Chapter 5

Digital signature scheme based on multivariate quadratic quasigroups

Post-Quantum Cryptography (PQC) includes cryptographic protocols based on lattices, isogenies, multivariate polynomial equations, error-correcting codes and cryptographic hash functions. Due to the fast computation and modest computational resource requirements, cryptographic schemes based on multivariate polynomials have emerged as promising candidates for PQC over the past two decades. In literature [127], various multivariate polynomials based public key cryptographic schemes (also referred as Multivariate Public Key Cryptosystems (MPKC)) have been proposed. Generally, they can be divided into four categories depending upon their non-linear quadratic part. These include: Matsumoto-Imai (MI) scheme [91], Hidden Field Equation (HFE) scheme [101], Oil-Vinegar (OV) scheme [102] and Stepwise Triangular System (STS) scheme [64]. MI cryptosystem [91] was proposed by Matsumoto and Imai in 1988. It was the first multivariate quadratic (MQ) problem based public-key cryptosystem. In 1995, Patarin [100] proved that the MI cryptosystem could be easily cryptanalyzed using the Linearization equation attack. In 1996, Patarin proposed a cryptosystem named Hidden Field Equation (HFE) cryptosystem [101] and its different variants including HFEv- cryptosystem which can be viewed as an extension of the MI cryptosystem that resists Linearization equation attack. In HFE cryptosystem, Patarin transform the central map of MI cryptosystem using a Frobenius automorphism map $x \mapsto x^{q^i},\ i \in \mathbb{N}$ of a finite field $\mathbb{F}_q$. In 1997, Patarin proposed a signature scheme named it *Oil-Vinegar (OV) signature scheme* [102]. The basic idea behind the OV signature scheme [102] was motivated by the Linearization equation attack on the MI cryptosystem. In this scheme, Patarin utilized the quadratic maps contain-

ing oil variables in linear form and the vinegar variables in quadratic form. In 1998, Kipnis and Shamir [72] proved that in OV signature scheme when number of oil and vinegar variables are same then it is not secure against an attack based on oil-vinegar separation technique. In 1999, Kipnis et al. [71] proposed a variant of OV signature scheme named it *Unbalanced Oil-Vinegar (UOV) signature scheme*. In UOV signature scheme number of vinegar variables is greater than number of oil variables. Additionally, they proved that it was very efficient and immune against oil-vinegar separation attack. However, the public key size in this scheme was still extremely large. In [103, 104], Patarin et al. proposed two digital signature schemes based on MQ problem, namely, *SFlash signature scheme* and *Quartz signature scheme* respectively. The idea of SFlash signature scheme was based on the MI cryptosystem and Quartz signature scheme was based upon the HFEv- cryptosystem. Dubois et al. [43] have practically shown that the SFlash signature scheme was not secure against direct attacks. In 2003, Chen and Yang [24] proposed a multivariate signature scheme called *Stepwise Triangular Scheme (STS)*, based on Tame Transformations or Tame map. In 2006, Ding et al. [40] gave the cryptanalysis of STS scheme.

In 2005, Ding et al. [39] came up with the idea of a multilayered Oil-Vinegar signature scheme, i.e. *Rainbow*. They have shown that it has shorter signature size and smaller public-key size than the classical OV signature scheme. The Rainbow signature scheme was submitted as a robust candidate for the NIST competition [50]. Thereafter, Petzoldt et al.[106] proposed a variant of Rainbow signature scheme, i.e. *Cyclic Rainbow*. The public key size of this scheme reduces by 62% and the number of required field multiplications operations in verifying the legitimate signature reduces by 30%. In 2015, Petzoldt et al. [108] proposed a signature scheme, referred as *GUI*, which was based upon the HFEv- framework. This scheme utilized the small finite fields to increase its efficiency; however, it couldn't resolve the problem of large key size. In 2017, Petzoldt et al. [107] tried to resolve that issue by applying vinegar variations to MultiHFE [22] scheme, resulted in the development of a HMFEv signature scheme, defined over any random finite field. Later on, Hashimoto [67] have shown that the HMFEv signature scheme is not secure. Parallely, in 2016 Chen et al. [23] proposed a multivariate signature scheme, namely *MQDSS*, which was proved to be secure with respect to random oracle model. This scheme was based upon 5-pass identification protocol explained in [51, 111].

In 2008, Gligoroski et al. [57, 58] proposed a public key cryptosystem based on multivariate quadratic quasigroup (MQQ). The central map of the proposed cryptosystem was based on the string transformations of quasigroups. They had

shown that the proposed cryptosystem was more efficient in both hardware and software compared to existing multivariate polynomial based public key cryptosystems. However, in the same year Mohamed et al. [93] analyzed the proposed cryptosystem and proved that it is susceptible to the MutantXL attack [38]. In 2010, Faugére et al. [49] proved that the proposed cryptosystem was also susceptible to Gröbner basis attack. Additionally, Faugére et al. [49] had shown that the central map in existing multivariate quadratic quasigroup based public key cryptosystem has a weakness in central map which can be exploited very easily. In the subsequent years, Gligoroski et al. improved their scheme and in 2012, they proposed a digital signature scheme, namely *MQQ-SIG*, [61]. In the scheme, authors used minus modifier variation in the public key to safeguard it against direct attacks. Additionally, they demonstrated that their scheme is ultra-fast compared to the existing MQ-based cryptosystems and proved that the it is secure against chosen message attacks (CMA). Following that, in the same year Gligoroski and Samardjiska [62] also proposed a probabilistic encryption scheme referred as *MQQ-ENC*. In this scheme, they utilized the left multivariate quadratic quasigroups (LMQQ) for the construction of central map. Despite the differences in central map of both MQQ-SIG and MQQ-ENC scheme, in 2015 Faugére et al. [48] proved that for both schemes, finding equivalent good keys is feasible in polynomial time.

This motivation drives us to do further research into refining the MQQ-SIG scheme aiming to address all existing drawbacks while ensuring the scheme's resilience against Faugére's attack [48].

This chapter is divided into several sections. In Section 5.1, we examine the conditions that must be satisfied for the T-function to define a MQQ over the Galois field $\mathbb{F}_{p^k}$ where p-prime and $k \in \mathbb{N}$. In Section 5.2, we propose the signature scheme named it "MQQ-Sigv" and its verification algorithm. In Section 5.3, we discuss the security analysis of the proposed signature scheme and prove that the scheme is secure against Direct attack, Min-rank attack, High-rank attack, Differential attack and EUF-CMA attack. Most importantly, we will prove that after applying the transformation proposed by Wang et al. [126] to the proposed scheme, it is infeasible to find an equivalent key in polynomial time. In Section 5.4, we give the operating characteristics and efficiency of the signature scheme.

In the following part of this chapter, a quasigroup $(Q, *)$ is represented as $(Q, \mathfrak{q})$ where $x * y$ represents the same meaning as $\mathfrak{q}(x, y)$. The parastrophe of quasigroup operation $\mathfrak{q}(x, y)$ is represented as $\mathfrak{q}_{\backslash}(x, y)$. Gligoroski et al. [56] presented that a quasigroup $(Q, \mathfrak{q})$ of order 2^d, $d \geq 2$ can be represented as a vector valued Boolean functions, explained briefly in Section 1.1.3.

5.1 Multivariate quadratic quasigroups over finite field

Definition 5.1.1. [113] Consider a map $f : (\mathbb{F}_2^d)^m \longrightarrow (\mathbb{F}_2^d)^l$ such that for every $x \in (\mathbb{F}_2^d)^m$, the k^{th} bit of the j^{th} component, denoted as $f(x)_k^j$, depends only on the rightmost k bits of each component of x, for all $j \in \{1, \ldots, l\}$. Then the function f is defined as *T-function*.

