Name of Proposal:

QR-UOV

Principal Submitter:
Hiroki Furue

email: hiroki.furue@ntt.com
organization: NIPPON TELEGRAPH AND TELEPHONE CORPORATION
postal address: 3-9-11, Midori-cho, Musashino-shi, Tokyo, 180-8585, Japan

Auxiliary Submitters:
Yasuhiko Ikematsu, Fumitaka Hoshino, Tsuyoshi Takagi,
Haruhisa Kosuge, Kimihiro Yamakoshi, Rika Akiyama,
Satoshi Nakamura, Shingo Orihara, Koha Kinjo

Inventors: All listed submitters

Owners: All listed submitters

Backup Point of Contact:
Koha Kinjo
email: kouha.kinjo@ntt.com
organization: NIPPON TELEGRAPH AND TELEPHONE CORPORATION
postal address: 3-9-11, Midori-cho, Musashino-shi, Tokyo, 180-8585, Japan

February 5, 2025

Contents

1 Introduction **4**
- 1.1 History . . . 4
- 1.2 QR-UOV at ASIACRYPT 2021 . . . 4
- 1.3 Our Purpose of this Document . . . 6

2 Notations and Parameters **6**
- 2.1 Notations . . . 7
- 2.2 Parameters . . . 8

3 Preliminaries **8**
- 3.1 Basic Description of UOV used for QR-UOV . . . 8
- 3.2 Matrix Representation of Quotient Ring Elements . . . 9

4 Algorithm Specification **10**
- 4.1 Key Generation . . . 11
- 4.2 Signature Generation . . . 13
- 4.3 Signature Verification . . . 15
- 4.4 Representation of Keys and Signature . . . 15
- 4.5 Parameter Sets . . . 16
- 4.6 Auxiliary Functions . . . 18
- 4.7 Generation of Pseudorandom Finite Field Elements . . . 20
- 4.8 Pseudorandom Generator . . . 22
- 4.9 Note on Basic Linear Algebra . . . 22

5 Performance Analysis **22**
- 5.1 Performance on the NIST Reference Platform . . . 22
- 5.2 Performance on Other Platforms . . . 25

6 Expected Security Strength **27**
- 6.1 Underlying Problems and Security Definitions . . . 27
- 6.2 Security Proof . . . 28
- 6.3 Security Estimation of Proposed Parameters . . . 33

7 Analysis of Attacks against QR-UOV **37**
- 7.1 Claw Finding Attack . . . 37
- 7.2 Direct Attack . . . 37
- 7.3 Key Recovery Attacks on UOV . . . 40
  - 7.3.1 Kipnis-Shamir Attack . . . 41
  - 7.3.2 Reconciliation Attack . . . 41
  - 7.3.3 Intersection Attack . . . 42
  - 7.3.4 Rectangular MinRank Attack . . . 43
- 7.4 Irreducibility of Polynomial f . . . 45
- 7.5 Lifting Method over Extension Field . . . 46
- 7.6 Multiplying $(\Phi_x^f)^{(N)}$ to the public key . . . 49

8 Advantages and Limitations **51**

Changelog (Round 1 → Round 2)

We mainly change our submissions in the following points:

- **Comprehensive algorithm specification and refined security proof:** In the first round, details on auxiliary functions such as seed expansion functions $\mathsf{Expand}_{\mathsf{pk}}$ and $\mathsf{Expand}_{\mathsf{sk}}$ were not specified. In this version, the specifications have been clarified to enhance the completeness of the algorithm specification. Since some auxiliary functions use a pseudorandom generator and rejection sampling, the security proof of QR-UOV has been refined to incorporate the use of these functions.

- **Implementations:** The NIST's status report on the first round additional signatures [NIS24] mentioned that the implementation of QR-UOV can be further optimized. In response to this comment, we improve our implementation from the first round version. Notably, our new implementations can significantly reduce the timing data from the original implementation.

- **Security:** The status report to the first round candidates expresses their concerns about the security of the quotient ring (QR) structure of QR-UOV [NIS24]. However, our proposed parameter sets from the first round remain secure to this day, demonstrating the reliability of QR-UOV's parameter selection. The security of QR-UOV is based on the hardness of UOV and QR-MQ problems. The UOV problem is a well-known problem, to which the security of other UOV variants is also reduced. We here assume the UOV problem without the QR structure used in QR-UOV. The QR-MQ problem is an original assumption used in QR-UOV, and its hardness must be the primary concern raised in the status report; however, we sufficiently analyzed its security in the first round submission, and its security has not been weakened since the first round. Also, we add comments on some recently proposed attacks on UOV variants in the second round submission. Especially, we analyze the effect of the technique used in the attack on SNOVA [IA24] on QR-UOV.

- **Parameters:** In the first round submission, we propose four parameters for each security level. In this submission, we select one recommended parameter set for each level based on its performance as follows:

$$\begin{aligned} \text{I: } & (q, v, m, \ell) = (127, 156, 54, 3), \\ \text{III: } & (q, v, m, \ell) = (127, 228, 78, 3), \\ \text{V: } & (q, v, m, \ell) = (127, 306, 105, 3). \end{aligned}$$

1 Introduction

1.1 History

Currently used public key cryptosystems such as RSA and ECC can be broken in polynomial time using a quantum computer executing Shor's algorithm [Sho99]. Thus, there has been growing interest in post-quantum cryptography (PQC), which is secure against quantum computing attacks. Indeed, the U.S. National Institute for Standards and Technology (NIST) has initiated a PQC standardization project [NIS].

Multivariate public key cryptography (MPKC), based on the difficulty of solving a system of multivariate quadratic polynomial equations over a finite field (the multivariate quadratic ($\mathcal{MQ}$) problem), is regarded as a strong candidate for PQC. The $\mathcal{MQ}$ problem is NP-complete [GJ90] and is thus likely to be secure in the post-quantum era.

The unbalanced oil and vinegar signature scheme (UOV) [KPG99], a multivariate signature scheme proposed by Kipnis et al. at EUROCRYPT 1999, has withstood various types of attacks for approximately 20 years. UOV is a well-established signature scheme owing to its short signature and short execution time. Indeed, a multilayer UOV variant Rainbow [DS05] was selected as a third-round finalist in the NIST PQC project [NIS20]. However, an attack on Rainbow proposed by Beullens at 2022 [Beu22] broke the security of third round parameters and makes the Rainbow scheme inefficient if it takes a countermeasure against the attack. Thus, the research following the approach to return to the original UOV has been accelerating. One drawback of UOV is that the public key is much larger than those of other PQC candidates such as lattice-based signature schemes. Indeed, Rainbow, whose public key size is close to that of the plain UOV, had the largest public key among the third-round-finalist signature schemes, and NIST's report [NIS20] stated that Rainbow is unsuitable as a general-purpose signature scheme owing to this problem.

One of the directions to solve the problem of UOV's large public key size is to utilize an algebraic structure. The CRYSTALS-DILITHIUM [DKL$^+$18] is a selected algorithm in the NIST PQC project, and it is standardized as ML-DSA [FIP24]. It is based on the hardness of the module learning with errors ((M)LWE) problem [BGV14]. Non-structural LWE [Reg09] is a well-studied hard problem in cryptography, and the MLWE problem is its generalization using a module comprising vectors over a ring. This suggests that improving non-structural cryptographic schemes by investigating further algebraic theory is a natural direction to realize compact schemes. We present QR-UOV following this direction to develop a UOV variant with a small public key.

1.2 QR-UOV at ASIACRYPT 2021

At ASIACRYPT 2021, Furue et al. [FIKT21] proposed a new variant of UOV, which is called *quotient ring UOV (QR-UOV)*. The public key of QR-UOV is represented by block matrices in which every component corresponds to an ele-

ment of a quotient ring. More precisely, we use an injective ring homomorphism from the quotient ring $\mathbb{F}_q[x]/(f)$ to the matrix ring $\mathbb{F}_q^{\ell\times\ell}$, where $f \in \mathbb{F}_q[x]$ is a polynomial with $\deg f = \ell$. In this study, the image Φ_g^f of the homomorphism for $g \in \mathbb{F}_q[x]/(f)$ is called the *polynomial matrix* of g. From this homomorphism, we can compress the ℓ^2 components in Φ_g^f to ℓ elements of $\mathbb{F}_q$ because the polynomial matrix Φ_g^f is determined by the ℓ coefficients of g. This can be considered as a generalization of the block-anti-circulant UOV (BAC-UOV) presented at SAC 2019 [SP20], which is the case for $f = x^\ell - 1$. Utilizing the elements of a quotient ring in block matrices is similar to the MLWE problem [BGV14] because the MLWE problem uses elements of a ring in vectors. Namely, the transition from UOV to QR-UOV (including BAC-UOV) can be regarded as analogous to the transition from LWE to MLWE. Therefore, as with the MLWE problem, this type of research deserves more attention than passing notice.

QR-UOV requires the public key to be generated considering the symmetry of the polynomial matrices Φ_g^f. In UOV, the public key $\mathcal{P} = (p_1, \ldots, p_m)$, which comprises quadratic polynomials p_i, is obtained by composing a central map $\mathcal{F} = (f_1, \ldots, f_m)$ and a linear map $\mathcal{S}$, that is, $\mathcal{P} = \mathcal{F} \circ \mathcal{S}$. Then, the corresponding matrices $P_1, \ldots, P_m$ of the public key $\mathcal{P}$ are given by $P_i = S^\top F_i S$, where $F_1, \ldots, F_m$, and S are matrices corresponding to $\mathcal{F}$ and $\mathcal{S}$, respectively. We cannot construct $P_1, \ldots, P_m$ as block matrices whose components are polynomial matrices Φ_g^f by naively choosing $F_1, \ldots, F_m$ and S as block matrices of Φ_g^f. This is because polynomial matrices Φ_g^f are generally unstable under the transpose operation. To solve this problem, we introduce the concept of an $\ell \times \ell$ invertible matrix W such that $W\Phi_g^f$ is symmetric for any $g \in \mathbb{F}_q[x]/(f)$; that is, $W\Phi_g^f$ is stable under the transpose operation. In Theorem 1, we prove that there exists such a symmetric W for any quotient ring $\mathbb{F}_q[x]/(f)$. Then, from the equations

$$(\Phi_{g_1}^f)^\top (W\Phi_{g_2}^f)\Phi_{g_1}^f = (W\Phi_{g_1}^f)^\top \Phi_{g_2}^f \Phi_{g_1}^f = W\Phi_{g_1 g_2 g_1}^f,$$

we can embed the $W\Phi_g^f$ structure into public key matrices by choosing $F_1, \ldots, F_m$ as block matrices with $W\Phi_g^f$ and S as a block matrix with Φ_g^f.

Moreover, we should consider how the choice of f affects the security of QR-UOV. Indeed, Furue et al. [FKI+20] broke BAC-UOV by transforming its anti-circulant matrices into diagonal concatenations of two smaller matrices. This transformation is obtained from the decomposition $x^\ell - 1 = (x-1)(x^{\ell-1} + \cdots + 1)$. Therefore, we investigate the relationship between the irreducibility of the polynomial f used to generate the quotient ring $\mathbb{F}_q[x]/(f)$ and the existence of such a transformation for symmetric matrices $W\Phi_g^f$. As shown in [FIKT21], if f is irreducible (*i.e.,* $\mathbb{F}_q[x]/(f)$ is a field), then there is no such transformation for matrices $W\Phi_g^f$, indicating that such an f is resistant to Furue et al.'s structural attack [FKI+20].

1.3 Our Purpose of this Document

In this document, we present a multivariate polynomial based digital signature scheme QR-UOV. The basic structure of QR-UOV is based on the original scheme at ASIACRYPT 2021 [FIKT21]. Moreover, we adopt the following developments:

- The EUF-CMA security proof in the QROM is given, and we modify the signature generation for this proof. (The proof is mainly based on the result by Kosuge and Xagawa [KX24].)
- We provide a variety of parameter sets.
- We offer more optimized implementation and analyze its performance.
- We give a new security analysis using some recently proposed results.

Organizations The rest of this document is organized as follows. Section 2 prepares some notations. Section 3 gives preliminaries on describing QR-UOV. Section 4 describes the algorithm specification of QR-UOV. Section 5 provides the performance analysis. Section 6 gives our security statements. Section 7 analyzes concrete attacks on QR-UOV. Section 8 discusses the advantages and limitations of QR-UOV.

Acknowledgements We are grateful for help from Makoto Yanagisawa and Atsuhito Nakase. We also acknowledge Noriki Mo for pointing out inconsistencies between the spec and implementation. We are grateful to Shuhei Nakamura for his useful technical comments. We thank Tetsutaro Kobayashi, Tomoyuki Okazaki, Ayumi Ito, and Sari Handa for their assistance in preparing the intermediate values. We also thank Naofumi Homma, Rei Ueno, and Hiroshi Amagasa for their comments on improving the reference implementations.

2 Notations and Parameters

This section describes the notations and parameters used in this document. We also define more specific basic methods used later in the document.

2.1 Notations

bit	one of the two symbols '0' or '1'.
bit string	an ordered sequence of bits.
byte	a bit string of length 8.
byte string	an ordered sequence of bytes.
$(\mathbf{a} \mid\mid \mathbf{b})$	a concatenation vector for given two vectors $\mathbf{a}$ and $\mathbf{b}$.
$\mathbf{0}_a$	the a dimensional zero vector over finite fields.
$\mathbb{F}_q$	finite field with q elements for a prime power q.
$\mathbb{B}$	the set $\{0, \ldots, 255\}$ of integers represented by a byte.
$\lceil x \rceil$	the smallest integer greater than or equal to a given real number x.
$\lfloor x \rfloor$	the largest integer less than or equal to a given real number x.
$[n]$	the set $\{1, \ldots, n\}$ for a given positive integer n.
$a \xleftarrow{\$} \mathrm{A}$	$a \in \mathrm{A}$ is chosen uniformly at random from A.
$\mathsf{IntegerToBits}(x, \alpha)$	computes a base-2 representation of x mod 2^α using big-endian order, ensuring the output is an α-bit string.
$\mathsf{IntegerToBytes}(x, \alpha)$	computes a base-256 representation of x mod 256^α using big-endian order, ensuring the output is an α-byte string.
$\mathsf{BitsToInteger}(x, \alpha)$	computes an integer value by α-bit string x using big-endian order.
$\mathsf{BytesToBits}(x)$	converts a byte string x to a bit string using big-endian order.
$\mathsf{Trunc}_l(x)$	truncates the leftmost l bytes from a byte string x.

We here also give some notations for representation matrices of elements of a quotient ring described in Subsection 3.2.

f	an irreducible polynomial in $\mathbb{F}_q[x]$ with degree ℓ.
Φ_g^f	an $\ell \times \ell$ matrix over $\mathbb{F}_q$ corresponding to $g \in \mathbb{F}_q[x]/(f)$ defined by equation (4) in Subsection 3.2.
$\mathcal{A}_f$	a set $\left\{\Phi_g^f \in \mathbb{F}_q^{\ell \times \ell} \;\middle|\; g \in \mathbb{F}_q[x]/(f)\right\}$.
W	an $\ell \times \ell$ matrix over $\mathbb{F}_q$ such that WX is symmetric for any $X \in \mathcal{A}_f$.
$W\mathcal{A}_f$	a set $\{WX \in \mathbb{F}_q^{\ell \times \ell} \mid X \in \mathcal{A}_f\}$.
$\mathcal{A}_f^{a,b}$	the set of $a\ell \times b\ell$ block matrices whose each component is an element of $\mathcal{A}_f$.
$W^{(a)}$	the $a\ell \times a\ell$ block diagonal matrix concatenating W diagonally a times.

2.2 Parameters

ℓ, V, M	positive integers.
v	number of vinegar variables: $v = \ell \cdot V$.
m	number of oil variables (equals to # of equations): $m = \ell \cdot M$.
n	number of variables: $n = v + m$.
N	$N = V + M$.
λ	secuirty parameter.
r	a random λ-bit string.
τ	length of random byte string input for rejection sampling, which takes three possible values depending on its specific usage.

3 Preliminaries

Since QR-UOV is an extension of plain UOV, we first review the construction of plain UOV in Subsection 3.1 to facilitate a description of QR-UOV in Section 4. Furthermore, as a preliminary step for the construction of QR-UOV, we introduce matrices representing elements of a quotient ring in Subsection 3.2.

3.1 Basic Description of UOV used for QR-UOV

This subsection describes the structure of the unbalanced oil and vinegar signature scheme (UOV) [KPG99]. For variables $\mathbf{x} = (x_1, \ldots, x_n)$ over $\mathbb{F}_q$, we call $x_1, \ldots, x_v$ *vinegar variables* and $x_{v+1}, \ldots, x_n$ *oil variables*.

We first recall the key generation of UOV as follows: We design $\mathcal{F} = (f_1, \ldots, f_m) : \mathbb{F}_q^n \to \mathbb{F}_q^m$, called a *central map*, such that each f_k with $k \in [m]$ is a quadratic polynomial of the form

$$f_k(x_1, \ldots, x_n) = \sum_{i=1}^{v} \sum_{j=i}^{n} \alpha_{i,j}^{(k)} x_i x_j, \tag{1}$$

where $\alpha_{i,j}^{(k)} \in \mathbb{F}_q$. Next, we choose a random linear map $\mathcal{S} : \mathbb{F}_q^n \to \mathbb{F}_q^n$ to hide the structure of $\mathcal{F}$. The public key map $\mathcal{P}$ is then provided as a polynomial map,

$$\mathcal{P} = \mathcal{F} \circ \mathcal{S} : \mathbb{F}_q^n \to \mathbb{F}_q^m, \tag{2}$$

and the private key comprises $\mathcal{F}$ and $\mathcal{S}$. We here omit linear and constant terms of $\mathcal{F}$ and constant terms of $\mathcal{S}$ for simplicity.

Next, we describe the inversion of the central map $\mathcal{F}$. When we try to find $\mathbf{x} \in \mathbb{F}_q^n$ satisfying $\mathcal{F}(\mathbf{x}) = \mathbf{t}$ for a given $\mathbf{t} \in \mathbb{F}_q^m$, we first choose random values $y_1, \ldots, y_v$ in $\mathbb{F}_q$ as the values of the vinegar variables. We can then easily obtain a solution for the equation $\mathcal{F}(y_1, \ldots, y_v, x_{v+1}, \ldots, x_n) = \mathbf{t}$, because this is a linear system of m equations in $n - v$ oil variables from the construction of the central map (1). If there is no solution to this equation, we choose new random values $y_1', \ldots, y_v'$, and repeat the above procedure.

By using this inversion approach, the signature is generated as follows: Given a message $\mathbf{t} \in \mathbb{F}_q^m$ to be signed, find a solution $\mathbf{y}$ to the equation $\mathcal{F}(\mathbf{x}) = \mathbf{t}$, and this gives a signature $\mathbf{s} = \mathcal{S}^{-1}(\mathbf{y}) \in \mathbb{F}_q^n$ for the message $\mathbf{t}$. The verification is performed by confirming whether $\mathcal{P}(\mathbf{s}) = \mathbf{t}$.

Finally, we introduce matrices representing the public and private keys of UOV. For each polynomial p_i of the public key map $\mathcal{P}$, there exists an $n \times n$ matrix P_i such that $p_i(\mathbf{x}) = \mathbf{x}^\top \cdot P_i \cdot \mathbf{x}$. We call these matrices $P_1, \ldots, P_m$ as the public key matrices. Similarly, an $n \times n$ matrix F_i can be taken for each f_i with $i \in [m]$, and an $n \times n$ matrix S is defined to satisfy $\mathcal{S}(\mathbf{x}) = S \cdot \mathbf{x}$. In general, these matrices P_i and F_i are taken as symmetric matrices if q is odd, and are taken as upper triangular matrices if q is even. For these representation matrices, based on equation (1), F_i has the following form

$$\begin{pmatrix} *_{v\times v} & *_{v\times m} \\ *_{m\times v} & 0_{m\times m} \end{pmatrix}. \tag{3}$$

Furthermore, from $\mathcal{P} = \mathcal{F} \circ \mathcal{S}$, we have

$$P_i = S^\top F_i S \ \ (i \in [m]).$$

3.2 Matrix Representation of Quotient Ring Elements

We here introduce polynomial matrices representing elements of a quotient ring. These matrices are utilized for the construction of QR-UOV in Section 4.

Let ℓ be a positive integer and $f \in \mathbb{F}_q[x]$ with $\deg f = \ell$. For any element g of the quotient ring $\mathbb{F}_q[x]/(f)$, we can uniquely define an $\ell \times \ell$ matrix Φ_g^f over $\mathbb{F}_q$ such that

$$\begin{pmatrix} 1 & x & \cdots & x^{\ell-1} \end{pmatrix} \Phi_g^f = \begin{pmatrix} g & xg & \cdots & x^{\ell-1}g \end{pmatrix}. \tag{4}$$

From this equation, we have

$$x^{j-1}g = \sum_{i=1}^{\ell} \left(\Phi_g^f\right)_{ij} \cdot x^{i-1} \quad (j \in [\ell]),$$

and $\left(\Phi_g^f\right)_{ij}$ is the coefficient of x^{i-1} in $x^{j-1}g$. We call the matrix Φ_g^f the *polynomial matrix* of g. The following lemma can be easily derived from this definition:

Lemma 1. *For any $g_1, g_2 \in \mathbb{F}_q[x]/(f)$, we have*

$$\Phi_{g_1}^f + \Phi_{g_2}^f = \Phi_{g_1+g_2}^f, \ \Phi_{g_1}^f \Phi_{g_2}^f = \Phi_{g_1 g_2}^f.$$

That is, the map $g \mapsto \Phi_g^f$ is an injective ring homomorphism from $\mathbb{F}_q[x]/(f)$ to the matrix ring $\mathbb{F}_q^{\ell\times\ell}$.

