Hybrid Approach for the Fast Verification for Improved Versions of the UOV and Rainbow Signature Schemes

Albrecht Petzoldt

Technische Universität Darmstadt, Department of Computer Science
Hochschulstraße 10, 64289 Darmstadt, Germany
apetzoldt@cdc.informatik.tu-darmstadt.de

Abstract. Multivariate cryptography is one of the main candidates to guarantee the security of communication in the post-quantum era. Especially in the area of digital signatures, multivariate cryptography offers a wide range of practical schemes. In [17] and [18] Petzoldt et al. showed a way to speed up the verification process of improved variants of the UOV and Rainbow signature schemes. In this paper we show how we can do even better by a slight variation of their algorithms.

Keywords: Multivariate Cryptography, UOV Signature Scheme, Rainbow Signature Scheme, Key Size Reduction, Fast Verification

1 Introduction

When quantum computers arrive, classical public-key cryptosystems such as RSA and ECC will be broken [1]. The reason for this is Shor's algorithm [19] which solves number theoretic problems like integer factorization and discrete logarithms in polynomial time on a quantum computer. So, to guarantee the security of communication in the post-quantum era, we need alternatives to those classical schemes. Besides lattice-, code-, and hash-based cryptosystems multivariate cryptography seems to be a candidate for this.
Additionally to its (believed) resistance against quantum computer attacks, multivariate cryptosystems are very fast, especially for signatures [2,3]. Furthermore they require only modest computational resources, which makes them attractive for the use on low-cost devices like smartcards and RFID chips.
In [17] and [18] Petzoldt et al. showed a way to speed up the verification process of improved versions of the UOV and Rainbow signature schemes. The key idea for this is to evaluate the public polynomials by computing matrix-vector products and using the structure of the public key to speed up these computations. By doing so, they achieved a speed up of the verification process by factors of 5 (UOV) and 2 (Rainbow) respectively.
In this paper we present a slight variation of their algorithms (called hybrid approach). The key idea is to evaluate the structured part of the public polynomials

by computing matrix-vector products and the random looking part by using the Macauley matrix of the public key. By our new approach, we get an additional speed up of the verification process of about 10-20 %. We derive our results both theoretically and show them using a C implementation of the schemes.
The structure of this paper is as follows: In Section 2 we give a short overview on multivariate cryptography and describe the UOV and Rainbow signature schemes. Section 3 reviews the approach of [14] and [16] to create UOV and Rainbow schemes with structured public keys. In Section 4 we demonstrate how we can use this special structure to speed up the verification process of the schemes. In Subsection 4.1 we look hereby on structured versions of the UOV scheme, whereas Subsection 4.2 deals with improved versions of Rainbow. Section 5 presents the results of our experiments and Section 6 concludes the paper.

2 Multivariate Public Key Cryptography

The basic idea behind multivariate cryptography is to choose a system $\mathcal{F}$ of m quadratic polynomials in n variables which can be easily inverted (central map). After that one chooses two affine invertible maps $\mathcal{S}$ and $\mathcal{T}$ to hide the structure of the central map. The public key of the cryptosystem is the composed quadratic map $\mathcal{P} = \mathcal{S} \circ \mathcal{F} \circ \mathcal{T}$ which is supposed to be difficult to invert. The private key consists of $\mathcal{S}$, $\mathcal{F}$ and $\mathcal{T}$ and therefore allows to invert $\mathcal{P}$.
Due to this construction, the security of multivariate cryptosystems is based on two mathematical problems:

Problem MQ: Solve the system $p^{(1)} = \ldots = p^{(m)} = 0$, where each $p^{(i)}$ is a quadratic polynomial in the n variables $x_1, \ldots, x_n$ with coefficients and variables in $\mathbb{F}$.

The MQ-problem is proven to be NP-hard even for quadratic polynomials over $GF(2)$ [8].

Problem EIP (Extended Isomorphism of Polynomials): Given a class of central maps $\mathcal{C}$ and a map $\mathcal{P}$ expressible as $\mathcal{P} = \mathcal{S} \circ \mathcal{F} \circ \mathcal{T}$, where $\mathcal{S}$ and $\mathcal{T}$ are affine maps and $\mathcal{F} \in \mathcal{C}$, find a decomposition of $\mathcal{P}$ of the form $\mathcal{P} = \mathcal{S}' \circ \mathcal{F}' \circ \mathcal{T}'$, with affine maps $\mathcal{S}'$ and $\mathcal{T}'$ and $\mathcal{F}' \in \mathcal{C}$.

In this paper we concentrate on the case of multivariate signature schemes. The standard process for signature generation and verification works as shown in Figure 1.

Signature Generation To sign a document d, we use a hash function $\mathcal{H} : \{0,1\}^* \rightarrow \mathbb{F}^m$ to compute the value $\mathbf{h} = \mathcal{H}(d) \in \mathbb{F}^m$. Then we compute $\mathbf{x} = \mathcal{S}^{-1}(\mathbf{h})$, $\mathbf{y} = \mathcal{F}^{-1}(\mathbf{x})$ and $\mathbf{z} = \mathcal{T}^{-1}(\mathbf{y})$. The signature of the document is $\mathbf{z} \in \mathbb{F}^n$. Here, $\mathcal{F}^{-1}(\mathbf{x})$ means finding one (of the possibly many) pre-image of $\mathbf{x}$ under the central map $\mathcal{F}$.

$$\begin{tikzcd} d \arrow[r, "\mathcal{H}"] & \mathbf{h} \in \mathbb{F}^m \arrow[r, "\mathcal{S}^{-1}"] & \mathbf{x} \in \mathbb{F}^m \arrow[r, "\mathcal{F}^{-1}"] & \mathbf{y} \in \mathbb{F}^n \arrow[r, "\mathcal{T}^{-1}"] & \mathbf{z} \in \mathbb{F}^n \arrow[lll, bend left, "\mathcal{P}"] \end{tikzcd}$$

Fig. 1: Signature generation and verification

Verification To verify the authenticity of a document, one simply computes $\mathbf{h}' = \mathcal{P}(\mathbf{z})$ and the hash value $\mathbf{h} = \mathcal{H}(d)$ of the document. If $\mathbf{h}' = \mathbf{h}$ holds, the signature is accepted, otherwise rejected.

There are several ways to build the central map $\mathcal{F}$ of multivariate schemes. In this paper we concentrate on the so called SingleField constructions. In contrast to BigField schemes like Matsumoto-Imai [11] and MiddleField schemes like ℓiC [6], here all the computations are done in one (relatively small) field. In the following two subsections we describe two well known examples of these schemes in detail.

2.1 The Unbalanced Oil and Vinegar (UOV) Signature Scheme

One way to create an easily invertible multivariate quadratic system is the principle of Oil and Vinegar, which was proposed by J. Patarin in [13].
Let $\mathbb{F}$ be a finite field. Let o and v be two integers and set $n = o + v$. We set $V = \{1, \ldots, v\}$ and $O = \{v + 1, \ldots, n\}$. We call $x_1, \ldots, x_v$ the Vinegar variables and $x_{v+1}, \ldots, x_n$ Oil variables. We define o quadratic polynomials $f^{(k)}(\mathbf{x}) = f^{(k)}(x_1, \ldots, x_n)$ of the form

$$f^{(k)}(\mathbf{x}) = \sum_{i \in V,\ j \in O} \alpha_{ij}^{(k)} x_i x_j + \sum_{i,j \in V,\ i \leq j} \beta_{ij}^{(k)} x_i x_j + \sum_{i \in V \cup O} \gamma_i^{(k)} x_i + \eta^{(k)} \ (1 \leq k \leq o). \tag{1}$$

Note that Oil and Vinegar variables are not fully mixed, just like oil and vinegar in a salad dressing.
The map $\mathcal{F} = (f^{(1)}(\mathbf{x}), \ldots, f^{(o)}(\mathbf{x}))$ can be easily inverted. First, we choose the values of the v Vinegar variables $x_1, \ldots, x_v$ at random. Therefore we get a system of o linear equations in the o variables $x_{v+1}, \ldots, x_n$ which can be solved e.g. by Gaussian Elimination. If the system does not have a solution, one has to choose other values of $x_1, \ldots, x_v$ and try again.
The public key of the scheme is given as $\mathcal{P} = \mathcal{F} \circ \mathcal{T}$, where $\mathcal{T}$ is an affine map from $\mathbb{F}^n$ to itself. The private key consists of the two maps $\mathcal{F}$ and $\mathcal{T}$ and therefore allows to invert the public key.

