Investigation on post-quantum digital signature systems based on multivariate equations

Matei Cristea-Enache[1], Kaitai Liang[1],Huanhuan Chen[1]
[1]TU Delft

June 27th, 2021

Abstract

The NIST Post-Quantum Cryptography standardisation process has called for new algorithms, for the purpose of finding and standardising new cryptographic algorithms, able to withstand attacks enabled by future quantum processing progress. Digital signature schemes are fundamental for validating authenticity and integrity of digital documents. In the pages that follow, algorithms currently submitted in the NIST process, which rely on multivariate equations, will be investigated. This thesis will examine their underlying structure, known attacks, as well as their required storage and efficiency.

1 Introduction

1.1 Background

Quantum computing is a topic actively being researched by both academic and private institutions. Quantum computers should be able to run any classical algorithm and, at the same time, provide exponential speedups for some specific "hard" problems.

One such algorithm is Shor's algorithm, exponentially faster than any known current algorithm for decomposing a number into its prime factors [1]. This exponential speedup is troublesome for a cryptographic system depending on the hardness of decomposing numbers, such as the public key system RSA.

Public key cryptographic systems are especially vulnerable to be broken by quantum algorithms, since they rely on the lack of a fast solution for NP hard problems (such as prime factor decomposition); quantum algorithms may be able to compute a fast solution, effectively breaking the encryption.

While the number of qubits that most quantum computers is still low and can't yet break RSA, ciphertexts that can't be maliciously decrypted now, can be stored by an attacker and decrypted at a later date; something known as Retrospective decryption. Thus, the search for a cryptographic system that is immune to attacks from quantum algorithms has been started by the United States National Institute for Standards and Technology, as well as various EU authorities and institutions [2].

A digital signature system is a cryptographic algorithm that enables a user to generate a signature for a given document and for another user to authenticate the author and verify the integrity of said document. The signer typically needs to keep a secret key, that enables him to quickly generate signatures, secret which keeps away a malicious attacker (who only has access the public key) for a long amount of time.

For the purpose of finding and standardising quantum-resistant public key cryptographic systems, NIST has started a standardisation process in 2017. It is currently in its third round, with 7 finalists and

8 alternate candidates [3]. As such, there is no consensus yet on the best digital signature cryptographic system.

This research will not cover all digital signature cryptographic systems, but we will focus on the category of schemes based on multivariate equations. Algorithms in this category base their security on the NP-hardness [4] of solving a general system of multivariate equations. Such schemes usually generate a trapdoor, a way of quickly solving the system of equations knowing the secret key, while keeping it difficult for an attacker to find a solution. In more detail, the solution X' of $\bar{F}(X') = D$ (where $\bar{F}$ is the system of multivariate equations and D is the document to be signed) is usually called the digital signature. In order to verify the validity of the signature, a verifier will simply check the validity of the solution X' in $\bar{F}(X') = D$.

The goal of our research is to analyse systems based on multivariate schemes. We aim to compare and classify them on multiple criteria, such as underlying features, memory and time efficiency, security and any known vulnerabilities. The following research questions follow from our goal.

1.2 Research Question

Investigate post-quantum multi-variate signature cryptographic systems as submitted in NIST. Compare security, storage and time efficiency. Analyse similar/distinguishing features, advantages and disadvantages.

1.3 Sub-questions

1. How can multivariate signature systems be classified, amongst those submitted to NIST Post-Quantum Cryptography Standardisation?
2. What are the distinguishing features between the systems?
3. What are the similar features between the systems?
4. How secure are the multivariate signature systems submitted for the NIST standardisation process?
5. How fast/memory efficient are the multivariate signature systems?
6. Are there any potential vulnerabilities and drawbacks of the multi-variate signature systems? Can we find ways to fix said drawbacks?

1.4 Methodology

First we will look for all multivariate signature schemes algorithms submitted in the NIST Post-Quantum Cryptography Standardisation [5] [6] [7] [8] [9] [10].

Next we will read the original papers, analyse other reviews, as well as analyse the related works the NIST algorithms are inspired by [3].

Next we will write a high level comparison between them (look for distinguishing features as well as similar structure). Furthermore, we will compare the algorithms based on their security and on any known vulnerabilities.

Next, we will compare the memory and time necessary to run the NIST algorithms. We will use our previous knowledge to explain and compare these results. We will use the found results in order to formulate a preliminary opinion on the quality of each algorithm.

We will be looking for drawbacks and propose a way of fixing said drawbacks.

1.5 Structure

In section 1 we have introduced the research background and have formulated the research questions that need to be answered. We have also described the research methodology, how we aim to collect data to motivate the answers to our research questions. In sections 2 and 3, we will analyse the NIST algorithms as well as the older algorithms they are inspired by. We will explain the mathematical principles of each algorithm, how signing and verifying works and what attacks have been discovered so far.

Sections 4 and 5 will contain a comparison between the time and memory constraints of all NIST multivariate digital signature algorithms. Using the previous knowledge, we will try and motivate the found results and talk about potential trade offs supported by the schemes.

Section 6 contains a proposed improvement of one of the schemes, while section 7 will contain a final discussion pondering the advantages and disadvantages.

Section 8 includes an ethical reflection upon the carried out research and section 9 contains future possible work and a conclusion.

2 Analysis of related work

As part of the analysis, we will be doing a high level description of the algorithms. We will focus on the underlying mathematical problem of each algorithm a general overview on how it generates public and secret keys as well as any known security vulnerabilities.

Before we start with the NIST submissions, we will analyse some older algorithms that have influenced present day work. This is because many of our present day algorithms can be described as some variant of an old algorithm.