We let the algebra of the matrices $\mathcal{A}_f := \left\{\Phi_g^f \in \mathbb{F}_q^{\ell\times\ell} \;\middle|\; g \in \mathbb{F}_q[x]/(f)\right\}$. This is a subalgebra in the matrix algebra $\mathbb{F}_q^{\ell\times\ell}$ from Lemma 1. Every $\ell\times\ell$ polynomial matrix Φ_g^f in $\mathcal{A}_f$ can be represented by only ℓ elements in $\mathbb{F}_q$, because Φ_g^f is determined by the ℓ coefficients of $g \in \mathbb{F}_q[x]/(f)$. QR-UOV compresses the public key size of UOV by utilizing this property of Φ_g^f.

For the construction of QR-UOV, we also introduce the concept of a matrix $W \in \mathbb{F}_q^{\ell\times\ell}$ such that $W\Phi_g^f$ is stable under the transpose operation. Note that any matrix in $W\mathcal{A}_f := \{WX \in \mathbb{F}_q^{\ell\times\ell} \mid X \in \mathcal{A}_f\}$ can also be represented by only ℓ elements in $\mathbb{F}_q$. In the following theorem, we prove that there exists an invertible matrix W for any f.

Theorem 1 (Theorem 1 in [FIKT21])**.** *Let $f \in \mathbb{F}_q[x]$ with $\deg f = \ell$. Then, there exists an invertible matrix $W \in \mathbb{F}_q^{\ell\times\ell}$ such that WX is a symmetric matrix for any $X \in \mathcal{A}_f$.*

In the proof of Theorem 1 in [FIKT21], the authors propose a way of constructing W satisfying the condition from a nonzero linear map $\bar{\phi} : \mathbb{F}_q[x]/(f) \to \mathbb{F}_q$ such that the ij-component of W is equal to $\bar{\phi}(x^{i+j-2})$.

For $\mathcal{A}_f$ and positive integers N, we define the the set $\mathcal{A}_f^{N,N}$ of block matrices in $\mathbb{F}_q^{\ell N\times\ell N}$ whose every component is an element of $\mathcal{A}_f$.

Example 1. For $f = x^3 - 3x - 1$ in $\mathbb{F}_7[x]$, we can take one element of $\mathcal{A}_f^{2,2}$ as follows

$$\begin{pmatrix} 2 & 0 & 2 & 0 & 0 & 1 \\ 2 & 2 & 6 & 1 & 0 & 3 \\ 0 & 2 & 2 & 0 & 1 & 0 \\ 3 & 2 & 5 & 3 & 6 & 5 \\ 5 & 2 & 3 & 5 & 0 & 0 \\ 2 & 5 & 2 & 6 & 5 & 0 \end{pmatrix}.$$

Every 3×3 block of this matrix can be represented as an element of $\mathbb{F}_q[x]/(f)$, that is, this matrix can be represented as a 2×2 matrix over $\mathbb{F}_q[x]/(f)$

$$\begin{pmatrix} 2+2x & x \\ 3+5x+2x^2 & 3+5x+6x^2 \end{pmatrix}.$$

We construct QR-UOV using $\mathcal{A}_f$ with an irreducible f and the extension field $\mathbb{F}_{q^\ell} \simeq \mathbb{F}_q[x]/(f)$. See Subsection 7.4 for the rationale behind using an irreducible polynomial f in QR-UOV.

4 Algorithm Specification

This section defines the algorithms for functions composing QR-UOV. First, we present the algorithms of key generation, signature generation, and signature verification in Subsecions 4.1 ,4.2, and 4.3. Next, we give the parameter sets of QR-UOV in Subsection 4.5. Following that, we list the auxiliary functions and the pseudorandom generator required for computing the main functions in

Subsection 4.6 and 4.8. Finally, we include a note on the basic linear algebra in Subsection 4.9.

4.1 Key Generation

The algorithm KeyGen (Algorithm 1) outputs a public key pk and a private key sk. We first recall some notations and give a naive way of generating the public and private keys of QR-UOV. We then provide a strategy to reduce the public/private key sizes by restricting the private key to a compact form and storing expandable data as a seed within the keys and expanding it when needed. It should be noted that not all data of the public key can be stored as a seed. Finally, we explain a method of representing the extended public and private keys by matrices over the extension field $\mathbb{F}_{q^\ell}$.

Algorithm 1 KeyGen()

Output: public key $\mathsf{pk} \in \{0,1\}^\lambda \times \left(\mathbb{F}_{q^\ell}^{M\times M}\right)^m$ and private key $\mathsf{sk} \in \{0,1\}^{2\lambda}$

1: $(\mathsf{seed}_{\mathsf{pk}}, \mathsf{seed}_{\mathsf{sk}}) \xleftarrow{\$} \{0,1\}^{2\lambda}$ ▷ $\mathsf{seed}_{\mathsf{pk}}, \mathsf{seed}_{\mathsf{sk}} \in \{0,1\}^\lambda$
2: $\bar{S}' \leftarrow \mathsf{Expand}_{\mathsf{sk}}(\mathsf{seed}_{\mathsf{sk}})$ ▷ $\bar{S}' \in \mathbb{F}_{q^\ell}^{V\times M}$
3: **for** i from 1 to m **do**
4: $(\bar{P}_{i,1}, \bar{P}_{i,2}) \leftarrow \mathsf{Expand}_{\mathsf{pk}}(\mathsf{seed}_{\mathsf{pk}}, i)$ ▷ $\bar{P}_{i,1} \in \mathbb{F}_{q^\ell}^{V\times V}$ (symmetric), $\bar{P}_{i,2} \in \mathbb{F}_{q^\ell}^{V\times M}$
5: $\bar{P}_{i,3} \leftarrow -\bar{S}'^\top \bar{P}_{i,1}\bar{S}' + \bar{P}_{i,2}^\top \bar{S}' + \bar{S}'^\top \bar{P}_{i,2}$ ▷ $\bar{P}_{i,3} \in \mathbb{F}_{q^\ell}^{M\times M}$
6: **end for**
7: **return** $(\mathsf{pk}, \mathsf{sk}) = \left(\left(\mathsf{seed}_{\mathsf{pk}}, \{\bar{P}_{i,3}\}_{i\in[m]}\right), \left(\mathsf{seed}_{\mathsf{sk}}, \mathsf{seed}_{\mathsf{pk}}\right)\right)$,

We begin with the notation and naive key generation of QR-UOV. As mentioned in Subsection 2.2, let v be the number of vinegar variables, m be the number of oil variables which is equal to the number of quadratic polynomials, and $n = v + m$. We set v, m, and n as multiples of the paremeter ℓ, namely $v = \ell \cdot V$, $m = \ell \cdot M$, and $n = \ell \cdot N$. The public and private keys of QR-UOV are represented by elements of $\mathcal{A}_f^{N,N}$ and $W^{(N)}\mathcal{A}_f^{N,N} := \left\{W^{(N)} \cdot X \,|X \in \mathcal{A}_f^{N,N}\right\}$ (see the definitions in Section 3.2). Note that we here use an irreducible polynomial as f of $\mathcal{A}_f$ from a security perspective stated in Subsection 7.4.

We give a naive way of generating the public and private keys of QR-UOV.

1. Choose F_i $(i \in [m])$ from $W^{(N)}\mathcal{A}_f^{N,N}$ as a symmetric matrix with the lower-right $m \times m$ zero-block as in (3).
2. Choose an invertible matrix S from $\mathcal{A}_f^{N,N}$ randomly.
3. Compute the public key matrices $P_i = S^\top F_i S$ $(i \in [m])$.

Then, P_i $(i \in [m])$ representing the public key map are elements of $W^{(N)}\mathcal{A}_f^{N,N}$ from the following proposition:

Proposition 1 (Proposition 1 in [FIKT21]). *For $X \in \mathcal{A}_f^{N,N}$ and $Y \in W^{(N)}\mathcal{A}_f^{N,N}$, we have*

$$X^\top Y X \in W^{(N)}\mathcal{A}_f^{N,N}.$$

Next, we present the key generation adopted in this specification, which can reduce the public/private key sizes. We apply a method restricting S to a specific compact form, which was first proposed by Czypek et al. [CHT12]. For P_i ($i \in [m]$) and F_i ($i \in [m]$), we define submatrices as follows:

$$P_i = \begin{pmatrix} P_{i,1} & P_{i,2} \\ P_{i,2}^\top & P_{i,3} \end{pmatrix},$$
$$F_i = \begin{pmatrix} F_{i,1} & F_{i,2} \\ F_{i,2}^\top & 0_{m\times m} \end{pmatrix},$$

where $P_{i,1}$ and $F_{i,1}$ are symmetric $v \times v$ matrices, $P_{i,2}$ and $F_{i,2}$ are $v \times m$ matrices, and $P_{i,3}$ is a symmetric $m \times m$ matrix. We then suppose to limit S to the following compact form

$$S = \begin{pmatrix} I_v & S' \\ 0_{m\times v} & I_m \end{pmatrix}, \tag{5}$$

where S' is a $v \times m$ matrix. Then, from $P_i = S^\top F_i S$ ($i \in [m]$), we obtain

$$\begin{aligned} F_{i,1} &= P_{i,1}, \\ F_{i,2} &= -P_{i,1}S' + P_{i,2}, \\ 0_{m\times m} &= {S'}^\top P_{i,1} S' - P_{i,2}^\top S' - {S'}^\top P_{i,2} + P_{i,3}. \end{aligned} \tag{6}$$

Therefore, we can compute

$$P_{i,3} = -{S'}^\top P_{i,1} S' + P_{i,2}^\top S' + {S'}^\top P_{i,2}.$$

from $P_{i,1} \in W^{(V)}\mathcal{A}_f^{V,V}$, $P_{i,2} \in W^{(V)}\mathcal{A}_f^{V,M}$ ($i \in [m]$), and $S' \in \mathcal{A}_f^{V,M}$, where $V = v/\ell$ and $M = m/\ell$. Since we can take $P_{i,1}$, $P_{i,2}$, and S' by expanding seeds, we can compress the public key into $m \times m$ matrices $P_{i,3}$ ($i \in [m]$) and the λ-bit public seed $\mathsf{seed}_{\mathsf{pk}}$ for $P_{i,1}$, $P_{i,2}$ ($i \in [m]$). Similary, we compress the private key into the λ-bit private seed $\mathsf{seed}_{\mathsf{sk}}$ for S'. Algorithm 1 uses two functions $\mathsf{Expand}_{\mathsf{sk}}$ and $\mathsf{Expand}_{\mathsf{pk}}$ given in Subsection 4.6 to expand the public and private keys from the seeds.

Finally, we explain the method of transformating matrices into the extension field $\mathbb{F}_{q^\ell}$ since Algorithm 1 computes matrix operations over $\mathbb{F}_{q^\ell}$ for efficiency. This method is called *pull-back method* [FIKT21]. Since the public key matrice P_k with $k \in [m]$ are elements of $W^{(N)}\mathcal{A}_f^{N,N}$, we can take ℓ matrices $\bar{P}_k^{(0)}, \ldots, \bar{P}_k^{(\ell-1)} \in \mathbb{F}_q^{N\times N}$ satisfying

$$P_k = \sum_{i=0}^{\ell-1} \left(\bar{P}_k^{(i)} \otimes W\Phi_{x^i}^f \right),$$

where $\otimes$ denotes the tensor product. We then can define an $N \times N$ matrix $\bar{P}_k$ over $\mathbb{F}_q[x]/(f)$ as follows:

$$\bar{P}_k = \sum_{i=0}^{\ell-1} x^i \bar{P}_k^{(i)}.$$

By using the same way, we can construct $\bar{F}_1, \ldots, \bar{F}_m$ and $\bar{S}$ over $\mathbb{F}_{q^\ell}$ corresponding to $F_1, \ldots, F_m$ and S as follows:

$$F_k = \sum_{i=0}^{\ell-1} \left(\bar{F}_k^{(i)} \otimes W\Phi_{x^i}^f \right) \;\Rightarrow\; \bar{F}_k = \sum_{i=0}^{\ell-1} x^i \bar{F}_k^{(i)},$$
$$S = \sum_{i=0}^{\ell-1} \left(\bar{S}^{(i)} \otimes \Phi_{x^i}^f \right) \;\Rightarrow\; \bar{S} = \sum_{i=0}^{\ell-1} x^i \bar{S}^{(i)}.$$

Then, it holds $\bar{P}_k = \bar{S}^\top \bar{F}_k \bar{S}$ from $P_k = S^\top F_k S$, and $\bar{F}_k$ has the form as in (3). Further, we define a function ϕ corresponding to the above transformation as follows:

$$\phi : \mathcal{A}_f^{a,b} \ni \begin{pmatrix} \Phi_{g_{1,1}}^f & \cdots & \Phi_{g_{1,b}}^f \\ \vdots & \ddots & \vdots \\ \Phi_{g_{a,1}}^f & \cdots & \Phi_{g_{a,b}}^f \end{pmatrix} \longmapsto \begin{pmatrix} g_{1,1} & \cdots & g_{1,b} \\ \vdots & \ddots & \vdots \\ g_{a,1} & \cdots & g_{a,b} \end{pmatrix} \in \mathbb{F}_{q^\ell}^{a \times b}.$$

For each $k \in [m]$, we then have

$$\phi\left(\left(W^{(N)} \right)^{-1} P_k \right) = \bar{P}_k, \quad \phi\left(\left(W^{(N)} \right)^{-1} F_k \right) = \bar{F}_k, \quad \phi(S) = \bar{S}.$$

4.2 Signature Generation

The algorithm Sign (Algorithm 2) takes a message $\mathbf{M}$ and a private key sk as input and outputs a signature σ. The procedure primarily follows the standard signature generation of the plain UOV: Invert the central map $\mathcal{F}$ by fixing v values of the vinegar variables, and then multiply S^{-1} in the form of

$$S^{-1} = \begin{pmatrix} I_v & -S' \\ 0_{m \times v} & I_m \end{pmatrix},$$

from equation (5). We here add a modification for the EUF-CMA security proof proposed by Sakumoto et al. [SSH11].

We describe the inversion of the central map $\mathcal{F}$ in the signature generation of QR-UOV. We first choose values for the vinegar variables $y_1, \ldots, y_v \in \mathbb{F}_q$ randomly. We then choose λ-bit random salt r and compute $\mathbf{t} \in \mathbb{F}_q^m$ by applying Hash on the input concatenating a bit string μ and the salt r, namely $\mathbf{t} := \mathsf{Hash}(\mu, r)$, where μ is a hash value of the public seed $\mathsf{seed}_{\mathsf{pk}}$ and the message $\mathbf{M}$. If the linear system in the oil variables $x_{v+1}, \ldots, x_n$,

$$\mathcal{F}(y_1, \ldots, y_v, x_{v+1}, \ldots, x_n) = \mathbf{t}, \tag{7}$$

Algorithm 2 $\mathsf{Sign}(\mathbf{M}, \mathsf{sk})$

Input: message $\mathbf{M} \in \mathbb{B}^*$ and private key $\mathsf{sk} \in \{0,1\}^{2\lambda}$
Output: signature $\sigma \in \{0,1\}^{\lambda} \times \mathbb{F}_q^m$

$(\mathsf{seed}_{\mathsf{sk}}, \mathsf{seed}_{\mathsf{pk}}) \leftarrow \mathsf{sk}$
$\bar{S}' \leftarrow \mathsf{Expand}_{\mathsf{sk}}(\mathsf{seed}_{\mathsf{sk}})$
$\mathbf{y} = (y_1, \ldots, y_v)^\top \xleftarrow{\$} \mathbb{F}_q^v$
for i from 1 to m **do**
  $(\bar{P}_{i,1}, \bar{P}_{i,2}) \leftarrow \mathsf{Expand}_{\mathsf{pk}}(\mathsf{seed}_{\mathsf{pk}}, i)$
  $\bar{\mathbf{y}} \leftarrow W^{(V)}\phi^{-1}\left(\bar{P}_{i,1}\right)\mathbf{y}$ ▷ $\bar{\mathbf{y}} \in \mathbb{F}_q^v$
  $\mathbf{L}_i \leftarrow -2\phi^{-1}\left(\bar{S}'\right)^\top \bar{\mathbf{y}} + 2W^{(V)}\phi^{-1}\left(\bar{P}_{i,2}^\top\right)\mathbf{y}$ ▷ $\mathbf{L}_i \in \mathbb{F}_q^m$
  $u_i \leftarrow \mathbf{y}^\top \bar{\mathbf{y}}$ ▷ $u_i \in \mathbb{F}_q$
end for
$L \leftarrow (\mathbf{L}_1, \ldots, \mathbf{L}_m)^\top$ ▷ $L \in \mathbb{F}_q^{m \times m}$
$\mathbf{u} \leftarrow (u_1, \ldots, u_m)^\top$ ▷ $\mathbf{u} \in \mathbb{F}_q^m$
$\mu \leftarrow \mathsf{SHAKE256}(\mathsf{seed}_{\mathsf{pk}} \| \mathsf{BytesToBits}(\mathbf{M}), 512)$
repeat
  $r \xleftarrow{\$} \{0,1\}^{\lambda}$
  $\mathbf{t} \leftarrow \mathsf{Hash}(\mu, r)$ ▷ $\mathbf{t} \in \mathbb{F}_q^m$
until $L\mathbf{x} = \mathbf{t} - \mathbf{u}$ has solutions for $\mathbf{x}$.
Choose one solution $(y_{v+1}, \ldots, y_n)^\top \in \mathbb{F}_q^m$ of $L\mathbf{x} = \mathbf{t} - \mathbf{u}$ randomly.
$\mathbf{s} \leftarrow (y_1, \ldots, y_v, y_{v+1}, \ldots, y_n)^\top - \left(\phi^{-1}\left(\bar{S}'\right) \cdot (y_1, \ldots, y_v)^\top \,||\, \mathbf{0}_m\right)$ ▷ $\mathbf{s} \in \mathbb{F}_q^n$
return $\sigma = (r, \mathbf{s})$

has at least one solution, then we obtain the signature by applying $\mathcal{S}^{-1}$ into $(y_1, \ldots, y_v,\ y_{v+1}, \ldots, y_n)$, where $(y_{v+1}, \ldots, y_n)$ is a randomly chosen solution of equation (7). If there exists no solution of equation (7), then we choose a new salt r and update $\mathbf{t}$ until equation (7) has at least one solution. Note that Algorithm 2 performs a part of operation over extension fields as in Algorithm 1 and uses the function ϕ defined in Subsection 4.1.

The main difference from the signature generation of the plain UOV is that if equation (7) has no solution, then it chooses a new random salt instead of selecting new values for the vinegar variable. By doing so, the signature $\mathbf{s}$ satisfying $\mathcal{P}(\mathbf{s}) = \mathsf{Hash}(\mu, r)$ is uniformly distributed over $\mathbb{F}_q^n$, and this fact enables us to prove the EUF-CMA security of QR-UOV (see Subsection 6.2). This loop iteration does not become a bottleneck since the expected number of iterations until equation (7) has at least one solution is approximately 2.0 for any parameter sets by assuming that equation (7) is a randomized system for $x_{v+1}, \ldots, x_n$.

4.3 Signature Verification

The algorithm Verify (Algorithm 3) takes a message $\mathbf{M}$, a public key pk, and a signature σ as input and outputs accept or reject. The procedure is the same as that of the plain UOV. The authenticity of the signature is checked as:

- Compute $\mathbf{t} = \mathsf{Hash}(\mu, r)$, where μ is a hash value of $\mathsf{seed}_{\mathsf{pk}}$ and $\mathbf{M}$.
- Compute $\mathbf{t}' \in \mathbb{F}_q^m$ by substituting the signature $\mathbf{s} \in \mathbb{F}_q^n$ into the public key map $\mathcal{P}$, namely $\mathbf{t}' = \mathcal{P}(\mathbf{s})$.
- If $\mathbf{t} = \mathbf{t}'$ holds, the signature σ is accepted, otherwise it is rejected.

Algorithm 3 $\mathsf{Verify}(\mathbf{M}, \mathsf{pk}, \sigma)$

Input: message $\mathbf{M} \in \mathbb{B}^*$, public key $\mathsf{pk} \in \{0,1\}^\lambda \times \left(\mathbb{F}_{q^\ell}^{M \times M}\right)^m$, and signature $\sigma \in \{0,1\}^\lambda \times \mathbb{F}_q^m$

Output: accept or reject

1: $\left(\mathsf{seed}_{\mathsf{pk}}, \{\bar{P}_{i,3}\}_{i\in[m]}\right) \leftarrow \mathsf{pk}$
2: $(r, \mathbf{s}) \leftarrow \sigma$
3: **for** i from 1 to m **do**
4: $\quad (\bar{P}_{i,1}, \bar{P}_{i,2}) \leftarrow \mathsf{Expand}_{\mathsf{pk}}(\mathsf{seed}_{\mathsf{pk}}, i)$
5: $\quad P_i \leftarrow W^{(N)}\phi^{-1}\begin{pmatrix} \bar{P}_{i,1} & \bar{P}_{i,2} \\ \bar{P}_{i,2}^\top & \bar{P}_{i,3} \end{pmatrix}$ ▷ $P_i \in \mathbb{F}_q^{n\times n}$
6: **end for**
7: $\mu \leftarrow \mathsf{SHAKE256}(\mathsf{seed}_{\mathsf{pk}} \| \mathsf{BytesToBits}(\mathbf{M}), 512)$
8: $\mathbf{t} \leftarrow \mathsf{Hash}(\mu, r)$
9: $\mathbf{t}' \leftarrow \left(\mathbf{s}^\top P_1 \mathbf{s}, \ldots, \mathbf{s}^\top P_m \mathbf{s}\right)^\top$ ▷ $\mathbf{t}' \in \mathbb{F}_q^m$
10: **return** accept if $\mathbf{t} = \mathbf{t}'$ and reject otherwise.