Remark: In opposite to other multivariate schemes the second affine map $\mathcal{S}$ is not needed for the security of UOV. So it can be omitted.

In his original paper [13] Patarin suggested to choose $o = v$ (Balanced Oil and Vinegar (OV)). After this scheme was broken by Kipnis and Shamir in [10], it was recommended in [9] to choose $v > o$ (Unbalanced Oil and Vinegar (UOV)). The UOV signature scheme over GF(256) is commonly believed to be secure for $o \geq 28$ equations [20] and $v = 2 \cdot o$ Vinegar variables. For UOV schemes over GF(31) we set $(o, v) = (33, 66)$ (80 bit security).

2.2 The Rainbow Signature Scheme

In [4] J. Ding and D. Schmidt proposed a signature scheme called Rainbow, which is based on the idea of (Unbalanced) Oil and Vinegar [9].

Let $\mathbb{F}$ be a finite field and V be the set $\{1, \dots, n\}$. Let $v_1, \dots, v_{u+1}, u \geq 1$ be integers such that $0 < v_1 < v_2 < \dots < v_u < v_{u+1} = n$ and define the sets of integers $V_i = \{1, \dots, v_i\}$ for $i = 1, \dots, u$. We set $o_i = v_{i+1} - v_i$ and $O_i = \{v_i + 1, \dots, v_{i+1}\}$ $(i = 1, \dots, u)$. The number of elements in V_i is v_i and we have $|O_i| = o_i$. For $k = v_1 + 1, \dots, n$ we define multivariate quadratic polynomials in the n variables $x_1, \dots, x_n$ by

$$f^{(k)}(\mathbf{x}) = \sum_{i \in O_l,\ j \in V_l} \alpha_{ij}^{(k)} x_i x_j + \sum_{i,j \in V_l,\ i \leq j} \beta_{ij}^{(k)} x_i x_j + \sum_{i \in V_l \cup O_l} \gamma_i^{(k)} x_i + \eta^{(k)}, \quad (2)$$

where l is the only integer such that $k \in O_l$. Note that these are Oil and Vinegar polynomials with $x_i,\ i \in V_l$ being the Vinegar variables and $x_j,\ j \in O_l$ being the Oil variables.

The map $\mathcal{F}(\mathbf{x}) = (f^{(v_1+1)}(\mathbf{x}), \dots, f^{(n)}(\mathbf{x}))$ can be inverted as follows. First, we choose the values of $x_1, \dots, x_{v_1}$ at random. Hence we get a system of o_1 linear equations (given by the polynomials $f^{(k)}$ $(k \in O_1)$) in the o_1 unknowns $x_{v_1+1}, \dots, x_{v_2}$, which can be solved by Gaussian Elimination. The so computed values of x_i $(i \in O_1)$ are substituted into the polynomials $f^{(k)}(\mathbf{x})$ $(k > v_2)$ and a system of o_2 linear equations (given by the polynomials $f^{(k)}$ $(k \in O_2)$) in the o_2 unknowns x_i $(i \in O_2)$ is obtained. By repeating this process we can get values for all the variables x_i $(i = 1, \dots, n)$ [1].

The public key of the scheme is given as $\mathcal{P} = \mathcal{S} \circ \mathcal{F} \circ \mathcal{T}$ with two invertible affine maps $\mathcal{S} : \mathbb{F}^m \rightarrow \mathbb{F}^m$ and $\mathcal{T} : \mathbb{F}^n \rightarrow \mathbb{F}^n$. The private key consists of $\mathcal{S}$, $\mathcal{F}$ and $\mathcal{T}$ and therefore allows to invert te public key.

In the following, we restrict ourselves to Rainbow schemes with two layers (i.e. $u = 2$). For this, $\mathbb{F} = GF(256)$, $(v_1, o_1, o_2) = (17, 13, 13)$ provides 80-bit security under known attacks [15]. For Rainbow schemes over GF(31), we choose $(v_1, o_1, o_2) = (14, 19, 14)$.

[1] It may happen, that one of the linear systems does not have a solution. If so, one has to choose other values of $x_1, \dots x_{v_1}$ and try again.

In this paper we restrict ourselves to Rainbow schemes with 2 layers. However, the results can be extended to Rainbow schemes with more layers in a natural way.

3 Improved versions of UOV and Rainbow

In [14] and [16] Petzoldt et al. presented an approach to create UOV- and Rainbow-based schemes with structured public keys, by which they could reduce the public key size of these schemes by up to 83 %. In this paper we describe only the key idea of their construction and refer to [14] and [16] for the details.

The main idea of the approach is to insert a structured matrix B into the Macauley matrix M_P of the public key. This matrix can be chosen by the user. In this paper we consider to types of structured matrices, namely

- partially circulant matrices (used for cyclicUOV and cyclicRainbow)
  To create an $m \times n$ matrix of this type, we choose randomly a vector $\mathbf{b} \in \mathbb{F}^n$. The rows of the matrix B are then given by

$$B[i] = \mathcal{R}^{i-1}(\mathbf{b}) \ (i = 1, \ldots, m), \tag{3}$$

  with $\mathcal{R}^i(\mathbf{b})$ being the cyclic right shift of the vector $\mathbf{b}$ by i positions.

- matrices generated by a Linear Recurring Sequence (LRS) (used for UOVLRS2 and RainbowLRS2)
  To create an $m \times n$ matrix of this type, we choose randomly a vector $\gamma = (\gamma_1, \ldots, \gamma_m) \in \mathbb{F}^m$. The elements of this vector have to be pairwise distinct. The rows of the matrix B are given by

$$B[i] = (1, \gamma_i, \gamma_i^2, \ldots, \gamma_i^{n-1}) \ (i = 1, \ldots, m). \tag{4}$$

To insert a structured matrix B into M_P, the authors of [14] used the relation $\mathcal{P} = \mathcal{F} \circ \mathcal{T}$ between a UOV public and private key, which translates into the matrix equation

$$M_P = M_F \cdot A \tag{5}$$

between the Macauley matrices of public key and central map. The elements of the matrix A in equation (5) are given as quadratic functions in the coefficients of the affine map $\mathcal{T}$. If this matrix is invertible, one can compute the matrix M_F in such a way that M_P has the form $M_P = (B|C)$ with a structured matrix B and a matrix C without visible structure. Figure 2 shows the layout of the resulting matrix M_P for UOV and Rainbow.
In Figure 2 we have

$$D := \frac{v \cdot (v+1)}{2} + o \cdot v$$

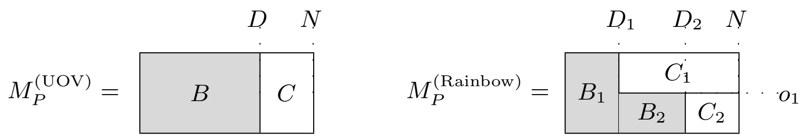

Fig. 2: Matrices M_P for structured versions of UOV (left) and Rainbow. The structured part is marked gray.

for UOV and

$$D_i = \frac{v_i \cdot (v_i + 1)}{2} + o_i \cdot v_i \ (i \in \{1, 2\})$$

for Rainbow. The number N is defined as

$$N := \frac{(n+1) \cdot (n+2)}{2}.$$

4 The verification process

The central part of the verification process for multivariate signature schemes is the evaluation of the public polynomials. Basically, there are two different strategies for this step.

Standard approach For a given (valid or invalid) signature $\mathbf{z} = (z_1, \ldots, z_n) \in \mathbb{F}^n$ one first computes an $\frac{(n+1)\cdot(n+2)}{2}$ vector mon, which contains the values of all monomials of degree ≤ 2, i.e.

$$\mathrm{mon} = (z_1^2, z_1 z_2, \ldots, z_n^2, z_1, \ldots, z_n, 1). \tag{6}$$

Then we have

$$\mathcal{P}(\mathbf{z}) = \begin{pmatrix} M_P[1] \cdot \mathrm{mon}^T \\ \vdots \\ M_P[m] \cdot \mathrm{mon}^T \end{pmatrix}, \tag{7}$$

with $M_P[i]$ being the i-th row of the Macauley matrix M_P and $\cdot$ being the standard scalar product.
Evaluating a system $\mathcal{P}$ of m equations in n variables in this way needs

- $\frac{n\cdot(n+1)}{2}$ field multiplications to compute the vector mon of equation (6) and
- $m \cdot \left(\frac{n\cdot(n+1)}{2} + n\right)$ multiplications to compute the scalar products of equation (7).