2.1 Bipolar systems

Most algorithms we will analyze fall into the category of bipolar multivariate systems [11]. This means that the public key is usually a map from K^n to K^m, meaning that the public key takes n variables and outputs m polynomials over a finite field K (each polynomial depending on n variables). Algorithms in this category usually have a low running time for computing $\bar{F}(X') = D$ (corresponding to the verifier checking the signature X' for a document D) and $F^{-1}(D) = X'$(corresponding to the signer using their secret key to generate a signature) for some secret equation F. After finding such a function, the signer usually picks 2 random affine maps S, T so that the final public key

$$\bar{F} = S \circ F \circ T$$

hides the secret inner polynomial F.

We will now briefly describe the signing and verifying process. As previously mentioned $\bar{F}$ is the public key with F, S, T composing the secret key. In order to sign a document $D = (x_1, \cdots, x_n)$, the signer uses the secret to calculate

$$X' = \bar{F}^{-1}(D) = T^{-1} \circ F^{-1} \circ S^{-1}(D)$$

which we claimed should be easy, knowing the secret key. The signer will offer X' as the signature accompanying D.

The verifier will use the public key $\bar{F}$ to check whether $\bar{F}(X')$ is indeed $\bar{F}(\bar{F}^{-1}(D)) = D$. The security of this signing process is usually assuming that a malicious party, knowing only the public key $\bar{F}$, will run into the difficulty of solving a multivariate systems of equations [4], thus will be unable to invert $\bar{F}$.

2.2 Matsumoto Imai

We will start with one of the first digital signature systems based on multivariate equations. As many others after it, this cryptosystem is based on the "hardness" of inverting, or solving a system of multivariate equations. The C* algorithm introduced by Matsumoto and Imai [6] describes how to "systematically" construct a system of low degree equations (so that verifying the plublic key is fast) with high degree inverses (so that a malicious party has a hard time finding the secret key).

2.2.1 Secret and Public keys

As detailed in the original paper, the secret key consists of multiple parts; first, S and T are two affine bijections $S, T : K_n \to K_n$, meant to hide the structure of our inner system of equations. μ and ϕ together represent K-isomorphisms, from the field K to the finite field K^n. At the core of this process, the system of equations K are raised to the power $1 + q^{\theta_i}$ where θ is chosen such that $\bar{F}$ will be an invertible map. This process can be summed up by the equation

$$p_k = \bar{F} = T^R \circ \mu \circ \phi \circ F \circ \phi^{-1} \circ \mu^{-1} \circ S^R(x)$$

while the secret key is $s_k = (S, T, \theta)$.

2.2.2 Signing and Verifying

The security is based on the difficulty of inverting the public key; if one owns the secret key, one can invert T, S, μ, ϕ and F, which will lead to a fast computation of the inverse and, thus, of generating a signature.

More specifically, in order to sign a document, the signer first generates the public and secret keys. In order to sign a document D, the signer finds the inverse of $\bar{F}$, which should be easy, knowing the secret (invertible) components T, S, μ, ϕ and F. Hence, the signature will be $X' = \bar{F}^{-1}(D)$. Anybody who wants to verify the signature will use the public key, $\bar{F}$, to compute $\bar{F}(X')$ and check whether it is or is not equal to the document D. The signature and verification process are fast, while trying to compute $\bar{F}^{-1}$ without the secret key should be hard for any malicious attacker [4].

2.2.3 Attacks

A paper by Patarin in 1995 [12] detailes a linearization attack in order to retrieve the secret key. There are variations of the Matsumoto Imai scheme, developed in order to patch up this vulnerability but we will not cover them in depth [13].

2.3 Oil Vinegar

In 1999, Patarin and Kipnis [14] develop a new multivariate signature scheme, based on the idea of "Oil-Vinegar" polynomials. An Oil-Vinegar polynomial is of the form

$$F_k(x) = \sum_{i=1}^{v}\sum_{j=1}^{o} a_{ijk} x'_i x_j + \sum_{i=1}^{v}\sum_{j=1}^{v} b_{ijk} x'_i x'_j + \sum_{i=1}^{o} c_{ik} x_i + \sum_{i=1}^{v} d_{ik} x'_i + e_k$$

The variables $a_{ijk}, b_{ijk}, c_{ik}, d_{ik}, e_k$ are the secret coefficients, $x_1, \cdots, x_o$ are called the "oil" variables and $x'_1, \cdots, x'_v$ are called the "vinegar" variables. The name hints at the fact that the oil variables do not mix, which will be helpful in solving this system of equations.

2.3.1 Secret and Public keys

Similarly to Matsumoto Imai [6], the scheme uses a secret bijective affine function $S : K^n \to K^n$, in order to hide the underlying structure of the equations.

The underlying difficulty of this scheme is once again the difficulty [4] of inverting the public key $\bar{F} = F \circ S$ without knowing both F and S separately.

Thus, Oil-Vinegar the secret key is the combination (F, S), enabling the signer to easily invert the public key $\bar{F}$.

2.3.2 Signing and Verifying

While Matsumoto Imai [6] relied on several other secret functions (μ, θ, ϕ), an Oil Vinegar polynomial can be inverted as is. The signer of a document $D = (y_1, \cdots, y_n)$ will try and find

$$F(x_1 \cdots x_o, x'_1 \cdots x'_v) = (y_1 \cdots y_n)$$

While this is a multivariate equation, the clever trick is to fix (guess) the values of the vinegar variables $(x'_1, \cdots, x'_v)$. Since our polynomials contain no oil-oil terms and because the terms are at most quadratic, guessing the vinegar variables will transform our problem into solving a system of linear univariate equations (where one could use Gaussian reduction). While some guesses might lead to no solutions, the signer could try again, knowing that the chance of no solutions is relatively low.

After finding a solution for

$$F(x_1, \cdots, x_o, x'_1, \cdots, x'_v) = (y_1, \cdots, y_n)$$

the signer outputs the signature $X' = S^{-1} \circ (x_1, \cdots, x_o, x'_1, \cdots, x'_v)$. The verifier will have to compute $F(X')$ and check if it is indeed equal to the signed document $D = (y_1, \cdots, y_n)$. As before, computing the signature X' without knowing the entire secret key is difficult to a malicious attacker [4].