4.4 Representation of Keys and Signature

As a preliminary step, we introduce the representation of polynomial matrices of the quotient ring. Recall that the block size of the representation matrices ℓ and an irreducible polynomial $f \in \mathbb{F}_q[x]$ with $\deg f = \ell$ are fixed. Each $g \in \mathbb{F}_q[x]/(f)$ has ℓ coefficients, allowing it to be represented as a concatenation of ℓ finite field elements, requiring $\lceil \log_2 q \rceil \cdot \ell$ bits in total. Since Φ_g^f is derived from g, it can also be represented in the same manner.

Then, we describe the representations of the public and private keys and the signature.

- The public key pk is a concatenation of byte strings representing $\mathsf{seed}_{\mathsf{pk}}$ and $P_{i,3}$ ($i \in [m]$). The $\mathsf{seed}_{\mathsf{pk}}$ is stored as a byte string. For $P_{i,3}$, each block matrix $W\Phi_g^f$ in $P_{i,3}$ is represented by a polynomial g with $\lceil \log_2 q \rceil \cdot \ell$ bits. Since each $P_{i,3}$ is a symmetric matrix, only the upper triangular part

needs to be stored. Therefore, $P_{i,3}$ is stored as a byte string converted from a bit string of length $(\lceil \log_2 q \rceil \cdot \ell) \cdot \frac{m(m+\ell)}{2\ell^2} \cdot m$, and $P_{i,3}$ $(i \in [m])$ is stored as concatenation of these byte strings.

- The private key sk is a concatenation of byte strings representing $\mathsf{seed}_{\mathsf{sk}}$ and $\mathsf{seed}_{\mathsf{pk}}$.
- The signature σ is a concatenation of byte strings representing r and $\mathbf{s}$. The salt r is simply stored as a byte string and $\mathbf{s}$ is stored as a byte string that is converted from a bit string of length $\lceil \log_2 q \rceil \cdot n$ representing n finite field elements in $\mathbb{F}_q$.

4.5 Parameter Sets

This subsection provides the parameter sets of QR-UOV. These parameter sets are proposed in accordance with security levels I, III, and V of the NIST PQC project [NIS22]. We take 7, 31, and 127 as the number q of the finite field $\mathbb{F}_q$. The reason that we do not use a finite field with even characteristics is as follows: If q is even, in a polynomial obtained as $\mathbf{x}^\top A\mathbf{x}$ where $A \in W^{(N)}\mathcal{A}_f^{N,N}$, the coefficients corresponding to the non-diagonal components of every diagonal block are zero owing to the symmetry of $W\Phi_g^f$.

Tables 1 and 2 provide four parameter sets for each security level, where we choose the one with $q = 127$ and $\ell = 3$ as the recommended parameters due to their efficiency and we set the security parameter λ as 128, 192, and 256 for the security levels I, III, and V, respectively.

In addition, Tables 3 and 4 show auxiliary parameters. The polynomial f used to define $\mathcal{A}_f$ is chosen to be irreducible. In Table 4, J_i denotes anti-identity matrix of size i (e.g. $J_2 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$).

Here, we give the sizes of the public key, private key, and signature for the parameter sets. These sizes are determined as follows:

- public key: $\left(\lceil \log q \rceil \cdot \frac{m^2(m+\ell)}{2\ell} + \lambda\right)$ bits
- private key: 2λ bits
- signature: $(\lceil \log q \rceil \cdot n + \lambda)$ bits

Therefore, the sizes of public/private keys and signature are computed as shown in Table 5.

Table 1: Main parameters for security level I, III, and V

SL	q	v	m	ℓ
I	127	156	54	3
III	127	228	78	3
V	127	306	105	3

Table 2: Additional parameters for security level I, III, and V

SL	q	v	m	ℓ
I	7	740	100	10
	31	165	60	3
	31	600	70	10
III	7	1100	140	10
	31	246	87	3
	31	890	100	10
V	7	1490	190	10
	31	324	114	3
	31	1120	120	10

Table 3: Part 1 of auxiliary parameters

SL	(q, v, m, ℓ)	n	N	V	M	τ_1	τ_2	τ_3
I	$(\mathbf{127}, \mathbf{156}, \mathbf{54}, \mathbf{3})$	210	70	52	18	4267	2916	82
	$(7, 740, 100, 10)$	840	84	74	10	32629	8947	201
	$(31, 165, 60, 3)$	225	75	55	20	4959	3571	104
	$(31, 600, 70, 10)$	670	67	60	7	19242	4518	116
III	$(\mathbf{127}, \mathbf{228}, \mathbf{78}, \mathbf{3})$	306	102	76	26	9020	6123	120
	$(7, 1100, 140, 10)$	1240	124	110	14	71432	18461	289
	$(31, 246, 87, 3)$	333	111	82	29	10878	7655	154
	$(31, 890, 100, 10)$	990	99	89	10	41974	9507	169
V	$(\mathbf{127}, \mathbf{306}, \mathbf{105}, \mathbf{3})$	411	137	102	35	16144	11018	162
	$(7, 1490, 190, 10)$	1680	168	149	19	130305	33694	391
	$(31, 324, 114, 3)$	438	146	108	38	18738	13145	203
	$(31, 1120, 120, 10)$	1240	124	112	12	66236	14326	210

Table 4: Part 2 of auxiliary parameters

(q, ℓ)	f	W
$(\mathbf{127}, \mathbf{3})$	$x^3 - x - 1$	$\begin{pmatrix} J_1 & \\ & J_2 \end{pmatrix}$
$(7, 10)$	$x^{10} - 2x - 1$	$\begin{pmatrix} J_1 & \\ & J_9 \end{pmatrix}$
$(31, 3)$	$x^3 - x - 1$	$\begin{pmatrix} J_1 & \\ & J_2 \end{pmatrix}$
$(31, 10)$	$x^{10} - 5x^3 - 1$	$\begin{pmatrix} J_3 & \\ & J_7 \end{pmatrix}$

Table 5: The public key, private key, and signature sizes in bytes

SL	(q, v, m, ℓ)	public key (bytes)	private key (bytes)	signature (bytes)
I	$(\mathbf{127}, \mathbf{156}, \mathbf{54}, \mathbf{3})$	24,255	32	200
	$(7, 740, 100, 10)$	20,641	32	331
	$(31, 165, 60, 3)$	23,641	32	157
	$(31, 600, 70, 10)$	12,266	32	435
III	$(\mathbf{127}, \mathbf{228}, \mathbf{78}, \mathbf{3})$	71,891	48	292
	$(7, 1100, 140, 10)$	55,149	48	489
	$(31, 246, 87, 3)$	70,983	48	232
	$(31, 890, 100, 10)$	34,399	48	643
V	$(\mathbf{127}, \mathbf{306}, \mathbf{105}, \mathbf{3})$	173,676	64	392
	$(7, 1490, 190, 10)$	135,407	64	662
	$(31, 324, 114, 3)$	158,421	64	306
	$(31, 1120, 120, 10)$	58,532	64	807

4.6 Auxiliary Functions

We introduce the following auxiliary functions. Note that these functions use RejSamp (Algorithm 10), which converts a byte string into a vector over $\mathbb{F}_q$, and RejSampPRG (Algorithm 11), which generates a pseudorandom vector over $\mathbb{F}_q$ from a seed shown in Subsection 4.7.

$\mathsf{Expand}_{\mathsf{sk}}$ (Algorithm 4) expands the private seed $\mathsf{seed}_{\mathsf{sk}}$ to $\bar{S}' \in \mathbb{F}_{q^\ell}^{V \times M}$. Using RejSampPRG, $(v_1, \ldots, v_{n_2}) \in \mathbb{F}_q^{n_2}$ with $n_2 := \ell VM$ is expanded from $\mathsf{seed}_{\mathsf{sk}}$. Then, ExpandMatrixVxM given in Algorithm 7 converts $(v_1, \ldots, v_{n_2})$ to $\bar{S}'$. Note that ExpandFql given in Algorithm 9 converts an element of $\mathbb{F}_q^\ell$ to one of $\mathbb{F}_{q^\ell}$. We sample the matrix in row-major order and sample each polynomial in $\mathbb{F}_q[x]/(f) = \mathbb{F}_{q^\ell}$ in reverse degree order from the constant term to the coefficient of $x^{\ell-1}$.

Algorithm 4 $\mathsf{Expand}_{\mathsf{sk}}(\mathsf{seed}_{\mathsf{sk}})$

Input: private seed $\mathsf{seed}_{\mathsf{sk}} \in \{0,1\}^\lambda$
Output: a matrix $\bar{S}' \in (\mathbb{F}_{q^\ell})^{V \times M}$

1: $n_2 := \ell VM$
2: $(v_1, \ldots, v_{n_2}) \leftarrow \mathsf{RejSampPRG}(\mathsf{seed}_{\mathsf{sk}}, 1, \tau_2, n_2)$ ▷ $\tau_2 = \tau_{q,\lambda}(n_2)$
3: $\bar{S}' \leftarrow \mathsf{ExpandMatrixVxM}(v_1, \ldots, v_{n_2})$
4: **return** $\bar{S}'$

$\mathsf{Expand}_{\mathsf{pk}}$ (Algorithm 5) expands the public seed $\mathsf{seed}_{\mathsf{pk}}$ to $\{\bar{P}_{i,1}\}_{i\in[m]}, \{\bar{P}_{i,2}\}_{i\in[m]}$ where $\bar{P}_{i,1}$ is a symmetric matrix in $\mathbb{F}_{q^\ell}^{V \times V}$ and $\bar{P}_{i,2} \in \mathbb{F}_{q^\ell}^{V \times M}$. To enable the public seed expansion to be executed in parallel, instead of generating all matrices at once, we define the function $\mathsf{Expand}_{\mathsf{pk}}$ as a function that produces $\bar{P}_{i,1}$ and $\bar{P}_{i,2}$ for a given counter i. Additionally, $\bar{P}_{i,1}$ and $\bar{P}_{i,2}$ are separately generated within the function to allow further parallel processing. Using RejSampPRG,

$(v_1, \ldots, v_{n_1}) \in \mathbb{F}_q^{n_1}$ with $n_1 := \ell V(V+1)/2$ and $(v'_1, \ldots, v'_{n_2}) \in \mathbb{F}_q^{n_2}$ with $n_2 := \ell VM$ are expanded from $\mathsf{seed}_{\mathsf{pk}}$. Then, ExpandSymmetricMatrixVxV given in Algorithm 8 and ExpandMatrixVxM convert these two vectors to $\bar{P}_{i,1}$ and $\bar{P}_{i,2}$, respectively. For each matrix, we sample in row-major order and sample each polynomial in $\mathbb{F}_q[x]/(f) = \mathbb{F}_{q^\ell}$ in reverse degree order. Note that for $\bar{P}_{i,1}$ we sample only the upper-triangular elements due to the symmetry.

Algorithm 5 $\mathsf{Expand}_{\mathsf{pk}}(\mathsf{seed}_{\mathsf{pk}}, i)$

Input: public seed $\mathsf{seed}_{\mathsf{pk}} \in \{0,1\}^\lambda$ and counter $i \in [m]$
Output: matrices $(\bar{P}_{i,1}, \bar{P}_{i,2}) \in (\mathbb{F}_{q^\ell})^{V \times V} \times (\mathbb{F}_{q^\ell})^{V \times M}$

1: $n_1 := \ell V(V+1)/2$
2: $(v_1, \ldots, v_{n_1}) \leftarrow \mathsf{RejSampPRG}(\mathsf{seed}_{\mathsf{pk}}, 2i-1, \tau_1, n_1)$ ▷ $\tau_1 = \tau_{q,\lambda}(n_1)$
3: $\bar{P}_{i,1} \leftarrow \mathsf{ExpandSymmetricMatrixVxV}(v_1, \ldots, v_{n_1})$
4: $n_2 := \ell VM$
5: $(v'_1, \ldots, v'_{n_2}) \leftarrow \mathsf{RejSampPRG}(\mathsf{seed}_{\mathsf{pk}}, 2i, \tau_2, n_2)$ ▷ $\tau_2 = \tau_{q,\lambda}(n_2)$
6: $\bar{P}_{i,2} \leftarrow \mathsf{ExpandMatrixVxM}(v'_1, \ldots, v'_{n_2})$
7: **return** $(\bar{P}_{i,1}, \bar{P}_{i,2})$

Hash (Algorithm 6) computes the hash value $\mathbf{t} \in \mathbb{F}_q^m$ of the message representative μ and the salt r using SHAKE256, where the output of SHAKE256 is converted to an element of $\mathbb{F}_q^m$ by RejSamp.

Algorithm 6 $\mathsf{Hash}(\mu, r)$

Input: message $\mu \in \mathbb{B}^{64}$ and salt $r \in \{0,1\}^\lambda$
Output: hash value $\mathbf{t} \in \mathbb{F}_q^m$

1: $(r_1, \ldots, r_{\tau_3}) \leftarrow \mathsf{SHAKE256}(\mathsf{BytesToBits}(\mu) \| r, 8\tau_3)$ ▷ $r_i \in \mathbb{B}$ ▷ $\tau_3 = \tau_{q,\lambda}(m)$
2: $\mathbf{t} \leftarrow \mathsf{RejSamp}(r_1, \ldots, r_{\tau_3}, \tau_3, m)$
3: **return** $\mathbf{t}$

Algorithm 7 $\mathsf{ExpandMatrixVxM}(v_1, \ldots, v_{n_2}) \mid_{n_2 := \ell VM}$

Input: vector $(v_1, \ldots, v_{n_2}) \in \mathbb{F}_q^{n_2}$
Output: matrix $A \in (\mathbb{F}_{q^\ell})^{V \times M}$

1: $k \leftarrow 1$
2: **for** i from 1 to V **do**
3: **for** j from 1 to M **do**
4: $A_{i,j} \leftarrow \mathsf{ExpandFql}(v_k, \ldots, v_{k+\ell-1})$
5: $k \leftarrow k + \ell$
6: **end for**
7: **end for**
8: **return** A

Algorithm 8 $\mathsf{ExpandSymmetricMatrixVxV}(v_1, \ldots, v_{n_1}) \mid_{n_1 := \ell V(V+1)/2}$

Input: vector $\mathbb{F}_q$ $(v_1, \ldots, v_{n_1}) \in \mathbb{F}_q^{n_1}$
Output: symmetric matrix $A \in (\mathbb{F}_{q^\ell})^{V \times V}$

$k \leftarrow 1$
for i from 1 to V **do**
  for j from 1 to V **do**
    if $j < i$ **then**
      $A_{i,j} \leftarrow A_{j,i}$
    else
      $A_{i,j} \leftarrow \mathsf{ExpandFql}(v_k, \ldots, v_{k+\ell-1})$
      $k \leftarrow k + \ell$
    end if
  end for
end for
return A

Algorithm 9 $\mathsf{ExpandFql}(v_1, \ldots, v_\ell)$

Input: vector $(v_1, \ldots, v_\ell) \in \mathbb{F}_q^\ell$
Output: polynomial $g \in \mathbb{F}_{q^\ell}$

$g \leftarrow \sum_{i=1}^{\ell} v_i \cdot x^{i-1}$
return g

4.7 Generation of Pseudorandom Finite Field Elements

Elements over $\mathbb{F}_q$ are sampled whithin auxiliary functions shown in Subsection 4.6. A pseudorandom byte string is generated and retrieved for every $\lceil \log_2 q \rceil$ bit to generate these elements. From each byte, $\lceil \log_2 q \rceil$ bits are extracted to retrieve a value in the range $[0, 2^{\lceil \log_2 q \rceil})$. In the range of $[0, 2^{\lceil \log_2 q \rceil})$, q is the only number that does not belong to $\mathbb{F}_q$, since q is chosen to be a Mersenne prime in this specification. Therefore, when q is obtained from the sequence of pseudorandom numbers, q should be skipped and not chosen. Also, when obtaining the element of $\mathbb{F}_q^m$, the first m numbers that are non q values should be selected.

The above procedure is called *rejection sampling*. Algorithm 10 shows the algorithm $\mathsf{RejSamp}$ to obtain a vector over $\mathbb{F}_q$ from a byte string $\mathbb{B}^\tau$, while $\mathsf{RejSampPRG}$ given in Algorithm 11 generates a pseudorandom byte string using a pseudorandom generator PRG (see Subsection 4.8) and converts the byte string into a vector over $\mathbb{F}_q$ using $\mathsf{RejSamp}$. Here, the value of $\tau = \tau_{q,\lambda}(n')$ represents the number of random selections from the set $\{0, \ldots, 2^{\lceil \log_2 q \rceil} - 1\}$ required to generate n' random elements over $\mathbb{F}_q$ with probability $1 - 2^{-\lambda}$. It is derived by

$$\tau_{q,\lambda}(n') := \min \left\{ t \in \mathbb{N} \mid P(n', t, q/2^{\lceil \log_2 q \rceil}) \leq 2^{-\lambda} \right\}.$$

$P(n', t, p)$ is the cumulative binomial distribution,

$$P(n', t, p) := \sum_{i=0}^{n'-1} \binom{t}{i} p^i (1-p)^{t-i} = I_{1-p}(t - n' + 1, n'),$$

which denotes the probability of less than n' successes in t independent Bernoulli trials of success probability p. $I_z(a, b)$ is called the regularized incomplete beta function, which is suitable for numerical evaluation.

Algorithm 10 $\mathsf{RejSamp}((r_1, \ldots, r_\tau), \tau, n')|_{\tau := \tau_{q,\lambda}(n')}$

Input: byte string $(r_1, \ldots, r_\tau) \in \mathbb{B}^\tau$, byte length τ, and vector length n'
Output: vector $\mathbb{F}_q$ $(v_1, \ldots, v_{n'}) \in \mathbb{F}_q^{n'}$

for j from 1 to τ **do**
  $v_j \leftarrow \mathsf{BitsToInteger}(\mathsf{BytesToBits}(r_j) \wedge \mathsf{IntegerToBits}(q, 8), \log_2(q+1))$
end for ▷ $\wedge$ is bitwise AND assuming q is Mersenne.
$k \leftarrow n' + 1$
while $v_k = q$ **and** $k < \tau + 1$ **do**
  $k \leftarrow k + 1$
end while
for j from 1 to n' **do**
  if $v_j = q$ **then**
    if $k < \tau + 1$ **then**
      $v_j \leftarrow v_k,\ k \leftarrow k + 1$
      while $v_k = q$ **and** $k < \tau + 1$ **do**
        $k \leftarrow k + 1$
      end while
    else
      $v_j \leftarrow 0$
    end if
  end if
end for
return $(v_1, \ldots, v_{n'})$ ▷ regard integer in $\{0, \ldots, q-1\}$ as element of $\mathbb{F}_q$.

Algorithm 11 $\mathsf{RejSampPRG}(\mathsf{seed}, i, \tau, n')$

Input: seed $\mathsf{seed} \in \{0,1\}^\lambda$, counter i, byte length τ, and vector length n'
Output: a pseudorandom sequence $(v_1, \ldots, v_{n'}) \in \mathbb{F}_q^{n'}$

$(r_1, \ldots, r_\tau) \leftarrow \mathsf{PRG}(\mathsf{seed}, i, 8\tau)$
$(v_1, \ldots, v_{n'}) \leftarrow \mathsf{RejSamp}((r_1, \ldots, r_\tau), \tau, n')$
return $(v_1, \ldots, v_{n'})$.

4.8 Pseudorandom Generator

Let PRG be a pseudorandom generator that takes a seed seed, a counter i and an output bit length $8k$ as inputs and outputs k-byte string, where the seed size equals to the security parameter λ. We offer two variants for the PRG, *SHAKE* [FIP15] and *AES counter mode* [FIP01].

SHAKE Option $\mathsf{PRG}(\mathsf{seed}, i, 8k)$ outputs $\mathsf{out} \in \mathbb{B}^k$ computed as:

$$\mathsf{out} \leftarrow \mathsf{SHAKE}(\mathsf{seed} \| \mathsf{IntegerToBits}(i-1, 16), 8k),$$

where $\mathsf{SHAKE} = \mathsf{SHAKE128}$ for the security level I and $\mathsf{SHAKE} = \mathsf{SHAKE256}$ for the securiy level III and V.

AES Option $\mathsf{PRG}(\mathsf{seed}, i, 8k)$ outputs $\mathsf{out} \in \mathbb{B}^k$ computed as:

1. $b \leftarrow \mathsf{NULL}$
2. **for** j from 1 to $\lceil k/16 \rceil$:

   $x \leftarrow \mathsf{IntegerToBits}(i-1, 64) \| \mathsf{IntegerToBits}(j-1, 64)$

   $b \leftarrow b \| \mathsf{AES}(\mathsf{seed}, x)$
3. $\mathsf{out} \leftarrow \mathsf{Trunc}_k(b)$

AES is set to AES128, AES196, and AES256 for the security levels I, III, and V, respectively.

