UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

Matheus Silva Pinheiro Bittencourt

REDUCING KEYS IN RAINBOW-LIKE SIGNATURE SCHEMES

Florianópolis

2019

Matheus Silva Pinheiro Bittencourt

REDUCING KEYS IN RAINBOW-LIKE SIGNATURE SCHEMES

Monografia submetida ao Programa de Graduação em Ciência da Computação para a obtenção do Grau de Bacharel em Ciência da Computação.
Orientador: Prof. Ricardo Felipe Custódio, Dr.
Coorientador: Gustavo Zambonin, Bel.

Florianópolis

2019

Matheus Silva Pinheiro Bittencourt

REDUCING KEYS IN RAINBOW-LIKE SIGNATURE SCHEMES

Esta Monografia foi julgada aprovada para a obtenção do Título de "Bacharel em Ciência da Computação", e aprovada em sua forma final pelo Programa de Graduação em Ciência da Computação.

Florianópolis, 11 de Novembro 2019.

Prof. José Francisco Danilo de Guadalupe Correa Fletes
Coordenador

Banca Examinadora:

Prof. Ricardo Felipe Custódio, Dr.
Orientador

Gustavo Zambonin, Bel.
Coorientador

Prof. Daniel Panario, Dr.
Carleton University

Lucas Pandolfo Perin, Me.
Universidade Federal de Santa Catarina

A Fátima, que infelizmente não pode acompanhar minha trajetória, mas tenho certeza de que estaria orgulhosa.

AGRADECIMENTOS

Agradeço aos meus pais que me fizeram capaz de entender e percorrer os caminhos do conhecimento e da vida. A toda a minha família que nunca me faltou com apoio sempre que precisei, em especial ao meu padrinho Fernando e a minha madrinha Daniela. Ao meu orientador Prof. Ricardo e ao Prof. Daniel pela ajuda no desenvolvimento deste trabalho com as inúmeras e produtivas reuniões. Ao Gustavo que me proporcionou a oportunidade de ser orientado por um amigo, e que contribui enormemente para este trabalho. Ao Douglas Silva e ao Douglas Martins que me orientaram em diversos aspectos da vida acadêmica e profissional. A todos os amigos que fiz dentro e fora do curso, que me acompanharam e me ajudaram direta ou indiretamente a percorrer esse trajeto. Em especial, agradeço ao Lucas e ao Vinicius que sempre me motivaram a vencer as batalhas enfrentadas com o ânimo e a dedicação necessária. Por último, e com certeza não menos importante, agradeço ao Mateus e ao Luís pela cumplicidade e companhia mesmo quando nada parecia bem, e por todas as pessoas incríveis que eles me proporcionaram conhecer.

"*Ce qui embellit le désert, c'est qu'il cache un puits quelque part...*"

Le Petit Prince

RESUMO

Os algoritmos clássicos de assinatura digital como RSA e ECDSA baseiam sua segurança na dificuldade da fatoração de inteiros, e no logaritmo discreto, respectivamente. Esses problemas já possuem algoritmos quânticos que os resolvem em tempo polinomial, ou seja, com computadores quânticos poderosos o suficiente, o uso dos algoritmos de assinatura digital mais difundidos tornará-se impraticável. Naturalmente, com o aumento do poder computacional quântico, o interesse por criptossistemas resistentes a ataques que utilizam-se de tais computadores também cresceu. A área que estuda esses criptossistemas é chamada de criptografia pós-quântica. Particularmente, esses algoritmos baseiam-se numa série de problemas que, por enquanto, permanecem difíceis, mesmo que computadores quânticos poderosos sejam utilizados, logo, despertam o interesse para substituir os criptossistemas clássicos. Este trabalho aborda criptossistemas baseados em sistemas de polinômios multivariados, que, baseiam-se em problemas como a solução de sistemas de polinômios e o isomorfismo de polinômios, os quais ainda são resistentes a algoritmos quânticos, e portanto, são candidatos para criptografia pós-quântica. Tais esquemas possuem tamanhos de chaves muito maiores que os algoritmos clássicos. Neste trabalho um novo método para redução de chaves privadas do esquema de assinatura digital Rainbow é proposto. Usando este método as chaves privadas podem ser reduzidas em até 84%. Ainda, este método pode ser combinado com outros de forma a reduzir tanto a chave privada como a chave pública.