2.3.3 Attacks

While the original scheme had the same number of oil and vinegar variables ("balanced"), security considerations indicate that the number of vinegar variables should be roughly $n^2/2$ [14].

2.4 HFE

Hidden Field Equations [5] are yet another variation of the Matsumoto and Imai scheme. It features a similar secret key, composed from two affine bijections S and T, meant to hide the structure of the inner multivariate equations

$$F(X) = \sum a_{ij} * X^{q^{\theta_{ij}} + q^{\phi_{ij}}} + \sum b_k * X^{q^{\epsilon_k}} + c$$

2.4.1 Secret and Public keys

Schemes relying on hidden field equations, generally keep S, T and F as the secret keys. This enables the signer, the owner of the secret, to be able to compute the inverse functions S^{-1}, T^{-1} in order to find the solution to the equation $F(X') = D'$. Finding a solution to this equation typically involves a specialised algorithm that runs in log polynomial time (e.g. Berlekamp algorithm [15]).

2.4.2 Signing and Verifying

The security of this scheme relies on the difficulty of inverting the public key $\bar{F} = T \circ F \circ S$, while the signer knowing the T, F and S has an easy time inverting and finding a solution $F(X') = D$ where D is the document to be signed. In order to invert the polynomials in F, Patarin points to the Berlekamp root finding algorithm [15], which is expected to run in log polynomial time [5].

The process for signing and verifying is fairly straightforward. The signer will use the secret keys T, F and S to find

$$\bar{F}^{-1}(D) = S^{-1} \circ F^{-1} \circ T^{-1}(D) = X'$$

where D is the document to be signed and X' is the resulting signature. The verifier will use the public key $\bar{F}$ to check whether $\bar{F}(X')$ is indeed equal to the signed document D.

2.4.3 Attacks

A point of weakness is detailed in the same paper by Patarin, where some choices for the polynomials F will leave the scheme more vulnerable to an affine multiple attack [5].

As a notable variant of this scheme, HFEv uses the HFE form of the equations F, while introducing "vinegar" variables. More precisely, Patarin introduces v variables $a' = (a'_1, \cdots, a'_v)$; the scheme uses these variables to express b_k as a secret random linear function depending on a' and c as a secret random quadratic function depending on a'. This will transform the HFE polynomial into a "oil vinegar" like polynomial.

$$F(X) = \sum a_{ij} * X^{q^{\theta_{ij}}+q^{\phi_{ij}}} + \sum b(a'_1, \cdots, a'_v) * X^{q^{\epsilon_k}} + c(a'_1, \cdots, a'_v)$$

This form is inspired by Oil-Vinegar schemes in that it enables the signer to invert the polynomial by guessing the vinegar variables and solving the resulting linear univariate polynomials.

3 Analysis of NIST algorithms

3.1 Rainbow

We will now start analysing multivariate digital signature schemes that have been submitted to the NIST Post-Quantum Cryptography standardisation process [3].

Rainbow [10], as the authors describe it, can be described as a generalised version of the Oil-Vinegar [14] scheme we have discussed earlier. It addresses security concerns, while at the same time promising a smaller signature size.

As was the case in oil-vinegar schemes, rainbow uses polynomials of the form

$$y_i = \sum_{i \in O_l, j \in S_l} a_{ij} * x_i * x_j + \sum_{i,j \in S_l} b_{ij} * x_i * x_j + \sum_{i \in S_{l+1}} c_i * x_j + e_i$$

where $x_i, i \in O_l$ are the oil variables and $x_i, i \in S_l$ are the vinegar ones.

To note here, is the construction of the O_l and S_l sets. More precisely, the cardinality of the vinegar sets S_l keeps increasing

$$S_1 \in S_2 \in, \cdots, \in S_u$$

and the oil sets O_l are constructed from the vinegar sets.

$$O_i = S_{i+1} - S_i, i = 1, \cdots, u-1$$

This can more easily be seen in Figure 1, taken from the original paper [10], where [] indicate the vinegar variables and () indicate the oil variables.

$$\begin{aligned}
&[x_1, \ldots, x_{v_1}]; \{x_{v_1+1}, \ldots, x_{v_2}\} \\
&[x_1, \ldots, x_{v_1}, x_{v_1+1}, \ldots, x_{v_2}]; \{x_{v_2+1}, \ldots, x_{v_3}\} \\
&[x_1, \ldots, x_{v_1}, x_{v_1+1}, \ldots, x_{v_2}, x_{v_2+1}, \ldots, x_{v_3}]; \{x_{v_3+1}, \ldots, x_{v_4}\} \\
&\ldots; \ldots \\
&[x_1, \ldots, \ldots, \ldots, \ldots, \ldots, \ldots, \ldots, \ldots, \ldots, \ldots, x_{v_{u-1}}]; \{x_{v_{u-1}+1}, \ldots, x_n\}
\end{aligned}$$

Figure 1: Rainbow layers

3.1.1 Secret and Public keys

As is the case for many of our schemes, the secret key consists of the polynomials F and two invertible affine linear maps S and T. In more detail, each rainbow layer (as seen in the figure above) has o_i polynomials with coefficients chosen randomly, where o_i is equal to the increase of vinegar variables ($o_i = v_{i+1} - v_i$). S and T are invertible linear maps (important for the signature process) which serve the purpose of hiding the structure of F. Thus, the secret key components are S, T and F and the public key is $\bar{F} = S \circ F \circ T$.