4.9 Note on Basic Linear Algebra

Like other UOV schemes, QR-UOV also requires solving a system of linear equations to generate a signature. A tremendous amount of research exists on algorithms for solving linear equations, including well-known constant-time implementations[CKY21, BCH+23]. Although we did not use such an algorithm in our reference implementation, it should be employed for critical applications.

5 Performance Analysis

Subsection 5.1 provides the performance analysis of QR-UOV on the NIST reference platform, and Subsection 5.2 provides the analysis on other platforms.

5.1 Performance on the NIST Reference Platform

Tables 6 and 7 show the timing data for optimized implementations, and Tables 8 and 9 show the timing data for reference implementations. The implementations of Tables 6 and 8 take the AES option for the pseudorandom generator, while those of Tables 7 and 9 take the SHAKE option. These implementations are written in C and use neither inline assembly nor intrinsics for special

processor instructions, but the compiler is not restricted from outputting such instructions. The optimized versions ignore the 32-bit environment. The experimental environment is as follows.

Processor: AMD EPYC 9654P
Clock Speed: Boost Clock : Up to 3.7GHz, Base Clock: 2.4GHz
Memory: 32GB (16GB RDIMM, 4800MT/s, Single Rank)
Operating System: Linux 5.15.0-112-generic, gcc version 11.4.0
Linux Distribution: Ubuntu 22.04.5 LTS
Compiler: gcc version 11.4.0
Library: OpenSSL 3.0.2 15 Mar 2022
Measured by: supercop-20221122

Table 6: Timing data for C optimized code with AES PRG (Mcycles)

SL	(q, v, m, ℓ)	KeyGen	Sign	Verify
I	$(\mathbf{127}, \mathbf{156}, \mathbf{54}, \mathbf{3})$	**21.491**	**1.636**	**1.341**
	$(7, 740, 100, 10)$	115.025	33.986	29.850
	$(31, 165, 60, 3)$	30.483	2.359	1.984
	$(31, 600, 70, 10)$	39.790	11.084	8.943
III	$(\mathbf{127}, \mathbf{228}, \mathbf{78}, \mathbf{3})$	**95.017**	**4.700**	**3.883**
	$(7, 1100, 140, 10)$	469.162	98.253	86.640
	$(31, 246, 87, 3)$	140.115	6.882	5.755
	$(31, 890, 100, 10)$	170.002	35.817	29.750
V	$(\mathbf{127}, \mathbf{306}, \mathbf{105}, \mathbf{3})$	**311.355**	**11.080**	**9.228**
	$(7, 1490, 190, 10)$	1506.767	235.123	207.230
	$(31, 324, 114, 3)$	418.352	16.020	13.515
	$(31, 1120, 120, 10)$	380.859	63.165	52.604

Table 7: Timing data for C optimized code with SHAKE PRG (Mcycles)

SL	(q, v, m, ℓ)	KeyGen	Sign	Verify
I	$(\mathbf{127}, \mathbf{156}, \mathbf{54}, \mathbf{3})$	**22.923**	**3.145**	**2.864**
	$(7, 740, 100, 10)$	125.204	44.492	40.469
	$(31, 165, 60, 3)$	32.360	4.270	3.861
	$(31, 600, 70, 10)$	45.509	16.874	14.722
III	$(\mathbf{127}, \mathbf{228}, \mathbf{78}, \mathbf{3})$	**100.160**	**9.926**	**9.079**
	$(7, 1100, 140, 10)$	519.508	148.382	137.174
	$(31, 246, 87, 3)$	147.834	14.635	13.466
	$(31, 890, 100, 10)$	190.692	56.523	50.449
V	$(\mathbf{127}, \mathbf{306}, \mathbf{105}, \mathbf{3})$	**324.808**	**24.810**	**22.727**
	$(7, 1490, 190, 10)$	1647.667	378.988	350.925
	$(31, 324, 114, 3)$	433.932	31.584	28.995
	$(31, 1120, 120, 10)$	421.961	104.666	94.098

Table 8: Timing data for C reference code with AES PRG (Mcycles)

SL	(q, v, m, ℓ)	KeyGen	Sign	Verify
I	$(\mathbf{127}, \mathbf{156}, \mathbf{54}, \mathbf{3})$	**17.402**	**2.233**	**1.695**
	$(7, 740, 100, 10)$	198.527	37.963	37.817
	$(31, 165, 60, 3)$	25.841	3.001	2.226
	$(31, 600, 70, 10)$	75.705	15.669	15.423
III	$(\mathbf{127}, \mathbf{228}, \mathbf{78}, \mathbf{3})$	**62.897**	**5.935**	**4.442**
	$(7, 1100, 140, 10)$	837.812	118.583	113.917
	$(31, 246, 87, 3)$	123.594	9.701	7.204
	$(31, 890, 100, 10)$	540.562	70.899	66.663
V	$(\mathbf{127}, \mathbf{306}, \mathbf{105}, \mathbf{3})$	**214.724**	**15.524**	**10.966**
	$(7, 1490, 190, 10)$	2931.010	319.581	292.849
	$(31, 324, 114, 3)$	279.161	19.549	13.868
	$(31, 1120, 120, 10)$	593.219	80.677	81.371

Table 9: Timing data for C reference code with SHAKE PRG (Mcycles)

SL	(q, v, m, ℓ)	KeyGen	Sign	Verify
I	$(\mathbf{127}, \mathbf{156}, \mathbf{54}, \mathbf{3})$	**18.646**	**3.573**	**3.025**
	$(7, 740, 100, 10)$	213.867	53.300	53.099
	$(31, 165, 60, 3)$	27.746	4.972	4.235
	$(31, 600, 70, 10)$	82.319	22.397	22.012
III	$(\mathbf{127}, \mathbf{228}, \mathbf{78}, \mathbf{3})$	**68.168**	**11.279**	**9.731**
	$(7, 1100, 140, 10)$	901.254	182.183	177.605
	$(31, 246, 87, 3)$	135.621	16.928	14.243
	$(31, 890, 100, 10)$	563.553	94.408	89.261
V	$(\mathbf{127}, \mathbf{306}, \mathbf{105}, \mathbf{3})$	**226.526**	**27.091**	**22.513**
	$(7, 1490, 190, 10)$	3253.714	458.374	430.449
	$(31, 324, 114, 3)$	296.113	36.755	30.987
	$(31, 1120, 120, 10)$	636.598	125.017	124.697

5.2 Performance on Other Platforms

Tables 10 and 11 show the timing data for the C implementations with the AVX2 intrinsics, and Table 12 and 13 do that for the AVX512. The experimental environment is the same as described in Section 5.1.

Table 10: Timing data for C code with AVX2 intrinsics and AES PRG (Mcycles)

SL	(q, v, m, ℓ)	KeyGen	Sign	Verify
I	$(\mathbf{127}, \mathbf{156}, \mathbf{54}, \mathbf{3})$	**8.587**	**1.586**	**1.224**
	$(7, 740, 100, 10)$	77.115	26.924	25.242
	$(31, 165, 60, 3)$	11.929	2.230	1.715
	$(31, 600, 70, 10)$	28.855	9.518	9.484
III	$(\mathbf{127}, \mathbf{228}, \mathbf{78}, \mathbf{3})$	**32.311**	**4.599**	**3.410**
	$(7, 1100, 140, 10)$	249.400	79.160	72.424
	$(31, 246, 87, 3)$	46.554	6.679	4.991
	$(31, 890, 100, 10)$	86.827	25.656	24.592
V	$(\mathbf{127}, \mathbf{306}, \mathbf{105}, \mathbf{3})$	**95.579**	**10.504**	**7.463**
	$(7, 1490, 190, 10)$	721.608	200.626	181.458
	$(31, 324, 114, 3)$	127.786	15.577	11.624
	$(31, 1120, 120, 10)$	168.875	47.262	44.696

Table 11: Timing data for C code with AVX2 intrinsics and SHAKE PRG (Mcycles)

SL	(q, v, m, ℓ)	KeyGen	Sign	Verify
I	$(\mathbf{127}, \mathbf{156}, \mathbf{54}, \mathbf{3})$	**10.077**	**3.118**	**2.741**
	$(7, 740, 100, 10)$	95.983	45.832	44.330
	$(31, 165, 60, 3)$	13.739	4.083	3.550
	$(31, 600, 70, 10)$	34.704	15.449	15.368
III	$(\mathbf{127}, \mathbf{228}, \mathbf{78}, \mathbf{3})$	**37.508**	**9.861**	**8.678**
	$(7, 1100, 140, 10)$	305.856	135.653	128.685
	$(31, 246, 87, 3)$	54.579	14.785	13.052
	$(31, 890, 100, 10)$	109.697	48.765	47.599
V	$(\mathbf{127}, \mathbf{306}, \mathbf{105}, \mathbf{3})$	**109.125**	**24.067**	**20.992**
	$(7, 1490, 190, 10)$	844.478	322.614	303.588
	$(31, 324, 114, 3)$	142.099	30.352	26.327
	$(31, 1120, 120, 10)$	212.364	91.098	87.995

Table 12: Timing data for C code with AVX512 intrinsics and AES PRG (Mcycles)

SL	(q, v, m, ℓ)	KeyGen	Sign	Verify
I	$(\mathbf{127}, \mathbf{156}, \mathbf{54}, \mathbf{3})$	**8.542**	**1.597**	**1.226**
	$(7, 740, 100, 10)$	79.799	27.868	25.922
	$(31, 165, 60, 3)$	11.842	2.237	1.718
	$(31, 600, 70, 10)$	25.602	9.077	9.191
III	$(\mathbf{127}, \mathbf{228}, \mathbf{78}, \mathbf{3})$	**31.643**	**4.621**	**3.424**
	$(7, 1100, 140, 10)$	246.281	78.941	72.336
	$(31, 246, 87, 3)$	45.758	6.738	5.033
	$(31, 890, 100, 10)$	88.485	26.668	25.560
V	$(\mathbf{127}, \mathbf{306}, \mathbf{105}, \mathbf{3})$	**88.518**	**10.289**	**7.334**
	$(7, 1490, 190, 10)$	693.961	193.768	175.021
	$(31, 324, 114, 3)$	118.992	15.289	11.431
	$(31, 1120, 120, 10)$	164.299	47.124	44.623

Table 13: Timing data for C code with AVX512 intrinsics and SHAKE PRG (Mcycles)

SL	(q, v, m, ℓ)	KeyGen	Sign	Verify
I	$(\mathbf{127}, \mathbf{156}, \mathbf{54}, \mathbf{3})$	**9.985**	**3.130**	**2.748**
	$(7, 740, 100, 10)$	98.667	46.819	44.924
	$(31, 165, 60, 3)$	13.794	4.222	3.691
	$(31, 600, 70, 10)$	31.841	15.010	15.113
III	$(\mathbf{127}, \mathbf{228}, \mathbf{78}, \mathbf{3})$	**36.717**	**9.824**	**8.604**
	$(7, 1100, 140, 10)$	302.556	134.929	128.632
	$(31, 246, 87, 3)$	53.540	14.469	12.736
	$(31, 890, 100, 10)$	111.722	49.818	48.728
V	$(\mathbf{127}, \mathbf{306}, \mathbf{105}, \mathbf{3})$	**102.334**	**23.880**	**20.950**
	$(7, 1490, 190, 10)$	847.914	347.789	329.495
	$(31, 324, 114, 3)$	133.457	30.072	26.157
	$(31, 1120, 120, 10)$	207.629	90.682	87.848

6 Expected Security Strength

In this section, we begin by introducing the underlying problems and the security definitions in Subsection 6.1. Next, we provide our security proof in Subsection 6.2. Finally, we estimate the complexity of plausible attacks against our proposed parameter sets in Subsection 6.3. See Section 7 for the details of each attack.

6.1 Underlying Problems and Security Definitions

We first introduce two problems for the security proof of QR-UOV as follows:

Definition 1 (UOV problem)**.** We let $\mathsf{MQ}_{q,n,m}$ the set of random quadratic maps with n variables and m equations over $\mathbb{F}_q$ and let $\mathsf{UOV}_{q,v,o,m}$ the set of public key maps of the plain UOV with v vinegar variables, o oil variables, and m equations over $\mathbb{F}_q$. *The UOV problem* asks to distinguish a random quadratic system from a UOV public key. If we let $\mathcal{A}$ be a UOV distinguisher algorithm, then we say the distinguishing advantage of $\mathcal{A}$ is

$$\begin{aligned}&\mathsf{Adv}^{\mathsf{UOV}}_{q,v,o,m}(\mathcal{A}) = \\ &\left|\Pr[\mathcal{A}(\mathcal{P}) = 1 \mid \mathcal{P} \leftarrow \mathsf{MQ}_{q,(v+o),m}] - \Pr[\mathcal{A}(\mathcal{P}) = 1 \mid \mathcal{P} \leftarrow \mathsf{UOV}_{q,v,o,m}]\right|.\end{aligned}$$

Definition 2 (QR-MQ problem)**.** Let f be an irreducible polynomial with $\deg f = \ell$. We then denote by $\mathsf{QR}_{q,n,m,\ell}$ the set of quadratic maps constructed as follows

$$\mathsf{QR}_{q,n,m,\ell} = \left\{ \left(\mathbf{x}^\top P_1 \mathbf{x}, \ldots, \mathbf{x}^\top P_m \mathbf{x}\right) : \mathbb{F}_q^n \to \mathbb{F}_q^m \mid P_1, \ldots, P_m \in W^{(N)} \mathcal{A}_f^{N,N} \right\},$$

where $N = n/\ell$. For a randomly chosen $\mathcal{P} \in \mathsf{QR}_{q,n,m,\ell}$ and $\mathbf{t} \in \mathbb{F}_q^m$, *the QR-MQ problem* asks to compute $\mathbf{s}$ such that $\mathcal{P}(\mathbf{s}) = \mathbf{t}$. If we let $\mathcal{A}$ be an adversary, then we say that the advantage of $\mathcal{A}$ against the QR-MQ problem is

$$\mathsf{Adv}^{\mathsf{QR\text{-}MQ}}_{q,n,m,\ell}(\mathcal{A}) = \Pr[\mathcal{P}(\mathbf{s}) = \mathbf{t} \mid \mathcal{P} \leftarrow \mathsf{QR}_{q,n,m,\ell}, \mathbf{t} \leftarrow \mathbb{F}_q^m, \mathbf{s} \leftarrow \mathcal{A}(\mathcal{P}, \mathbf{t})].$$

We prove the security of QR-UOV in Subsection 6.2, assuming the advantages against the above two problems are negligible. The first assumption is originally utilized for the security of the plain UOV, and thus seems relatively well understood. By contrast, the second assumption is inherent in QR-UOV. Therefore, assessing the hardness of the QR-MQ problem is crucial for provable security.

Subsequently, we give the definition of the EUF-CMA security, which is the standard security definition for digital signature schemes.

Definition 3 (EUF-CMA security)**.** Let $\mathcal{O}$ be a random oracle, and let $\mathcal{A}$ be an adversary. We define the advantage of $\mathcal{A}$ against the EUF-CMA game of a signature scheme $\mathsf{DSS} = (\mathsf{KeyGen}, \mathsf{Sign}^{\mathcal{O}}, \mathsf{Verify}^{\mathcal{O}})$ in the (quantum) random

oracle model as

$$\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{DSS}}(\mathcal{A}) =$$
$$\Pr[\mathrm{Verify}^{\mathcal{O}}(\mathsf{pk}, m, \sigma) = 1 \mid (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathrm{KeyGen}(), (m, \sigma) \leftarrow \mathcal{A}^{\mathcal{O}, \mathrm{Sign}^{\mathcal{O}}(\mathsf{sk}, \cdot)}(\mathsf{pk})],$$

where $\mathsf{Sign}^{\mathcal{O}}(\mathsf{sk}, \cdot)$ was not queried on input m. We say DSS is *EUF-CMA secure* if its advantage is negligible for any efficient adversary in the security parameter.

Our EUF-CMA security proof mainly depends on a result by Kosuge and Xagawa [KX24], which shows the EUF-CMA security of the modified UOV [SSH11] assuming the non-invertibility (INV) of the underlying trapdoor function. The INV is defined as a security property, where given a hard-to-invert function $\mathsf{F} : \mathcal{X} \to \mathcal{Y}$, it is computationally hard for an adversary to find a preimage $x \in \mathcal{X}$ of a random $y \in \mathcal{Y}$.

If we denote by $\mathsf{QRUOV}_{q,v,m,\ell}$ the set of public key maps of QR-UOV with parameters (q, v, m, ℓ), then the advantage of $\mathcal{A}$ against the INV game of the trapdoor function T underlying QR-UOV is given by

$$\mathsf{Adv}^{\mathsf{INV}}_{\mathsf{T}}(\mathcal{A}) = \Pr[\mathcal{P}(\mathbf{s}) = \mathbf{t} \mid \mathcal{P} \leftarrow \mathsf{QRUOV}_{q,v,m,\ell}, \mathbf{t} \leftarrow \mathbb{F}_q^m, \mathbf{s} \leftarrow \mathcal{A}(\mathcal{P}, \mathbf{t})].$$

6.2 Security Proof

The proof is divided into three parts: first, we demonstrate INV $\Rightarrow$ EUF-CMA; next, we show UOV + QR-MQ $\Rightarrow$ INV; and finally, we establish UOV + QR-MQ $\Rightarrow$ EUF-CMA by simply combining the two reductions[1].

INV $\Rightarrow$ EUF-CMA Kosuge and Xagawa [KX24] showed the EUF-CMA security of the modified UOV signature scheme.

Lemma 2 (Proposition 4 in the ePrint version of [KX24], INV $\Rightarrow$ EUF-CMA (Modified UOV Signature))**.** *For any quantum EUF-CMA adversary $\mathcal{A}_{\mathsf{cma}}$ of the modified UOV mUOV issuing at most q_{sign} classical queries to the signing oracle and q_{qro} (quantum) random oracle queries to H, there exist an INV adversary $\mathcal{B}_{\mathsf{inv}}$ of the trapdoor function T such that*

$$\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{mUOV}}(\mathcal{A}_{\mathsf{cma}}) \leq (2q_{\mathsf{qro}} + 1)^2\, \mathsf{Adv}^{\mathsf{INV}}_{\mathsf{T}}(\mathcal{B}_{\mathsf{inv}}) + \frac{3}{2} q'_{\mathsf{sign}} \sqrt{\frac{q'_{\mathsf{sign}} + q_{\mathsf{qro}} + 1}{2^\lambda}}$$
$$+ 2\,(q_{\mathsf{qro}} + 2) \sqrt{\frac{q'_{\mathsf{sign}} - q_{\mathsf{sign}}}{2^\lambda}},$$

where q'_{sign} is a bound on the total number of queries to H in all the signing queries and the running time of $\mathcal{B}_{\mathsf{inv}}$ is about that of $\mathcal{A}_{\mathsf{cma}}$.

[1] X $\Rightarrow$ Y denotes a reduction from X to Y.

Due to the usage of the pseudorandom generator with the rejection sampling, Lemma 2 cannot be directly applied to QR-UOV, necessitating modifications to parts of the proof. By defining a random function modeling $\mathsf{SHAKE256}$ as $\mathsf{H} : \mathbb{B}^{64} \times \{0,1\}^{\lambda} \to \mathbb{B}^{\tau_3}$, and PRG as $\mathsf{H}_{\mathsf{prg}} : \{0,1\}^{\lambda} \times [m] \to \mathbb{B}^{\tau_1}$[2][3], we establish the following lemma.

Lemma 3 (INV $\Rightarrow$ EUF-CMA (QR-UOV))**.** *For any quantum EUF-CMA adversary $\mathcal{A}_{\mathsf{cma}}$ of QR-UOV issuing at most q_{sign} classical queries to the signing oracle and q_{qro} (quantum) random oracle queries to H, there exists an INV adversary $\mathcal{B}_{\mathsf{inv}}$ of the trapdoor function T such that*

$$\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{QR\text{-}UOV}}(\mathcal{A}_{\mathsf{cma}}) \leq (2q_{\mathsf{qro}}+1)^2\,\mathsf{Adv}^{\mathsf{INV}}_{\mathsf{T}}(\mathcal{B}_{\mathsf{inv}}) + \frac{3}{2}q'_{\mathsf{sign}}\sqrt{\frac{q'_{\mathsf{sign}}+q_{\mathsf{qro}}+1}{2^{\lambda}}} + 2\,(q_{\mathsf{qro}}+2)\sqrt{\frac{q'_{\mathsf{sign}}-q_{\mathsf{sign}}}{2^{\lambda}}} + \frac{q'_{\mathsf{sign}}+m+1}{2^{\lambda}},$$

where q'_{sign} is a bound on the total number of queries to H in all the signing queries and the running time of $\mathcal{B}_{\mathsf{inv}}$ is about that of $\mathcal{A}_{\mathsf{cma}}$.

Proof. We divide the proof into three parts, identify the parts requiring considerations specific to QR-UOV, and rewrite them accordingly. Note that a major difference from the modified UOV signature lies in the random number generation and the usage of rejection sampling. In QR-UOV, random number generation is performed using the pseudorandom generator PRG, and its output is converted into a vector over $\mathbb{F}_q$ through the rejection sampling $\mathsf{RejSamp}$.