We discuss a construction of MQQ over the finite field $\mathbb{F}_{p^k}$, where p-prime and $k \geq 1$ [112]. As, the generation of MQQ involves the use of T-functions and therefore, it is referred as "T-multivariate quadratic quasigroups (T-MQQ)". Additionally, we explore an efficient method of determining the parastrophes of the T-multivariate quadratic quasigroup (T-MQQ).

The conditions under which the T-function defines a quasigroup are outlined in Theorem 5.1.2. For better comprehension, we specifically discuss the case for $p = 2$.

Theorem 5.1.2. [113] *A Boolean T-function $\mathfrak{q} : \mathbb{F}_2^{2d} \longrightarrow \mathbb{F}_2^d$, defines a quasigroup operation if and only if it is of the form $\mathfrak{q} = (\mathfrak{q}^{(d)}, \mathfrak{q}^{(d-1)}, \ldots, \mathfrak{q}^{(1)})$ and for $(x, y) = (x_d, \ldots, x_1, y_d, \ldots, y_1)$, the component $\mathfrak{q}^{(s)}(x, y)$ for all $s \in \{1, \ldots, d\}$ is defined as:*

$$\mathfrak{q}^{(s)}(x,y) = x_s \oplus y_s \oplus \left(\bigoplus_{\substack{j=(j_{s-1},\ldots,j_0)\in\mathbb{F}_2^s,\\ k=(k_{s-1},\ldots,k_0)\in\mathbb{F}_2^s}} b_{jk} x_{s-1}^{j_{s-1}} \ldots x_1^{j_1} y_{s-1}^{k_{s-1}} \ldots y_1^{k_1} \right) \tag{5.1}$$

Theorem 5.1.3. [112] *Let $(Q, \mathfrak{q})$ be a quasigroup defined using T-function over $\mathbb{F}_{p^k}$. If a map $\mathfrak{q} : \mathbb{F}_{p^k}^{2d} \longrightarrow \mathbb{F}_{p^k}^d$ such that $\mathfrak{q} = (\mathfrak{q}^{(1)}, \ldots, \mathfrak{q}^{(d)})$, the components $\mathfrak{q}^{(s)}$ for all $s = 1, \ldots, d$, and $(x, y) = (x_d, \ldots, x_1, y_d, \ldots, y_1)$ is represented as:*

$$\begin{aligned}\mathfrak{q}^{(s)}(x,y) = p_1^{(s)}(x_s) + p_2^{(s)}(y_s) + \sum_{l,m>s} \alpha_{l,m}^{(s)} x_l x_m + \sum_{l,m>s} \beta_{l,m}^{(s)} y_l y_m + \sum_{l,m>s} \gamma_{l,m}^{(s)} x_l y_m \\ + \sum_{l>s} \delta_l^{(s)} x_l + \sum_{l>s} \epsilon_l^{(s)} y_l + \eta^{(s)}.\end{aligned} \tag{5.2}$$

where $p_1^{(s)}(x_s)$ and $p_2^{(s)}(y_s)$ is a quadratic permutation polynomials over $\mathbb{F}_{p^k}$. $(\mathbb{F}_{p^k}^d, \mathfrak{q})$ defines a MQQ having order p^{kd}.

Proposition 5.1.4. [112] *Consider $(Q, \mathfrak{q})$ be an MQQ over $\mathbb{F}_{p^k}$ having order p^{kd} and defined using Theorem 5.1.3. Let D, D_1 and D_2 are three $d \times d$ size non-singular matrices, whereas c, c_1 and c_2 are non-zero vectors of dimension d. The*

quasigroups $(Q, \mathfrak{q}_0)$ *and* $(Q, \mathfrak{q})$ *can be considered isotopic to each other, if the following condition holds:*

$$\mathfrak{q}_0(x, y) = D \cdot \mathfrak{q}(D_1 \cdot x + c_1, D_2 \cdot y + c_2) + c \tag{5.3}$$

Consider a quasigroup $(Q, \mathfrak{q})$ of order p^{kd}. In the cryptographic algorithms, finding the left $\mathfrak{q}_\backslash$ or right $\mathfrak{q}_/$ parastrophes of the given quasigroup operation $\mathfrak{q}$ is necessary for decryption or signature process. However, determining the explicit form of the left parastrophe $\mathfrak{q}_\backslash$ can be computationally intensive in terms of space and time, especially when the degree of parastrophe can be arbitrary $(2 \leq deg \leq d)$. In this context, we explore two distinct approaches for obtaining the parastrophe of $\mathfrak{q}$, with the choice depending on the system architecture:

1. Finding and storing the multiplication table (Latin square) of the quasigroup $\mathfrak{q}$, and compute parastrophe $\mathfrak{q}_\backslash$ by referring the multiplication table (Latin square). But in this process architecture machine utilize the bigger memory of the system. Since, this approach is independent of the type of quasigroup, so we can compute the parastrophe $\mathfrak{q}_\backslash$ for any type of MQQ.

2. Another way of computing the parastrophe $\mathfrak{q}_\backslash$ of the quasigroup $\mathfrak{q}$ is by solving the system of d equations mentioned by Equations (5.4) and to find out the unknowns $\{y_1, \cdots, y_d\}$ for memory constrained architectures. By this approach we can avoid to find the explicit form of the parastrophe $\mathfrak{q}_\backslash$.

Our primary focus here is on the second approach for determining the parastrophe $\mathfrak{q}_\backslash$ of quasigroup operation $\mathfrak{q}$. Instead of directly obtaining the explicit form of $\mathfrak{q}_\backslash$ and evaluating $y = \mathfrak{q}_\backslash(u, v)$ for given $u, v \in \mathbb{F}_{p^k},\ k \geq 1$, we choose to find y for the bilinear MQQ operation $(\mathfrak{q})$ by leveraging the identity $\mathfrak{q}_\backslash(u, v) = y$ implies $\mathfrak{q}(u, y) = v$. In simpler terms, we transform the task of evaluating $\mathfrak{q}_\backslash$ into solving a system of d equations in d variables $y_1, \ldots, y_d$ over $\mathbb{F}_{p^k}$ as:

$$\mathfrak{q}(u, y) = v. \tag{5.4}$$

Finding solutions to this system of equations is usually a non-trivial problem. Fortunately, due to the unique structure of MQQ, this system can be solved efficiently in polynomial time. The reason lies in the conversion of Equation (5.4) into a system of linear equations containing d variables.

5.1.1 Existential unforgeability under chosen-message attack

In this section we discuss a notion of security designed for digital signature scheme. This standard notion of security is referred as *Existential Unforgeability Under Chosen-Message Attack (EUF-CMA)* and this concept is established through an experiment or game conducted between an adversary $\mathcal{A}$ and a challenger Ch.

Consider a digital signature scheme $Sig = \{\mathcal{K}, Gen, Ver\}$ where $\mathcal{K}$ is *Key-Generation algorithm*, Gen is *Signature-Generation algorithm* and Ver is *Signature Verification algorithm*. Suppose $\mathcal{A}$ represents an adversary that can access the signature protocol as black box (means it takes message as an input and gives the signature as an output) and Ch represents as challenger of game. Now we define the following experiment $Exp^{EUF-CMA}_{Sig(\mathtt{k})}$ which runs between $\mathcal{A}$ and Ch:

Algorithm 9 Experiment $Exp^{EUF-CMA}_{Sig(\mathtt{k})}$

1: The challenger (Ch) generates keys $(sk, pk) \leftarrow \mathcal{K}(\mathtt{k})$ by running Key-Generation algorithm which takes security parameter $\mathtt{k}$ as an input. Additionally, gives pk to $\mathcal{A}$.
2: The adversary $\mathcal{A}$ requests signatures for chosen n messages $\{m_i\}_{i=1}^n$ and obtains the valid signatures $\{\sigma_i\}_{i=1}^n$ in response, where $\sigma_i \leftarrow Gen(sk, m_i)$ for $i = 1, \ldots, n$.
3: The adversary $\mathcal{A}$ adaptively produces the message-signature pair (σ^*, m^*).
4: The output of the experiment $Exp^{EUF-CMA}_{Sig(\mathtt{k})}$ is

$$\begin{cases} 1, \text{ if } Ver(pk, m^*, \sigma^*) = \text{True and } \ m^* \notin \{m_i\}_{i=1}^n \\ 0, \ otherwise \end{cases}$$

The success probability (or Advantage (**Adv**)) of $\mathcal{A}$ can be defined as:

$$\mathbf{Adv}(\mathcal{A}^{EUF-CMA}_{Sig(\mathtt{k})}) = Prob[Exp^{EUF-CMA}_{Sig(\mathtt{k})} = 1].$$

The signature scheme $Sig(\mathtt{k})$ is EUF-CMA secure if the advantage $\mathbf{Adv}(\mathcal{A}^{EUF-CMA}_{Sig(\mathtt{k})})$ of any probabilistic polynomial time (PPT) adversary $\mathcal{A}$ is negligible concerning security parameter $\mathtt{k}$.