Palavras-chave: criptografia pós-quântica, assinatura digital, Rainbow

ABSTRACT

Classic digital signature algorithms base their security upon the difficulty of the integer factorization problem, and the discrete logarithm problem, respectively. These problems already have quantum algorithms that solve them in polynomial time, consequently, with sufficiently powerful quantum computers, the use of the most common digital signature algorithms would become impractical. Naturally, with the rise in quantum computational power, the interest in cryptosystems resistant to attacks that make use of such computers has raised as well. The area that studies such cryptosystems is called post-quantum cryptography. Particularly, these algorithms are based upon a series of problems that, at this time, continue to be hard, even with quantum computers available, hence, provoke interest to substitute the classical schemes. This work approaches cryptosystems based on systems of multivariate polynomials. They base their security upon problems like the polynomial system solving and the isomorphism of polynomials, which are still resistant to quantum computers, henceforth are candidates to post-quantum cryptography. Such schemes have much larger keys than classical algorithms. In this work a new method that allows the reduction of private keys of the Rainbow digital signature scheme is proposed. Using this method, private keys can be reduced by up to 84%. Still, this method can be combined with others to reduce the private key and the public key simultaneously.

Keywords: post-quantum cryptography, digital signatures, Rainbow

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

TABLE OF CONTENTS

1 INTRODUCTION

Classic asymmetric cryptography is threatened as a result of the advance in quantum algorithms development. The hardness of recovering private keys, for instance, RSA and ECDSA keys, relies, respectively, on the hardness of the integer factorization problem and the discrete logarithm problem. Quantum polynomial-time algorithms that solve those problems already exist (SHOR, 1999). Using such algorithms, recovering private keys can be done efficiently by a sufficiently powerful quantum computer. Hence, the most used digital signature algorithms would become insecure.

In such a scenario, cryptosystems that run on classical computers and cannot be broken by quantum computers should be used for handling digital signatures. The area that studies such cryptosystems is called post-quantum cryptography, and the interest in this has emerged with the development of quantum computers. The security of these algorithms relies on problems that are not known to be solvable in polynomial-time, therefore, they appear to be good candidates for use in a scenario of quantum adversaries.

There are several classes of post-quantum cryptosystems proposed in the literature, and each of them relies on one kind of hard problem. This work aims to study Multivariate Public-Key Cryptosystems (MPKCs) based on Rainbow (DING; SCHMIDT, 2005). These cryptosystems are constructed using multivariate polynomials systems. A polynomial system is usually composed of polynomial equations with single variable monomials, and can be easily solved using, for instance, Gaussian elimination. However, with the inclusion of more variables into the monomials, it becomes a multivariate system. The problem of solving multivariate systems is NP-Hard (GAREY; JOHNSON, 1979). Therefore, it may be interesting to build cryptosystems based on this trait.

Several digital signature schemes were developed based on the structure of multivariate systems. One of the first schemes presented was the Oil and Vinegar (OV) signature scheme (PATARIN, 1997), which was broken by Kipnis and Shamir in its original specification (KIPNIS; SHAMIR, 1998). A subsequent work (KIPNIS; PATARIN; GOUBIN, 1999) reparametrized it, leading to a scheme called Unbalanced Oil and Vinegar. The trapdoor introduced in the original OV is used in many other schemes. In essence, they are similar to OV but ended up optimizing the signature, and key sizes, while maintaining security levels. These schemes are still considered secure.

MPKCs have very efficient signature generation and verification algorithms, as well as small signatures, in some cases, smaller than classic algorithms. The main caveat of such schemes is that they have large public

and private keys. Various efforts were made to reduce such parts of these cryptosystems, but, to the best of our knowledge, none of them reduced both public and private keys. This work aims to understand the Rainbow signature scheme, its subsequent optimized schemes and the key reduction techniques proposed in the literature. Finally, a new framework that allows for a reduction of both the private and the public keys in Rainbow-like schemes is proposed.

1.1 GOALS AND SCOPE

General goal: Study and describe Rainbow-like digital signature schemes, understand the optimizations that reduce keys and signature sizes, and their impact on the security of the classic schemes. Observe and analyze the impact of parameter selection for such algorithms, as this plays an important role in efficient and fast implementations of the schemes. Analyze the state-of-the-art schemes, to understand the strategies being used to optimize the cryptosystems.