3.1.2 Signing and Verifying

To sign a document, the signer needs to use his secret key, in order to find a solution for the equation $\bar{F}(X') = D$, where D is the document to be signed. The secret key is helpful, in the sense that the signer can invert both S and T, which reduces the problem to solving $F(X') = D'$. Inverting this system of multivariate equations will rely on the special layered Oil-Vinegar structure we have described earlier. The inverting process starts at the first layer of o_1 equations and, similarly to simple Oil-Vinegar, randomly chooses values for the vinegar variables. This will turn the equations at this layer into easy to solve linear, univariate equations [16].

After solving the first layer, the algorithm now knows the previously guessed $x_1, \cdots, x_{v_1}$ as well as the previous oil variables $x_{v_1+1}, \cdots, x_{v_2}$. These variables are precisely the new vinegar variables of the second layer, which will enable us to find solutions for these now linear, univariate equations. This will go on, either until we find values for all $x_1, \cdots, x_n$, or until there exists no solution at a certain layer, case in which we restart the process with a new guess for the first layer $x_1, \cdots, x_{v_1}$.

The final signature X' for a document D is composed of the variables $x_1, \cdots, x_n$. In order to check the validity of the signature for a given document, the verifier will simply use the public key $\bar{F}$ and check whether $\bar{F}(X')$ is indeed equal to the document D.

3.1.3 Attacks

The Rainbow scheme is one of the most secure algorithms in terms of attacks targeting it. As it is based on Oil-Vinegar schemes, multiple attacks [17] [18] have been analysed and determined to be no more threatening than a brute force attack. It is surely still an open question, but the security of the well studied Oil-Vinegar scheme gives us confidence.

3.2 LUOV

LUOV [9] is another multivariate digital signature scheme that is based on classical Oil-Vinegar schemes. LUOV aims to drastically reduce the size of the public keys, by "lifting" the public key from the field $\mathbb{F}_2$ to $\mathbb{F}_2^r$. This extension will lead to all coefficient being 0 or 1, thus lowering the public key size. The paper authors point to Beullens and Preneel [19] for the theoretical background of this "lift" not affecting security, but attacks exploiting this property have been found, increasing the parameters by at least 40 percent [20].

3.2.1 Secret and Public keys

Similarly to other Oil-Vinegar schemes, LUOV uses a trapdoor $\bar{F} = F \circ T$ where T is a secret map and F is a collection of Oil-Vinegar polynomials of the form

$$F_k(x) = \sum_{i=1}^{v}\sum_{j=1}^{o} a_{ijk}x_i x_j + \sum_{i=1}^{v}\sum_{j=1}^{v} a_{ijk}x_i x_j + \sum_{i=1}^{o} b_{ik}x_i + \sum_{i=1}^{v} b_{ik}x_i + c_k$$

where v represents the number of the vinegar variables and o variables are called the oil variables.

Regular Oil-Vinegar schemes store the coefficients of F, T as the secret key, and P as the public key, but that might be an issue for memory limited devices. LUOV fixes this issue by using a psuedorandom

number generator, which enables to algorithm to generate the needed coefficients by storing a seed and a small part of the public key $\bar{F}$. We will analyse this in more detail in sections 4 and 5. More specifically, the secret key is a private seed, which is used by the pseudorandom number generator to generate a public seed and the linear secret map T. The public key consists of the public seed and a matrix Q_2, both used by the algorithm in order to calculate the public map $\bar{F}$.

3.2.2 Signing and Verifying

The security of the algorithm relies on the difficulty of finding a solution X' for the system of multivariate equations $\bar{F}(X') = D$ where D is the document to be signed, without knowing the secret components of $\bar{F}$. The signer, possessing the secrets F, T will be able use the inverse T^{-1} in order to solve $X' = F^{-1} \circ T^{-1}(D)$. In order to find a solution to $X' = F^{-1}(D')$ the signer will use the property of the Oil-Vinegar polynomial that the oil variables do not "mix" together. This will enable the signer to guess the v vinegar variables and then quickly solve a system of now linear univariate equations [16]. The solution X' is the signature of the document D and can be verified by using the public map $\bar{F}$ and checking whether $\bar{F}(X') = D$.

3.2.3 Attacks

One of the improvements LUOV brings is "lifting" the public key: usual Oil-Vinegar public keys are defined as $P : \mathbb{F}_2^n \to \mathbb{F}_2^m$ (meaning a system of m equations with n variables over a finite field of size 2); LUOV defines its public key as $P : \mathbb{F}_{2^r}^n \to \mathbb{F}_{2^r}^m$, using a larger extension field $\mathbb{F}_{2^r}$. This is inspired by [19]: the algorithm still chooses its public and secret keys over the field $\mathbb{F}_2$, but when representing it over the larger field $\mathbb{F}_{2^r}$, the coefficients become easier to store, while the security of the scheme should stay unaffected.

An attack by Ding [20] shows that this "lift" can be generally insecure, as for some r, a smaller subfield $\mathbb{F}_{2^d}$ can be found where d is a divisor of r. This will reduce the complexity of our attack and thus decrease the security of our scheme.

3.3 GeMSS

GeMSS is another digital signature algorithm based on multivariate equations. It is most closely related to the HFE variant, HFEv, promising fast signing and verifying times at the cost of a larger public key size. HFE schemes have been studied in depth without the discovery of any generic attacks against them.

3.3.1 Secret and public keys

The secret key of GeMSS [7] is composed of two invertible maps S, T, meant to hide the structure of the HFEv-like polynomial

$$F(X) = \sum_{0 \le j < i < n, 2^i + 2^j < D} A_{ij} X^{2^i + 2^j} + \sum_{0 \le i < n, 2^i \le D} \beta_i(v_1, \cdots, v_v) X^{2^i} + \gamma(v_1, \cdots, v_v)$$

The v variables $v_1, \cdots, v_v$ are called the vinegar variables and, importantly, all β_i are linear transformations while γ is quadratic. As the original paper authors point out, if the vinegar variables are fixed, the polynomial becomes a univariate HFE[5] polynomial, that can be solved in quasi-linear time: GeMSS uses the Berlekamp algorithm for finding the roots of a univariate polynomial [15].