5.2 Construction of central map using multivariate quadratic quasigroup

We introduce a new way to construct a central map using the MQQ with vinegar variables. The motivation for this idea stems from the structure of central map in the HFEv- signature scheme [28]. We will modify the Equation (5.2) by incorporating extra variables, known as "vinegar variables". These additional variables play a crucial role in defining the central map of the proposed signature scheme.

Algorithm 10 Construction of central map by utilizing the vinegar variation of MQQ

Input: $d, k \in \mathbb{N}$ and p-prime.

1: Generate coefficients $\alpha_{l,m}^{(s)}, \beta_{l,m}^{(s)}, \gamma_{l,m}^{(s)}, c \in \mathbb{F}_{p^k}$ for all $l, m > s$ and $s \in \{1, \ldots, d\}$ randomly.

2: For all $s \in \{1, \ldots, d\}$,

- Construct the coefficients $\delta_i^{(s)} = \sum_{i=1}^{v} a_i z_i$ and $\epsilon_i^{(s)} = \sum_{i=1}^{v} b_i z_i$, where a_i and b_i are chosen randomly from $\mathbb{F}_{p^k}$.
- Construct the coefficient $\eta^{(s)} = \sum_{i=0}^{v} c_i z_i^2$, where c_i is chosen randomly from $\mathbb{F}_{p^k}$.

3: For all $s \in \{1, \ldots, d\}$,

- Generate two random bits $r_1, r_2 \in \mathbb{F}_2$ if $p = 2$, otherwise set $r_1 = 0, r_2 = 0$.
- If $r_l = 0$ construct $p_l^{(s)} = a_l^{(s)} x_s + b_l^{(s)}$, otherwise construct $p_l^{(s)} = a_l^{(s)} x_s^2 + b_l^{(s)}$, for $l \in \{1, 2\}$.

4: Construct the polynomial map $\mathfrak{q}^{(s)}(x_1, \ldots, x_d, y_1, \ldots, y_d, z_1, \ldots, z_d)$ for all $s \in \{1, \ldots, d\}$ by utilizing (5.5) and the map

$$\mathfrak{q} = (\mathfrak{q}^{(1)}, \mathfrak{q}^{(2)}, \ldots, \mathfrak{q}^{(d)}).$$

5: Generate randomly three non-singular $d \times d$ matrices D, D_1, D_2 and vectors c, c_1, c_2 with dimension d over $\mathbb{F}_{p^k}$.

Output: The tuple $(\mathfrak{q}, D^{-1}, D_1^{-1}, D_2^{-1})$ and central map: $\mathfrak{q}_0(x, y, z) = D \cdot \mathfrak{q}(D_1 \cdot x + c_1, D_2 \cdot y + c_2, z) + c$.

In above algorithm, if we choose random values for the vinegar variables $\{z_1, \ldots, z_v\}$ where each $z_i \in \mathbb{F}_{p^k}$ for $i = 1, \ldots, v$, then the coefficients $\delta_l^{(s)}, \epsilon_l^{(s)}$ and $\eta^{(s)}$ become constant terms and belongs to the field $\mathbb{F}_{p^k}$. As a consequence, we give the following theorem. This theorem is essential for finding an inverse function of the central map.

Theorem 5.2.1. *Consider a map* $\mathfrak{q} : \mathbb{F}_{p^k}^{2d+v} \longrightarrow \mathbb{F}_{p^k}^{d}$ *such that* $\mathfrak{q} = (\mathfrak{q}^{(1)}, \ldots, \mathfrak{q}^{(d)})$

and the components $\mathfrak{q}^{(s)}$ *for all* $s = 1, \ldots, d$ *is represented as:*

$$\mathfrak{q}^{(s)}(x_1, \cdots, x_d, y_1, \cdots, y_d, z_1, \ldots, z_v) = p^{(s)}(x_s) + p^{(s)}(y_s) + \sum_{l,m>s} \alpha_{l,m}^{(s)} x_l x_m + \sum_{l,m>s} \beta_{l,m}^{(s)} y_l y_m + \sum_{l,m>s} \gamma_{l,m}^{(s)} x_l y_m + \sum_{l>s} \delta_l^{(s)}(z_1, \ldots, z_v) x_l + \sum_{l>s} \epsilon_l^{(s)}(z_1, \ldots, z_v) y_l + \eta^{(s)}(z_1, \ldots, z_v) + c \tag{5.5}$$

where $p^{(s)}(y_s)$ *and* $p^{(s)}(x_s)$ *are quadratic permutations over* $\mathbb{F}_{p^k}$ *and the coefficients* $\alpha_{l,m}^{(s)}, \beta_{l,m}^{(s)}$ *and* $\gamma_{l,m}^{(s)} \in \mathbb{F}_{p^k}$*. The coefficients* $\delta_l^{(s)}(z_1, \ldots, z_v)$ *and* $\epsilon_l^{(s)}(z_1, \ldots, z_v)$ *must be linear where the coefficient* $\eta^{(s)}(z_1, \ldots, z_v)$ *must be quadratic in vinegar variable. After choosing random values of the vinegar variables* $\{z_1, \ldots, z_v\}$ *where each* $z_i \in \mathbb{F}_{p^k}$ *for all* $i = 1, \ldots, v$*.* $(\mathbb{F}_{p^k}^d, \mathfrak{q})$ *defines a MQQ of order* p^{kd}*.*

5.2.1 Generation of private-key

This section describes an algorithm for generating two different affine maps: S and T. To significantly reduce memory consumption and speed up the signing process, we utilize a symmetric Toeplitz matrix instead of a random permutation matrix. The idea of generating the private key pair (S, T) is motivated by MQQ-SIG signature scheme [62].

According to Algorithm 11, to construct the pair of affine maps (S, T) we just need to store $(\sigma_1, \sigma_2, M_0, (\alpha_i)_n)$ parameters. This requires only $\left(2n + \frac{r \cdot n}{8} + \frac{k \cdot n}{8}\right)$ bytes space where n, k, r are input parameters which is proportional to $\mathcal{O}(n)$.

5.2.2 Generation of public-key

We construct a public key by utilizing the central map defined by Algorithm 10 and the private key pair generated by Algorithm 11. For the public key, we also utilized the minus modifier technique to decrease its size. The public key construction is motivated by the underlying principles of the *Rainbow signature scheme.*

Algorithm 11 Affine maps S and T

Input: $n, u_1, u_2, r, k \in \mathbb{N}$ and p-prime

1: Generate two permutation matrices p_{σ_1} and p_{σ_2} over field $\mathbb{F}_2$ corresponding to two different random permutations σ_1 and σ_2 respectively on the set $\{1, \ldots, n\}$.