Before the other modifications, we first modify the game so that it aborts if $\mathsf{RejSamp}$ fails to generate a random vector over $\mathbb{F}_q$. Since the parameters are set so that the probability of failure in $\mathsf{RejSamp}$ is bounded by $2^{-\lambda}$, the product of $2^{-\lambda}$ and the number of $\mathsf{RejSamp}$ calls can bound the advantage gap.

Next, we modify the game to enable the simulation of the signing oracle. To clarify how the simulation of the signing oracle is required, we provide an overview of the proof of Lemma 2.

Step 1: Using tight adaptive reprogramming [GHHM21], the signing oracle is modified to repeatedly reprogram the randomly chosen $(r, \mathbf{t})$ satisfy $\mathsf{Hash}(\mu, r) = \mathbf{t}$, until $\mathbf{t}$ is successfully inverted.

Step 2: The reprogramming for $(r, \mathbf{t})$ resulting in inversion failure is canceled. Here, semi-classical O2H [AHU19] is used to ensure that the adversary cannot query H with (μ, r), where r was used during the failed inversion. This ensures that canceling the reprogramming does not affect the adversary's view, making this modification feasible.

[2]The assumption of the random oracle model for PRG instantiated by SHAKE or AES-CTR is justified by the indifferentiability of these functions from the random function [NAB+20].

[3]During the execution of $\mathsf{Expand}_{\mathsf{sk}}$, the output length of PRG is τ_2; therefore, the output of $\mathsf{H}_{\mathsf{prg}}$ must be truncated accordingly.

Step 3: To simulate signatures without using the private key, a random preimage $\mathbf{s}$ is selected, and reprogramming is performed to ensure $\mathsf{H}(\mu, r) = \mathcal{P}(\mathbf{s})$, outputting $(r, \mathbf{s})$ as the signature. This reprogramming ensures that $(r, \mathbf{s})$ is recognized as a valid signature.

For QR-UOV, which uses the rejection sampling on the output of the hash function, there is a mismatch between the output of H (i.e., $\mathbb{B}^{\tau_3}$) and $\mathcal{P}(\mathbf{s}) \in \mathbb{F}_q^m$. We modify the proof for the modified UOV signature to fit QR-UOV as follows.

Step 1: Random values are sampled from $\mathbb{B}^{\tau_3}$ and used as the output of H during reprogramming.

Step 2: Reprogramming executed during inversion failure is canceled, as in the original proof. Then, the signing oracle is performed as follows.

1. Repeat random selection of $\mathbf{r}$ until the inversion of $\mathbf{t} = \mathsf{RejSamp}(\mathbf{r})$ becomes feasible and let $\mathbf{s}$ be the preimage of $\mathbf{t}$ ($\mathbf{t} = \mathcal{P}(\mathbf{s})$ holds.).
2. Reprogram H such that $\mathsf{H}(\mu, r) = \mathbf{r}$ holds, where r is randomly chosen.
3. Outputs $(r, \mathbf{s})$.

Step 3: When simulating the signing oracle, the following steps are performed:

1. Choose $\mathbf{s}$ randomly from $\mathbb{F}_q^n$ and let $\mathbf{t} = \mathcal{P}(\mathbf{s})$.
2. Select $\mathbf{r}$ randomly from $\mathcal{R}_{\mathbf{t}} = \{\mathbf{r} \in \mathbb{B}^{\tau_3} : \mathsf{RejSamp}(\mathbf{r}) = \mathbf{t}\}$.
3. Reprogram H such that $\mathsf{H}(\mu, r) = \mathbf{r}$ holds, where r is randomly chosen.
4. Output $(r, \mathbf{s})$.

We demonstrate that the chosen $(\mathbf{r}, \mathbf{s}, \mathbf{t})$ in Step 3 follows the same distribution as those selected by the signing oracle in Step 2. As shown by Sakumoto et al. [SSH11], the preimage $\mathbf{s}$ generated through retries in UOV follows a uniform distribution; therefore, $\mathbf{s}$ in Step 3 is sampled from the same distribution as in Step 2. As a result, $\mathbf{t} = \mathcal{P}(\mathbf{s})$ in Steps 2 and 3 also follows the same distribution. Furthermore, given a specific $(\mathbf{s}, \mathbf{t})$, the conditional probability of $\mathbf{r}$ in Step 2 is $\Pr[\mathbf{r} \mid (\mathbf{s}, \mathbf{t})] = \frac{1}{|\mathcal{R}_{\mathbf{t}}|}$, as $\mathbf{r}$ is chosen uniformly until $\mathbf{t} = \mathcal{P}(\mathbf{s})$ becomes invertible. This conditional probability also holds in Step 3 since $\mathbf{r}$ is uniformly chosen from $\mathcal{R}_{\mathbf{t}}$. Hence, $(\mathbf{r}, \mathbf{s}, \mathbf{t})$ is distributed identically in both Step 2 and Step 3. Since the feasibility of randomly selecting from $\mathcal{R}_{\mathbf{t}}$ is not trivial, we provide a sampling algorithm, that is, $\mathsf{InvRejSamp}$ given in Algorithm 12. The simulator executes $\mathsf{InvRejSamp}(\mathbf{t}, \tau_3, m)$ to obtain $\mathbf{r}$ in Step 3. Due to the modifications to the EUF-CMA game, the simulation of the signing oracle has become feasible in QR-UOV.

Finally, the INV adversary $\mathcal{B}_{\mathsf{inv}}$ simulates the modified game as follows. Given the public key map $\mathcal{P}$, $\mathcal{B}_{\mathsf{inv}}$ randomly selects $\mathsf{seed}_{\mathsf{pk}}$ and provides $\mathsf{pk} = (\mathsf{seed}_{\mathsf{pk}}, \{P_{i,3}\}_{i \in [m]})$ to $\mathcal{A}_{\mathsf{cma}}$. To ensure that the adversary can recover $\mathcal{P}$ from

Algorithm 12 $\mathsf{InvRejSamp}((v_1, \ldots, v_{n'}), \tau, n')|_{\tau := \tau_{q,\lambda}(n')}$

Input: vector $(v_1, \ldots, v_{n'}) \in \mathbb{F}_q^{n'}$, vector length τ, and byte length n'
Output: byte string $(r_1, \ldots, r_\tau) \in \mathbb{B}^\tau$

1: $k \leftarrow 1$
2: **for** i from 1 to τ **do**
3: $\quad w_i \xleftarrow{\$} \{0, 1, \ldots q\}$
4: **end for**
5: **for** i from 1 to n' **do**
6: $\quad$ **while** $w_k = q$ **and** $k < \tau$ **do**
7: $\quad\quad k \leftarrow k + 1$
8: $\quad$ **end while**
9: $\quad$ **if** $k = \tau$ **then**
10: $\quad\quad$ **go to** Line 1 ▷ Retry the procedure since it is failed.
11: $\quad$ **end if**
12: $\quad w_k \leftarrow v_i$
13: $\quad k \leftarrow k + 1$
14: **end for**
15: **for** i from 1 to $k - 1$ **do**
16: $\quad r_i \xleftarrow{\$} \{r \in \mathbb{B} : \mathsf{BytesToBits}(r) \wedge \mathsf{IntegerToBits}(q, 8) = \mathsf{IntegerToBits}(w_i, 8)\}$
17: **end for**
18: **for** i from k to τ **do**
19: $\quad r_i \xleftarrow{\$} \mathbb{B}$
20: **end for**
21: **return** $(r_1, \ldots, r_\tau)$

pk, $\mathcal{B}_{\mathsf{inv}}$ programs $\mathsf{H}_{\mathsf{prg}}$ such that $\mathsf{Expand}_{\mathsf{pk}}(\mathsf{seed}_{\mathsf{pk}}, i)$ outputs $(\bar{P}_{i,1}, \bar{P}_{i,2})$ contained within $\mathcal{P}$. Except for this modification, the reduction proceeds in the same manner as the proof of Lemma 2. □

UOV + QR-MQ ⇒ INV We demonstrate the INV of the underlying trapdoor function of QR-UOV. Before providing the proof, we prepare a lemma showing a bijection from quadratic maps with the QR structure over $\mathbb{F}_q$ to quadratic maps over $\mathbb{F}_{q^\ell}$. We call this transformation a *pull-back method*. The correctness of this statement is trivially derived from the existence of the function ϕ defined in Subsection 4.1.

Lemma 4. *For any irreducible polynomial f with* $\deg f = \ell$*, there exists a bijection from* $\mathsf{QR}_{q,n,m,\ell}$ *to* $\mathsf{MQ}_{q^\ell,n/\ell,m/\ell}$*. Similarly, there exists a bijection from* $\mathsf{QRUOV}_{q,v,m,\ell}$ *to* $\mathsf{UOV}_{q^\ell,v/\ell,m/\ell,m}$*.*

We then show the security of the INV game of QR-UOV assuming the difficulty of the UOV and QR-MQ problems.

Lemma 5 (UOV and QR-MQ $\Rightarrow$ INV)**.** *For any quantum INV adversary $\mathcal{A}_{\mathsf{inv}}$ of the trapdoor function T underlying QR-UOV, there exist adversaries $\mathcal{B}_{\mathsf{uov}}$ and $\mathcal{B}_{\mathsf{mq}}$ against the UOV problem with parameters $(q^\ell, v/\ell, o/\ell, m)$ and the QR-MQ problem with parameters $(q, (v+m), m, \ell)$, respectively, such that*

$$\mathsf{Adv}^{\mathsf{INV}}_{\mathsf{T}}(\mathcal{A}_{\mathsf{inv}}) \leq \mathsf{Adv}^{\mathsf{UOV}}_{q^\ell, v/\ell, o/\ell, m}(\mathcal{B}_{\mathsf{uov}}) + \mathsf{Adv}^{\mathsf{QR\text{-}MQ}}_{q,(v+m),m,\ell}(\mathcal{B}_{\mathsf{mq}}),$$

where the running times of $\mathcal{B}_{\mathsf{uov}}$ and $\mathcal{B}_{\mathsf{mq}}$ are about that of $\mathcal{A}_{\mathsf{inv}}$.

Proof. We prove the above statement by the following sequence of games.

1. Let Game_0 be $\mathcal{A}_{\mathsf{inv}}$'s INV game against the trapdoor function of QR-UOV, where we have $\Pr[\mathsf{Game}_0() = 1] = \mathsf{Adv}^{\mathsf{INV}}_{\mathsf{T}}(\mathcal{A}_{\mathsf{inv}})$.

2. Let Game_1 be the same as Game_0 except that the challenger chooses a random $\mathcal{P} \in \mathsf{QR}_{q,(v+m),m,\ell}$ without the UOV structure. We can construct an adversary $\mathcal{B}_{\mathsf{uov}}$ on the UOV problem with parameters $(q^\ell, v/\ell, m/\ell, m)$ as follows: when $\mathcal{B}_{\mathsf{uov}}$ is given a multivariate quadratic map $\bar{\mathcal{P}}$ with $N = (v+m)/\ell$ variables and m equations over $\mathbb{F}_{q^\ell}$, it transforms $\bar{\mathcal{P}}$ into $\mathcal{P}$ with $(v+m)$ variables and m equations over $\mathbb{F}_q$ composed of m matrices in $W^{(N)}\mathcal{A}_f^{N,N}$. This transformation is performed by the inverse of ϕ defined in Subsection 4.1. Then, $\mathcal{B}_{\mathsf{uov}}$ runs $\mathcal{A}_{\mathsf{inv}}$ on input $\mathcal{P}$. From Lemma 4, if $\mathcal{B}_{\mathsf{uov}}$ is given a $\bar{\mathcal{P}} \in \mathsf{UOV}_{q^\ell, v/\ell, m/\ell, m}$, then $\mathcal{B}^{\mathcal{A}_{\mathsf{inv}}}_{\mathsf{uov}}$ simulates Game_0, and if $\mathcal{B}_{\mathsf{uov}}$ is given a $\bar{\mathcal{P}} \in \mathsf{MQ}_{q^\ell, N, m}$, then $\mathcal{B}^{\mathcal{A}_{\mathsf{inv}}}_{\mathsf{uov}}$ simulates Game_1. Therefore, we have

$$|\Pr[\mathsf{Game}_0() = 1] - \Pr[\mathsf{Game}_1() = 1]| \leq \mathsf{Adv}^{\mathsf{UOV}}_{q^\ell, v/\ell, m/\ell, m}(\mathcal{B}_{\mathsf{uov}}).$$

3. From the construction, Game_1 is the QR-MQ game with patameters $(q, (v+m), m, \ell)$, and thus we have $\Pr[\mathsf{Game}_1() = 1] = \mathsf{Adv}^{\mathsf{QR\text{-}MQ}}_{q,(v+m),m,\ell}(\mathcal{B}_{\mathsf{mq}})$.

We can confirm the correctness of the statement by combining inequalities from the game transitions as follows:

$$\begin{aligned} & \left|\mathsf{Adv}^{\mathsf{INV}}_{\mathsf{T}}(\mathcal{A}_{\mathsf{inv}}) - \mathsf{Adv}^{\mathsf{QR\text{-}MQ}}_{q,(v+m),m,\ell}(\mathcal{B}_{\mathsf{mq}})\right| \leq \mathsf{Adv}^{\mathsf{UOV}}_{q^\ell, v/\ell, o/\ell, m}(\mathcal{B}_{\mathsf{uov}}) \\ \Leftrightarrow \quad & \mathsf{Adv}^{\mathsf{INV}}_{\mathsf{T}}(\mathcal{A}_{\mathsf{inv}}) \leq \mathsf{Adv}^{\mathsf{UOV}}_{q^\ell, v/\ell, o/\ell, m}(\mathcal{B}_{\mathsf{uov}}) + \mathsf{Adv}^{\mathsf{QR\text{-}MQ}}_{q,(v+m),m,\ell}(\mathcal{B}_{\mathsf{mq}}). \end{aligned}$$

□

UOV + QR-MQ $\Rightarrow$ EUF-CMA Combining Lemmas 3 and 5, we obtain the EUF-CMA security proof of QR-UOV.

Theorem 2 (QR-MQ+UOV $\Rightarrow$ EUF-CMA)**.** *For any quantum EUF-CMA adversary $\mathcal{A}_{\mathsf{cma}}$ of QR-UOV issuing at most q_{sign} classical queries to the signing oracle and q_{qro} (quantum) random oracle queries to H, there exist adversaries*

$\mathcal{B}_{\mathsf{uov}}$ and $\mathcal{B}_{\mathsf{mq}}$ against the UOV problem with parameters $(q^\ell, v/\ell, o/\ell, m)$ and the QR-MQ problem with parameters $(q, (v+m), m, \ell)$, respectively, such that

$$\begin{aligned}\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{QR\text{-}UOV}}(\mathcal{A}_{\mathsf{cma}}) \leq (2q_{\mathsf{qro}}+1)^2 \left(\mathsf{Adv}^{\mathsf{UOV}}_{q^\ell, v/\ell, o/\ell, m}(\mathcal{B}_{\mathsf{uov}}) + \mathsf{Adv}^{\mathsf{QR\text{-}MQ}}_{q,(v+m),m,\ell}(\mathcal{B}_{\mathsf{mq}})\right) \\ + \frac{3}{2} q'_{\mathsf{sign}} \sqrt{\frac{q'_{\mathsf{sign}} + q_{\mathsf{qro}} + 1}{2^\lambda}} + 2\left(q_{\mathsf{qro}} + 2\right) \sqrt{\frac{q'_{\mathsf{sign}} - q_{\mathsf{sign}}}{2^\lambda}} \\ + \frac{q'_{\mathsf{sign}} + m + 1}{2^\lambda},\end{aligned}$$

where q'_{sign} is a bound on the total number of queries to H in all the signing queries and the running times of $\mathcal{B}_{\mathsf{uov}}$ and $\mathcal{B}_{\mathsf{mq}}$ are about that of $\mathcal{A}_{\mathsf{cma}}$.

Remark 1. In [CDP23], Chatterjee et al. claimed that there exist some issues in the EUF-CMA security proof in the ROM given by Sakumoto et al. [SSH11]. QR-UOV uses the modification of the signature generations for the security proof used in the proof by Sakumoto et al. However, our security reduction is based on the proof by Kosuge and Xagawa [KX24] that made corrections to the proof of Sakumoto et al. Thus, the result by Chatterjee et al. does not affect our security proof.

6.3 Security Estimation of Proposed Parameters

In this subsection, we confirm that the proposed parameters in Subsection 4.5 satisfy security levels I, III, and V of the NIST PQC project by estimating the complexity of relevant attacks on QR-UOV described in Section 7.

We begin by describing the criteria used to evaluate the security levels. The security levels I, III, and V indicate that a classical attacker needs more than 2^{143}, 2^{207}, and 2^{272} classical gates to break the parameters, whereas a quantum attacker needs more than 2^{61}, 2^{125}, and 2^{189} quantum gates, respectively, from the call for additional digital signature schemes [NIS22]. The number of gates required for each attack can be computed using

$$\#\,\text{gates} = \#\,\text{field multiplications} \cdot \left(2 \cdot (\log_2 q)^2 + \log_2 q\right).$$

Next, we provide a list of the relevant attacks on QR-UOV along with their associated complexities. As stated in Subsection 6.2, the EUF-CMA security of QR-UOV can be reduced to the difficulty of the QR-MQ problem and the UOV problems with parameters $(q^\ell, v/\ell, m/\ell, m)$, namely UOV with v/ℓ vinegar variables, m/ℓ oil variables, and m equations over $\mathbb{F}_{q^\ell}$. We here consider the hybrid approach [BFP09] with Wiedemann XL (WXL) [YCBC07] and the polynomial XL (PXL) [FK24] as attackers against the (QR-)MQ problem. (See Subsection 7.2 for the relationship between the difficulty of the QR-MQ problem and the plain MQ problem.) Also, we consider the Kipnis-Shamir [KS98], reconciliation [DYC+08], intersection [Beu21], and rectangular MinRank [Beu21] attacks as attackers against the UOV problem with parameters $(q^\ell, v/\ell, m/\ell, m)$. In

addition to these attacks, we consider the claw finding attack. The complexity of these attacks is evaluated in Table 14.

Furthermore, key recovery attacks on QR-UOV can be performed by treating QR-UOV as plain UOV with parameters (q, v, m, m) without considering its structure. In Table 15, we estimate the complexity of three key recovery attacks, the Kipnis-Shamir, reconciliation, and intersection attacks, on the plain UOV with parameters (q, v, m, m) for each proposed parameter set. (See Subsection 7.3.4 for the reason that the rectangular MinRank is not applicable to the plain UOV with parameters (q, v, m, m).) Then, one can confirm that for each attack, the complexity of key recovery attacks on the plain UOV with (q, v, m, m) is larger than or equal to the one on the plain UOV with $(q^\ell, v/\ell, m/\ell, m)$.

Finally, we assess the security levels of the QR-UOV parameter sets. In Table 14 and 15, for each parameter set, the upper entry shows the number of classical gates, whereas the lower entry shows the number of quantum gates. Furthermore, the values in bold indicate the complexity of the best attack against each parameter set. These tables show that the proposed parameters satisfy the requirements for each security level I, III, and V. One can confirm that our proposed parameters for security levels I and III also satisfy security levels II and IV, respectively.

Table 14: The complexity of the classical and quantum attacks, the claw finding attack, and the Hashimoto's method with WXL (WXL) and PXL on the MQ problem, and the Kipnis-Shamir (KS), reconciliation (Recon.), intersection (Inter.), and rectangular MinRank (RM) attacks on UOV($q^\ell, v/\ell, m/\ell, m$) against the proposed parameter sets

SL	(q, v, m, ℓ)	$\log_2$(#gates) in classical (upper)/quantum (lower)						
		Claw	WXL	PXL	KS	Recon.	Inter.	RM
I	$(\mathbf{127}, \mathbf{156}, \mathbf{54}, \mathbf{3})$	201	160	**150**	718	164	460	158
		201	132	**128**	372	164	282	158
	$(7, 740, 100, 10)$	155	184	201	1793	**148**	1641	157
		155	**105**	136	908	148	869	157
	$(31, 165, 60, 3)$	162	163	152	531	**151**	343	153
		162	**117**	127	279	151	224	153
	$(31, 600, 70, 10)$	187	164	162	2600	**152**	2415	157
		187	**111**	134	1312	152	1251	157
III	$(\mathbf{127}, \mathbf{228}, \mathbf{78}, \mathbf{3})$	286	222	**211**	1056	231	653	219
		286	182	**180**	542	227	390	219
	$(7, 1100, 140, 10)$	**211**	262	283	2693	219	2452	229
		211	**152**	188	1359	219	1287	229
	$(31, 246, 87, 3)$	229	226	**215**	801	221	508	220
		229	**171**	180	415	220	323	220
	$(31, 890, 100, 10)$	261	235	232	3890	**216**	3585	223
		261	**164**	193	1958	216	1851	223
V	$(\mathbf{127}, \mathbf{306}, \mathbf{105}, \mathbf{3})$	380	288	279	1414	291	851	**277**
		380	**237**	238	722	289	505	277
	$(7, 1490, 190, 10)$	281	354	384	3649	**277**	3291	287
		281	**213**	253	1838	277	1719	287
	$(31, 324, 114, 3)$	296	286	**279**	1055	283	658	**279**
		296	**218**	232	543	280	413	279
	$(31, 1120, 120, 10)$	311	280	**275**	4931	283	4540	290
		311	**197**	230	2479	283	2335	290

Table 15: The complexity of the classical and quantum key recovery attacks on UOV(q, v, m, m), the Kipnis-Shamir (KS), reconciliation (Recon.), and intersection (Inter.) attacks, against the proposed parameter sets

SL	(q, v, m, ℓ)	$\log_2$(#gates) in classical (upper)/quantum (lower)		
		KS	Recon.	Inter.
I	$(\mathbf{127}, \mathbf{156}, \mathbf{54}, \mathbf{3})$	736	421	772
		383	348	527
	$(7, 740, 100, 10)$	1825	1350	2089
		928	891	1140
	$(31, 165, 60, 3)$	545	408	666
		287	314	450
	$(31, 600, 70, 10)$	2651	1368	2788
		1341	1038	1528
III	$(\mathbf{127}, \mathbf{228}, \mathbf{78}, \mathbf{3})$	1073	596	1115
		553	491	753
	$(7, 1100, 140, 10)$	2725	1980	3088
		1379	1302	1673
	$(31, 246, 87, 3)$	814	591	980
		423	451	653
	$(31, 890, 100, 10)$	3941	2001	4129
		1987	1516	2250
V	$(\mathbf{127}, \mathbf{306}, \mathbf{105}, \mathbf{3})$	1431	782	1477
		732	644	997
	$(7, 1490, 190, 10)$	3681	2661	4167
		1858	1745	2253
	$(31, 324, 114, 3)$	1069	763	1278
		551	581	848
	$(31, 1120, 120, 10)$	4983	2502	5201
		2508	1893	2820

7 Analysis of Attacks against QR-UOV

This section describes relevant attacks on QR-UOV. The rest of this section is organized as follows. Subsection 7.1 explains the claw finding attack which finds a collision point. Subsection 7.2 explains the direct attack which directly finds a signature for a given message. Subsection 7.3 recalls known key recovery attacks on the plain UOV. Subsection 7.4 discusses the relation between the irreducibility of the polynomial f constructing the quotient ring of QR-UOV and its security. Subsection 7.5 describes a way to transform the public key matrices of QR-UOV into the extension field $\mathbb{F}_{q^\ell}$. Subsection 7.6 provides a way of transforming the public key matrices over the base field $\mathbb{F}_q$ using the quotient ring structure.