Altogether, we need therefore

$$\frac{n}{2} \cdot ((m+1) \cdot (n+1) + 2 \cdot m) \tag{8}$$

field multiplications to evaluate the system $\mathcal{P}$.

Alternative approach For each of the public polynomials

$$p^{(k)}(x_1,\ldots,x_n) = \sum_{i=1}^{n}\sum_{j=i}^{n} p_{ij}^{(k)} \cdot x_i x_j + \sum_{i=1}^{n} p_i^{(k)} \cdot x_i + p_0^{(k)} \quad (k=1,\ldots,m) \tag{9}$$

we define an upper triangular matrix $\widetilde{MP}^{(k)}$ by

$$\widetilde{MP}^{(k)} = \begin{pmatrix} p_{11}^{(k)} & p_{12}^{(k)} & p_{13}^{(k)} & \ldots & p_{1n}^{(k)} & p_1^{(k)} \\ 0 & p_{22}^{(k)} & p_{23}^{(k)} & \ldots & p_{2n}^{(k)} & p_2^{(k)} \\ 0 & 0 & p_{33}^{(k)} & & p_{3n}^{(k)} & p_3^{(k)} \\ \vdots & & \ddots & & \vdots & \vdots \\ 0 & 0 & \ldots & 0 & p_{nn}^{(k)} & p_n^{(k)} \\ 0 & 0 & \ldots & 0 & 0 & p_0^{(k)} \end{pmatrix}. \tag{10}$$

For a (valid or invalid) signature $\mathbf{z} = (z_1,\ldots,z_n)$ of the message we define the extended signature vector

$$\mathrm{sign} = (z_1,\ldots,z_n,1). \tag{11}$$

With this notation we can write the evaluate the polynomial $p^{(k)}$ by computing the matrix vector product

$$\mathrm{sign} \cdot \widetilde{MP}^{(k)} \cdot \mathrm{sign}^T \quad (k \in \{1,\ldots,m\}). \tag{12}$$

Evaluating a single polynomial in this way needs

- $\frac{(n+1)\cdot(n+2)}{2} - 1$ field multiplications to compute the matrix-vector product $\mathrm{sign} \cdot \widetilde{MP}^{(k)}$ and
- $n+1$ multiplications to compute value h_k.

Altogether, we need therefore

$$m \cdot \left(\frac{(n+1)\cdot(n+4)}{2} - 1 \right) \tag{13}$$

field multiplications to evaluate the system $\mathcal{P}$.

In this paper we propose a new way of doing the evaluation process called hybrid approach which combines standard and alternative approach.

Hybrid approach For structured versions of the UOV and Rainbow signature schemes, we combine both strategies as follows: While the random looking part of the public polynomials is evaluated by the standard approach, we evaluate the structured part using the alternative approach. By using the rich structure of our polynomials, this step can be sped up significantly. In the following two subsections we show how this can be done for improved versions of UOV and Rainbow.

4.1 UOV

Let $\mathbf{z} = (z_1, \ldots, z_n)$ be a (valid or invalid) signature.
For $k = 1, \ldots, o$ we define $v \times n$ matrices $MP^{(k)}$ containing the public coefficients of the quadratic monomials $x_i \cdot x_j$ $(1 \leq i \leq v,\ i \leq j \leq n)$ by

$$MP^{(k)} = \begin{pmatrix} p_{1,1}^{(k)} & p_{1,2}^{(k)} & \cdots & p_{1,v}^{(k)} & p_{1,v+1} & \cdots & p_{1,n-1}^{(k)} & p_{1,n}^{(k)} \\ 0 & p_{2,2}^{(k)} & \cdots & p_{2,v}^{(k)} & p_{2,v+1}^{(k)} & \cdots & p_{2,n-1}^{(k)} & p_{2,n}^{(k)} \\ 0 & 0 & \ddots & & & & & \vdots \\ 0 & \ldots & 0 & p_{v,v}^{(k)} & p_{v,v+1}^{(k)} & \cdots & p_{v,n-1}^{(k)} & p_{v,n}^{(k)} \end{pmatrix}. \tag{14}$$

Additionally we compute a vector mon $\in \mathbb{F}^{N-D}$ containing the values of the quadratic monomials $x_i x_j$ $(v+1 \leq i \leq j \leq n)$, the values of the linear monomials x_i $(1 \leq i \leq n)$ and the value of the constant monomial, i.e.

$$\text{mon} = (z_{v+1}^2, z_{v+1}z_{v+2}, z_{v+1}z_{v+3}, \ldots, z_{v+1}z_n, z_{v+2}^2, \ldots, z_n^2, z_1, z_2, \ldots, z_n, 1). \tag{15}$$

With this notation we have

$$p^{(k)}(x_1, \ldots, x_n) = \underbrace{(x_1, \ldots, x_v) \cdot MP^{(k)} \cdot (x_1, \ldots, x_n)^T}_{\text{structured part}} + \underbrace{C[k] \cdot \text{mon}^T}_{\text{random part}} \tag{16}$$

where C is the submatrix consisting of the last $N - D$ columns of the Macauley matrix M_P (see Figure 2).
In the following, we show how the structured part can be evaluated more efficiently for cyclicUOV and UOVLRS2.

cyclicUOV In the case of cyclicUOV [14], the matrices $MP^{(k)}$ are of the form shown in Figure 3. We have

$$MP_{ij}^{(k)} = MP_{i,j-1}^{(k-1)}\ \forall i = 1, \ldots, v,\ j = i+1, \ldots, n,\ k = 2, \ldots, o. \tag{17}$$

Therefore we get

$$(z_1, \ldots, z_i) \cdot \begin{pmatrix} MP_{1,j}^{(k)} \\ MP_{2,j}^{(k)} \\ \vdots \\ MP_{i,j}^{(k)} \end{pmatrix} = (z_1, \ldots, z_i) \cdot \begin{pmatrix} MP_{1,j-1}^{(k-1)} \\ MP_{2,j-1}^{(k-1)} \\ \vdots \\ MP_{i,j-1}^{(k-1)} \end{pmatrix} \quad \begin{matrix} \forall i = 1, \ldots, v \\ j = i+1, \ldots, n, \\ k = 2, \ldots, o. \end{matrix} \tag{18}$$

The boxes in Figure 3 illustrate this equation. The black boxes show the vector $(MP_{1,j-1}^{(k-1)}, \ldots, MP_{i,j-1}^{(k-1)})^T$ on the right hand side of the equation, whereas the blue boxes represent the vector $(MP_{1,j}^{(k)}, \ldots, MP_{i,j}^{(k)})^T$ on the left hand side. As one can see, the blue boxes in the matrix $MP^{(k)}$ are exactly the same as the black boxes in the matrix $MP^{(k-1)}$ $(k = 2, \ldots, o)$. We can use this fact to speed

$$MP^{(1)} = \begin{pmatrix} s_1 & s_2 & s_3 & \cdots & s_{v-1} & s_v & s_{v+1} & \cdots & s_{n-1} & s_n \\ 0 & s_{n+1} & s_{n+2} & \cdots & s_{n+v-2} & s_{n+v-1} & s_{n+v} & \cdots & s_{2n-2} & s_{2n-1} \\ 0 & 0 & s_{2n} & \cdots & s_{2n+v-4} & s_{2n+v-3} & s_{2n+v-2} & \cdots & s_{3n-4} & s_{3n-3} \\ \vdots & & \ddots & & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & & \cdots & 0 & s_{D-2o-1} & s_{D-2o} & s_{D-2o+1} & \cdots & s_{D-o-2} & s_{D-o-1} \\ 0 & & \cdots & & 0 & s_{D-o} & s_{D-o+1} & \cdots & s_{D-1} & s_D \end{pmatrix}$$

$$MP^{(2)} = \begin{pmatrix} s_D & s_1 & s_2 & \cdots & s_{v-2} & s_{v-1} & s_v & \cdots & s_{n-2} & s_{n-1} \\ 0 & s_n & s_{n+1} & \cdots & s_{n+v-3} & s_{n+v-2} & s_{n+v-1} & \cdots & s_{2n-3} & s_{2n-2} \\ 0 & 0 & s_{2n-1} & \cdots & s_{2n+v-5} & s_{2n+v-4} & s_{2n+v-3} & \cdots & s_{3n-5} & s_{3n-4} \\ \vdots & & \ddots & & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & & \cdots & 0 & s_{D-2o-2} & s_{D-2o-1} & s_{D-2o} & \cdots & s_{D-o-3} & s_{D-o-2} \\ 0 & & \cdots & & 0 & s_{D-o-1} & s_{D-o} & \cdots & s_{D-2} & s_{D-1} \end{pmatrix}$$