2: Generate matrices such that $p_{\rho_i^{(1)}} = [p_{\rho_i}[\mathtt{i}, \mathtt{j}]]$, $p_{\rho_j^{(2)}} = [p_{\rho_j}[\mathtt{i}, \mathtt{j}]]$ randomly over $\mathbb{F}_{p^k}$ for all $i \in \{0, \ldots, u_1\}, j \in \{0, \ldots, u_2\}$ satisfying the condition $[p_{\rho_i}[\mathtt{i}, \mathtt{j}]] = [p_{\rho_i}[\mathtt{i}+1, \mathtt{j}+1]]$, $[p_{\rho_j}[\mathtt{i}, \mathtt{j}]] = [p_{\rho_j}[\mathtt{i}+1, \mathtt{j}+1]]$ respectively with the symmetric condition of matrix.

3: Generate a random matrix $M_0 = [m^0_{i,j}]_{r \times n}$ over the set $\{0, 1\} \subset \mathbb{F}_{p^k}$, such that for each column j, there exits at least one row i where $m_{i,j} \neq 0$.

4: For the fixed ordering of $\mathbb{F}_{p^k} \backslash \{0\}$, construct a repeating sequence $(\alpha_i)_n$ of length n. Form a new matrix $M = [\alpha_{\sigma_1(j)} \cdot m^{(0)}_{i,j}]_{r \times n}$ and then obtain the matrix I_M by replacing the last r rows of the identity matrix I_n by M.

5: Compute the matrices

$$S'_{inv} = \sum_{i=0}^{u_1} p_{\rho_i^{(1)}} \cdot p_{\sigma_2} + \sum_{j=0}^{v_1} p_{\rho_j^{(2)}} \cdot P_{\sigma_1}, \tag{5.6}$$

$$T'_{inv} = \sum_{i=0}^{u_1} p_{\rho_i^{(1)}} \cdot p_{\sigma_1} \cdot I_M + \sum_{j=0}^{v_1} p_{\rho_j^{(1)}} \cdot p_{\sigma_2} \cdot I_M. \tag{5.7}$$

6: Let $\mathbf{SubT}'_{inv}$ be the $r \times n$ matrix containing first r rows of T'_{inv}. If $\mathbf{SubT}'_{inv}$ has zero column or $det(T'_{inv}) = 0$, $det(S'_{inv}) = 0$, then go to step 1. Else create the matrices $S = ((S'_{inv})^{transpose})^{-1}$ and $T = ((T'_{inv})^{transpose})^{-1}$. Additionally, generate a column vector $v_s = (\alpha_{\sigma_1(1)} \cdot \alpha_{\sigma_2(1)}, \ldots, \alpha_{\sigma_1(n)}\alpha_{\sigma_2(n)})^T$.

7: The affine maps are: $S(x) = S \cdot x + v_s$ and $T(x) = T \cdot x$.

Output: The tuple $(\sigma_1, \sigma_2, M_0, (\alpha_i)_n)$ and the affine map tuple (S, T).

Algorithm 12 Generation of public key

Input: leader element $b = (11\ldots1)$

1: Generate the private key pair (S, T) using the Algorithm 11.

2: Construct a central map $\mathfrak{q}_0$ using Algorithm 10.

3: Suppose $x = (x_1, \ldots, x_n)$ be the vector with $x_i \in \mathbb{F}_{p^k}$, we transform vector x into $X = (X_1, \ldots, X_d)$ over $\mathbb{F}_{p^k}^8$, where $n = 8d$ and each $X_i = (x_{8i-7}, \ldots, x_{8i})$.

4: Construct a map $F : \mathbb{F}_{p^k}^n \longrightarrow \mathbb{F}_{p^k}^n$ such that:

$$\begin{aligned} F(x_1, \ldots x_n) = (y_1, \ldots, y_n) &\Longleftrightarrow (Y_1, \ldots, Y_d) \\ = ((\mathfrak{q}_0(b, X_1, Z_1), \mathfrak{q}_0(X_1, X_2, Z_2), \ldots, \mathfrak{q}_0(X_{d-1}, X_d, Z_d))) & \end{aligned} \tag{5.8}$$

where $b = (11\ldots1)$ be the leader element and $Z = (Z_1, \ldots, Z_d)$, $Z_i = (z_{8i-7}, \ldots, z_{8i})$ be the vector containing vinegar variables.

5: Construct a map $P' : \mathbb{F}_{p^k}^n \longrightarrow \mathbb{F}_{p^k}^n$ such that $P' = T \circ F \circ S = (P_1, \ldots, P_n)$, where P_i is $P_i(x_1, \ldots, x_n)$ for all $1 \leq i \leq n$.

6: Construct the public key map $P : \mathbb{F}_{p^k}^n \longrightarrow \mathbb{F}_{p^k}^{n-r}$ as $P = (P_{r+1}, \ldots, P_n)$ by removing first r quadratic equations from the map P'.

7: Choose a cryptographically secure hash map $Hash : \{0,1\}^* \longrightarrow \{0,1\}^n$

Output: Public key pair is $pk = (P, Hash)$

5.2.3 Signature scheme

This section introduces a signing method for a given document $M \in \mathbb{F}_{p^k}^n$. The signing process involves finding the inverse of a tuple (S, F, T). Notably, to compute the inverse of central map F efficiently, we utilize Theorem 5.2.1. The first step involves converting the central map into a T-MQQ by choosing random values for the vinegar variables from the field $\mathbb{F}_{p^k}$. Then, we calculate the parastrophe of the given quasigroup operation $\mathfrak{q}$ using Proposition 5.1.4.

Algorithm 13 Inverse of the function

Input: $d, k \in \mathbb{N}$, p-prime.

1: Choose the vinegar variables $\{z_1, \ldots, z_v\}$ from $\mathbb{F}_{p^k}$ randomly. Substitute these variables into the map $\mathfrak{q}(x, y, z)$ defined by Equation (5.5). Then this map converts into Equation (5.2) as described by Theorem 5.2.1.

2: Compute $u_1 = D_1 \cdot u + c_1$ and $v_1 = D^{-1} \cdot (v - c)$.

3: Solve the system of equations $\mathfrak{q}(u_1, y_1) = v_1$ for the unknown y_1.

4: If the aforementioned system of equations has no solution, a new set of vinegar variables is chosen, resulting a new system of equations which needs to be solved. If the solution is unique then proceed to next step. In case of more than one solution, use other form of technique to find the correct set of y_1 like hash functions.

5: Compute $y = D_2^{-1} \cdot (y_1 - c_2)$

Output: $y =$ inverse of function F (denoted as F^{-1}).

To enhance the security of the scheme against Direct attack, the signing process follows a two steps approach. First, we concatenate the hash of the message with random values from the field $\mathbb{F}_{p^k}$. Then, we generate the signature for the resultant combined string.

Algorithm 14 Signature algorithm

Input: Message $M = (m_1, \ldots, m_s, m_{s+1}, \ldots, m_n) \in \mathbb{F}_{p^k}^n$, secret affine map pair (S, T) and the central function F.

1: Compute $h = Hash(m_1, m_2, \ldots, m_s, m_{s+1}, \ldots, m_n) = h_0 \,||\, h_1$, here h_0 is first $n - s$ bits of h and h_1 is remaining s-bits long string.

2: Choose $r_0 = (r_{01}, r_{02}, \ldots, r_{0s}) \in \mathbb{F}_{p^k}^s$ and $r_1 = (r_{11}, r_{12}, \ldots, r_{1(n-s)}) \in \mathbb{F}_{p^k}^{n-s}$ randomly. Compute $x_0 = r_0 \,||\, h_0$ and $x_1 = r_1 \,||\, h_1$.