7.1 Claw Finding Attack

This subsection considers the claw finding attack which is also called the birthday attack. Very recently, Saarinen [Saa24] proposed a technique to improve the complexity estimation originally given in [BCC$^+$23, BCH$^+$23]. We apply this technique for the complexity estimation of claw finding attack in Subsection 6.3. Note that the reduction factor in the complexity estimation by the technique of [Saa24] is only $2/(q-1)$, and thus this technique does not weaken the proposed parameters.

For a message representative μ obtained from a messege $\mathbf{M}$, an attacker computes $\mathcal{P}(\mathbf{s}_i)$ for X inputs $\{\mathbf{s}_i\}_{i\in[X]}$ and $\mathsf{Hash}(\mu, r_j)$ for Y salts $\{r_j\}_{j\in[Y]}$. If $XY = q^m$, then there is a collision with probability $\approx 1 - e^{-1}$, and the attacker can output the signature $(r_j, \mathbf{s}_i)$ for the massage $\mathbf{M}$. Further, as mentioned in [Saa24], we can revise this attack by utilizing the property that $\mathcal{P}(c\mathbf{s}_i) = c^2\mathcal{P}(\mathbf{s}_i)$ for any $c \in \mathbb{F}_q$. This can reduce the image of $\mathcal{P}$ and Hash where a collision must occur by a factor of $2/(q-1)$ in the case of odd q. Following [BCC$^+$23, BCH$^+$23], we estimate the number of gates required for this attack considering the cost of multiplication and addition in $\mathbb{F}_q$ as follows:

$$2\left(\frac{2}{q-1}\cdot q^m \cdot m \cdot 2^{17} \cdot \left(2\cdot(\log_2 q)^2 + \log_2 q\right)\right)^{\frac{1}{2}}. \tag{8}$$

In this estimation, we suppose that computing Hash has a bit cost of 2^{17} and applying a fast enumeration algorithm [FT23] in $\mathbb{F}_q$ to evaluate $\mathcal{P}$ successively.

For the quantum claw finding attack, it is shown that attackers with limited time will prefer the classical attacks in [JS19]. Thus, in Subsection 6.3, we estimated the complexity of the quantum claw finding attack by equation (8).

7.2 Direct Attack

This subsection describes the direct attack, which can be defined as follows: Given a public key $\mathcal{P}$ with n variables and m equations over $\mathbb{F}_q$ and a target $\mathbf{t} \in \mathbb{F}_q^m$, the direct attack tries to solve the MQ system $\mathcal{P}(\mathbf{s}) = \mathbf{t}$ to find a signature $\mathbf{s}$.

We first explain the complexity of solving the MQ system with $n \leq m$ (overdetermined), and then show a way of reducing the MQ system with $n > m$ (underdetermined) into a smaller overdetermined system. Finally, we discuss the difficulty of the QR-MQ problem to which the security of QR-UOV is reduced in Theorem 2.

Overdetermined case We provide a way of estimating the complexity of solving the overdetermined MQ system, since an underdetermined system can be transformed into an overdetermined system by specifying $n - m$ variables without disturbing the existence of a solution with high probability. One of the best-known approaches for algebraically solving the quadratic system is the hybrid approach [BFP09], which randomly guesses k with $0 \leq k \leq n$ variables before applying an MQ solver such as F4 [Fau99], F5 [Fau02], and XL [CKPS00]. The guessing process is repeated until a solution is obtained. The complexity of this approach with the Wiedemann XL (WXL) [YCBC07], which is a variant of XL, for a classical adversary is given by

$$\min_k \left(O\left(q^k \cdot 3 \cdot \binom{n-k+2}{2} \cdot \binom{d_{reg}+n-k}{d_{reg}}^2 \right) \right), \tag{9}$$

where d_{reg} denotes the degree of regularity of the system. The degree of regularity d_{reg} for a certain class of polynomial systems called *semi-regular systems* [Bar04, BFS03, BFSY05] is known to be the degree of the first non-positive term in the following series [BFSY05, YC05]:

$$\frac{\left(1-z^2\right)^m}{(1-z)^{n-k+1}}. \tag{10}$$

Empirically, the public key system of UOV is considered to be a semi-regular system. Therefore, this series (10) can be used to estimate the degree of regularity. By using Grover's algorithm [Gro96], the complexity of a quantum direct attack is estimated as

$$\min_k \left(O\left(q^{k/2} \cdot 3 \cdot \binom{n-k+2}{2} \cdot \binom{d_{reg}+n-k}{d_{reg}}^2 \right) \right). \tag{11}$$

Furthermore, a new variant of the hybrid approach with XL, which is called polynomial XL (PXL), was proposed in 2021 [FK24]. This PXL reduces the complexity by performing Gaussian elimination on the matrix over a polynomial ring, and the complexity of PXL for classical and quantum attackers is given by

$$O\left(k^2 \cdot \alpha \cdot \tbinom{n-k+d_{reg}}{d_{reg}} \cdot \tbinom{n+d_{reg}}{d_{reg}} + q^k \cdot \left(\alpha^2 \cdot \tbinom{k+d_{reg}}{d_{reg}} + \alpha^\omega\right)\right)$$
$$\text{and} \quad O\left(k^2 \cdot \alpha \cdot \tbinom{n-k+d_{reg}}{d_{reg}} \cdot \tbinom{n+d_{reg}}{d_{reg}} + q^{\frac{k}{2}} \cdot \left(\alpha^2 \cdot \tbinom{k+d_{reg}}{d_{reg}} + \alpha^\omega\right)\right),$$

respectively, where k is the number of guessed variables and $\omega = 2.37$ is the constant in the complexity of matrix multiplication. Furthermore, α is given by

$$\alpha = \sum_{d=0}^{d_{reg}} \max\left\{\text{coeff}\left((1-z)^{m-(n-k)}(1+z)^m, z^d\right), 0\right\},$$

where $\text{coeff}(f, t)$ denotes the coefficient of a term t in a polynomial f.

Underdetermined case We explain a way of solving the underdetermined MQ system efficiently. Thomae and Wolf [TW12] proposed a technique for reducing the number of variables and equations when $n > m$. For $\alpha = \lfloor \frac{n}{m} \rfloor - 1$, they reduce the $(n - m + \alpha)$ variables and α equations and thereby obtain a quadratic system with $m - \alpha$ variables and equations. Furue et al. [FNT21] improved Thomae and Wolf's technique supposing to guess values of k variables as in the hybrid approach, and Hashimoto [Has23] proposed two techniques by modifying the technique of Furue et al. to enhance efficiency. The complexities of Hashimoto's techniques on the MQ system can be estimated using the complexity of solving the MQ system with n variables and m equations in $\mathbb{F}_q$, denoted by $MQ(q, n, m)$. Then, the complexity of the Hashimoto's first technique is given as

$$q^k \cdot MQ(q, m - \alpha - k, m - \alpha) + (m - k) \cdot MQ(q, \alpha, \alpha),$$

under the condition $n - m + k \geq \alpha \cdot (m - k)$, and that of the Hashimoto's second technique is given as

$$q^k \cdot (MQ(q, m - \alpha - k, m - \alpha) + MQ(q, \alpha, \alpha)) + (m - \alpha - k) \cdot MQ(q, \alpha, \alpha),$$

under the condition $n - m \geq \alpha \cdot (m - k - \alpha)$. In Subsection 6.3, we confirm the security of the proposed parameters by the complexity of the hybrid approach with WXL given by equation (9) using one of these Hashimoto's techniques which has smaller complexity. Note that it is difficult to combine PXL and Hashimoto's techniques since both algorithms utilize the guessed k variables before substituting k values, and thus we apply Thomae and Wolf [TW12] technique to PXL to estimate the complexity in Subsection 6.3.

QR-MQ probelm We finally discuss the security of the QR-MQ problem. In Table 16, for a QR-UOV public key system, we compare the theoretical d_{reg} and experimental d_{reg} using the F4 algorithm. The theoretical d_{reg} is the degree of regularity of F4 as the smallest degree with a non-positive coefficient in $\left(1 - z^2\right)^m / (1 - z)^{m-k}$, assuming that the system is semi-regular. The experimental d_{reg} is the highest degree among the step degrees, where nonzero polynomials are generated in experiments of F4 using the Magma algebra system [BCP97]. In our experiment, we set m to sufficiently large values so that our computation for one parameter was performed within one day, and v is set equal to $2m$. For the public key of QR-UOV with $(v + m)$ variables and m

equations, we fix the last v variables and execute the hybrid approach by fixing k variables additionally. That is, the direct attack is executed on the system of m equations in $m-k$ variables. Table 16 shows that the degrees of regularity obtained experimentally are the same as those obtained theoretically. These results indicate that the difficulty of solving the public key system of QR-UOV is equivalent to that of solving the randomized MQ system.

Table 16: Theoretical and experimental degree of regularity of public key system of QR-UOV obtained using the Magma algebra system [BCP97].

(q, v, m, ℓ, k)	theoretical d_{reg}	experimental d_{reg}
$(7, 24, 12, 3, 0)$	13	13
$(7, 24, 12, 3, 1)$	7	7
$(7, 24, 12, 3, 2)$	6	6
$(7, 30, 15, 3, 0)$	16	16
$(7, 30, 15, 3, 1)$	8	9
$(7, 30, 15, 3, 2)$	7	7

Remark 2. There is a difference between the theoretical and experimental degrees in Table 16; however, it is not unique to QR-UOV. In the case of $(q, v, m, \ell, k) = (7, 30, 15, 3, 1)$ in Table 16, the experimental $d_{reg} = 9$ is larger than the theoretical $d_{reg} = 8$. We conduct an experiment to derive d_{reg} for a randomized quadratic system with the same parameters, and the result is $d_{reg} = 9$. This indicates that the observed difference between theoretical and experimental d_{reg} is not unique to QR-UOV.

7.3 Key Recovery Attacks on UOV

This subsection recalls some existing key recovery attacks, the Kipnis-Shamir [KS98], reconciliation [DYC+08], intersection [Beu21], and rectangular MinRank [Beu21] attacks. For the security of QR-UOV, these key recovery attacks can be performed on the following two problems:

- UOV$(q^\ell, v/\ell, m/\ell, m)$,
- UOV(q, v, m, m),

where UOV(q, v, o, m) denotes the plain UOV with v vinegar variables, o oil variables, and m equations over $\mathbb{F}_q$. The first one corresponds to one of the underlying problems of our security proof obtained by the pull-back transformation described in Lemma 4, and the second one is enabled by ignoring the quotient ring structure of QR-UOV. This subsection describes the behavior of the key recovery attacks on UOV(q, v, o, m), and thus, by substituting $(q^\ell, v/\ell, m/\ell, m)$ and (q, v, m, m) for (q, v, o, m) in the following estimations, we can obtain the complexity of the key recovery attacks on UOV$(q^\ell, v/\ell, m/\ell, m)$ and UOV(q, v, m, m), respectively.

Recall that the key recovery attacks aim to obtain the subspace $\mathcal{S}^{-1}(\mathcal{O})$ of $\mathbb{F}_q^n$, where $\mathcal{O}$ is the oil subspace defined as

$$\mathcal{O} := \left\{(0,\ldots,0,\alpha_1,\ldots,\alpha_o)^\top \mid \alpha_i \in \mathbb{F}_q\right\}.$$

7.3.1 Kipnis-Shamir Attack

The Kipnis-Shamir attack [KS98] chooses two invertible matrices W_i, W_j from the set of linear combinations of the representation matrices $P_1,\ldots,P_m$ for the public key. Then, it probabilistically recovers a part of the subspace $\mathcal{S}^{-1}(\mathcal{O})$ by computing the invariant subspace of $W_i^{-1}W_j$. The complexity of the Kipnis-Shamir attack is estimated as

$$O\left(q^{v-o-1}\cdot o^4\right).$$

Grover's algorithm [Gro96] can be used to reduce the complexity for a quantum adversary to

$$O\left(q^{\frac{v-o-1}{2}}\cdot o^4\right).$$

Then, the complexity of the Kipnis-Shamir attack for classical and quantum adversaries against $\mathrm{UOV}(q^\ell, v/\ell, m/\ell, m)$ is given as

$$O\left(q^{v-m-\ell}\cdot(m/\ell)^4\right) \quad \text{and} \quad O\left(q^{\frac{v-m-\ell}{2}}\cdot(m/\ell)^4\right).$$

Furthermore, the complexity of the Kipnis-Shamir attack for classical and quantum adversaries against $\mathrm{UOV}(q, v, m, m)$ is given as

$$O\left(q^{v-m-1}\cdot m^4\right) \quad \text{and} \quad O\left(q^{\frac{v-m-1}{2}}\cdot m^4\right).$$

7.3.2 Reconciliation Attack

The reconciliation attack [DYC+08] treats a vector y of $\mathcal{S}^{-1}(\mathcal{O})$ as variables and solves the quadratic system $y^\top P_i y = 0$ $(i \in [m])$. Here, the dimension of $\mathcal{S}^{-1}(\mathcal{O})$ is o, and thus if we impose affine constraints, we then solve a system of m equations in $n-o=v$ variables and still have a solution with high probability.

We show how to estimate the complexity of the reconciliation attack by considering the cases $v > m$ and $v < m$ separately. In the case of $v > m$, which is generally satisfied in UOV to prevent the Kipnis-Shamir attack, the system of equations $y^\top P_i y = 0$ has a large number of solutions. Therefore, to determine a solution uniquely, we need to solve the following system to find multiple vectors $y_1,\ldots,y_k$ of $\mathcal{S}^{-1}(\mathcal{O})$:

$$\begin{cases} y_j^\top P_i y_j = 0 & (i \in [m], 1 \le j \le k), \\ y_j^\top P_i y_\ell = 0 & (i \in [m], 1 \le j < \ell \le k). \end{cases}$$

We here lower bound the complexity of solving this problem by that of solving the MQ problem with v variables and equations. In the case of $v < m$, the

complexity of the reconciliation attack is estimated as that of solving a quadratic system of m equations in v variables. In both cases, we estimate the complexity of solving these problems using the hybrid approach with WXL, as given in equations (9) and (11).

Then, the complexity agianst UOV$(q^\ell, v/\ell, m/\ell, m)$ is given by

$$\min_k \left(O\left(q^{\ell \cdot k} \cdot 3 \cdot \binom{v/\ell - k + 2}{2} \cdot \binom{d_{reg} + v/\ell - k}{d_{reg}}^2 \right) \right),$$

where $0 \leq k \leq v/\ell$, since $v/\ell < m$. Furthermore, the complexity agianst UOV(q, v, m, m) is given by

$$\min_k \left(O\left(q^{k} \cdot 3 \cdot \binom{v - k + 2}{2} \cdot \binom{d_{reg} + v - k}{d_{reg}}^2 \right) \right),$$

where $0 \leq k \leq v$, since $v > m$.

7.3.3 Intersection Attack

In [Beu21], Beullens proposed a new key recovery attack against UOV, called an intersection attack. In the case of $v < 2o$, for an integer $k \geq 2$ satisfying $k < \frac{v}{v-o}$, let $L_1, \ldots, L_k$ be k invertible matrices randomly chosen from a set of linear combinations of the representation matrices $P_1, \ldots, P_m$ for the public key. This attack then solves the following equations for $\mathbf{y} \in \mathbb{F}_q^n$:

$$\begin{cases} (L_j^{-1}\mathbf{y})^\top P_i (L_j^{-1}\mathbf{y}) = 0 & (i \in [m], 1 \leq j \leq k), \\ (L_j^{-1}\mathbf{y})^\top P_i (L_\ell^{-1}\mathbf{y}) = 0 & (i \in [m], 1 \leq j < \ell \leq k). \end{cases} \tag{12}$$

Note that, for a solution $\mathbf{z}$ for this system, $\mathbf{z}$ is not a vector in $\mathcal{S}^{-1}(\mathcal{O})$, but $L_j^{-1}\mathbf{z}$ is a vector in $\mathcal{S}^{-1}(\mathcal{O})$. The solution space obtained from the above equation has $ko - (k-1)v$ dimensions. Thus, its complexity is equivalent to that of solving the quadratic system with $n - (ko - (k-1)v) = kv - (k-1)o$ variables and $\binom{k+1}{2}m - 2\binom{k}{2}$ equations owing to its linear dependency. The value of k is generally chosen such that the complexity of solving the above system takes the minimum value under the condition of $k < \frac{v}{v-o}$. On the other hand, in the case of $v \geq 2o$, which is the case of the proposed parameters of QR-UOV, the intersection attack becomes a probabilistic algorithm, which solves the system of equation (12) as $k = 2$ with n variables and $(3m - 2)$ equations and one of solutions is a target vector with a probability of approximately $q^{-v+2o-1}$. Therefore, its complexity is estimated by q^{v-2o+1} times the complexity of solving the quadratic system with n variables and $(3m-2)$ equations. We estimate the complexity of solving these problems with the complexity of the hybrid approach with WXL in equations (9) and (11).

Then, the complexity agianst UOV$(q^\ell, v/\ell, m/\ell, m)$ is given by

$$\min_k \left(O\left(q^{v-2m+\ell} \cdot q^{\ell \cdot k} \cdot 3 \cdot \binom{n/\ell - k + 2}{2} \cdot \binom{d_{reg} + n/\ell - k}{d_{reg}}^2 \right) \right),$$

where $0 \leq k \leq n/\ell$, since $v/\ell > 2m/\ell$. Furthermore, the complexity agianst $\mathrm{UOV}(q, v, m, m)$ is given by

$$\min_k \left(O\left(q^{v-2m+1} \cdot q^k \cdot 3 \cdot \binom{n-k+2}{2} \cdot \binom{d_{reg}+n-k}{d_{reg}}^2 \right) \right),$$

where $0 \leq k \leq n$, since $v > 2m$.

7.3.4 Rectangular MinRank Attack

The rectangular MinRank attack [Beu21] was originally proposed for the Rainbow scheme by Beullens, and it tries to solve a MinRank problem obtained by transforming the public key matrices. Furue and Ikematsu [FI23] showed that the rectangular MinRank attack is applicable to $\mathrm{UOV}(q^\ell, V, M, m)$ which is one of the underlying problems of the security of QR-UOV. Note that we here suppose that $(P_1, \ldots, P_m)$, $(F_1, \ldots, F_m)$, and S are matrices representing the public and private keys of UOV with parameters (q^ℓ, V, M, m).

