme, we need to compute the square root of a finite field element, which is just a 2-power in fields of even characteristic; see Step 2(a) of the Signature Generation in Section 4.1. Table 2 compares the execution time in second ([s]) of our schemes with those of UOV, Rainbow and CUOV at a security level of 80 bit. The experiments were performed by using a straightforward MAGMA [4] implementation (version 2.19-7) on a processor Intel(R)

Table 1. Comparison of key sizes and signature lengths for parameters at 80-bit, 100-bit and 128-bit security level

security level (bit)	scheme parameters	hash length (bit)	signature length (bit)	public key size (KB)	private key size (KB)
80	UOV$(2^8, 28, 56)$	224	672	99.9	93.5
	Rainbow$(2^8, 17, 13, 13)$	208	344	25.1	19.1
	CUOV$(2^8, 26, 13)$	208	312	47.6	6.5
	Our CSSv$(\mathbf{2^8, 26, 13})$	**208**	**312**	**29.7**	**6.9**
	Our SVSv$(\mathbf{2^8, 26, 13, 1})$	**208**	**320**	**21.9**	**6.0**
100	UOV$(2^8, 35, 70)$	280	840	193.8	179.5
	Rainbow$(2^8, 26, 16, 17)$	264	472	59.0	45.0
	CUOV$(2^8, 34, 17)$	272	408	106.8	12.7
	Our CSSv$(\mathbf{2^8, 34, 17})$	**272**	**408**	**66.1**	**13.1**
	Our SVSv$(\mathbf{2^8, 34, 17, 1})$	**272**	**416**	**47.5**	**11.3**
128	UOV$(2^8, 45, 90)$	360	1080	409.4	375.9
	Rainbow$(2^8, 36, 21, 22)$	344	632	136.1	102.5
	CUOV$(2^8, 44, 22)$	352	528	232.0	24.8
	Our CSSv$(\mathbf{2^8, 44, 22})$	**352**	**528**	**142.6**	**24.6**
	Our SVSv$(\mathbf{2^8, 44, 22, 2})$	**352**	**544**	**103.8**	**21.4**

Table 2. Comparison of execution time for parameters at 80-bit security level

scheme parameters	key generation [s]	signature generation [s]	signature verification [s]
UOV$(2^8, 28, 56)$	6.186	0.421	1.685
Rainbow$(2^8, 17, 13, 13)$	3.824	0.370	0.808
SVSv$(2^8, 26, 13)$	**1.638**	**0.081**	**0.292**
CSSv$(2^8, 26, 13)$	2.128	0.141	0.453
CUOV$(2^8, 26, 13)$	6.041	0.248	1.076

Core(TM) i5-4300U CPU @ 2.50GHz with 8 GB RAM in Windows 7 Professional. Here, we use MAGMA commands `IsConsistent()` for solving linear systems, `Factorization()` for solving univariate quadratic equations, `Sqrt()` for computing square-root of numbers over finite fields and `Cputime()` for computing the execution time.

In the signature generation process of both the CSSv and the SVSv scheme we require all variables $x_1, \ldots, x_{o-1}$ to be different from zero. However this holds, in the case of $q = 256$, with a high probability of $\left(\frac{255}{256}\right)^{o-1}$. For the parameter sets proposed in Table 1, this probability is at least 84.5 %. Therefore, the probability of finding a signature in the first try (without choosing other values for the vinegar variables) is very high.

6 Conclusion

In this paper we revisited the recently proposed multivariate signature scheme CUOV of Nie et al. [22] and the attack of Hashimoto against this scheme. We carefully analyzed which design properties make the scheme insecure and proposed two new multivariate signature schemes called CSSv and SVSv which avoid Hashimoto's attack. We showed that our schemes are secure not only against Hashimoto's attack, but also against all known attacks on multivariate cryptosystems, including direct, rank and differential attacks. Especially the SVSv scheme is very efficient and outperforms current multivariate constructions such as UOV and Rainbow in terms of key and signature size.

Acknowledgments

The first and second author thank the Japanese Society for the Promotion of Science (JSPS) for financial support under grant KAKENHI 16K17644 and 15F15350.