$$\vdots$$

$$MP^{(o-1)} = \begin{pmatrix} s_{D-o+3} & s_{D-o+4} & s_{D-o+5} & \cdots & s_{o+1} & s_{o+2} & s_{o+3} & \cdots & s_{v+1} & s_{v+2} \\ 0 & s_{v+3} & s_{v+4} & \cdots & s_{n+o} & s_{n+o+1} & s_{n+o+2} & \cdots & s_{n+v} & s_{n+v+1} \\ 0 & 0 & s_{n+v+2} & \cdots & s_{2n+o-2} & s_{2n+o-1} & s_{2n+o} & \cdots & s_{2n+v-2} & s_{2n+v-1} \\ \vdots & & \ddots & & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & & \cdots & 0 & s_{D-3o+1} & s_{D-3o+2} & s_{D-3o+3} & \cdots & s_{D-2o} & s_{D-2o+1} \\ 0 & & \cdots & & 0 & s_{D-2o+2} & s_{D-2o+3} & \cdots & s_{D-o+1} & s_{D-o+2} \end{pmatrix}$$

$$MP^{(o)} = \begin{pmatrix} s_{D-o+2} & s_{D-o+3} & s_{D-o+4} & \cdots & s_o & s_{o+1} & s_{o+2} & \cdots & s_v & s_{v+1} \\ 0 & s_{v+2} & s_{v+3} & \cdots & s_{n+o-1} & s_{n+o} & s_{n+o+1} & \cdots & s_{n+v-1} & s_{n+v} \\ 0 & 0 & s_{n+v+1} & \cdots & s_{2n+o-3} & s_{2n+o-2} & s_{2n+o-1} & \cdots & s_{2n+v-3} & s_{2n+v-2} \\ \vdots & & \ddots & & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & & \cdots & 0 & s_{D-3o} & s_{D-3o+1} & s_{D-3o+2} & \cdots & s_{D-2o-1} & s_{D-2o} \\ 0 & & \cdots & & 0 & s_{D-2o+1} & s_{D-2o+2} & \cdots & s_{D-o} & s_{D-o+1} \end{pmatrix}$$

Fig. 3: Matrices $MP^{(k)}$ for cyclicUOV

Algorithm 1 Verification process for cyclicUOV

Input: public system of cyclicUOV, signature $\mathbf{z} = (z_1, \ldots, z_n)$, hash value $h \in \mathbb{F}^m$
Output: Boolean value TRUE or FALSE
1: mon $\leftarrow (z_{v+1}^2, z_{v+1}z_{v+2}, z_{v+1}z_{v+3}, \ldots, z_{v+1}z_n, z_{v+2}^2, \ldots, z_n^2, z_1, \ldots, z_n, 1)$
2: **for** $i = 1$ to n **do** ▷ first polynomial
3:     $\text{temp}_i \leftarrow \sum_{j=1}^{\min(i,v)} MP_{ji}^{(1)} \cdot z_j$
4: **end for**
5: $h_1' \leftarrow \sum_{j=1}^{n} \text{temp}_j \cdot z_j$
6: $h_1' \leftarrow h_1' + \sum_{i=1}^{N-D} C_{1,i} \cdot \text{mon}_i$
7: **for** $k = 2$ to o **do** ▷ polynomials $2, \ldots, o$
8:     **for** $i = n$ to $v+1$ by -1 **do**
9:         $\text{temp}_i \leftarrow \text{temp}_{i-1}$
10:     **end for**
11:     **for** $i = v$ to 2 by -1 **do**
12:         $\text{temp}_i \leftarrow \text{temp}_{i-1} + MP_{ii}^{(k)} \cdot z_i$
13:     **end for**
14:     $\text{temp}_1 \leftarrow MP_{11}^{(k)} \cdot z_1$
15:     $h_k' \leftarrow \sum_{j=1}^{n} \text{temp}_j \cdot z_j$
16:     $h_k' \leftarrow h_k' + \sum_{i=1}^{N-D} C_{k,i} \cdot \text{mon}_i$
17: **end for**
18: **if** $h_l = h_l' \; \forall l \in \{1, \ldots, o\}$ **then return** TRUE ▷ TEST
19: **else return** FALSE
20: **end if**

up the evaluation of the structured part of the cyclicUOV public key by a large factor.
The whole verification process of cyclicUOV is shown by Algorithm 1.

Algorithm 1 works as follows. In line 1 the algorithm computes the vector mon of equation (15). From line 2 to 6 we evaluate the first polynomial. From line 2 to 5 we hereby deal with the structured part of the polynomial, which is evaluated by the alternative approach. Finally, line 6 of the algorithm deals with the random looking part of the first polynomial, which is evaluated using the standard approach.
In the loop (line 7 to 17) the remaining polynomials are evaluated. From line 8 to 15 we hereby deal with the structured part of the polynomials. By using the value of the vector temp computed in the previous iteration of the loop, we can evaluate the structured part of each polynomial $p^{(i)}$ $(i = 2, \ldots, o)$ by using only $n + v$ field multiplications. Finally, in line 16 of the algorithm, we deal with the random looking part of the polynomials, which is evaluated by the standard approach.

Computational effort To evaluate the system $\mathcal{P}$, Algorithm 1 needs

- $\frac{o\cdot(o+1)}{2}$ field multiplications to compute the vector mon (line 1).

To evaluate the first polynomial, the algorithm needs

- in step 3 $\frac{v\cdot(v+1)}{2}+o\cdot v$ field multiplications,
- in step 5 n field multiplications,
- and in step 6 $\frac{o\cdot(o+1)}{2}+n$ field multiplications.

Therefore, to compute the value of h_1', the algorithm needs $\frac{n}{2}\cdot(n+5)$ field multiplications.
In the loop (line 7 to 17) the algorithm needs

- v field multiplications to compute the vector temp (line 12 and 14),
- in line 15 n field multiplications,
- and in line 16 $\frac{o\cdot(o+1)}{2}+n$ field multiplications.

So, for every iteration of the loop the algorithm needs $2\cdot n+v+\frac{o\cdot(o+1)}{2}$ field multiplications.
Altogether, we need therefore

$$o\cdot\frac{o\cdot(o+1)}{2}+\frac{n}{2}\cdot(n+5)+(o-1)\cdot(2\cdot n+v) \tag{19}$$

field multiplications to evaluate equation (16).

For $\mathbb{F}$ =GF(256), $(o,v)=(28,56)$ this means a reduction of the number of field multiplications needed during the verification process by 80 % or a factor of 5.0 (compared to the evaluation of $\mathcal{P}$ using the standard approach; see equation (8)). For a UOV scheme over GF(31), $(o,v)=(33,66)$, we get a reduction factor of 5.4.

UOVLRS2 In the case of UOVLRS2, the matrices $MP^{(k)}$ are of the form shown in Figure 4.

We have

$$MP_{ij}^{(k)}=\gamma_k\cdot MP_{i,j-1}^{(k)}\ \forall i\in\{1,\ldots,v\},\ j\in\{i+1,\ldots,n\},\ k\in\{1,\ldots,o\}. \tag{20}$$

Therefore we get

$$(z_1,\ldots,z_i)\cdot\begin{pmatrix}MP_{1,j}^{(k)}\\ MP_{2,j}^{(k)}\\ \vdots\\ MP_{i,j}^{(k)}\end{pmatrix}=\gamma_k\cdot(z_1,\ldots,z_i)\cdot\begin{pmatrix}MP_{1,j-1}^{(k)}\\ MP_{2,j-1}^{(k)}\\ \vdots\\ MP_{i,j-1}^{(k)}\end{pmatrix} \tag{21}$$
$$\forall i\in\{1,\ldots v\},\quad j\in\{i+1,\ldots,n\},\ k\in\{1,\ldots,o\}.$$

$$MP^{(k)} = \begin{pmatrix} 1 & \gamma_k & \gamma_k^2 & \cdots & \gamma_k^{v-2} & \gamma_k^{v-1} & \gamma_k^{v} & \cdots & \gamma_k^{n-2} & \gamma_k^{n-1} \\ 0 & \gamma_k^{n} & \gamma_k^{n+1} & \cdots & \gamma_k^{n+v-3} & \gamma_k^{n+v-2} & \gamma_k^{n+v-1} & \cdots & \gamma_k^{2n-3} & \gamma_k^{2n-2} \\ 0 & 0 & \gamma_k^{2n-1} & \cdots & \gamma_k^{2n+v-5} & \gamma_k^{2n+v-4} & \gamma_k^{2n+v-3} & \cdots & \gamma_k^{3n-5} & \gamma_k^{3n-4} \\ \vdots & & \ddots & & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & & \cdots & 0 & \gamma_k^{D-2o-2} & \gamma_k^{D-2o-1} & \gamma_k^{D-2o} & \cdots & \gamma_k^{D-o-4} & \gamma_k^{D-o-3} \\ 0 & & \cdots & & 0 & \gamma_k^{D-o-1} & \gamma_k^{D-o} & \cdots & \gamma_k^{D-2} & \gamma_k^{D-1} \end{pmatrix}$$