3: Compute $y_0 = T^{-1}(x_0)$ and $y_1 = T^{-1}(x_1)$

4: Compute $z_0 = F^{-1}(y_0)$ and $z_1 = F^{-1}(y_1)$, and

5: Compute $\sigma = (\sigma_0, \sigma_1)$ where $\sigma_0 = S^{-1}(z_0)$ and $\sigma_1 = S^{-1}(z_1)$.

Output: Signature of the message M is $\sigma = (\sigma_0, \sigma_1)$.

Algorithm 15 Verification algorithm

Input: Message $M \in \mathbb{F}_{p^k}^n$, $\sigma = (\sigma_0, \sigma_1)$ and public key $pk = (P, Hash)$.

1: Compute $h = Hash(m_1, m_2, \ldots, m_s, m_{s+1}, \ldots, m_n) = h_0 \mid\mid h_1$, here h_0 is first $n - s$ bits of h and h_1 is remaining s-bits long.

2: Compute h_0' is the last $n - s$ bits of $P(\sigma_0)$ and h_1' is the last s bits of $P(\sigma_1)$.

3: Check if $h_0' = h_0$ and $h_1' = h_1$ return True; Else, return False.

Output: True or False

5.3 Security analysis

We will prove that the MQQ-Sigv signature scheme is EUF-CMA secure under the hardness of multivariate quadratic (MQ) problem. Additionally, we present a comprehensive analysis demonstrating its resistance to Min-rank attack, High-rank attack, Direct attack and Differential attack for different parameters. Finally, we will show that after applying the transformation proposed by Wang et al. [126] on the proposed signature scheme, it will be infeasible to find an equivalent good key in polynomial time.

Theorem 5.3.1. *Consider a cryptographically secure collision-resistant hash function $\mathcal{H}$ as a random oracle. Then the proposed MQQ-Sigv signature scheme is EUF-CMA secure under the hardness of the multivariate quadratic problem.*

Proof. The proof of this theorem follows a similar approach to the one used in the OHV scheme [77] relying on the difficulty of solving the multivariate cubic problem. We will prove the result by contradiction. Suppose there exists an adversary $\mathcal{A}$ with a non-negligible winning probability in the EUF-CMA game. This means $\mathbf{Adv}(\mathcal{A}_{Sig(\mathtt{k})}^{\text{EUF-CMA}}) = Prob[Exp_{Sig(\mathtt{k})}^{\text{EUF-CMA}} = 1]$ is non-negligible, where $\mathtt{k}$ is the security parameter. We will show that an oracle machine $\mathcal{O}^{\mathcal{A}}$ can be constructed to solve the MQ problem by presenting the series of games G_0, G_1 and G_2. Each G_i introduces a control modification to the preceding game G_{i-1} for $i = 1,\ 2$. We denote the probability of an adversary $\mathcal{A}$ for winning the game G_i as $Prob[G_i]$. An oracle machine, $\mathcal{O}^{\mathcal{A}}$ runs $\mathcal{A}$ and manage the outputs of the random oracle $\mathcal{H}$ within these games.

- $\mathbf{G}_0$: This game resembles to the EUF-CMA game for the MQQ-Sigv signature scheme. Therefore,
  $\mathbf{Adv}(\mathcal{A}_{Sig(\mathtt{k})}^{\text{EUF-CMA}}) = Prob[Exp_{Sig(\mathtt{k})}^{\text{EUF-CMA}} = 1] = Prob[G_0]$.

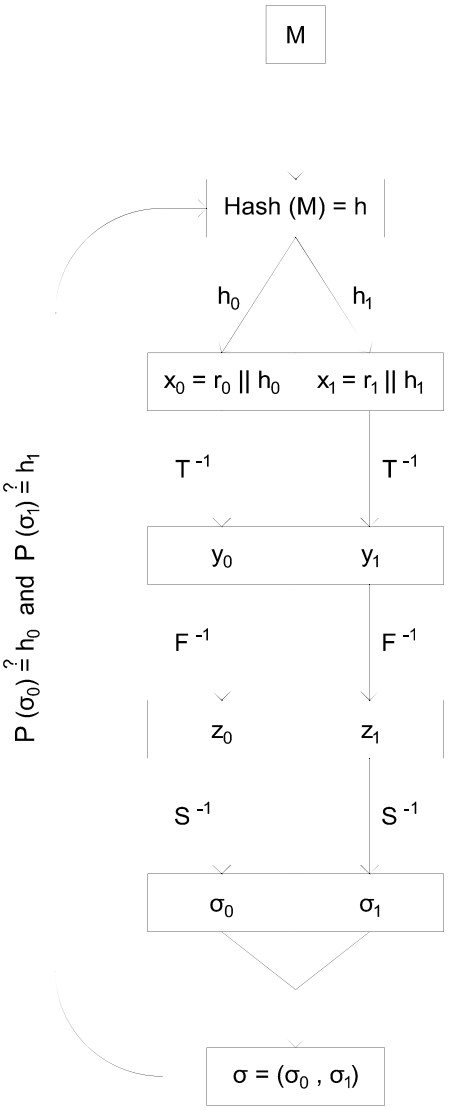

Figure 5.1: Pictorial representation of signature scheme and its verification

- $\mathbf{G}_1$: This game is identical to $\mathbf{G}_0$ except for $j \in \{1, \ldots, n\}$, $\mathcal{O}^{\mathcal{A}}$ provides the output of the hash $\mathcal{H}$ query of msg_j by $m_j = P(\sigma_0) \parallel P(\sigma_1) = S \circ F \circ T(\sigma_0) \parallel S \circ F \circ T(\sigma_1) \in \mathbb{F}_{p^k}^n$ and the signature query by σ_j, where σ_j is randomly selected from $\mathbb{F}_{p^k}^n$. Note that if $|Prob(G_1) - Prob(G_0)|$ is non-negligible, it would imply that $\mathcal{A}$ can distinguish the outputs generated by the hash function $\mathcal{H}$. However, it is infeasible since $\mathcal{H}$ is a cryptographically secure collision-resistant hash function. Therefore, $|Prob(G_1) - Prob(G_0)| = \epsilon_1(\mathtt{k})$ is negligible.

- $\mathbf{G}_2$: This game is identical to $\mathbf{G}_1$ except that oracle machine $\mathcal{O}^{\mathcal{A}}$ substitutes the output of the hash query of msg^* by some random element m^* from $\mathbb{F}_{p^k}^n$. Similar to the argument of $\mathbf{G}_1$, it can be asserted that $|Prob(G_2) - Prob(G_1)| = \epsilon_2(\mathtt{k})$ is negligible.

Now we have,

$$|Prob[G_2] - Prob[Exp_{Sig(\mathtt{k})}^{EUF-CMA} = 1]| = |Prob[G_2] - Prob[G_0]|$$
$$\leq |Prob[G_2] - Prob[G_1]| + |Prob[G_1] - Prob[G_0]| = \epsilon_1(\mathtt{k}) + \epsilon_2(\mathtt{k}) = \epsilon(\mathtt{k}).$$

Here, $\epsilon(\mathtt{k})$ is a negligible function. Consequently, the probability $Prob[G_2]$ that $\mathcal{A}$ wins the game G_2 is equal to the probability $\mathbf{Adv}(\mathcal{A}_{Sig(\mathtt{k}))}^{\text{EUF-CMA}}) = Prob[Exp_{Sig(\mathtt{k})}^{\text{EUF-CMA}} = 1]$ that $\mathcal{A}$ wins EUF-CMA game. It means, if $Prob[G_2]$ is non-negligible, then $\mathbf{Adv}(\mathcal{A}_{Sig(\mathtt{k})}^{\text{EUF-CMA}})$ is also non-negligible. It is important to note that $Prob[G_2]$ being non-negligible implies that $\mathcal{A}$ can generate the signature σ^* for msg^* such that $P(\sigma^*) = S \circ F \circ T(\sigma^*) = m^*$ with non-negligible probability. Thus with the assistance of $\mathcal{A}$, the machine $\mathcal{O}^{\mathcal{A}}$ can solve the MQ problem $P(x) = S \circ F \circ T(x) = \sigma^*$, contradicting our assumption that the MQ problem is hard to solve. Therefore, the $\mathbf{Adv}(\mathcal{A}_{Sig(\mathtt{k})}^{EUF-CMA}) = Prob[Exp_{Sig(\mathtt{k})}^{EUF-CMA} = 1]$ is negligible. Hence, we conclude that MQQ-Sigv signature scheme is EUF-CMA secure. □