References

1. D.J. Bernstein, J. Buchmann, E. Dahmen (Eds.): Post-Quantum Cryptography. Springer, 2009.
2. O. Billet, H. Gilbert: Cryptanalysis of Rainbow. SCN 2006, LNCS vol. 4116, pp. 336 - 347. Springer, 2006.
3. A. Bogdanov, T. Eisenbarth, A. Rupp, C. Wolf. Time-area optimized public-key engines: MQ-cryptosystems as replacement for elliptic curves? CHES 2008, LNCS vol. 5154, pp. 45–61. Springer, 2008.
4. W. Bosma, J. Cannon, C. Playoust: The Magma algebra system. I. The user language. J. Symbolic Comput. 24, 3-4 (1997), pp. 235 - 265.
5. A.I.T. Chen, M.-S. Chen, T.-R. Chen, C.-M. Cheng, J. Ding, E. L.-H. Kuo, F. Y.-S. Lee, B.-Y. Yang. SSE implementation of multivariate PKCs on modern x86 cpus. CHES 2009, LNCS vol. 5747, pp. 33 - 48. Springer, 2009.
6. D. Coppersmith, J. Stern, S. Vaudenay: Attacks on the birational permutation signature schemes. CRYPTO '93, LNCS vol. 773, pp. 435 - 443. Springer, 1993.
7. J. Ding, J. E. Gower, D. S. Schmidt: Multivariate Public Key Cryptosystems. Springer, 2006.
8. J. Ding, D. S. Schmidt: Rainbow, a new multivariate polynomial signature scheme. ACNS 2005, LNCS vol. 3531, pp. 164-175. Springer 2005.
9. J. Ding, B.Y. Yang, C.H.O Chen, M.S. Chen, C.M. Cheng: New Differential-Algebraic attacks and Reparametrization of Rainbow. ACNS 2008, LNCS vol. 5037, pp. 242–257. Springer, 2008.
10. V. Dubois, P. Fouque, A. Shamir, J. Stern. Practical cryptanalysis of SFLASH. CRYPTO 2007, LNCS vol. 4622, pp. 1 - 12. Springer 2007.
11. J.C. Faugère: A new efficient algorithm for computing Gröbner bases (F4). Journal of Pure and Applied Algebra 139, pp. 61-88 (1999).
12. P. A. Fouque, L. Granboulan, J. Stern: Differential Cryptanalysis for Multivariate Schemes. EUROCRYPT 2005, LNCS vol. 3494, pp. 249 - 265. Springer, 2005.