Fig. 4: Matrices $MP^{(k)}$ for UOVLRS2

The boxes in Figure 4 illustrate this equation: The black boxes show the vector $(MP^{(k)}_{1,j-1}, \ldots, MP^{(k)}_{i,j-1})^T$ on the right hand side of equation (21), while the blue boxes represent the vector $(MP^{(k)}_{1,j}, \ldots, MP^{(k)}_{i,j})^T$ on the left hand side. Any blue box can be computed by multiplying the corresponding black box by γ_k.

We can use this fact to speed up the verification process of UOVLRS2 by a large factor (see Algorithm 2).

Algorithm 2 Verification process for UOVLRS2

Input: public key of UOVLRS2, signature $\mathbf{z} = (z_1, \ldots, z_n) \in \mathbb{F}^n$, hash value $\mathbf{h} \in \mathbb{F}^m$
Output: Boolean value TRUE or FALSE

mon $\leftarrow (z_{v+1}^2, z_{v+1}z_{v+2}, z_{v+1}z_{v+3}, \ldots, z_{v+1}z_n, z_{v+2}^2, \ldots, z_n^2, z_1, \ldots, z_n, 1)$
for $k = 1$ to o **do**
    temp$_1 \leftarrow z_1$
    for $j = 2$ to v **do**
        temp$_j \leftarrow \gamma_k \cdot$ temp$_{j-1} + MP^{(k)}_{jj} \cdot z_j$
    end for
    for $j = v + 1$ to n **do**
        temp$_j \leftarrow \gamma_k \cdot$ temp$_{i-1}$
    end for
    $h'_k \leftarrow \sum_{i=1}^{n}$ temp$_i \cdot z_i$
    $h'_k \leftarrow h'_k + \sum_{i=1}^{N-D} C_{k,i} \cdot$ mon$_i$
end for
if $h_k = h'_k\ \forall k \in \{1, \ldots, o\}$ **then return** TRUE
else return FALSE
end if

Algorithm 2 works as follows:
In line 1 the vector mon of equation (15) is computed. From line 2 to 12 the polynomials are evaluated. Each polynomial is evaluated individually. From line 3 to 10 we deal with the structured part of the polynomials. Due to the special design of our polynomials we can perform this step by using only $2 \cdot n + v - 2$ field multiplications. In line 11 we finally evaluate the random looking part of the polynomials.

Computational effort Algorithm 2 needs

- $\frac{o \cdot (o+1)}{2}$ field multiplications to compute the vector mon (line 1) and, in every iteration of the main loop (line 2 to 12)
- $n + v - 2$ field multiplications to compute the vector temp (line 5 and 8)
- and $2 \cdot n + \frac{o \cdot (o+1)}{2}$ field multiplications to compute the hash value h'_k (line 10 and 11).

Therefore, to evaluate equation (12) (o iterations of the main loop), Algorithm 2 needs

$$(o+1) \cdot \frac{o \cdot (o+1)}{2} + o \cdot (3 \cdot n + v - 2) \text{ field multiplications.} \tag{22}$$

For $\mathbb{F} = GF(256)$, $(o, v) = (28, 56)$ this means a reduction of the number of field multiplications needed during the verification process by a factor of 5.2 (compared to evaluating the system $\mathcal{P}$ using the standard approach; see equation (8)). For UOV schemes over GF(31), $(o, v) = (33, 66)$, the reduction factor is 5.5.

4.2 Rainbow

The verification process of the improved versions of Rainbow is mainly done as for the improved versions of UOV. However we have to consider the different structure of the polynomials.
For Rainbow, the matrices $MP^{(k)}$ are defined as follows. For the public polynomials of the first Rainbow layer $MP^{(k)}$ is a $v_1 \times v_2$ matrix of the form

$$MP^{(k)} = \begin{pmatrix} p_{11}^{(k)} & p_{12}^{(k)} & \cdots & p_{1,v_1}^{(k)} & p_{1,v_1+1} & \cdots & p_{1,v_2-1}^{(k)} & p_{1,v_2}^{(k)} \\ 0 & p_{22}^{(k)} & \cdots & p_{2,v_1}^{(k)} & p_{2,v_1+1}^{(k)} & \cdots & p_{2,v_2-1}^{(k)} & p_{2,v_2}^{(k)} \\ 0 & 0 & \ddots & & & & & \vdots \\ 0 & 0 & 0 & p_{v_1,v_1}^{(k)} & p_{v_1,v_1+1}^{(k)} & \cdots & p_{v_1,v_2-1}^{(k)} & p_{v_1,v_2}^{(k)} \end{pmatrix} (v_1 + 1 \leq k \leq v_2), \tag{23}$$

for the public polynomials of the second layer we get

$$MP^{(k)} = \begin{pmatrix} p_{11}^{(k)} & p_{12}^{(k)} & \cdots & p_{1,v_2}^{(k)} & p_{1,v_2+1} & \cdots & p_{1,n-1}^{(k)} & p_{1,n}^{(k)} \\ 0 & p_{22}^{(k)} & \cdots & p_{2,v_2}^{(k)} & p_{2,v_2}^{(k)} & \cdots & p_{2,n-1}^{(k)} & p_{2,n}^{(k)} \\ 0 & 0 & \ddots & & & & & \vdots \\ 0 & 0 & 0 & p_{v_2,v_2}^{(k)} & p_{v_2,v_2+1}^{(k)} & \cdots & p_{v_2,n-1}^{(k)} & p_{v_2,n}^{(k)} \end{pmatrix} \in \mathbb{F}^{v_2 \times n} \; (v_2+1 \leq k \leq n). \tag{24}$$

For each layer $\ell \in \{1,2\}$ we define a vector $\mathrm{mon}^{(\ell)}$ containing the monomials of the non structured part of the public key (with respect to the graded lexicographic order of monomials), i.e.

$$\begin{aligned}\mathrm{mon}^{(1)} = (z_1 z_{v_2+1}, z_1 z_{v_2+2}, \ldots, z_1 z_n, z_2 z_{v_2+1}, \ldots z_{v_1} z_n,\\ z_{v_1+1}^2, z_{v_1+1} z_{v_1+2}, \ldots, z_{v_1+1} z_n, z_{v_1+2}^2, \ldots, z_n^2, z_1, \ldots, z_n, 1)\end{aligned} \quad (25)$$

and

$$\mathrm{mon}^{(2)} = (z_{v_2+1}^2, z_{v_2+1} z_{v_2+2}, \ldots, z_{v_2+1} z_n, z_{v_2+2}^2, \ldots, z_n^2, z_1, \ldots, z_n, 1). \quad (26)$$

Such we get

$$p^{(k)}(x_1, \ldots, x_n) = \underbrace{(x_1, \ldots, x_{v_1}) \cdot MP^{(k)} \cdot (x_1, \ldots, x_{v_2})^T}_{\text{structured part}} + \underbrace{C_1[k - v_1] \cdot (\mathrm{mon}^{(1)})^T}_{\text{random part}} \ (k = v_1{+}1, \ldots, v_2), \quad (27)$$

and

$$p^{(k)}(x_1, \ldots, x_n) = \underbrace{(x_1, \ldots, x_{v_2}) \cdot MP^{(k)} \cdot (x_1, \ldots, x_n)^T}_{\text{structured part}} + \underbrace{C_2[k - v_2] \cdot (\mathrm{mon}^{(2)})^T}_{\text{random part}} \ (k = v_2{+}1, \ldots, n) \quad (28)$$

where the matrices C_1 and C_2 are defined as shown in Figure 2.