Let $M_1, M_2, \ldots, M_k \in \mathbb{F}_{p^k}^{n \times m}$ and $r \in \mathbb{N}$, find a non-trivial linear combination (with $(\omega_1, \ldots, \omega_k) \in \mathbb{F}_{p^k}$) so that

$$Rank(\sum_{i=1}^{k} \omega_i M_i) \leq r. \tag{5.9}$$

is proven to be a NP-hard and famously known as **Min-rank problem** [38, 127]. Using Min-rank problem an attacker tries to find a linear combination of public matrices which sums up to a low-rank matrix. Based on Min-rank problem two types of attacks have been proposed:

- In **Min-rank attack**, an adversary tries to finds the large kernel shared by a small number of linear combinations (with $(\omega_1, \ldots, \omega_k)$) of the system represented by (5.9).
- Following a similar approach, in **High-rank attack**, an adversary attempts to finds the large kernel shared by a large number of linear combinations (with $(\omega_1, \ldots, \omega_k)$) for (5.9).

Theorem 5.3.2. *The proposed MQQ-Sigv signature scheme achieves ℓ-bits security against Min-rank attack, if the number of variables $v_{min\text{-}rank} \geq \lceil r_1 + 2^{val} \rceil$, where $val = \frac{\ell - r_1 \log_2 q}{3}$.*

Proof. In [130], Yang and Chen gave the Min-rank attack's complexity over $\mathbb{F}_q$ for any multivariate digital signature where $q = p^k$. The complexity is equal to $q^{s\lceil \frac{m}{n} \rceil} m^3$, where m is the number of equations, n is the numbers of variables and s is the minimum rank as described in (5.9). For MQQ-Sigv signature scheme, we consider $n = v_{\text{min-rank}}, m = v_{\text{min-rank}} - r_1$ and $s \approx n - (n - r_1) = r_1$. As a result, to obtain ℓ-bits security against Min-rank attack, $q^{s\lceil \frac{m}{n} \rceil} m^3 \geq 2^\ell$, which implies $q^{r_1}(v_{\text{min-rank}} - r_1)^3 \geq 2^\ell$. It means $v_{\text{min-rank}} \geq \lceil r_1 + 2^{val} \rceil$, where $val = \frac{\ell - r_1 \log_2 q}{3}$. Therefore, MQQ-Sigv signature scheme achieves ℓ-bits security against Min-rank attack, if $v_{\text{min-rank}} \geq \lceil r_1 + 2^{val} \rceil$, where $val = \frac{\ell - r_1 \log_2 q}{3}$. □

Remark 5.3.3. For finite field $\mathbb{F}_{2^k}$, the MQQ-Sigv signature scheme would be immune to Min-rank attack with ℓ-bit security if $v_{\text{min-rank}} \geq \lceil r + 2^{val_1} \rceil$, where $val_1 = \frac{\ell - r_1 k}{3}$

Theorem 5.3.4. *The proposed MQQ-Sigv signature scheme achieves ℓ-bits security against High-rank attack over $\mathbb{F}_q$ where $q = 2^k$, if the number of variables $v_{high\text{-}rank} \approx \frac{\ell}{2 \log_2 q}$.*

Proof. In [130], Yang and Chen mentioned the complexity of High-rank attack as $q^{\bar{\omega}}(\bar{\omega} n^2 + \frac{n^3}{6})$, where n is the number of plaintext variables and $\bar{\omega}$ is the minimal number of appearance of a plaintext variables in central map. For MQQ-Sigv signature scheme, to obtain ℓ-bits security against High-rank attack, we require $q^{\bar{\omega}}(4\bar{\omega} d^2 + \frac{8d^3}{6}) \geq 2^\ell$. In particular, if $q = 2^k$ and $d \approx \frac{\ell}{2k} = \frac{\ell}{2 \log_2 q}$, then the High-rank attack's complexity is greater than 2^ℓ for any $\bar{\omega} \in \mathbb{N}$. This implies that MQQ-Sigv signature scheme achieves ℓ-bits security against High-rank attack if $v_{\text{high-rank}} \approx \frac{\ell}{2 \log_2 q}$. □

Security level ℓ(in bits)	Number of removed equations (r_1)	Minimum number of variables to resist Min-rank attack	Minimum number of variables to resist High-rank attack
80	8	264	5
	9	59	5
	10	20	5
96	9	265	6
	10	50	6
	11	17	6
112	10	1635	7
	11	267	7
	12	52	7
128	11	10332	8
	12	1637	8
	15	21	8

Table 5.1: Least number of variables required to resist Min-rank and High-rank attack over the field $\mathbb{F}_{2^8}$

Direct attack. In Direct attack an adversary tries to solve the system of multivariate quadratic equations over finite field. This means, to find the valid signature for the given document and fixed $(y_1, \ldots, y_n)$ adversary attempts to solve the following system of equations over finite field:

$$\begin{cases} P_1(x_1, \ldots, x_n) &= y_1 \\ P_2(x_1, \ldots, x_n) &= y_2 \\ &\vdots \\ P_n(x_1, \ldots, x_n) &= y_n \end{cases} \tag{5.10}$$

It is also known as *Algebraic attack*. Adversary tries different approaches like Gröbner basis technique such as the Buchberger's algorithm (often referred as F_5 algorithm), MutantXl technique [120] to solve the given system of equations.

Theorem 5.3.5. *The MQQ-Sigv signature scheme achieves ℓ-bits security against Direct attack under the hardness of MQ problem.*

Proof. In [105], Petzoldt determine the minimum number of equations required to achieve ℓ-bits security against Direct attack experimentally and it is mentioned in Table 5.2.

Security level ℓ (in bits)	Number of equations		
	$\mathbb{F}_{2^4}$	$\mathbb{F}_{31}$	$\mathbb{F}_{2^8}$
80	30	28	26
100	39	36	33
128	51	48	43
192	80	75	68
256	110	103	93

Table 5.2: Minimal number of equations needed to achieve given security level

Based on the assumption that the MQ problem is hard to solve. The proposed MQQ-Sigv signature scheme is likely to achieve ℓ-bits security against Direct attack provided the number of equations are chosen as discussed in Table 5.2. □

Consider a function $G : \mathbb{F}_{p^k}^n \to \mathbb{F}_{p^k}^m$ and its differential at $\mathtt{a} \in \mathbb{F}_{p^k}^n$ is the linear mapping defined as $dG_{\mathtt{a}}(x) = G(x + \mathtt{a}) - G(x)$. If G is a quadratic function, then its differential will an affine function. Let us consider $L_{G,\mathtt{a}}(x) = dG_{\mathtt{a}}(x) - dG_{\mathtt{a}}(0)$, the linear part of the differential. Clearly, $L_{G,\mathtt{a}}(x)$ is a bilinear function and can be defined by $L_{G,\mathtt{a}}(x) = B_G(x, \mathtt{a}) = G(x + \mathtt{a}) - G(x) - G(\mathtt{a}) + G(0)$. It is also called a polar form of function G.