Specific goals: Describe the classic OV and the UOV digital signature schemes; Describe the Rainbow signature scheme; Introduce relevant optimizations on Rainbow, like CyclicRainbow (PETZOLDT; BULYGIN; BUCHMANN, 2010); Compare and analyze the performance of the aforementioned schemes in terms of operations needed to generate and verify signatures as well as storage requirements. Finally, propose new optimizations on top of those schemes.

Scope: This work is focused on the Rainbow digital signature scheme, its ancestors, and the optimizations made on top of it. Other classes of post-quantum algorithms such as code-, lattice- and hash-based cryptosystems are not covered. Classical asymmetric algorithms are not covered as well. Quantum algorithms are also not discussed.

1.2 METHOD

The work was developed using the infrastructure and resources provided by the Computer Security Laboratory (LabSEC/UFSC). A literature review was made to determine the state-of-the-art in MPKCs. Recently proposed schemes were studied, as well as broken ones for a better understanding of the constructions used that optimize the classic multivariate schemes. The performance of all the schemes studied were observed along with the impact of the optimizations.

2 CRYPTOGRAPHY AND ALGEBRA BACKGROUND

In this chapter some basic background on cryptography and algebra is given for a complete understanding of the following chapters. All definitions and descriptions are based on (STINSON, 2006).

2.1 PUBLIC KEY CRYPTOGRAPHY

Symmetric cryptography consists of ciphering and deciphering a message with the same key. This can be very useful if two parties, say Alice and Bob, have agreed upon using the same key through a secure channel. But if Alice and Bob are distant and do not have a secure channel to share a symmetric key, this cannot be used. Public key cryptography solves this issue, as there exists two keys. One of the keys is public, everyone can have access, and it is used to encrypt a message. The other key is private and can be held only by its owner, no one else can access it. Deciphering a message can only be made with the private key. Also, getting the private key from the public key should be computationally unfeasible. Now, if Bob wants to send a private message to Alice, he gets her public key through any channel of communication, encrypts the message, and sends it to Alice. Alice, which possesses the private key, is the only one capable of decrypting Bob's message.

The idea of public key cryptosystems was first introduced in (DIFFIE; HELLMAN, 1976), but the first practical scheme of such kind was proposed in (RIVEST; SHAMIR; ADLEMAN, 1977). These algorithms can be thought as a trapdoor one-way function, i.e. the encryption process "traps" the message, and only with the possession of the private key, one can untangle the original message from the trap. The security of these algorithms rely on the difficulty of getting the private key from the public one, that is, it should be hard to unravel the message from the trap.

2.2 DIGITAL SIGNATURES

Conventional signatures are used everyday to provide authenticity of documents, like letters and contracts. A digital signature is a signature to a digital document, that can be transmitted across a digital medium. Digitally signed documents have considerable differences to conventional signatures. First, a digital signature is not physically bonded to the signed document. Second, the verification procedure of the signature is done in a different way

than classical signatures. Conventional signatures are verified by similarity to other signatures that are trusted. Digital signatures are verified using publicly known algorithms. Also, a digital signature can be copied several times, and it remains authentic. Care should be taken, such that a document is not used more times than it should be. For instance, a signed document saying that Alice wants to transfer some amount of money to Bob should be used only once, as Bob could force the bank to transfer this same amount multiple times. Adding the time the signature was made to the signed document can mitigate this problem.

A digital signature scheme is composed of three efficient[1] algorithms:

- **Key generation:** Alice generates a key pair, publishes the public key to anyone who wants to verify her signatures, and keeps the private key secret.

- **Signature generation:** Alice, possessing the private key, signs a document d, yielding a signature σ, and publishes the pair (d, σ). For each different document, a different signature is produced.

- **Signature verification:** Bob wants to verify Alice's signature (d, σ). With the public key, Bob can verify that, indeed, only Alice's private key could generate σ for d. If someone altered d to insert malicious information and sent this modified document d', to Bob, the signature σ would not be verified by the verification algorithm.