cyclicRainbow For the polynomials $p^{(v_1+2)}, \ldots, p^{(v_2+1)}$ we get

$$MP_{ij}^{(k)} = MP_{i,j-1}^{(k-1)} \ \forall i = 1, \ldots, v_1, \ j = i+1, \ldots, v_2, \ k = v_1+2, \ldots, v_2+1 \quad (29)$$

or

$$(\mathrm{sign}_1, \ldots, \mathrm{sign}_i) \cdot \begin{pmatrix} MP_{1,j}^{(k)} \\ MP_{2,j}^{(k)} \\ \vdots \\ MP_{i,j}^{(k)} \end{pmatrix} = (\mathrm{sign}_1, \ldots, \mathrm{sign}_i) \cdot \begin{pmatrix} MP_{1,j-1}^{(k-1)} \\ MP_{2,j-1}^{(k-1)} \\ \vdots \\ MP_{i,j-1}^{(k-1)} \end{pmatrix} \quad \begin{array}{c} \forall i = 1, \ldots, v_1, \\ j = i+1, \ldots, v_2, \\ k = v_1+2, \ldots, v_2+1. \end{array} \quad (30)$$

For the polynomials $p^{(v_2+2)}, \ldots, p^{(n)}$ we get

$$MP_{ij}^{(k)} = MP_{i,j-1}^{(k-1)} \ \forall i = 1, \ldots, v_2, \ j = i+1, \ldots, n, \ k = v_2+2, \ldots, n \quad (31)$$

or

$$(\mathrm{sign}_1, \ldots, \mathrm{sign}_i) \cdot \begin{pmatrix} MP_{1,j}^{(k)} \\ MP_{2,j}^{(k)} \\ \vdots \\ MP_{i,j}^{(k)} \end{pmatrix} = (\mathrm{sign}_1, \ldots, \mathrm{sign}_i) \cdot \begin{pmatrix} MP_{1,j-1}^{(k-1)} \\ MP_{2,j-1}^{(k-1)} \\ \vdots \\ MP_{i,j-1}^{(k-1)} \end{pmatrix} \quad \begin{array}{c} \forall i = 1, \ldots, v_2, \\ j = i+1, \ldots, n, \\ k = v_2+2, \ldots, n. \end{array} \quad (32)$$

Algorithm 3 Verification process for cyclicRainbow

Input: public system of cyclicRainbow, signature $\mathbf{z} = (z_1, \ldots, z_n)$,
hash value $h \in \mathbb{F}^m$

Output: Boolean value TRUE or FALSE

$\text{mon}^{(1)} = (z_1 z_{v_2+1}, z_1 z_{v_2+2}, \ldots z_1 z_n, z_2 z_{v_2+1}, \ldots, z_{v_1} z_n,$
$z_{v_1+1}^2, z_{v_1+1} z_{v_1+2}, \ldots, z_n^2, z_1, \ldots, z_n, 1)$

for $i = 1$ to v_2 **do** ▷ First polynomial ($p^{(v_1+1)}$)

    $\text{temp}_i \leftarrow \sum_{j=1}^{\min(i,v_1)} MP_{ji}^{(v_1+1)} \cdot z_j$

end for

$h'_1 \leftarrow \sum_{j=1}^{v_2} \text{temp}_j \cdot z_j$

$h'_1 \leftarrow h'_1 + \sum_{j=1}^{N-D_1} C_{1,j}^{(1)} \cdot \text{mon}_j^{(1)}$

for $k = v_1 + 2$ to v_2 **do** ▷ Polynomials $p^{(v_1+2)}$ to $p^{(v_2)}$

    for $i = v_2$ to $v_1 + 1$ by -1 **do**

        $\text{temp}_i \leftarrow \text{temp}_{i-1}$

    end for

    for $i = v_1$ to 2 by -1 **do**

        $\text{temp}_i \leftarrow \text{temp}_{i-1} + MP_{ii}^{(k)} \cdot z_i$

    end for

    $\text{temp}_1 \leftarrow MP_{11}^{(k)} \cdot z_1$

    $h'_k \leftarrow \sum_{j=1}^{v_2} \text{temp}_j \cdot z_j$

    $h'_k \leftarrow h'_k + \sum_{j=1}^{N-D_1} C_{k-v_1,j}^{(1)} \cdot \text{mon}_j^{(1)}$

end for

$\text{mon}^{(2)} \leftarrow (z_{v_2+1}^2, z_{v_2+1} z_{v_1+2}, \ldots, z_{v_2+1} z_n, z_{v_2+2}^2, \ldots z_n^2, z_1, \ldots, z_n, 1)$

for $i = n$ to $v_2 + 1$ by -1 **do** ▷ polynomial $p^{(v_2+1)}$

    $\text{temp}_i \leftarrow \sum_{j=1}^{v_2} MP_{ji}^{(v_2+1)} \cdot z_j$

end for

for $i = v_2$ to $v_1 + 1$ by -1 **do**

    $\text{temp}_i \leftarrow \text{temp} + \sum_{j=v_1+1}^{i} MP_{ji}^{(v_2+1)} \cdot z_j$

end for

for $i = v_1$ to 2 by -1 **do**

    $\text{temp}_i \leftarrow \text{temp}_{i-1} + MP_{ii}^{(v_2+1)} \cdot z_i$

end for

$\text{temp}_1 \leftarrow MP_{11}^{(v_2+1)} \cdot z_1$

$h'_{v_2+1} \leftarrow \sum_{j=1}^{n} \text{temp}_j \cdot z_j$

$h'_{v_2+1} \leftarrow h'_{v_2+1} + \sum_{j=1}^{N-D_2} C_{1,j}^{(2)} \cdot \text{mon}_j^{(2)}$

for $k = v_2 + 2$ to n **do** ▷ Polynomials $p^{(v_2+2)}$ to $p^{(n)}$

    for $i = n$ to $v_2 + 1$ by -1 **do**

        $\text{temp}_i \leftarrow \text{temp}_{i-1}$

    end for

    for $i = v_2$ to 2 by -1 **do**

        $\text{temp}_i \leftarrow \text{temp}_{i-1} + MP_{ii}^{(k)} \cdot z_i$

    end for

    $\text{temp}_1 \leftarrow MP_{11}^{(k)} \cdot z_1$

    $h'_k \leftarrow \sum_{j=1}^{n} \text{temp}_j \cdot \text{sign}_j$

    $h'_k \leftarrow \sum_{j=1}^{N-D_2} C_{k-v_2,j}^{(2)} \cdot \text{mon}_j^{(2)}$

end for

if $h_k = h'_k \ \forall k \in \{v_1 + 1, \ldots, n\}$ **then return** TRUE ▷ TEST

else return FALSE

end if

We can use this fact to speed up the verification process of cyclicRainbow by a large factor (see Algorithm 3).

Algorithm 3 works as follows. In line 1 the algorithm computes the vector $\text{mon}^{(1)}$ of equation (25). From line 2 to 6 the first polynomial is evaluated. From line 2 to 5 we hereby deal with the structured part of the polynomial, whereas line 6 evaluates the random looking part of the polynomial. In the loop (line 7 to 17) we then deal with the remaining polynomials of the first layer. From line 8 to 15 we evaluate the structured part. Due to the cyclic structure of the polynomials we can compute each vector temp using only v_1 field multiplications. Finally, line 16 handles the random looking part of the polynomials.
In line 18 of the algorithm the vector $\text{mon}^{(2)}$ of equation (26) is computed. From line 19 to 30 the algorithm evaluates the first polynomial of the second Rainbow layer. From line 19 to 29 we deal with the structured part of the polynomials. Due to the rich structure of the partially circulant polynomials the vector temp can be computed by using only $o_2 \cdot v_2 + \frac{o_1 \cdot (o_1+1)}{2} + v_1$ field multiplications. Finally, in line 30, we evaluate the random looking part of the polynomial. In the loop (line 31 to 41) we finally deal with the remaining polynomials of the second Rainbow layer. From line 32 to 39 the structured part of the polynomials is evaluated. Note that, to perform this part, the algorithm needs only $v_2 + n$ field multiplications. Finally, in line 40 of the algorithm, we deal with the random looking part of the polynomials is evaluated.

Computational cost To evaluate the first polynomial, Algorithm 3 needs

- $v_1 \cdot o_2 + \frac{m \cdot (m+1)}{2}$ field multiplications to compute the vector $\text{mon}^{(1)}$ (line 1),
- $\frac{v_1 \cdot (v_1+1)}{2} + v_1 \cdot o_1$ field multiplications to compute the vector temp (line 3) and
- $v_2 + v_1 \cdot o_2 + \frac{m \cdot (m+1)}{2} + n$ field multiplications to compute the value h'_{v_1+1}.