13. M. R. Garey and D. S. Johnson: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman and Company 1979.
14. D. Goodin: NSA preps quantum-resistant algorithms to head off crypto-apocalypse. http://arstechnica.com/security/2015/08/nsa-preps-quantum-resistant-algorithms-to-head-off-crypto-apocolypse/.
15. L. Goubin, N. Courtois: Cryptanalysis of the TTM cryptosystem. ASIACRYPT 2000, LNCS vol. 1976, pp. 44 - 57. Springer, 2000.
16. Y. Hashimoto: On the security of Cubic UOV. IACR eprint archive, http://eprint.iacr.org/2016/788
17. A. Kipnis, L. Patarin, L. Goubin: Unbalanced Oil and Vinegar Schemes. EUROCRYPT 1999, LNCS vol. 1592, pp. 206–222. Springer 1999.
18. N. Koblitz: Elliptic curve cryptosystems. Math. Comp. 48, 177 (1987), pp. 203 - 209.
19. D. Kravitz: Digital Signature Algorithm. US patent 5231668 (July 1991).
20. T. Matsumoto, H. Imai: Public quadratic polynomial-tuples for efficient signature-verification and message-encryption. EUROCRYPT 1988. LNCS vol. 330, pp. 419-453. Springer 1988.
21. D. Moody, R. A. Perlner, D. Smith-Tone: An Asymptotical Optimal Attack on the ABC Multivariate Encryption Scheme. PQCrypto 2014, LNCS vol. 8772, pp. 180 - 196. Springer 2014.
22. X. Nie, B. Liu, H. Xiong, G. Lu: Cubic unbalance oil and vinegar signature scheme. Inscrypt 2015, LNCS vol. 9589, pp. 47 - 56. Springer, 2016.
23. National Institute of Standards and Technology: Report on Post Quantum Cryptography. NISTIR draft 8105, `http://csrc.nist.gov/publications/drafts/nistir-8105/nistir_8105_draft.pdf`.
24. J. Patarin: Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocrypt 88. CRYPTO 1995, LNCS vol. 963, pp. 248-261. Springer, 1995.
25. J. Patarin: Hidden Field Equations (HFE) and Isomorphisms of Polynomials (IP): Two New Families of Asymmetric Algorithms. EUROCRYPT, LNCS vol. 1070, pp. 33 - 48. Springer, 1996.
26. A. Petzoldt, S. Bulygin, J. Buchmann: Linear recurring sequences for the UOV key generation. PKC 2011, LNCS vol. 6571, pp. 335–350. Springer, 2011.
27. A. Petzoldt, S. Bulygin, J. Buchmann: CyclicRainbow - a multivariate signature scheme with a partially cyclic public key. INDOCRYPT 2010, LNCS vol. 6498, pp. 33 - 48. Springer, 2010,
28. J. Patarin, N. Courtois, L. Goubin: Flash, a fast multivariate signature algorithm. CTRSA 2001, LNCS vol. 2020, pp. 298 - 307. Springer, 2001.
29. A. Petzoldt, M.S. Chen, B.Y. Yang, C. Tao, J. Ding: Design Principles for HFEv-based Signature Schemes. ASIACRYPT 2015 - Part I, LNCS vol. 9452, pp. 311 - 334. Springer 2015.
30. R. L. Rivest, A. Shamir, L. Adleman: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Commun. ACM 21 (2), pp. 120-126 (1978).
31. P. Shor: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer, SIAM J. Comput. 26 (5), pp. 1484 - 1509 (1997).
32. C. Tao, A. Diene, S. Tang, J. Ding. Simple matrix scheme for encryption. PQCrypto 2013, LNCS vol. 7932, pp. 231 – 242. Springer, 2013.

A Experiments with MAGMA

In this section we present the results of our experiments with the direct attack against the CSSv and SVSv schemes. For our experiments we created, for

K =GF(256) and different values of o and v, public systems of CSSv and SVSv in MAGMA [4] code. We then fixed v (resp. $v + r$ in the case of SVSv) of the variables to create determined systems and solved these using the F4 algorithm [11] integrated in MAGMA. Table 2 and 3 show the degree of regularity of the corresponding systems. For each of the parameter sets listed in the table we performed 10 experiments.

Table 3. Experiments with the direct attack against CSSv

	o	8	9	10	11	12	13	14	15
CSSv with $v = \frac{o}{3}$	v	-	3	-	-	4	-	-	5
	d_{reg}	-	8	-	-	9	-	-	11
CSSv with $v = \frac{o}{2}$	v	4	-	5	-	6	-	7	-
	d_{reg}	11	-	13	-	15	-	17	-
random system [1]	d_{reg}	11	12	13	14	15	16	17	18

[1] determined system with 1 cubic and $(o - 1)$ quadratic equations

Table 4. Experiments with the direct attack against the SVSv scheme

	o	8	9	10	11	12	13	14	15
SVSv with $v = \frac{o}{3}$	(v,r)	-	(3,1)	-	-	(4,2)	-	-	(5,1)
	d_{reg}	-	8	-	-	9	-	-	11
SVSv with $v = \frac{o}{2}$	(v,r)	(4,2)	-	(5,1)	-	(6,2)	-	(7,1)	-
	d_{reg}	10	-	12	-	14	-	16	-
random system	d_{reg}	10	11	12	13	14	15	16	17

As the experiments show, the public systems of both CSSv and SVSv behave, for $o = 2 \cdot v$, very similar to random systems. On the other hand, for smaller values of v, the public systems are significantly easier to solve.