This can be seen as the inverse of sending private messages using public key cryptography, as the owner of the private key ciphers the message to be signed. Recall that it should be hard to find the private key from the public one, thus it is hard to forge new signatures without possessing the private key.

A secure digital signature scheme should provide three properties to the signed documents, as long as they are valid:

- **Integrity:** The document is untouched compared to the originally signed document. Flipping one bit of the document would render the signature invalid.

- **Authenticity:** Only the owner of the private key can generate valid signatures for the corresponding public key. So, if a document is signed, it could only be signed be the private key owner. Generating new signatures, without the private key, should be computationally unfeasible for secure schemes.

[1] Can be computed in a reasonable time by the parties involved.

- **Non-repudiation:** The signer cannot deny having created a valid signature σ for a document d. This is due to the fact that, it is unfeasible to forge signatures.

2.3 CRYPTOGRAPHIC HASH FUNCTIONS

In the scope of this work, a hash function is a function that is used to map arbitrarily sized data to a fixed length value, i.e. $\mathcal{H} : \{0,1\}^* \to \{0,1\}^n$. Such functions are useful to maintain integrity of documents, for instance, a document d has this fingerprint $h = \mathcal{H}(d)$, which can be verified whenever one wants to ensure that d was not modified. If for some reason the document was changed, this modified version d' would have a different fingerprint $h' = \mathcal{H}(d')$, and the integrity of the original document would not be verified because $h \neq h'$.

Of course, due to the "compression" made by $\mathcal{H}$, there ought to exist two documents that generate the same fingerprint, that is $h = \mathcal{H}(d_1) = \mathcal{H}(d_2)$ and $d_1 \neq d_2$. This happens due to the fact that there exists infinitely more documents than fingerprints, so by the pigeonhole principle there are infinitely many documents with the same fingerprint. This can be a problem, because the integrity of those documents could be erroneously verified. However, in practice, for secure hash functions, this happens with negligible probability for a large enough n.

Hash functions are extensively used in digital signatures. It enables the algorithms to sign arbitrarily big documents by signing only the fingerprint h. However, a hash function should satisfy some properties to be considered cryptographically secure. Specially, three problems should be unfeasible to solve. Let $\mathcal{H} : X \to Y$ be a hash function:

- **Preimage:** Given $y \in Y$, find $x \in X$ such that $y = \mathcal{H}(x)$.
- **Second preimage:** Given $x \in X$, find $x' \in X$, $x' \neq x$ such that $\mathcal{H}(x) = \mathcal{H}(x')$.
- **Collision:** Find x, $x' \in X$ such that $\mathcal{H}(x) = \mathcal{H}(x')$.

Suppose that Alice is using $\mathcal{H}$ to sign a document d, this is, Alice only signs $h = \mathcal{H}(d)$. If a malicious party, namely Eve, could solve any of the preimage problems efficiently, let's say, Eve finds d' such that $h = \mathcal{H}(d) = \mathcal{H}(d')$, then she could publish the same signature yielded by Alice as a valid signature for d' without Alice's private key. If Eve solves the collision problem efficiently, she could hand Alice a document d to be signed, while having ge-

nerated a document with malicious content d'. The signature created by Alice would be valid for both d and d'.

2.4 FINITE FIELDS

These section's definitions are based on (LIDL; NIEDERREITER, 1983), and a basic understanding of the algebraic structures used in multivariate schemes is given.

Definition 1. *A group $(G,*)$ with the set G and the binary operation $*$ satisfies the following conditions:*

- *$*$ is associative, that is, $a*(b*c) = (a*b)*c$ for any $a,b,c \in G$.*
- *There exists an identity element $e \in G$ such that $a*e = e*a = a$.*
- *There exists an inverse for every $a \in G$ such that $a*a^{-1} = a^{-1}*a = e$.*

If the group also satisfies the condition $a*b = b*a$ for all $a,b \in G$ then it is called an abelian or commutative group.