Thus, the secret key consists of the invertible secret maps S, T and the n HFEv polynomials in $n + v$ variables. The public key $\bar{F}$ is obtained by taking the first m polynomials of

$$T \circ F \circ S(X, v_1, \cdots, v_{n+v})$$

The public key hides the vinegar variables in the central HFEv polynomials, thus making it difficult for an attacker to solve a system of multivariate equations [4].

3.3.2 Signing and verifying

As is the case in most digital signature systems based on multivariate equations, signing a document D involves the signer to find a solution to $\bar{F}(X') = D$. This is normally an NP hard problem [4], but the signer should be able to make the problem easier by knowing the secret key, F, S and T. Specifically, GeMSS first appends $n - m$ values to the document D (so that there are n equations again) and computes T^{-1}. Now, in order to solve the multivariate equation

$$F(X, v_1, \cdots, v_v) = D'$$

the signer randomly guesses the vinegar variables $V = v_1, \cdots, v_v$ and is now tasked with solving the univariate HFE equation $F(X, V) = D'$. GeMSS uses the Berlekamp algorithm for quickly solving univariate equations [15]. A found solution $\bar{X}$ is finally composed with S^{-1} to give the final signature

$$X' = S^{-1} \circ F^{-1} \circ T^{-1}(D)$$

In order to verify the signature X' accompanying the document D the verifier will use the public key $\bar{F}$ and check whether it is a solution to the equation $\bar{F}(X') = D$.

3.3.3 Attacks

As GeMSS is heavily influenced by HFE, one of the most studied families of multivariate digital signature schemes, none of the proposed attacks are able to harm its security. NIST mentions progress in the efficiency of the MinRank attack [18], which still does not contradict the theoretical security of GeMSS. As such, the security of this algorithm is one of its advantages.

3.4 MQDSS

MQDSS [8] is a digital signature scheme based on multivariate equations, but is dissimilar to all other algorithm we have analyzed so far. It does rely on the hardness of solving a system of multivariate equations F, but its main idea relies on the polar form $G(x, y)$ where

$$G(x, y) = F(x + y) - F(x) - F(y)$$

This polar form enables the algorithm to make use of the Sakumoto-Shirai-Hiwatari 5-pass IDS [21], where the secret and public keys, s and v respectively, are split, resulting in two "summands" which, by themselves, don't give any information of the secret s.

3.4.1 Secret and public keys

Similar to other digital signature schemes aiming to reduce the size of their keys, MQDSS uses a pseudorandom number generator, in order to be able to store the secret key s_k as a seed. This is used as input for the random generator, which outputs a seed S_F which further generates the multivariate system F, the input of the system s and the vectors r_0, t_0 and e_0 which will be necessary for the signing process.

Thus, the public key is $(S_F, F(s))$ where S_F is the seed used to generate F (seeds are used repeatedly in order to reduce key sizes).

The secret key remains the secret s_k, meaning the variables r_0, t_0, e_0 and the system input s.

3.4.2 Signing and verifying

The process of signing a document and verifying said signature is unlike other multivariate digital signature schemes we have analysed so far. It is based on identification scheme protocols, protocols in which the verifier and signer exchange challenges and responses in order to validate the authenticity of the signature.

In particular, MQDSS is inspired by the Fiat-Shamir transform [22] : knowing a predetermined challenge function $H(pk, M)$, the signer will be able to use the secret key s_k in order to return as signature a valid response *resp* (its validity depending on the nature of the challenge function).

In order to verify a signature, the Fiat-Shamir transform processes $H(p_k, M)$ and *resp* and determines if the challenge was satisfied.

In the case of MQDSS, the challenge revolves around the polar form and the secret splitting of the secret key $s_k = r_0 + r_1$ as well as subsequent splits $\alpha F(r_0) = e_0 + e_1$ and $\alpha r_0 = t_0 + t_1$. The security of this signature relies on the ability to send some coefficients, without revealing anything about the secret key s_k.

3.4.3 Attacks

While MQDSS still relies on the hardness of solving a system of multivariate equations, its construction leaves it open to a set of different attacks. In particular, a recent forgery attack [23] takes advantage of a vulnerability of identification schemes, namely the ability of the attack to guess one of the two challenges.

This attack increases both the necessary key sizes and number of needed rounds by almost 40 percent [23], thus severely harming the efficiency of this scheme.

4 Comparison of key sizes

name	security level	parameters	|pk| (KB)	|sk| (KB)	|sig| (byte)
GeMSS128	1	(128, 513, 174, 12, 12, 4)	417	15	32
Rainbow	1	(GF(16), 36, 32, 32)	154	99	66
LUOV	1	(7, 57, 197)	12	0.03	239
MQDSS	1-2	(128, 31, 48, 135)	0.05	0.02	20854
GeMSS192	3	(192, 513, 265, 22, 20, 4)	1304	40	51
Rainbow	3	(GF(256), 68, 32, 36)	841	597	164
LUOV	3	(7, 83, 283)	35	0.03	337
MQDSS	3-4	(192, 31, 64, 202)	0.06	0.02	43728
GeMSS256	5	(256, 513, 354, 30, 33, 4)	3046	84	72
Rainbow	5	(GF(256), 96, 36, 64)	1841	1343	212
LUOV	5	(7, 110, 374)	82	0.03	440

Table 1: Key sizes

We will now be looking at the different key sizes of all NIST multivariate digital signature algorithms. We have separated them by the NIST security levels and we have included the the parameter sets chosen by the authors of the respective papers [8] [10] [9] [7]. For most schemes there are additional modes of operation with some kind of trade off between performance and key sizes; we will address them but haven't included all of them in our table. Our findings can be found in Table 1.