Before describing the rectangular MinRank attack, we introduce a way of transforming sets of matrices used in the attack. Let $(G_1, \ldots, G_m)$ be a set of n-by-n matrices over $\mathbb{F}_q$, and $\mathbf{g}_i^{(j)}$ denotes the j-th column vector of G_i, namely,

$$G_i = \begin{pmatrix} \mathbf{g}_i^{(1)} & \mathbf{g}_i^{(2)} & \cdots & \mathbf{g}_i^{(n)} \end{pmatrix} \in \mathbb{F}_q^{n \times n}.$$

Then, we define the new set $(\tilde{G}_1, \ldots, \tilde{G}_n)$ of n-by-m matrices as follows:

$$\begin{aligned} \tilde{G}_1 &:= \begin{pmatrix} \mathbf{g}_1^{(1)} & \mathbf{g}_2^{(1)} & \cdots & \mathbf{g}_m^{(1)} \end{pmatrix}, \\ &\vdots \\ \tilde{G}_n &:= \begin{pmatrix} \mathbf{g}_1^{(n)} & \mathbf{g}_2^{(n)} & \cdots & \mathbf{g}_m^{(n)} \end{pmatrix}. \end{aligned}$$

We then obtain $(\tilde{P}_1, \ldots, \tilde{P}_n)$ and $(\tilde{F}_1, \ldots, \tilde{F}_n)$ by applying this deformation to $(P_1, \ldots, P_m)$ and $(F_1, \ldots, F_m)$, respectively. Further, we have

$$(\tilde{P}_1, \ldots, \tilde{P}_n) = (S^\top \tilde{F}_1, \ldots, S^\top \tilde{F}_n) \cdot S.$$

For the proposed parameter sets shown in Subsection 4.5, we have $m > V > M$. From this relation, it is easily seen that the deformation matrices $\tilde{F}_{V+1}, \ldots, \tilde{F}_N \in \mathbb{F}_q^{N \times m}$ are of rank $\leq V$ since they have the following form:

$$\begin{pmatrix} *_{V \times m} \\ 0_{M \times m} \end{pmatrix}.$$

Then, there exists a linear combination of $\tilde{P}_1, \ldots, \tilde{P}_N \in \mathbb{F}_q^{N \times m}$ whose rank is $\leq V$. The rectangular MinRank attack utilizes this property to recover the private key.

The rectangular MinRank attack tries to find a non-zero element of $\mathcal{S}^{-1}(\mathcal{O})$. As in the case of Rainbow, the rectangular MinRank attack against UOV with (q^ℓ, V, M, m) is constructed as follows. Since $\dim\left(\mathcal{S}^{-1}(\mathcal{O})\right) = M$, there exists a non-zero N-by-1 vector with the following form:

$$\mathbf{a} = (a_1, a_2, \ldots, a_{V+1}, 0, \ldots, 0)^\top \in \mathcal{S}^{-1}(\mathcal{O}).$$

Then, it is shown that

$$\sum_{i=1}^{V+1} a_i \tilde{P}_i = (\tilde{P}_1, \ldots, \tilde{P}_N) \cdot \mathbf{a} = (S^\top \tilde{F}_1, \ldots, S^\top \tilde{F}_N) \cdot (S \cdot \mathbf{a})$$

is a linear combination of $S^\top \tilde{F}_{v+1}, \ldots, S^\top \tilde{F}_N$. Thus, this linear combination is of rank $\leq V$. Namely, the vector $\mathbf{a}$ gives a solution to the MinRank problem for $(\tilde{P}_1, \ldots, \tilde{P}_{V+1})$ with the target rank V. Moreover, we have

$$p_1(\mathbf{a}) = \cdots = p_m(\mathbf{a}) = 0.$$

As a result, the vector $\mathbf{a} = (a_1, a_2, \ldots, a_{V+1}, 0, \ldots, 0)^\top$ we want to find is a common solution to the following problems:

$$\text{(i) } \mathrm{Rank}\left(\sum_{i=1}^{V+1} a_i \tilde{P}_i\right) \leq V,$$
$$\text{(ii) } p_1(\mathbf{a}) = \cdots = p_m(\mathbf{a}) = 0.$$

Complexity analysis We then describe the estimation of the complexity to solve the above problems (i) and (ii). This is done along Beullens' estimation [Beu21] for the rectangular MinRank attack against Rainbow. Note that the characteristic of $\mathbb{F}_{q^\ell}$ is always odd in QR-UOV. See [FI23] for more details on complexity estimation.

First, we only consider the problem (i). Fix an integer m' such that $V+1 \leq m' \leq m$. Let $\tilde{P}'_i$ be the $N \times m'$ matrix obtained by removing the column vectors from $(m'+1)$-th to m-th of $\tilde{P}_i$. Then one considers to apply the support minors modeling method [BBC+20] to the MinRank problem $(\tilde{P}'_1, \ldots, \tilde{P}'_{V+1})$ with the target rank V. Let I' be the ideal in $\mathbb{F}_{q^\ell}[\mathbf{a}, \mathbf{c}]$ generated by the bilinear equations obtained from the support minors modeling, where $\mathbf{c}$ is the set of $\binom{m'}{V+1}$ minor variables. For $b \in \mathbb{N}$, set

$$R'(b) := \sum_{i=1}^{b} (-1)^{i+1} \binom{m'}{V+i} \binom{N+i-1}{i} \binom{V+1+b-i}{b-i}.$$

Let $I'_{b,1}$ be the subspace of $(b,1)$-degree homogeneous polynomials of I' in $\mathbb{F}_{q^\ell}[\mathbf{a}, \mathbf{c}]$. Then, Bardet et al. [BBC+20] calculated that $\dim_{\mathbb{F}_{q^\ell}} I'_{b,1} = R'(b)$ for $1 \leq b \leq V+1$.

Next, we extend our analysis to include the problem (ii). We assume that $p_1(\mathbf{a}), \ldots, p_m(\mathbf{a})$ behave as a semi-regular system, where

$$\mathbf{a} = (a_1, a_2, \ldots, a_{V+1}, 0, \ldots, 0)^\top.$$

Let I be the ideal generated by I' and $p_1(\mathbf{a}), \ldots, p_m(\mathbf{a})$, namely,

$$I := I' + \langle p_1(\mathbf{a}), \ldots, p_m(\mathbf{a}) \rangle \subset \mathbb{F}_{q^\ell}[\mathbf{a}, \mathbf{c}].$$

Moreover, set

$$\begin{aligned} G'(t_1, t_2) &:= \binom{m'}{V} t_2 + \sum_{b=1}^{V+1} \left(\binom{m'}{V} \binom{V+b-1}{b} - R'(b) \right) t_1^b t_2, \\ G(t_1, t_2) &:= G'(t_1, t_2) \cdot (1 - t_1^2)^m. \end{aligned}$$

Let $b_{\min} \in \mathbb{N}$ be the minimum of b such that

$$\dim_{\mathbb{F}_{q^\ell}} I_{b,1} = \dim_{\mathbb{F}_{q^\ell}} \mathbb{F}_{q^\ell}[\mathbf{a}, \mathbf{c}]_{b,1} - 1.$$

Then, following Beullens' estimation, we can state that $b_{\min}$ is predicted by

$$b_{\min}^{(\mathrm{predict})} := \min \{ b \mid G(t_1, t_2)_{b,1} \leq 1 \}, \tag{13}$$

where $G(t_1, t_2)_{b,1}$ is the coefficient of $t_1^b t_2$.

Finally, by applying to $I_{b_{\min},1}$ the bilinear XL algorithm with Wiedemann algorithm [Wie86], we can find a solution $\mathbf{a}$ to problem (i) and (ii) with the following complexity:

$$3 \binom{m'}{V}^2 \binom{V + b_{\min} - 1}{b_{\min}}^2 (V+1)^2. \tag{14}$$

In [FI23], we experimented that $b_{\min}$ is equal to $b_{\min}^{(\mathrm{predict})}$ for some small parameters. Thus, we use $b_{\min}^{(\mathrm{predict})}$ instead of $b_{\min}$ to estimate the complexity of the rectangular MinRank attack against QR-UOV theoretically in Subsection 6.3.

Note that it is more efficient to apply the support minors method to rectangular matrices where the number of rows is larger than the number of columns. Thus, for the parameters with $N < m$, we estimate the complexity of the attack by considering to apply the MinRank attack after transposing the matrices $\tilde{P}_1, \ldots, \tilde{P}_N$. This operation is clearly possible since the rank of matrix is stable under transposition.

7.4 Irreducibility of Polynomial f

The public key matrices of QR-UOV are given as block matrices, where each component is an element of $W\mathcal{A}_f$. For the security of QR-UOV, we here discuss the relation between the irreducibility of the polynomial f of $\mathcal{A}_f$ and the

existence of transformation on symmetric matrices $W\Phi_g^f$ into a specific form matrix. Indeed, the security of BAC-UOV [SP20] whose public key is represented as block anti-circulant matrices was weakened by transforming anti-circulant matrices into a specific form with zero submatrices [FKI+20]. Therefore, we have to find f such that there exists no such a transformation on $W\Phi_g^f$.

In [FIKT21], the authors provided the following three theorems for the transformation on $W\Phi_g^f$ which show the suitability of an irreducible f for QR-UOV.

Theorem 3 (Theorem 4 in [FIKT21]). *Let $f \in \mathbb{F}_q[x]$ be a **reducible** polynomial with $\deg f = \ell$ and W be an invertible matrix such that every element of $W\mathcal{A}_f$ is a symmetric matrix. If $f = f_1 \cdots f_k$ ($k \in \mathbb{N}$), where $f_1, \ldots, f_k$ are distinct and irreducible, and $\deg f_1 \leq \cdots \leq \deg f_k$, then there exists an invertible matrix $L \in \mathbb{F}_q^{\ell \times \ell}$ and $i \in [\ell - 1]$ such that for any $X \in W\mathcal{A}_f$,*

$$L^\top X L = \begin{pmatrix} *_{i\times i} & 0_{i\times(\ell-i)} \\ 0_{(\ell-i)\times i} & *_{(\ell-i)\times(\ell-i)} \end{pmatrix}.$$

Theorem 4 (Theorem 5 in [FIKT21]). *With the same notation as in Theorem 3, if there exists $f' \in \mathbb{F}_q[x]$ such that $f'^2 \mid f$, there exists an invertible matrix $L \in \mathbb{F}_q^{\ell \times \ell}$ such that, for any $X \in W\mathcal{A}_f$,*

$$(L^\top X L)_{\ell\ell} = 0.$$

Theorem 5 (Theorem 2 in [FIKT21]). *Let $f \in \mathbb{F}_q[x]$ be an **irreducible** polynomial with $\deg f = \ell$ and W be an invertible matrix such that every element of $W\mathcal{A}_f$ is a symmetric matrix. Then, there is no invertible matrix $L \in \mathbb{F}_q^{\ell \times \ell}$ and $i, j \in [\ell]$ such that for any $X \in W\mathcal{A}_f$,*

$$(L^\top X L)_{i,j} = 0.$$

Theorems 3 and 4 show that if f is reducible, for any $X \in W\mathcal{A}_f$, X can be transformed into a matrix with zero submatrices by multiplying an invertible matrix and its transposition from both sides. In contrast, Theorem 5 shows that if f is irreducible, there exists no such transformation on $W\Phi_g^f$. Therefore, we select an irreducible polynomial as the f of $\mathcal{A}_f$ used in our proposed QR-UOV.

7.5 Lifting Method over Extension Field

The lifting method was proposed in [FIKT21] as a method of attacking QR-UOV by diagonalizing the matrices in $\mathcal{A}_f$ over the extension field $\mathbb{F}_{q^\ell}$. In this subsection, we show that the lifting method is essentially the same as the pullback method given in Subsection 4.1. As a preliminary step, we introduce the following theorem.

Theorem 6 (Theorem 3 in [FIKT21]). *With the same notation as in Theorem 5,*

(i) There exists an invertible matrix $L \in \mathbb{F}_{q^\ell}^{\ell\times\ell}$ such that

$$L^{-1}\Phi_x^f L = \begin{pmatrix} x & & & & \\ & x^q & & & \\ & & x^{q^2} & & \\ & & & \ddots & \\ & & & & x^{q^{\ell-1}} \end{pmatrix}.$$

In particular, this L diagonalizes any matrix in $\mathcal{A}_f$.

(ii) The matrix L described in (i) satisfies the condition that $L^\top W L$ is diagonal. Therefore, we can write

$$L^\top W L = \begin{pmatrix} \alpha_0 & & & & \\ & \alpha_1 & & & \\ & & \ddots & & \\ & & & \alpha_{\ell-1} \end{pmatrix}.$$

We recall the idea of the lifting method stated in [FIKT21]. The first and second statements in Theorem 5 show that for any $g \in \mathbb{F}_q[x]/(f) \cong \mathbb{F}_{q^\ell}$ the matrix $L^\top W \Phi_g^f L$ is diagonal. This indicates that $P_1, \ldots, P_m$ of QR-UOV can be transformed into block diagonal matrices for which the block size is $N \times N$. This indicates that $P_1, \ldots, P_m$ in QR-UOV can be transformed into block diagonal matrices with block size $N \times N$. Let $L^{(N)} = I_N \otimes L$ be an $n \times n$ block diagonal matrix with block size ℓ ($n = \ell \cdot N$), for which the N diagonal blocks are L. Then, $(L^{(N)})^\top P_i L^{(N)}$ ($i \in [m]$) become block matrices wherein every component is in a diagonal form. Furthermore, there exists a permutation matrix A such that $(L^{(N)}A)^\top P_i (L^{(N)}A)$ is a block diagonal matrix with block size N, and let $\bar{L} := L^{(N)}A$. The transformed matrices $\bar{L}^\top P_i \bar{L}$ can be represented by $(\bar{L}^{-1}S\bar{L})^\top (\bar{L}^\top F_i \bar{L})\ (\bar{L}^{-1}S\bar{L})$. Then, $\bar{L}^\top F_i \bar{L}$ is the diagonal concatenation of ℓ smaller matrices, similar to $\bar{L}^\top P_i \bar{L}$. Furthermore, $\bar{L}^{-1}S\bar{L}$ is also the diagonal concatenation of ℓ smaller matrices from (i) in Theorem 6. Then, owing to the structure of F_i, every diagonal block of $\bar{L}^\top F_i \bar{L}$ has an $M \times M$ zero block, similar to F_i. Therefore, each diagonal block of $\bar{L}^\top P_i \bar{L}$ has the same form as the matrix representing the public key of UOV with V vinegar variables and M oil variables over $\mathbb{F}_{q^\ell}$. The lifting method proposed in [FIKT21] executes the key recovery attacks on one of such diagonal blocks.

We show that the lifting method is essentially the same as the pull-back method. Let $A \in \mathbb{F}_q^{n\times n}$ be a permutation matrix such that

$$A^\top (X \otimes Y) A = Y \otimes X$$

for any $X \in \mathbb{F}_q^{N\times N}$ and $Y \in \mathbb{F}_q^{\ell\times\ell}$. Also, we recall the equation in Subsection 4.1

$$P_k = \sum_{i=0}^{\ell-1} \bar{P}_k^{(i)} \otimes W\Phi_{x^i}^f.$$

Then the transformation in the above lifting method is described as follows:

$$
\begin{aligned}
&A^\top (L^{(N)})^\top P_k L^{(N)} A = A^\top (L^{(N)})^\top \left(\sum_{i=0}^{\ell-1} \bar{P}_k^{(i)} \otimes W \Phi^f_{x^i} \right) L^{(N)} A \\
=&A^\top \left(\sum_{i=0}^{\ell-1} \bar{P}_k^{(i)} \otimes L^\top W \Phi^f_{x^i} L \right) A = A^\top \left(\sum_{i=0}^{\ell-1} \bar{P}_k^{(i)} \otimes L^\top W L \cdot L^{-1} \Phi^f_{x^i} L \right) A \\
=&A^\top \left(\sum_{i=0}^{\ell-1} \bar{P}_k^{(i)} \otimes \begin{pmatrix} \alpha_0 x^i & & & \\ & \alpha_1 x^{qi} & & \\ & & \ddots & \\ & & & \alpha_{\ell-1} x^{q^{\ell-1} i} \end{pmatrix} \right) A \\
=&\sum_{i=0}^{\ell-1} \begin{pmatrix} \alpha_0 x^i & & & \\ & \alpha_1 x^{qi} & & \\ & & \ddots & \\ & & & \alpha_{\ell-1} x^{q^{\ell-1} i} \end{pmatrix} \otimes \bar{P}_k^{(i)} \\
=&\begin{pmatrix} \alpha_0 \sum_{i=0}^{\ell-1} \bar{P}_k^{(i)} x^i & & & \\ & \alpha_1 \sum_{i=0}^{\ell-1} \bar{P}_k^{(i)} x^{qi} & & \\ & & \ddots & \\ & & & \alpha_{\ell-1} \sum_{i=0}^{\ell-1} \bar{P}_k^{(i)} x^{q^{\ell-1} i} \end{pmatrix} \\
=&\begin{pmatrix} \alpha_0 \bar{P}_k & & & \\ & \alpha_1 \bar{P}_{k,q} & & \\ & & \ddots & \\ & & & \alpha_{\ell-1} \bar{P}_{k,q^{\ell-1}} \end{pmatrix}.
\end{aligned}
$$

Here, we have set $\bar{P}_{k,q^a} := (p_{i,j}^{q^a})_{i,j}$, where $\bar{P}_k = (p_{i,j})$. Therefore, $\bar{P}_{k,q}, \ldots, \bar{P}_{k,q^{\ell-1}}$ are easily recovered from $\bar{P}_k$. Thus, when we consider a key recovery attack using the lifting method, it is enough to treat only $\bar{P}_k$ ($k \in [m]$). Since the pull-back method is also to execute a key recovery attack on $\bar{P}_k$, we conclude that a key recovery attack using the pull-back method is the same as that using the lifting method. The only difference from the pull-back method is that we can apply the direct attack on the system of $\bar{L}^\top P_i \bar{L}$ ($i \in [m]$) obtained by applying the lifting method. However, for most cases, this lifting direct attack is not more efficient than the plain direct attack, since the large finite field $\mathbb{F}_{q^\ell}$ disturbs guessing some variables in the hybrid approach. Therefore, we list only the complexity of the plain direct attack in Subsection 6.3.

Remark 3. The attacks on SNOVA recently proposed by [CLVVP24] combine a transformation into the extension field and some existing attacks. One can confirm that the transformation used in the attacks on SNOVA is equivalent to the lifting method. As mentioned above, we confirmed that the attack on the lifted QR-UOV does not weaken the security of QR-UOV. Therefore, these attacks on SNOVA do not affect the security of QR-UOV.

7.6 Multiplying $(\Phi_x^f)^{(N)}$ to the public key

This subsection introduces a technique to increase the number of public key matrices by multiplying $(\Phi_x^f)^{(N)}$ and demonstrates that this technique cannot be mounted as an effective attack on QR-UOV. The technique is based on the following observation, which is primarily derived from the method used in the attack on SNOVA [IA24]. When we multiply $(\Phi_x^f)^{(N)} \in \mathbb{F}_q^{n\times n}$ to public key matrices P_i with $i \in [m]$ from the right side, we have

$$\begin{aligned} P_i \cdot (\Phi_x^f)^{(N)} &= S^\top \cdot F_i \cdot S \cdot (\Phi_x^f)^{(N)} \\ &= S^\top \cdot F_i \cdot (\Phi_x^f)^{(N)} \cdot S, \end{aligned}$$

due to the commutativity of Φ_g^f with $g \in \mathbb{F}_q[x]/(f)$. Since $(\Phi_x^f)^{(N)}$ is a block diagonal matrix, the lower-right $m \times m$ submatrix of $F_i \cdot (\Phi_x^f)^{(N)}$ is the zero matrix as in the original F_i.

From the above observation, we can use $m \cdot \ell$ matrices $P_i \cdot (\Phi_{x^{j-1}}^f)^{(N)}$ with $i \in [m]$ and $j \in [\ell]$ in applying the key recovery attacks. Note that it is enough to consider $m \cdot \ell$ matrices $P_i \cdot (\Phi_{x^{j-1}}^f)^{(N)}$ since we have

$$(\Phi_{x^a}^f)^{(N)^\top} \cdot P_i \cdot (\Phi_{x^b}^f)^{(N)} = P_i \cdot (\Phi_{x^{a+b}}^f)^{(N)}.$$

By applying this method, we can use more information to recover the private key compared to the original key recovery attack on $\mathrm{UOV}(q, v, m, m)$ over the base field $\mathbb{F}_q$.