In **Differential attack** [53], an adversary tries to find the distribution of the rank of the differential and deduce information about the internal structure of the polar form $L_{P,\mathtt{a}}(x) = B_P(x, \mathtt{a}) = P(x + \mathtt{a}) - P(x) - P(\mathtt{a}) + P(0)$, where P is the system of multivariate polynomials known as a public key. Subsequently, adversary aims to recover the private key. According to existing literature [38, 53, 61, 62, 126], the Differential attack on multivariate polynomial based schemes is most effective against the Matsumoto-Imai (MI) family of cryptosystems. Additionally, Rainbow schemes [38] have been proven resistant to Differential attack when their central maps are chosen randomly.

The public key construction of the proposed MQQ-Sigv signature scheme is based on the underlying principle of the Rainbow scheme. The non-linear component of the MQQ-Sigv scheme is not a function of type $F : x \to x^{q^l+1}, l \in \mathbb{N}$. Additionally, the structure of quasigroups is masked using the private key, making it inaccessible to potential attackers. Consequently, we have the following result:

Theorem 5.3.6. *The proposed MQQ-Sigv signature scheme is secure against Differential attack under the condition that the MQQ operations* $\mathfrak{q}^{(i)}$, $i = 1, \ldots, n$ *are chosen uniformly at random.*

5.3.1 Resistance against good-key attack

In [49], Faugére et al. have shown that the MQQ construction is a layered single field scheme and has Rainbow like structure. In [48], Faugére et al. proved that due to the structure of MQQ with only 2-variables the MQQ-SIG [61] is vulnerable against the equivalent good-key attack.

Definition 5.3.7. [48] Both keys (F, S, T) and $(F^{'}, S^{'}, T^{'})$ are equivalent keys, if and only if $(T \circ F \circ S) = (T^{'} \circ F^{'} \circ S^{'}) \wedge (F \mid_I = F^{'} \mid_I)$, where $\wedge$ is 'and' operator, $F, F^{'} \in \mathbb{F}_{p^k}^m[x_1, \ldots, x_n]$, $S, S^{'} \in GL_n(\mathbb{F}_{p^k})$ and $T, T^{'} \in GL_m(\mathbb{F}_{p^k})$. Equivalently, the central functions F and $F^{'}$ have a similar structure when restricted to a limited set $I = \{I^{(1)}, \ldots, I^{(m)}\}$.

Definition 5.3.8. [48] The key $(F^{'}, S^{'}, T^{'})$ is a good key for (F, S, T) if and only if $(T \circ F \circ S = T^{'} \circ F^{'} \circ S^{'}) \wedge (F \mid_J = F^{'} \mid_J)$ for the fixed set J such that J satisfies the following criteria:

- For a given two fixed sets $I = \{I^{(1)}, \ldots, I^{(m)}\}$ and $J = \{J^{(1)}, \ldots, J^{(m)}\}$ with $J^{(k)} \subsetneq I^{(k)}$ for all $k \in \{1, \ldots, m\}$, then atleast one $J^{(k)} \neq \phi$.

Essentially, the equivalent key notion reduces the number of variables by introducing two more linear maps (K, L), where $K \in GL_m(\mathbb{F}_{p^k})$ and $L \in GL_n(\mathbb{F}_{p^k})$ such that public key $P = T \circ K^{-1} \circ K \circ F \circ L \circ L^{-1} \circ S$. If F and $F^{'} = K \circ F \circ L$ have similar structure according to Definition 5.3.7, then $T^{'} = T \circ K^{-1}$ and $S^{'} = L^{-1} \circ S$ will be the equivalent keys. Furthermore, the good key concept [121] lowers the number of unknowns or unfixed coefficients in $(S^{'}, T^{'})$.

In the proposed MQQ-Sigv signature scheme, we introduced the vinegar variables to the MQQ structure and utilize Algorithm 14 to construct the signature scheme. However, the scheme is based on structure of MQQ and according to Faugére et al [48], MQQ based signature schemes is vulnerable to equivalent good key attack.

To potentially improve the resistance of the MQQ-Sigv signature scheme against good key attacks, we use transformation introduced by Wang et al. [126]. This transformation involves randomly selecting a quadratic map over $\mathbb{F}_{p^k}$. The goal is to make it computationally infeasible for an adversary to find an equivalent key in polynomial time.

Transformation of MQQ-Sigv signature scheme

Consider a MQQ-Sigv=$\{\mathcal{K}, Sig, Ver\}$ as a digital signature scheme described in Section 5.2.3. To secure the MQQ-Sigv against good key attack, user needs to randomly select an invertible quadratic polynomial $G : \mathbb{F}_{p^k}^{2n} \rightarrow \mathbb{F}_{p^k}^{2n}$ as a part

of private key and two random irreversible hash functions $H : \mathbb{F}_{p^k}^n \to \mathbb{F}_{p^k}^n$ and $\widetilde{H} : \mathbb{F}_{p^k}^n \to \mathbb{F}_{p^k}^n$.

Suppose the transformed version of MQQ-Sigv signature scheme is $\widetilde{\text{MQQ-Sigv}} = \{\widetilde{\mathcal{K}}, \widetilde{Sig}, \widetilde{Ver}\}$, where $\widetilde{\mathcal{K}}$ is key generation algorithm, $\widetilde{Sig}$ is signature generation algorithm and $\widetilde{Ver}$ is verification algorithm.

1. **Key Generation algorithm**: The affine maps S and T can be generated by following the Algorithm 11. Subsequently, $pk = (P, Hash)$ can be generated by following an Algorithm 12. The secret key consists a map $G : \mathbb{F}_{p^k}^{2n} \to \mathbb{F}_{p^k}^{2n}$ which is an invertible map. Consequently, the public key tuple is: $(P, Hash, H \circ S, H \circ G^{-1} \circ S, \widetilde{H} \circ F \circ G^{-1} \circ S, \widetilde{H} \circ T^{-1})$ and the private key tuple is: (S, T, G) for $\widetilde{\text{MQQ-Sigv}}$.

2. **Signature Generation algorithm**: Consider $M = (m_1, m_2, \ldots, m_s, m_{s+1}, \ldots, m_n) \in \mathbb{F}_{p^k}^n$ needs to be signed using the private key pair (S, T, G). The Algorithm 16 can be used to generate the signature of M.

Algorithm 16 Transformed signature scheme $\widetilde{\text{MQQ-Sigv}}$

Input: Message $M = (m_1, m_2, \ldots, m_s, m_{s+1}, \ldots, m_n) \in \mathbb{F}_{p^k}^n$, secret key tuple (S, T, G) and the central function F.

1: Compute $h = Hash(m_1, \ldots, m_n) = h_0 \,||\, h_1$, here h_0 is first $n - s$ bits of h and h_1 is remaining s-bits string;

2: Choose $r_0 = (r_{01}, r_{02}, \ldots, r_{0s}) \in \mathbb{F}_{p^k}^s$ and $r_1 = (r_{11}, r_{12}, \ldots, r_{1(n-s)}) \in \mathbb{F}_{p^k}^{n-s}$ randomly and uniformly. Compute $x_0 = r_0 \,||\, h_0$ and $x_1 = r_1 \,||\, h_1$;

3: Calculate the following:

(i) $y_0 = T^{-1}(x_0)$ and $y_1 = T^{-1}(x_1)$;

(ii) $z_0 = F^{-1}(y_0)$ and $z_1 = F^{-1}(y_1)$,

(a) $\sigma_0 = S^{-1}(z_0)$ and $\sigma_1 = S^{-1}(z_1)$;

(b) $g(z_0, z_1) = g_0 \,||\, g_1$, $\sigma_{g_0} = S^{-1}(g_0)$ and $\sigma_{g_1} = S^{-1}(g_1)$.

Output: The signature of the Message M is (σ, σ_g) where $\sigma = (\sigma_0, \sigma_1)$ and $\sigma_g = (\sigma_{g_0}, \sigma_{g_1})$ is referred as *Forward signature* and *Backward signature* respectively.