4.1 GeMSS

Starting with GeMSS, it offers very small signature sizes, a characteristic shared with the other oil-vinegar scheme Rainbow. This is due to only needing to find a solution in $\mathbb{F}_2^{o+v}$ to the equation $\bar{F}(X') = D$. The secret key is relatively small, the difference between it and Rainbow being the smaller inner polynomial F. The size of the public key is the biggest drawback of GeMSS, disadvantage that is not fixed by the BlueGeMSS and RedGeMSS variants (which drastically decrease signing performance at the cost of marginally bigger keys).

4.2 Rainbow

Rainbow, the other unbalanced oil-vinegar scheme, has similar key sizes to GeMSS. Its public key size is smaller, due to the simpler form of the inner polynomial F (oil-vinegar polynomial versus a HFEv polynomial). We note that the secret key is larger, but this is less of a drawback: the memory of a device signing a document is less stringent than the memory of the many devices who wish to verify it. Additionally, the Rainbow variant CZ-Rainbow is able to further reduce the public key size by a factor of 3 at the cost of marginally longer key generation times.

4.3 LUOV

LUOV offers the smallest public key size, amongst the oil-vinegar schemes. This is due to the lifting assumption, the fact that the public key $\bar{F}$ is lifted from $\mathbb{F}_2^n \to \mathbb{F}_2^m$ to $\mathbb{F}_{2^r}^n \to \mathbb{F}_{2^r}^m$. This lift reduces the size of the public key coefficients, but has been a relatively unstudied security assumption. More so, LUOV uses both private and public seeds to generate the respective keys. This further reduces the size of the otherwise lengthy public key, as well as the secret key. This improvement comes at the cost of lengthier signature and verification procedures, trade off which we will investigate in section 5.3.

4.4 MQDSS

While the authors of MQDSS did not include parameters for the most difficult security level, we can see the key sizes are not like other typical multivariate digital signature schemes. The underlying Fiat-Shamir transform [22] for 5-pass Identification Schemes includes attaching all challenges and responses in the signature, thereby increasing its size significantly, compared to the other schemes. While, it does offer promisingly small public and secret key sizes, NIST [24] deemed it most similar but inferior to the best symmetric-based signature schemes, category of schemes we have not analysed in our research.

5 Comparison of running times

name	security level	parameters	key generation (cycles)	sign (cycles)	verification (cycles)
GeMSS128	1	(128, 513, 174, 12, 12, 4)	1.9 G	6.7 G	29 M
Rainbow	1	(GF(16), 36, 32, 32)	32 M	319 K	41 K
LUOV	1	(7, 57, 197)	13 M	4 M	3 M
MQDSS	1-2	(128, 31, 48, 135)	1 M	27 M	20 M
GeMSS192	3	(192, 513, 265, 22, 20, 4)	7.9 G	15.1 G	89 M
Rainbow	3	(GF(256), 68, 32, 36)	197 M	1 M	203 K
LUOV	3	(7, 83, 283)	60 M	14 M	10 M
MQDSS	3-4	(192, 31, 64, 202)	3 M	85 M	62 M
GeMSS256	5	(256, 513, 354, 30, 33, 4)	20.5 G	25.3 G	172 M
Rainbow	5	(GF(256), 96, 36, 64)	436 M	2 M	362 K
LUOV	5	(7, 110, 374)	136 M	28 M	21 M

Table 2: Number of cycles on Reference platform

We will now analyse the different performance of the multivariate digital signature schemes presented in NIST. As in the previous section, we have included the reference parameters the author of the original papers have chosen themselves [8] [10] [9] [7]. We have included only the standard schemes, but will briefly talk about some of the variants the authors have proposed. All running times have been taken from the original papers, running on processors with a similar architecture. We will be looking at the number of CPU cycles, averaged out over a large amount of runs by the authors (due

name	security level	parameters	key generation (cycles)	sign (cycles)	verification (cycles)
GeMSS128	1	(128, 513, 174, 12, 12, 4)	52 M	1.2 G	150 K
Rainbow	1	(GF(16), 36, 32, 32)	11 M	85 K	42 K
LUOV	1	(7, 57, 197)	2 M	1 M	1 M
MQDSS	1-2	(128, 31, 48, 135)	1 M	4 M	2 M
GeMSS192	3	(192, 513, 265, 22, 20, 4)	273 M	3.5 G	439 K
Rainbow	3	(GF(256), 68, 32, 36)	60 M	348 K	162 K
LUOV	3	(7, 83, 283)	6 M	3 M	3 M
MQDSS	3-4	(192, 31, 64, 202)	3 M	9 M	6 M
GeMSS256	5	(256, 513, 354, 30, 33, 4)	844 M	7.1 G	943 K
Rainbow	5	(GF(256), 96, 36, 64)	217 M	857 K	423 K
LUOV	5	(7, 110, 374)	13 M	6 M	4 M

Number of cycles on architecture supporting AVX2

Table 3: Number of cycles on architecture supporting AVX2

to the deterministic nature of some schemes). The results can be found in Table 2. Additionally, we have included the performance of the schemes on a newer Haswell/Skylake CPU architecture. This new architecture enables AVX2 instructions, which optimize some vector operations. In the following subsection we will only discuss the results of the optimised Table 3.

5.1 GeMSS

GeMSS offers key generation and signature verification similar to Rainbow. They similarly have to generate the invertible maps S, T and the inner polynomial F. While F requires more storage, generating it is comparably fast. The signature verification process similarly has to check the validity of a signature X' in the equation $\bar{F}(X') = D$. The drawback of this schemes lies in the complicated signature process, due to the application of the Berlekamp algorithm in finding a solution to the HFE polynomial (after guessing the vinegar variables) [7] [15].