Subsequently, we show that the method increasing the number of public key matrices is not more efficient than the key recovery attacks over the extension field $\mathbb{F}_{q^\ell}$. More specifically, we show that the matrices $P_i \cdot (\Phi_{x^{j-1}}^f)^{(N)}$ are essentially the same as the matrices obtained by naturally transforming the matrices $\bar{P}_i \in \mathbb{F}_{q^\ell}^{N\times N}$ in Subsection 4.1 onto the base field $\mathbb{F}_q$. For a variable set $\bar{\mathbf{y}} = (\bar{\mathbf{y}}_1, \ldots, \bar{\mathbf{y}}_N)^\top$, we consider the following m quadratic polynomials over $\mathbb{F}_{q^\ell}$:

$$\bar{\mathbf{y}}^\top \cdot \bar{P}_i \cdot \bar{\mathbf{y}} \quad (i \in [m]). \tag{15}$$

These form a public key of an instance of $\mathrm{UOV}(q^\ell, V, M, m)$. For each $\bar{\mathbf{y}}_j$, we set $\bar{\mathbf{y}}_j = \sum_{k=0}^{\ell-1} \mathbf{y}_k^{(j)} x^k$, that is,

$$\bar{\mathbf{y}}_j = \begin{pmatrix} 1 & x & \cdots & x^{\ell-1} \end{pmatrix} \cdot \begin{pmatrix} \mathbf{y}_0^{(j)} \\ \mathbf{y}_1^{(j)} \\ \vdots \\ \mathbf{y}_{\ell-1}^{(j)} \end{pmatrix}. \tag{16}$$

Moreover, we set $\mathbf{y}^{(j)} = (\mathbf{y}_0^{(j)}, \ldots, \mathbf{y}_{\ell-1}^{(j)})^\top$ and $\mathbf{y} = (\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(N)})^\top$. Then the polynomial $\bar{\mathbf{y}}^\top \cdot \bar{P}_i \cdot \bar{\mathbf{y}}$ is represented using the variables $\mathbf{y}$ by substituting (16) into (15). Namely, there exist ℓ quadratic polynomials $p_i^{(0)}(\mathbf{y}), \ldots, p_i^{(\ell-1)}(\mathbf{y})$ in n variables $\mathbf{y}$ over $\mathbb{F}_q$ such that

$$\bar{\mathbf{y}}^\top \cdot \bar{P}_i \cdot \bar{\mathbf{y}} = p_i^{(0)}(\mathbf{y}) + x \cdot p_i^{(1)}(\mathbf{y}) + \cdots + x^{\ell-1} \cdot p_i^{(\ell-1)}(\mathbf{y}). \tag{17}$$

It is clear that the polynomials $p_i^{(k)}(\mathbf{y})$ obtained in this way form a public key of $\mathrm{UOV}(q, v, m, m \cdot \ell)$, since $\bar{P}_1, \ldots, \bar{P}_m$ form a public key of $\mathrm{UOV}(q^\ell, V, M, m)$. We show the following relation between $p_i^{(k)}(\mathbf{y})$ and $P_i \cdot (\Phi^f_{x^{j-1}})^{(N)}$:

Theorem 7. *For each $i \in [m]$, we have*

$$\left\langle p_i^{(0)}(\mathbf{y}), \ldots, p_i^{(\ell-1)}(\mathbf{y}) \right\rangle_{\mathbb{F}_q} = \left\langle \mathbf{y}^\top \cdot P_i \cdot (\Phi^f_{x^{j-1}})^{(N)} \cdot \mathbf{y} \;\middle|\; j \in [\ell] \right\rangle_{\mathbb{F}_q}.$$

Proof. First, we set $X = (1, x, \ldots, x^{\ell-1})$. Then we obtain the equation $\bar{\mathbf{y}} = (I_N \otimes X) \cdot \mathbf{y}$. Thus, by the definition, we have the following computations:

$$\begin{aligned}
\bar{\mathbf{y}}^\top \cdot \bar{P}_i \cdot \bar{\mathbf{y}} &= \mathbf{y}^\top \cdot (I_N \otimes X)^\top \cdot \bar{P}_i \cdot (I_N \otimes X) \cdot \mathbf{y} \\
&= \mathbf{y}^\top \cdot \left(\bar{P}_i \otimes X^\top X\right) \cdot \mathbf{y} \\
&= \mathbf{y}^\top \cdot \left(\sum_{h=0}^{\ell-1} x^h \bar{P}_i^{(h)} \otimes X^\top X\right) \cdot \mathbf{y} \\
&= \mathbf{y}^\top \cdot \left(\sum_{h=0}^{\ell-1} \bar{P}_i^{(h)} \otimes x^h X^\top X\right) \cdot \mathbf{y}.
\end{aligned}$$

Moreover, by the result for $X^\top X$ in the lemma below, we obtain

$$\begin{aligned}
&\mathbf{y}^\top \cdot \left(\sum_{h=0}^{\ell-1} \bar{P}_i^{(h)} \otimes X^\top X \cdot \Phi^f_{x^h}\right) \cdot \mathbf{y} \\
=&\mathbf{y}^\top \cdot \left(\sum_{h=0}^{\ell-1} \bar{P}_i^{(h)} \otimes \left(\sum_{k=0}^{\ell-1} x^k W \Phi^f_{z_k} \cdot \Phi^f_{x^h}\right)\right) \cdot \mathbf{y} \\
=&\sum_{k=0}^{\ell-1} x^k \cdot \mathbf{y}^\top \cdot \left(\sum_{h=0}^{\ell-1} \bar{P}_i^{(h)} \otimes W \Phi^f_{x^h}\right) \cdot (\Phi^f_{z_k})^{(N)} \cdot \mathbf{y} \\
=&\sum_{k=0}^{\ell-1} x^k \cdot \mathbf{y}^\top \cdot P_i \cdot (\Phi^f_{z_k})^{(N)} \cdot \mathbf{y}.
\end{aligned}$$

As a result, for each i and k, we have

$$p_i^{(k)}(\mathbf{y}) = \mathbf{y}^\top \cdot P_i \cdot (\Phi^f_{z_k})^{(N)} \cdot \mathbf{y}.$$

Since $z_0, z_1, \ldots, z_{\ell-1}$ in $\mathbb{F}_{q^\ell}$ are linearly independent over $\mathbb{F}_q$ from the lemma below, the theorem holds. □

Lemma 6. *(i) $xX^\top X = X^\top X \cdot \Phi^f_x$.*

(ii) There exist $z_0, \ldots, z_{\ell-1} \in \mathbb{F}_{q^\ell}$ such that $X^\top X = \sum_{k=0}^{\ell-1} x^k W \Phi^f_{z_k}$.

(iii) $\{z_0, \ldots, z_{\ell-1}\}$ are linearly independent over $\mathbb{F}_q$.

Proof. (i) It is easy to show statement (i) from the definition of Φ^f_x.

(ii) We set $X^\top X = \sum_{k=0}^{\ell-1} x^k Y_k$, where $Y_k \in \mathbb{F}_q^{\ell\times\ell}$. Since $X^\top X$ and $X^\top X \cdot \Phi^f_x$ are symmetric, Y_k and $Y_k\Phi^f_x$ are also symmetric. Thus, we have

$$W^{-1}(Y_k\Phi^f_x) = W^{-1}{\Phi^f_x}^\top Y_k = \Phi^f_x W^{-1} Y_k.$$

This implies that $W^{-1}Y_k$ and Φ^f_x are commutative. Therefore, there exists $z_k \in \mathbb{F}_q^\ell$ such that $W^{-1}Y_k = \Phi^f_{z_k}$.

(iii) By the definition of X,

$$X^\top X = \begin{pmatrix} X \\ X\Phi^f_x \\ \vdots \\ X\Phi^f_{x^{\ell-1}} \end{pmatrix} = \sum_{k=0}^{\ell-1} x^i \begin{pmatrix} e_{k+1} \\ e_{k+1}\cdot\Phi^f_x \\ \vdots \\ e_{k+1}\cdot\Phi^f_{x^{\ell-1}} \end{pmatrix},$$

where $e_{k+1} = (0,\ldots,0,\overset{k+1}{1},0\ldots,0)$. Therefore, we have $W\Phi^f_{z_k} = \begin{pmatrix} e_{k+1} \\ e_{k+1}\cdot\Phi^f_x \\ \vdots \\ e_{k+1}\cdot\Phi^f_{x^{\ell-1}} \end{pmatrix}$.
Since the first row of $W\Phi^f_{z_k}$ is e_{k+1}, it is clear that $W\Phi^f_{z_0},\ldots,W\Phi^f_{z_{\ell-1}}$ are linearly independent. Thus, (iii) holds. □

From Theorem 7, we see that the $\mathbb{F}_q$-vector space spanned by $m\cdot\ell$ polynomials $\mathbf{y}^\top \cdot P_i \cdot (\Phi^f_{x^{j-1}})^{(N)} \cdot \mathbf{y}$ discussed above is the same as that spanned by the public key $p_i^{(k)}(\mathbf{y})$ of UOV $(q, v, m, m\cdot\ell)$ obtained from a public key of UOV(q^ℓ, V, M, m) by transforming onto the base field $\mathbb{F}_q$. At present, it is not known effective attacks on UOV over $\mathbb{F}_q^\ell$ by utilizing the transformation of its public key into polynomials over the base field $\mathbb{F}_q$. Indeed, we confirmed that the key recovery attacks on the polynomials $p_i^{(k)}(\mathbf{y})$ for our proposed parameter sets are not efficient. In conclusion, the key recovery attacks using $P_i\cdot(\Phi^f_{x^{j-1}})^{(N)}$ or $p_i^{(k)}(\mathbf{y})$ do not weaken the security of QR-UOV.

8 Advantages and Limitations

The main advantages of QR-UOV are

- **Public key and signature sizes**. The plain UOV is known as a scheme with a small signature and a large public key. (The first round parameters in security level I have approximately 50KB public key and 100B signature [BCD+23].) Our proposed parameter sets with $q = 127$ and $\ell = 3$ reduce the public key size by approximately 50-60% compared with the plain UOV (approximately 20KB in security level I) with a few hundred bits of signature.

- **Efficiency**. The signature generation and verification processes consist of simple linear algebra operations over small finite fields (e.g. $\mathbb{F}_7$, $\mathbb{F}_{31}$, and $\mathbb{F}_{127}$) and thus QR-UOV can be implemented very efficiently.

- **Security**. The EUF-CMA security of QR-UOV is formally proven in the QROM assuming the difficulty of two problems, the UOV and QR-MQ problems. The security of the plain UOV is based on the UOV problem and thus it seems relatively well understood. By contrast, the QR-MQ problem is a new assumption generated by us to construct our security proof. Though there exists no formal reduction from the QR-MQ problem into the plain MQ problem, we provide some experimental facts that indicate that the difficulty of solving the QR-MQ problem is equivalent to that of solving the plain MQ problem.

- **Simplicity.** The design of the plain UOV is extremely simple, and QR-UOV is a natural extension of UOV utilizing the quotient rings structure. We can consider that the research undertaken to obtain from UOV to QR-UOV corresponds to that obtained from LWE to MLWE, where MLWE problem is a generalization of LWE using a module comprising vectors over a ring. Therefore, it requires only a minimum knowledge of algebra to understand and implement QR-UOV.

The main disadvantage of QR-UOV is the large size of the public key compared with other post-quantum signature schemes such as lattice-based signatures. As we mentioned above, the public key size of QR-UOV is reduced from that of the plain UOV. However, some selected lattice-based signature schemes have further small public keys with approximately 1000B in security level I. This disadvantage might make it difficult to apply QR-UOV to constrained devices such as smart cards. However, the increase in memory capabilities in the future will relax the impact of this disadvantage.

References

[AHU19] Andris Ambainis, Mike Hamburg, and Dominique Unruh. Quantum security proofs using semi-classical oracles. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019*, pages 269–295, Cham, 2019. Springer International Publishing.

[Bar04] Magali Bardet. *Étude des systèms algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie.* PhD thesis, Université Pierre et Marie Curie-Paris VI, 2004.

[BBC+20] Magali Bardet, Maxime Bros, Daniel Cabarcas, Philippe Gaborit, Ray Perlner, Daniel Smith-Tone, Jean-Pierre Tillich, and Javier Verbel. Improvements of algebraic attacks for solving the rank decoding and minrank problems. In *Advances in Cryptology–ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7–11, 2020, Proceedings, Part I 26*, pages 507–536. Springer, 2020.

[BCC+23] Ward Beullens, Fabio Campos, Sofía Celi, Basil Hess, and Matthias J. Kannwischer. MAYO specification document. `https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/mayo-spec-web.pdf`, 2023.

[BCD+23] Ward Beullens, Ming-Shing Chen, Jintai Ding, Boru Gong, Matthias J. Kannwischer, Jacques Patarin, Bo-Yuan Peng, Dieter Schmidt, Chengdong Tao Cheng-Jhih Shih, and Bo-Yin Yang. UOV: unbalanced oil and vinegar: Algorithm specifications and supporting documentation version 1.0. `https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/UOV-spec-web.pdf`, 2023.

[BCH+23] Ward Beullens, Ming-Shing Chen, Shih-Hao Hung, Matthias J. Kannwischer, Bo-Yuan Peng, Cheng-Jhih Shih, and Bo-Yin Yang. Oil and vinegar: Modern parameters and implementations. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2023(3):321–365, 2023.

[BCP97] Wieb Bosma, John Cannon, and Catherine Playoust. The Magma algebra system. I. The user language. *J. Symbolic Comput.*, 24(3-4):235–265, 1997. Computational algebra and number theory (London, 1993).

[Beu21] Ward Beullens. Improved cryptanalysis of UOV and Rainbow. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021*, pages 348–373, Cham, 2021. Springer International Publishing.

[Beu22] Ward Beullens. Breaking Rainbow takes a weekend on a laptop. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022*, pages 464–479, Cham, 2022. Springer Nature Switzerland.

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

[BFS03] Magali Bardet, Jean-Charles Faugere, and Bruno Salvy. *Complexity of Gröbner basis computation for Semi-regular Overdetermined sequences over F_2 with solutions in F_2*. PhD thesis, INRIA, 2003.

[BFSY05] Magali Bardet, Jean-Charles Faugère, Bruno Salvy, and Bo-Yin Yang. Asymptotic behavior of the index of regularity of quadratic semi-regular polynomial systems. In *8th International Symposium on Effective Methods in Algebraic Geometry*, 2005.

[BGV14] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):1–36, 2014.

[CDP23] Sanjit Chatterjee, M Prem Laxman Das, and Tapas Pandit. Revisiting the security of salted UOV signature. In *Progress in Cryptology–INDOCRYPT 2022: 23rd International Conference on Cryptology in India, Kolkata, India, December 11–14, 2022, Proceedings*, pages 697–719. Springer, 2023.

[CHT12] Peter Czypek, Stefan Heyse, and Enrico Thomae. Efficient implementations of MQPKS on constrained devices. In Emmanuel Prouff and Patrick Schaumont, editors, *Cryptographic Hardware and Embedded Systems – CHES 2012*, pages 374–389, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[CKPS00] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In Bart Preneel, editor, *Advances in Cryptology — EUROCRYPT 2000*, pages 392–407, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[CKY21] Tung Chou, Matthias J. Kannwischer, and Bo-Yin Yang. Rainbow on cortex-m4. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(4):650–675, 2021.

[CLVVP24] Daniel Cabarcas, Peigen Li, Javier Verbel, and Ricardo Villanueva-Polanco. Improved attacks for SNOVA by exploiting stability under a group action. Cryptology ePrint Archive, Paper 2024/1770, 2024.

[DKL+18] Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 238–268, 2018.

[DS05] Jintai Ding and Dieter Schmidt. Rainbow, a new multivariable polynomial signature scheme. In John Ioannidis, Angelos Keromytis, and Moti Yung, editors, *Applied Cryptography and Network Security*, pages 164–175, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[DYC+08] Jintai Ding, Bo-Yin Yang, Chia-Hsin Owen Chen, Ming-Shing Chen, and Chen-Mou Cheng. New differential-algebraic attacks and reparametrization of Rainbow. In Steven M. Bellovin, Rosario Gennaro, Angelos Keromytis, and Moti Yung, editors, *Applied Cryptography and Network Security*, pages 242–257, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[Fau99] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases (F4). *J. Pure Appl. Algebra*, 139(1-3):61–88, 1999.

[Fau02] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F5). In *ISSAC 2002*, pages 75–83. ACM, 2002.

[FI23] Hiroki Furue and Yasuhiko Ikematsu. A new security analysis against MAYO and QR-UOV using rectangular MinRank attack. In Junji Shikata and Hiroki Kuzuno, editors, *Advances in Information and Computer Security*, pages 101–116, Cham, 2023. Springer Nature Switzerland.

[FIKT21] Hiroki Furue, Yasuhiko Ikematsu, Yutaro Kiyomura, and Tsuyoshi Takagi. A new variant of unbalanced oil and vinegar using quotient ring: QR-UOV. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021*, pages 187–217, Cham, 2021. Springer International Publishing.

[FIP01] FIPS PUB 202, Advanced Encryption Standard (AES). National Institute of Standards and Technology, NIST FIPS PUB 197, U.S. Department of Commerce, November 2001. U.S.Department of Commerce/National Institute of Standards and Technology.

[FIP15] FIPS PUB 202, SHA-3 standard: Permutation-based hash and extendable-output functions. National Institute of Standards and Technology, NIST FIPS PUB 202, U.S. Department of Commerce, August 2015. U.S.Department of Commerce/National Institute of Standards and Technology.

[FIP24] FIPS PUB 204, module-lattice-based digital signature standard. National Institute of Standards and Technology, NIST FIPS PUB 204, U.S. Department of Commerce, October 2024. U.S.Department of Commerce/National Institute of Standards and Technology.

[FK24] Hiroki Furue and Momonari Kudo. Polynomial XL: A variant of the XL algorithm using Macaulay matrices over polynomial rings. In Markku-Juhani Saarinen and Daniel Smith-Tone, editors, *Post-Quantum Cryptography*, pages 109–143, Cham, 2024. Springer Nature Switzerland.

[FKI+20] Hiroki Furue, Koha Kinjo, Yasuhiko Ikematsu, Yacheng Wang, and Tsuyoshi Takagi. A structural attack on block-anti-circulant UOV at SAC 2019. In Jintai Ding and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography*, pages 323–339, Cham, 2020. Springer International Publishing.

[FNT21] Hiroki Furue, Shuhei Nakamura, and Tsuyoshi Takagi. Improving Thomae-Wolf algorithm for solving underdetermined multivariate quadratic polynomial problem. In Jung Hee Cheon and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography*, pages 65–78, Cham, 2021. Springer International Publishing.

[FT23] Hiroki Furue and Tsuyoshi Takagi. Fast enumeration algorithm for multivariate polynomials over general finite fields. In Thomas Johansson and Daniel Smith-Tone, editors, *Post-Quantum Cryptography*, pages 357–378, Cham, 2023. Springer Nature Switzerland.

[GHHM21] Alex B. Grilo, Kathrin Hövelmanns, Andreas Hülsing, and Christian Majenz. Tight adaptive reprogramming in the qrom. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021*, pages 637–667, Cham, 2021. Springer International Publishing.

[GJ90] Michael R. Garey and David S. Johnson. *Computers and intractability; a guide to the theory of NP-completeness*. W. H. Freeman & Co., USA, 1990.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, pages 212–219, New York, NY, USA, 1996. Association for Computing Machinery.

[Has23] Yasufumi Hashimoto. An improvement of algorithms to solve under-defined systems of multivariate quadratic equations. *JSIAM Letters*, 15:53–56, 2023.

[IA24] Yasuhiko Ikematsu and Rika Akiyama. Revisiting the security analysis of SNOVA. In *Proceedings of the 11th ACM Asia Public-Key Cryptography Workshop*, APKC '24, page 54–61, New York, NY, USA, 2024. Association for Computing Machinery.

[JS19] Samuel Jaques and John M. Schanck. Quantum cryptanalysis in the RAM model: Claw-finding attacks on SIKE. In *Advances in Cryptology – CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part I*, page 32–61, Berlin, Heidelberg, 2019. Springer-Verlag.

[KPG99] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced oil and vinegar signature schemes. In Jacques Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, pages 206–222, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

[KS98] Aviad Kipnis and Adi Shamir. Cryptanalysis of the oil and vinegar signature scheme. In Hugo Krawczyk, editor, *Advances in Cryptology — CRYPTO '98*, pages 257–266, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[KX24] Haruhisa Kosuge and Keita Xagawa. Probabilistic hash-and-sign with retry in the quantum random oracle model. In *IACR International Conference on Public-Key Cryptography*, pages 259–288. Springer, 2024.

[NAB+20] Michael Naehrig, Erdem Alkim, Joppe Bos, Léo Ducas, Karen Easterbrook, Brian LaMacchia, Patrick Longa, Ilya Mironov, Valeria Nikolaenko, Christopher Peikert, Ananth Raghunathan, and Douglas Stebila. FrodoKEM. Technical report, National Institute of Standards and Technology, 2020. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions`.

[NIS] NIST: Post-quantum cryptography CSRC. `https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization`.

[NIS20] NIST: Status report on the second round of the NIST post-quantum cryptography standardization process. `https://nvlpubs.nist.gov/nistpubs/ir/2020/NIST.IR.8309.pdf`, 2020.

[NIS22] NIST: Call for additional digital signature schemes for the post-quantum cryptography standardization process. `https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/call-for-proposals-dig-sig-sept-2022.pdf`, 2022.

[NIS24] NIST: Status report on the first round of the additional digital signature schemes for the NIST post-quantum cryptography standardization process. https://nvlpubs.nist.gov/nistpubs/ir/2024/NIST.IR.8528.pdf, 2024.

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):1–40, 2009.

[Saa24] Markku-Juhani O. Saarinen. Bit security of UOV 1.0 (and MAYO), 2024. https://groups.google.com/a/list.nist.gov/g/pqc-forum/c/SlBu7_oITN4/m/1MW-OVyfBgAJ.

[Sho99] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[SP20] Alan Szepieniec and Bart Preneel. Block-anti-circulant unbalanced oil and vinegar. In Kenneth G. Paterson and Douglas Stebila, editors, *Selected Areas in Cryptography – SAC 2019*, pages 574–588, Cham, 2020. Springer International Publishing.

[SSH11] Koichi Sakumoto, Taizo Shirai, and Harunaga Hiwatari. On provable security of UOV and HFE signature schemes against chosen-message attack. In Bo-Yin Yang, editor, *Post-Quantum Cryptography*, pages 68–82, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[TW12] Enrico Thomae and Christopher Wolf. Solving underdetermined systems of multivariate quadratic equations revisited. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *Public Key Cryptography – PKC 2012*, pages 156–171, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[Wie86] Douglas Wiedemann. Solving sparse linear equations over finite fields. *IEEE transactions on information theory*, 32(1):54–62, 1986.

[YC05] Bo-Yin Yang and Jiun-Ming Chen. All in the XL family: Theory and practice. In Choon-sik Park and Seongtaek Chee, editors, *Information Security and Cryptology – ICISC 2004*, pages 67–86, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[YCBC07] Bo-Yin Yang, Owen Chia-Hsin Chen, Daniel J. Bernstein, and Jiun-Ming Chen. Analysis of QUAD. In Alex Biryukov, editor, *Fast Software Encryption*, pages 290–308, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.