3. **Signature Verification algorithm**: For verification protocol, the input is Message $M \in \mathbb{F}_{p^k}^n$, signature $(\sigma = (\sigma_0, \sigma_1), \sigma_g = (\sigma_{g_0}, \sigma_{g_1}))$ and the public key tuple $(P, Hash, H \circ S, H \circ G^{-1} \circ S, \widetilde{H} \circ F \circ G^{-1} \circ S, \widetilde{H} \circ T^{-1})$. By using Algorithm 17, user can verify whether the signature is legitimate or not.

Algorithm 17 Verification algorithm

Input: The Message $M = (m_1, m_2, \ldots, m_s, m_{s+1}, \ldots, m_n) \in \mathbb{F}_{p^k}^n$, the signature pair $(\sigma = (\sigma_0, \sigma_1), \sigma_g = (\sigma_{g_0}, \sigma_{g_1}))$ and the public key tuple $(P, Hash, H \circ S, H \circ G^{-1} \circ S, \widetilde{H} \circ F \circ G^{-1} \circ S, \widetilde{H} \circ T^{-1})$.

1: Compute $h = Hash(m_1, m_2, \ldots, m_n) = h_0 \,||\, h_1$, here h_0 is first $n - s$ bits of h and h_1 is remaining s-bits string.

2: Compute h_0' is the last $n - s$ bits of $P(\sigma_0)$ and h_1' is the last s bits of $P(\sigma_1)$.

3: Check whether $h_0' = h_0$ and $h_1' = h_1$, if yes then

(a) Check $\widetilde{H} \circ T^{-1}(h_0) \,||\, \widetilde{H} \circ T^{-1}(h_1) = \widetilde{H} \circ F \circ G^{-1} \circ S(\sigma_g)$; if yes, then

(i) Check $\widetilde{H} \circ G^{-1} \circ S(\sigma_g) = H \circ S(\sigma)$.

If any one condition in step (2) is not true, then the verification fails.

Output: Either True or False.

The pictorial representation of transformed MQQ-Sigv signature scheme is given in Figure 5.2. In transformed MQQ-Sigv signature scheme, there are two signature pairs (σ, σ_g), where σ is the forward signature and σ_g is the backward signature. Suppose an adversary $\mathcal{A}$ tries to find an equivalent key (S', F', T') of (S, F, T) for the given MQQ-Sigv signature scheme. There may be a possibility that $\mathcal{A}$ can find these pairs of the equivalent key corresponding to forward signature σ, but to find the random map $G : \mathbb{F}_{p^k}^{2n} \to \mathbb{F}_{p^k}^{2n}$ is not feasible in polynomial time according to [126]. Therefore, for an adversary to find the equivalent good key corresponding to the tuple (S, F, T, G) is not feasible in polynomial time. As a consequence, we have the following result:

Theorem 5.3.9. *The transformed digital signature scheme* $\widetilde{\text{MQQ-Sigv}}$ *based on MQQ is secure against Equivalent good key attack.*

5.4 Operating characteristics

In this section, we discuss the operating characteristics like public key size and signature size of the MQQ-Sigv scheme. Additionally, we compare these characteristics with the existing schemes like MQQ-SIG and Rainbow experimentally.

The proposed scheme includes public key $pk = (P, Hash)$ where pk consists $(n - r)$ polynomials in n variables over $\mathbb{F}_{p^k}$ and a standardized collision free hash function $Hash$. We analyzed the public key size of MQQ-Sigv over $\mathbb{F}_{2^k}$.

$$\text{Size of public key } (P) := \begin{cases} \frac{k}{8}(n-r)\left(1 + \frac{(n+v)(n+v+1)}{2}\right) & \text{if } k = 1 \\ \frac{k}{8}(n-r)\left(1 + \frac{(n+v)(n+v+3)}{2}\right) & \text{if } k > 1 \end{cases} \tag{5.11}$$

Finite field	Security level ℓ (in bits)	Number of variables (n)	Number of removed equations (r)	Number of vinegar variables (v)	Public key size (in bytes)
128	128	25	15	8	5206
		35	15	10	18917
256	128	45	20	15	47275
		45	20	20	102410

Table 5.3: Size of public key for 128-bit security

Security level ℓ (in bits)	Algorithm	Number of variables (n)	Number of vinegar variables (v)	Public key size (in bytes)	Signature Size (in bytes)
80	MQQ-SIG	50	20	137,408	40
	Rainbow	50	20	30,240	42
	MQQ-Sigv	50	20	12,780	100
96	MQQ-SIG	70	20	222,360	48
	MQQ-Sigv	70	20	32,441	140
112	MQQ-SIG	90	30	352,828	56
	MQQ-Sigv	90	30	73,810	180
128	MQQ-SIG	100	40	526,368	64
	MQQ-Sigv	100	40	100,110	200

Table 5.4: Comparative analysis of MQQ-SIG, Rainbow and MQQ-Sigv scheme in terms of key size and signature size

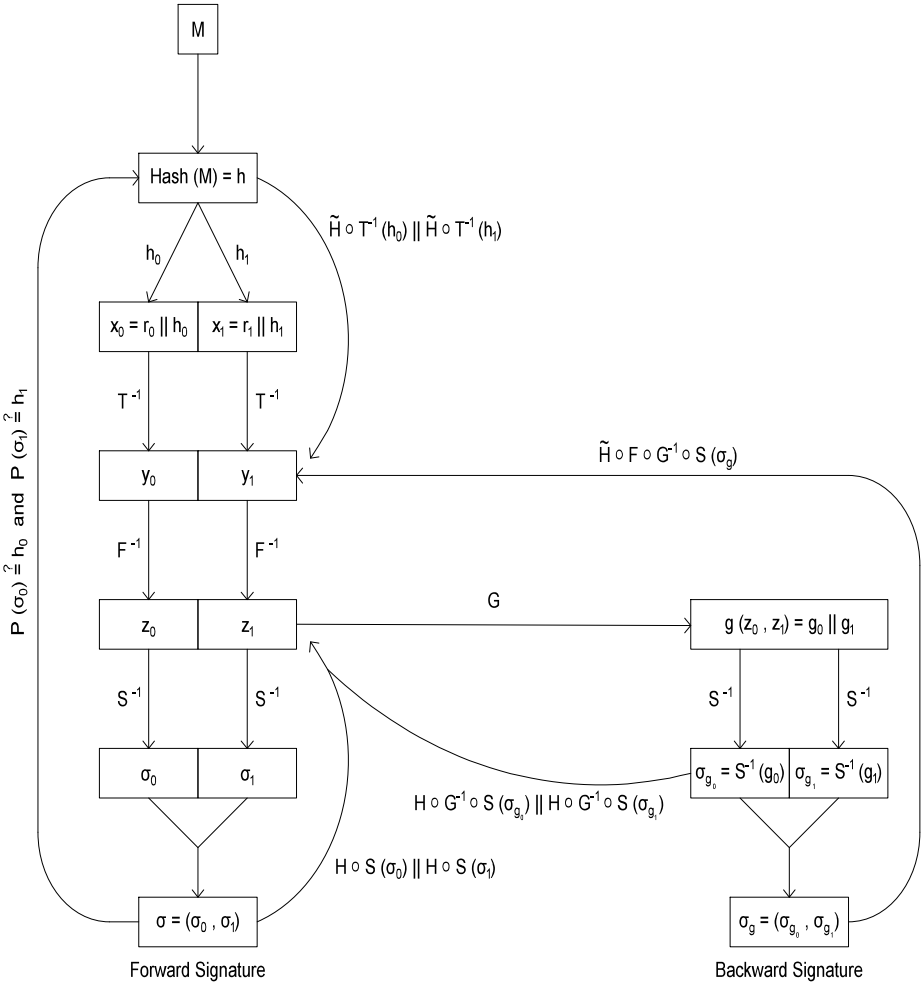

Figure 5.2: Pictorial representation of transformed MQQ-Sigv signature scheme