During the evaluation of each of the remaining polynomials of the first layer, the algorithm needs

- v_1 field multiplications to compute the vector temp and
- $v_2 + v_1 \cdot o_2 + \frac{m \cdot (m+1)}{2} + n$ field multiplications to compute the vector h'_k ($k = v_1 + 2, \ldots, v_2$).

To evaluate the polynomial $p^{(v_2+1)}$, the algorithm needs

- $o_2 \cdot v_2 + \frac{o_1 \cdot (o_1+1)}{2} + v_1$ field multiplications to compute the vector temp (line 20, 23, 26 and 28) and
- $n + \frac{o_2 \cdot (o_2+1)}{2} + n$ field multiplications to compute the value h'_{v_2+1}.

The vector $\text{mon}^{(2)}$ is a subvector of the vector $\text{mon}^{(1)}$ and has not to be computed again.

During the evaluation of each of the remaining polynomials of the second Rainbow layer, the algorithm needs

- v_2 field multiplications to compute the vector temp and
- $n + \frac{o_2 \cdot (o_2+1)}{2} + n$ field multiplications to compute the value h'_k $(k \in \{v_2 + 2, \ldots, n\})$

For the parameters $(q, v_1, o_1, o_2) = (256, 17, 13, 13)$, this means a reduction of the number of field multiplications needed during the verification process by 56 % or a factor of 2.3 (with respect to the evaluation with the standard approach, see (8)). For a Rainbow scheme over GF(31), $(v_1, o_1, o_1) = (14, 19, 14)$ the reduction factor is 2.2.

RainbowLRS2 For the polynomials $p^{(v_1+1)}, \ldots, p^{(v_2)}$ of the RainbowLRS2 public key we get

$$MP^{(k)}_{ij} = \gamma_k \cdot MP^{(k)}_{i,j-1} \;\forall i = 1, \ldots, v_1,\; j = i+1, \ldots, v_2,\; k = 2, \ldots, o_1 + 1 \quad (33)$$

or

$$(\mathrm{sign}_1, \ldots, \mathrm{sign}_i) \cdot \begin{pmatrix} MP^{(k)}_{1,j} \\ MP^{(k)}_{2,j} \\ \vdots \\ MP^{(k)}_{i,j} \end{pmatrix} = \gamma_k \cdot (\mathrm{sign}_1, \ldots, \mathrm{sign}_i) \cdot \begin{pmatrix} MP^{(k)}_{1,j-1} \\ MP^{(k)}_{i,j-1} \\ \vdots \\ MP^{(k)}_{i,j-1} \end{pmatrix} \quad \begin{array}{c} \forall i = 1, \ldots, v_1, \\ j = i+1, \ldots, v_2, \\ k = 2, \ldots, o_1 + 1. \end{array} \quad (34)$$

For the polynomials $p^{(v_2+1)}, \ldots, p^{(n)}$ we get

$$MP^{(k)}_{ij} = MP^{(k-1)}_{i,j-1} \;\forall i = 1, \ldots, v_2,\; j = i+1, \ldots, n,\; k = o_1+2, \ldots, o_1+o_2 \quad (35)$$

or

$$(\mathrm{sign}_1, \ldots, \mathrm{sign}_i) \cdot \begin{pmatrix} MP^{(k)}_{1,j} \\ MP^{(k)}_{2,j} \\ \vdots \\ MP^{(k)}_{i,j} \end{pmatrix} = (\mathrm{sign}_1, \ldots, \mathrm{sign}_i) \cdot \begin{pmatrix} MP^{(k-1)}_{1,j-1} \\ MP^{(k-1)}_{2,j-1} \\ \vdots \\ MP^{(k-1)}_{i,j-1} \end{pmatrix} \quad \begin{array}{c} \forall i = 1, \ldots, v_2, \\ j = i+1, \ldots, n, \\ k = o_1 + 2, \ldots, o_1 + o_2. \end{array} \quad (36)$$

We can use this fact to speed up the the verification process of RainbowLRS2 by a significant factor (see Algorithm 4).

Algorithm 4 works as follows. From line 1 to 12 we evaluate the polynomials of the first Rainbow layer. In line 1 we define the vector $\mathrm{mon}^{(1)}$ containing the values of the monomials of the random looking part of the polynomials. From line 3 to 9 we compute the matrix vector product $(z_1, \ldots, z_{v_1}) \cdot MP^{(k)}$. Due to the special structure of our polynomials we achieve this by doing only $v_2 + v_1 - 2$ field multiplications. In line 10 and 11 we finally compute the value $h'_k = p^{(k)}(\mathbf{z})$.

Algorithm 4 Verification process for RainbowLRS2

Input: public key of RainbowLRS2, signature $\mathbf{z} = (z_1, \ldots, z_n) \in \mathbb{F}^n$
hash value $\mathbf{h} \in \mathbb{F}^m$

Output: Boolean value TRUE or FALSE

$\text{mon}^{(1)} \leftarrow (z_1 z_{v_2+1}, z_1 z_{v_2+2}, \ldots, z_1 z_n, z_2 z_{v_2+1}, \ldots, z_{v_1} z_n,$
$\quad z_{v_1+1}^2, z_{v_1+1} z_{v_1+2}, \ldots, z_{v_1+1} z_n, z_{v_1+2}^2, \ldots z_n^2, z_1 \ldots, z_n, 1)$
for $k = v_1 + 1$ to v_2 **do**
  $\text{temp}_1 \leftarrow z_1$
  for $j = 2$ to v_1 **do**
    $\text{temp}_j \leftarrow \gamma_k \cdot \text{temp}_{j-1} + MP_{jj}^{(k)} \cdot z_j$
  end for
  for $j = v_1 + 1$ to v_2 **do**
    $\text{temp}_j \leftarrow \gamma_k \cdot \text{temp}_{j-1}$
  end for
  $h'_k \leftarrow \sum_{i=1}^{v_2} \text{temp}_i \cdot z_i$
  $h'_k \leftarrow h'_k + \sum_{i=1}^{N-D_1} C_{k-v_1,i}^{(1)} \cdot \text{mon}^{(1)}$
end for
$\text{mon}^{(2)} \leftarrow (z_{v_2+1}^2, z_{v_2+1} z_{v_2+2}, \ldots, z_{v_2+1} z_n, z_{v_2+2}^2, \ldots z_n^2, z_1 \ldots, z_n, 1)$
for $k = v_2 + 1$ to n **do**
  $\text{temp}_1 \leftarrow z_1$
  for $j = 2$ to v_2 **do**
    $\text{temp}_j \leftarrow \gamma_k \cdot \text{temp}_{j-1} + MP_{jj}^{(k)} \cdot z_j$
  end for
  for $j = v_2 + 1$ to n **do**
    $\text{temp}_j \leftarrow \gamma_k \cdot \text{temp}_{j-1}$
  end for
  $h'_k \leftarrow \sum_{i=1}^{n} \text{temp}_i \cdot z_i$
  $h'_k \leftarrow h'_k + \sum_{i=1}^{N-D_2} C_{k-v_2,i}^{(2)} \cdot \text{mon}^{(2)}$
end for
if $h_k = h'_k \ \forall k \in \{v_1 + 1, \ldots, n\}$ **then return** TRUE
else return FALSE
end if

From line 13 to 24 we evaluate the public polynomials of the second layer. Again, we first compute the vector $\mathrm{mon}^{(2)}$ (line 13). The matrix vector product $(z_1,\ldots,z_2)\cdot MP^{(k)}$ (line 15 to 21) can be computed by using only $n+v_2-2$ field multiplications. In line 22 and 23 we finally compute the value $h'_k = p^{(k)}(\mathbf{z})$ $(k = v_2+1,\ldots,n)$.

Computational effort To evaluate a single polynomial of the first layer, Algorithm 4 needs

- v_2+v_1-2 field multiplications to compute the product $(z_1,\ldots,z_{v_1})\cdot MP^{(k)}$ and
- $v_2+v_1\cdot o_2+\frac{m\cdot(m+1)}{2}+n+1$ field multiplications to compute the value h'_k.