5.2 Rainbow

On the other hand the Rainbow scheme offers a very small signature time due to the simpler nature of the underlying polynomial $\bar{F}$. After guessing the vinegar variables in the first layer, the algorithm repeatedly performs Gaussian elimination in order to get the final signature X'. We attribute the speed of the signature process to the conceptual simplicity of the performed Gaussian elimination [16]. We note that the performance of CZ-Rainbow, variant which reduces the size of the public key by a factor of 3, is similarly fast, with the exception of a much slower verification time.

5.3 LUOV

LUOV offers mixed performance, when compared to the fast Rainbow scheme. Key generation is fast, due to the usage of pseudorandom number generators and storing the used seeds. This comes at the cost of higher signature and verification times: the generation of the invertible matrix T and the inner polynomial F is delayed to these later phases. This is similar to the performance of Compressed Rainbow, which also uses pseudorandom number generators in order to obtain smaller key sizes at the cost of slower performance.

5.4 MQDSS

MQDSS offers fast key generation times, due the small amount of variables (r_0, t_0, e_0) that need to be generated but also due to the pseudo random number generator that is used to delay the creation of some vectors in the signature phase. Verification time is slower than other schemes we have looked at: this is due to the verifier no longer checking the signature X' in $\bar{F}(X') = D$, but rather a more complex 5-pass identification scheme. The verifier receives all challenges and answers and must parse them several times in order to check its validity.

6 Proposed improvement of GeMSS

In the previous sections, we covered Oil-Vinegar schemes, category which contains two NIST schemes, Rainbow and GeMSS. In this section, we will focus on the similarities between the 2 algorithms and propose an improvement to the GeMSS public key generation.

GeMSS is a digital signature algorithm based on an HFEv polynomial. The underlying structure is different than that of Rainbow in the structure of the inner polynomial F. Some of the drawbacks include large signing times (due to the complexity of solving $F(X') = D'$) as well as large public key sizes.

Rainbow is an Oil-Vinegar scheme: it has a similar structure (two invertible secrets S, T and a polynomial F) which contributes to a large public key.

We would like to draw attention to the Compressed Rainbow variant, variant which offers greatly reduced public and secret key sizes at the cost of slower signature generation and verification. This trade off is accomplished by randomly choosing S, T using a pseudo random number generator. The public key becomes the seed used in the generator and a small part (30 percent) of the previous public key. This procedure is similarly used in LUOV.

We propose a variant of GeMSS which has smaller key sizes at the cost of a slower signing procedure, similar to the version of Compressed Rainbow. We note that the key generation algorithm detailed in the original paper already samples S, T, F at random, giving us confidence in the ability to use a pseudo random generator. This would include changing the secret and public keys to store the used seeds s_{pub}, s_{priv}. Furthermore, the signing and verification subroutines need to be changed such that the scheme generates S, F, T and $\bar{F}$ respectively, from the given public and private seeds.

This approach would further hinder the performance of the GeMSS signing and verification process, but could be very important in instances which necessitate small public keys (e.g. Certificate chains).

7 Discussion

In this section, we will analyse all multivariate digital signature algorithms submitted to NIST and some of their variants. We will summarise our findings and formulate advantages and disadvantages to each. We will mention any found attacks and will reason which use cases suit which algorithm.

7.0.1 Rainbow

The Rainbow scheme [10] is based on the well studied Oil-Vinegar scheme. It is fast and by building upon such a well studied problem, its security has not been exploited. Its main disadvantage is the large size of its public key, disadvantage that is addressed by its slightly slower variant CZ-Rainbow. CZ-Rainbow uses a pseudorandom number generator in order to not store the entire public key $\bar{F}$, but rather the used seed and a small portion of it, reducing the size of the key by 70 percent. Being able to choose this trade off, in the cases where storage is limited, gives us confidence that Rainbow can function as a general purpose digital signature scheme.

7.0.2 LUOV

LUOV [9] is another Oil-Vinegar scheme, with its main advantage being the very small size of the (usually large) public key. In order to achieve this, LUOV uses a pseudorandom number generator in order to not store the entire public key, but rather a part of it and the used seed. This increases signature and verification performance, which might be important in some cases. Additionally LUOV lifts the public key from $\mathbb{F}_2^n \to \mathbb{F}_2^m$ to $\mathbb{F}_{2^r}^n \to \mathbb{F}_{2^r}^m$. This further reduces the size of the public key, but the security of this assumption is not well studied. As a response to the NIST submission, a differential attack [20] takes advantage on the possibility of a smaller subfield for some parameter choices. The authors have mitigated this attack in the latest version of LUOV, but NIST did consider this assumption insufficiently studied [24]. We consider this algorithm to be similar to Compressed Rainbow, but the lift security assumption needs to be additionally studied.

7.0.3 GeMSS

GeMSS [7] is a scheme based on Hidden Field Equations. The small size of its signature and the security of the well studied HFE scheme are two main advantages.

The key sizes are similar to Rainbow, but its BlueGeMSS and RedGeMSS variants do not offer a similar trade off between a smaller public key size at the cost of slower performance. The two variants only affect the choices of the parameters, and enable a trade off between an even larger public key for the purpose of a faster signing process.

Thus, as other Oil-Vinegar schemes, GeMSS suffers from large public keys. GeMSS also suffers from slow signature time, due to the complicated operations needed to find a solution to the underlying equation $F(X') = D$. We consider this algorithm to be inferior to Rainbow, with similar but more accentuated drawbacks.

7.0.4 MQDSS

MQDSS [8] is a multivariate digital signature scheme, constructed upon a SSH 5-pass Identification Scheme [21]. Thus, it does not possess some of the common characteristics of the other schemes. Its key sizes are small and performance is some of the best in the group. Having said this, its signature size is vastly higher than the other multivariate schemes: NIST compares [24] it most closely with symmetrical key signature schemes, outside the scope of our study. Furthermore, the security of the scheme is mathematically proved, but it is vulnerable to a forgery attack, common attack against algorithms relying on Identification Schemes. Without having done further investigation in another category of digital signature schemes, we believe the opinion of NIST [24], that the algorithm has inferior performance to other candidates, while also being insecure.