Corollary 1. *The set of integers modulo n, denoted $\mathbb{Z}/n$, along with the ordinary addition, form a finite abelian group denoted $\mathbb{Z}_n$.*

Proof. The associativity and commutativity come from the regular addition operation. The identity element is 0 and the inverse element of any $a \in \mathbb{Z}/n$ is $-a \mod n$. Hence, $(\mathbb{Z}/n,+)$ is an abelian group. □

Definition 2. *A ring $(R,+,*)$ with the set R and the two binary operations $+$ and $*$ satisfies the conditions:*

- *$(R,+)$ forms an abelian group.*
- *$*$ is associative, that is, $a*(b*c) = (a*b)*c$ for any $a,b,c \in R$.*
- *$*$ is distributive, that is, $a*(b+c) = a*b+a*c$ for any $a,b,c \in R$.*

It must be emphasized that not only the ordinary multiplication and addition can be used to form groups or rings.

Definition 3. *A ring $(R,+,*)$ can be further classified as:*

- ***Commutative** if $*$ is commutative, that is, $a*b = b*a$ for all $a,b \in R$.*
- *A **division ring** if the nonzero elements of R form a group under $*$.*
- *A **field** if it is a commutative division ring.*

Corollary 2. *The set of integers modulo a prime p, denoted $\mathbb{Z}/p$, along with the ordinary addition and multiplication, form a finite field denoted $\mathbb{F}_p$.*

Proof. As stated above $(\mathbb{Z}/n,+)$ is an abelian group, therefore $(\mathbb{Z}/p,+)$ is a commutative group as well. The ordinary multiplication is associative, distributive and commutative. Now it must be shown that $(\mathbb{Z}\setminus\{0\},*)$ forms a group. The identity element of this group is 1. It can be observed that $a*b$ is never zero, because if $a*b \equiv 0 \mod p$ then either a or b is zero and divides p, which is a contradiction because p is prime. Now to show that all elements have inverses, the Bézout's identity can be used. For every $a \in \mathbb{Z}/p$ there exists $x,y \in \mathbb{Z}/p$ such that $a*x+p*y \equiv 1 \mod p$, due to the fact that $\gcd(a,p)=1$. Note that $p*y \equiv 0 \mod p$, hence $a*x \equiv 1 \mod p$ and x is the inverse of a. It can be concluded that $(\mathbb{Z}/p,+,*)$ is a finite field. □

Definition 4. *Given a ring $(R,+,*)$ there exists a positive integer n such that $n*r=0$ for every $r \in R$. The least such positive integer is the characteristic of the ring. If n does not exist, the characteristic is 0.*

It can be noted that finite fields have prime characteristic. If a finite field had a characteristic $n=p*q$ with $p,q \in \mathbb{Z}$ and $1<p,q<n$, then $n*e=(p*q)*e=(p*e)*(q*e)=0$ which implies that either $p*e=0$ or $q*e=0$ and it is a contradiction, because n should be the least positive integer such that $n*e=0$.

Corollary 3. *Let $\mathbb{F}_p[x]/f$ be the set of polynomials with coefficients in a finite field $\mathbb{F}_p$, modulo an irreducible polynomial[2] f of degree n. $\mathbb{F}_p[x]/f$ along with the ordinary polynomial addition and multiplication modulo f, form an extension field denoted $\mathbb{F}_{p^n}$.*

For illustration, the finite field $\mathbb{F}_{2^2}$ can be defined with its elements being $P=\{0,1,x,x+1\}$, and the operations, the ordinary addition and multiplication of polynomials modulo $f(x)=x^2+x+1$. Note that all coefficients are in $\mathbb{F}_2$. The addition operation is given in Table 1 and the multiplication in Table 2. The addition in $\mathbb{F}_{2^2}$ is the trivial polynomial addition with coefficients modulo 2, and there is no need for polynomial reductions. The multiplication though needs reductions modulo the irreducible polynomial. For instance, $x*(x+1) \equiv x^2+x \equiv 1+1*f(x) \equiv 1 \mod f$.

Proof. To prove Corollary 3 the field properties should hold for the finite fields of this fashion. Let $P=\mathbb{F}_p[x]/f$. Then, $(P,+)$ forms an abelian group with $e=0$ and the inverse of any $a \in P$ being $-a$. Also, $(P\setminus\{0\},*)$ forms a group, with the identity element being 1. The inverse element of any $a \in$

[2] An irreducible polynomial cannot be divided by a polynomial other than $f(x)=1$.