Additionally we need $v_1\cdot o_2+\frac{m\cdot(m+1)}{2}$ field multiplications to compute the vector $\mathrm{mon}^{(1)}$. To evaluate a polynomial of the second layer, the algorithm needs

- $n+v_2-2$ field multiplications to compute the product $(z_1,\ldots,z_{v_1})\cdot MP^{(k)}$ and
- $n+\frac{o_2\cdot(o_2+1)}{2}+n$ field multiplications to compute the value h'_k.

Since the vector $\mathrm{mon}^{(2)}$ is a subvector of $\mathrm{mon}^{(1)}$, it has not to be computed again. Therefore, to evaluate the whole system $\mathcal{P}$, Algorithm 4 needs

$$v_1\cdot o_2+\frac{m\cdot(m+1)}{2}+o_1\cdot v_1\cdot o_2+\sum_{\ell=1}^{2} o_\ell\cdot\left(\frac{(n-v_\ell)\cdot(n-v_\ell+1)}{2}+2\cdot v_{\ell+1}+n+v_\ell-2\right) \tag{37}$$

field multiplications.

For the parameters $(q,v_1,o_1,o_2) = (256,17,13,13)$, this means a reduction by 55 % or a factor of 2.2 (with respect to the evaluation with the standard approach, see (8)). For a Rainbow scheme over GF(31), $(v_1,o_1,o_1) = (14,19,14)$ the reduction factor is 2.2.

5 Experiments

We checked our theoretical results on a straightforward C implementation of our schemes. Table 1 shows the results. The parameters in this table are chosen for 80 bit security.

The differences between the results of our theoretical analysis (see Section 4) and the actual runtime of the verification process is mainly caused by the heavy use of control structures in our algorithms.

As the table shows, the running time of the verification process can be sped up by 10 to 20 % by using the hybrid approach for the evaluation of the public systems (compared to the results of [17] and [18]). However, the additional speed up varies drastically for the different schemes.

Scheme	public key		verification			
	size (kB)	red. factor		#multiplications	time (ms)	speed up factor
$\mathrm{UOV}(31, 33, 66)$	108.5	-		171,567	1.75	-
$\mathrm{cyclicUOV}(31, 33, 66)$	17.1	6.3	[18]	31,614	0.34	5.2
			this paper	32,010	0.32	5.5
$\mathrm{UOVLRS2}(31, 33, 66)$	17.1	6.3	[17]	30,492	0.30	5.8
			this paper	30,987	0.31	5.7
$\mathrm{UOV}(256, 28, 56)$	99.9	-		105,882	0.98	-
$\mathrm{cyclicUOV}(256, 28, 56)$	16.5	6.1	[18]	20,804	0.20	4.9
			this paper	21,070	0.19	5.4
$\mathrm{UOVLRS2}(256, 28, 56)$	16.5	6.1	[17]	19,992	0.18	5.4
			this paper	20,342	0.17	5.8
$\mathrm{Rainbow}(31, 14, 19, 14)$	25.3	-		48,568	0.44	-
$\mathrm{cyclicRainbow}(31, 14, 19, 14)$	9.5	2.6	[18]	22,064	0.13	2.0
			this paper	21,718	0.12	2.1
$\mathrm{RainbowLRS2}(31, 14, 19, 14$	9.5	2.6	[17]	22,293	0.13	2.0
			this paper	22,064	0.12	2.3
$\mathrm{Rainbow}(256, 17, 13, 13)$	25.1	-		26,660	0.26	-
$\mathrm{cyclicRainbow}(256, 17, 13, 13)$	9.5	2.6	[18]	12,343	0.12	2.1
			this paper	12,178	0.12	2.1
$\mathrm{RainbowLRS2}(256, 17, 13, 13)$	9.5	2.6	[17]	12,415	0.13	2.0
			this paper	12,792	0.11	2.2

Table 1: Improved versions of UOV and Rainbow

6 Conclusion

In this paper we presented improved algorithms for the verification process of structured versions of the UOV and Rainbow signature schemes. The key idea of these algorithms is to evaluate the structured and the random looking part of the public system separately. By doing so we achieve a speed up of 10 to 20 % compared to the algorithms presented in [17] and [18].

Acknowledgements

We thank the anonymous referees of PQCrypto 2013 who suggested the improvements shown in this paper. We furthermore want to thank the Horst Görtz Foundation for financial support.

References

[1] Bernstein, D.J., Buchmann, J., Dahmen, E. (eds.): Post Quantum Cryptography. Springer, Heidelberg (2009)

[2] A. Bogdanov, T. Eisenbarth, A. Rupp, and C. Wolf. Time-area optimized public-key engines: -cryptosystems as replacement for elliptic curves? CHES 2008, LNCS vol. 5154, pp. 45-61. Springer, 2008.

[3] A.I.T. Chen, M.-S. Chen, T.-R. Chen, C.-M. Cheng, J. Ding, E. L.-H. Kuo, F. Y.-S. Lee, and B.-Y. Yang. SSE implementation of multivariate pkcs on modern x86 cpus. CHES 2009, LNCS vol. 5747, pp. 33–48. Springer, 2009.

[4] Ding J., Schmidt D.: Rainbow, a new multivariate polynomial signature scheme. In Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS vol. 3531, pp. 164–175 Springer, Heidelberg (2005)

[5] Ding, J., Yang, B.-Y., Chen, C.-H. O., Chen, M.-S., and Cheng, C.M.: New Differential-Algebraic Attacks and Reparametrization of Rainbow. In: LNCS 5037, pp.242–257, Springer, Heidelberg (2005)

[6] Ding, J., Wolf, C., Yang, B.-Y.: ℓ-invertible Cycles for Multivariate Quadratic Public Key Cryptography. In: Okamoto, T., Wang, X., (eds.): PKC 2007, LNCS, vol. 4450, pp. 266–281, Springer, Heidelberg (2007)

[7] Faugère, J.C.: A new efficient algorithm for computing Groebner bases (F4). Journal of Pure and Applied Algebra, 139:61–88 (1999)

[8] M. R. Garey and D. S. Johnson. Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, 1979

[9] Kipnis, A., Patarin, L., Goubin, L.: Unbalanced Oil and Vinegar Schemes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS vol. 1592, pp. 206–222 Springer, Heidelberg (1999)

[10] Kipnis, A., Shamir, A.: Cryptanalysis of the Oil and Vinegar Signature scheme. In: Krawzyck, H. (ed.) CRYPTO 1998, LNCS vol. 1462, pp. 257–266 Springer, Heidelberg (1998)

[11] Matsumoto, T., Imai, H.: Public Quadratic Polynomial-Tuples for efficient Signature-Verification and Message-Encryption. Advances in Cryptology - EUROCRYPT 1988, LNCS vol. 330, pp. 419–453, Springer, Heidelberg (1988)

[12] Patarin, J.: Hidden Field equations (HFE) and Isomorphisms of Polynomials (IP). In: Proceedings of EUROCRYPT'96, LNCS vol. 1070, pp. 38–48, Springer, Heidelberg (1996)
[13] Patarin, J,: The oil and vinegar signature scheme, presented at the Dagstuhl Workshop on Cryptography (September 97)
[14] Petzoldt, A. Bulygin, S., Buchmann, J.: A Multivariate Signature Scheme with a partially cyclic public key. In Proceedings of SCC 2010, pp. 229 - 235
[15] Petzoldt, A., Bulygin, S., Buchmann, J.: Selecting Parameters for the Rainbow Signature Scheme. In: Proceedings of PQCrypto'10, LNCS vol. 6061, pp. 218 -240, Springer, Heidelberg (2010)
[16] Petzoldt, A., Bulygin, S., Buchmann, J.: CyclicRainbow - A Multivariate Signature Scheme with a Partially Cyclic Public Key. In: Proceedings of INDOCRYPT'10, LNCS vol. 6498, pp. 33-48, Springer, Heidelberg (2010)
[17] Petzoldt, A., Bulygin, S.: Linear Recurring Sequences for the UOV Key Generation Revisited. In: Proceedings of ISISC 2012, LNCS vol. 7839, pp. 441-455, Springer, Heidelberg (2012)
[18] Petzoldt, A., Bulygin S., Buchmann, J.: Fast Verification for Improved Versions of the UOV and Rainbow Signature Schemes. PQCrypto 2013, to appear.
[19] Shor, P.: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer, SIAM J. Comput. 26 (5): pp. 1484–1509.
[20] E. Thomae, C. Wolf: Solving underdetermined Systems of Multivariate Quadratic Equations Revisited. PKC 2012, LNCS vol. 7293, pp. 156 - 171. Springer 2012.