8 Responsible Research

The authors of this research were not affiliated directly or indirectly with any of the stakeholders of the analysed algorithms or with the government agency carrying out the standardisation process. Nonetheless we acknowledge some factors which might influence the integrity of the carried out research. Namely, the discussion weighing the advantages and disadvantages of each algorithm could have been influenced by the plethora of other opinions from related work. The data used in the research is publicly available and reproducible, but there could be implicit bias favoring the algorithms which have performed better in other analyses.

As far as the ethical aspect of the performed research, the investigation was done theoretically, with no human experiments. The goal of the research is to improve digital signature schemes, schemes which play role in validating authorship of documents, thereby preventing instances of fraud, forgery or tampering. The research has investigated attacks against said algorithms, for the purpose of dis-

qualifying potentially insecure algorithms. More so, none of the analysed algorithms are in present use, so the investigation of possible attacks cannot, yet, enable unethical agents.

9 Conclusions and Future Work

The purpose of this report was to investigate digital signature schemes based on multivariate equations as submitted in the NIST standardisation process. This paper contains an in depth overview of related work, especially on schemes which have influenced the current NIST schemes. The research has focused on the underlying mathematical problems, highlighting similarities and differences between the algorithms. Furthermore, the schemes were compared on the basis of their key sizes, performance, as well as known attacks damaging their security. A point of improvement was suggested for one of the algorithms, leaving the implementation and documentation of the results to future research. This study made some preliminary recommendation on the overall quality of analysed algorithms, while acknowledging the need for further investigation.

References

[1] Peter W Shor. "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer". In: *SIAM review* 41.2 (1999), pp. 303–332.

[2] Lange T Smart N. *Post-Quantum Cryptography: Current state and quantum mitigation*. Tech. rep. European Union Agency for Cybersecurity, 2021.

[3] Gorjan Alagic et al. *Status report on the first round of the NIST post-quantum cryptography standardization process*. US Department of Commerce, National Institute of Standards and Technology âŠ, 2019.

[4] Nicolas Courtois et al. "Efficient algorithms for solving overdefined systems of multivariate polynomial equations". In: *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2000, pp. 392–407.

[5] Jacques Patarin. "Hidden fields equations (HFE) and isomorphisms of polynomials (IP): Two new families of asymmetric algorithms". In: *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 1996, pp. 33–48.

[6] Tsutomu Matsumoto and Hideki Imai. "Public quadratic polynomial-tuples for efficient signature-verification and message-encryption". In: *Workshop on the Theory and Application of of Cryptographic Techniques*. Springer. 1988, pp. 419–453.

[7] Antoine Casanova et al. "GeMSS: a great multivariate short signature". PhD thesis. UPMC-Paris 6 Sorbonne Universités; INRIA Paris Research Centre, MAMBA Team âŠ, 2017.

[8] Ming-Shing Chen et al. "MQDSS specifications". In: *NIST PQC Round* 2 (2020), p. 13.

[9] Ward Beullens et al. "LUOV: Signature scheme proposal for NIST PQC project". In: (2019).

[10] Jintai Ding and Dieter Schmidt. "Rainbow, a new multivariable polynomial signature scheme". In: *International Conference on Applied Cryptography and Network Security*. Springer. 2005, pp. 164–175.

[11] Jintai Ding, Jason E Gower, and Dieter S Schmidt. *Multivariate public key cryptosystems*. Vol. 25. Springer Science & Business Media, 2006.

[12] Jacques Patarin. "Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocryptâ88". In: *Annual International Cryptology Conference*. Springer. 1995, pp. 248–261.

[13] Jacques Patarin, Louis Goubin, and Nicolas Courtois. "C-+* and HM: Variations around two schemes of T. Matsumoto and H. Imai". In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 1998, pp. 35–50.

[14] Aviad Kipnis, Jacques Patarin, and Louis Goubin. "Unbalanced oil and vinegar signature schemes". In: *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 1999, pp. 206–222.

[15] Elwyn R Berlekamp. "Factoring polynomials over finite fields". In: *Bell System Technical Journal* 46.8 (1967), pp. 1853–1859.

[16] Daniel J Bernstein, Tung Chou, and Peter Schwabe. "McBits: fast constant-time code-based cryptography". In: *International Conference on Cryptographic Hardware and Embedded Systems*. Springer. 2013, pp. 250–272.

[17] Jintai Ding et al. "New differential-algebraic attacks and reparametrization of rainbow". In: *International Conference on Applied Cryptography and Network Security*. Springer. 2008, pp. 242–257.

[18] Magali Bardet et al. "Improvements of Algebraic Attacks for solving the Rank Decoding and MinRank problems". In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2020, pp. 507–536.

[19] Ward Beullens and Bart Preneel. "Field lifting for smaller UOV public keys". In: *International Conference on Cryptology in India*. Springer. 2017, pp. 227–246.

[20] Jintai Ding et al. "Cryptanalysis of The Lifted Unbalanced Oil Vinegar Signature Scheme". In: *Annual International Cryptology Conference*. Springer. 2020, pp. 279–298.

[21] Koichi Sakumoto, Taizo Shirai, and Harunaga Hiwatari. "Public-key identification schemes based on multivariate quadratic polynomials". In: *Annual Cryptology Conference*. Springer. 2011, pp. 706–723.

[22] Amos Fiat and Adi Shamir. "How to prove yourself: Practical solutions to identification and signature problems". In: *Conference on the theory and application of cryptographic techniques*. Springer. 1986, pp. 186–194.

[23] Daniel Kales and Greg Zaverucha. *Forgery Attacks on MQDSSv2. 0*. 2019.

[24] Gorjan Alagic et al. "Status report on the second round of the NIST post-quantum cryptography standardization process". In: *US Department of Commerce, NIST* (2020).