$+$	0	1	x	$x+1$
0	0	1	x	$x+1$
1	1	0	$x+1$	x
x	x	$x+1$	0	1
$x+1$	$x+1$	x	1	0

Table 1 – Addition operation of $\mathbb{F}_{2^2}$

$*$	0	1	x	$x+1$
0	0	0	0	0
1	0	1	x	$x+1$
x	0	x	$x+1$	1
$x+1$	0	$x+1$	1	x

Table 2 – Multiplication operation of $\mathbb{F}_{2^2}$

$P \setminus \{0\}$ can be found using Bézout's indentity because $\gcd(a, f) = 1$, this comes from the fact that f is irreducible. So, there exists $g, h \in P \setminus \{0\}$ such that $a * g + f * h \equiv 1 \mod f$. Note that $f * h \equiv 0 \mod f$, hence g is the inverse of a. Observe that $a * b \equiv 0 \mod f$ never happens in the group $(P \setminus \{0\}, *)$, because either a or b would need to be zero and would divide f, which is a contradiction because f is irreducible. All other properties can be trivially extended from the ordinary polynomial addition and multiplication. Therefore, $(P, +, *)$ is a finite field. □

Extensions of the binary field $\mathbb{F}_2$ are specially interesting because representing and operating with them can be very computationally efficient. An element of $\mathbb{F}_{p^n}$ needs n elements of $\mathbb{F}_p$ to be represented, hence an element of $\mathbb{F}_{2^8}$ needs 8 bits, or 1 byte. For instance, the element $x^7 + x^5 + x + 1 \in \mathbb{F}_{2^8}$ can be represented by the binary 8-uple $(1,0,1,0,0,0,1,1)$. The addition, when elements are represented in such way, is simply the bitwise exclusive or XOR operation. The multiplication operation can be performed efficiently using a modification to the Peasant's algorithm, which takes advantage of the binary representation. The reduction modulo f can also be done by a series of bitwise XOR operations. For small finite fields, a lookup table such as Table 1 and 2, can be precomputed in order to dramatically improve the efficiency of the operations, as only one query to such table would be needed to obtain the result.

3 MULTIVARIATE CRYPTOGRAPHY

In this chapter, basic foundations are given for the comprehension of the techniques proposed in the current work. Section 3.1 contains a description of the basic mathematical structure used to build MPKCs. Section 3.2 presents the construction used in the schemes discussed in this work. Section 3.3 depicts the underlying problems in which MPKCs rely their security on. Section 3.4 describes the schemes addressed in this study. Section 3.5 illustrates some variants of the Rainbow digital signature scheme present in the literature. Section 3.6 demonstrates a method to find equivalent private keys in Rainbow-like schemes and how it can be used to reduce the outer maps.

3.1 SYSTEMS OF MULTIVARIATE EQUATIONS

Standard polynomials are simply a sum of monomials, each monomial consists of a variable and a constant that multiplies it. Polynomials can be represented using a vector that stores those constants. With multivariate polynomials, each monomial consists of a multiplication of more than one variable and, again, a constant. This inclusion of multiple variables to the monomials is what makes them interesting to use in cryptosystems, since solving a system of such polynomials is computationally hard. For the purpose of multivariate cryptography, multivariate quadratic equations are sufficient, hence most commonly used.

Definition 5. *A multivariate quadratic polynomial over a finite field* $\mathbb{F}$ *is defined as:*

$$p(x_1,\ldots,x_n) = \sum_{i=1}^{n}\sum_{j=i}^{n} \alpha_{ij} x_i x_j + \sum_{i=1}^{n} \beta_i x_i + \gamma, \tag{3.1}$$

where all $x, \alpha, \beta, \gamma \in \mathbb{F}$.

A system of polynomials is a set of polynomials that share the same variables. It is known that, for polynomials with n variables, systems that have at least n linear polynomials, may be solvable, and the existence of solutions can be checked efficiently. One of the most common methods for solving such systems is the Gaussian elimination. Note that, not all systems with n polynomials and n variables have a unique solution, some of them may even have infinitely many solutions. It happens due to the fact that a polynomial can be linearly dependent of others in the systems.

Systems of multivariate polynomials can be constructed as well. Opposed to the systems explained above, solving multivariate systems is a NP-

Hard problem (GAREY; JOHNSON, 1979), even for the simplest case of quadratic polynomials, therefore they are interesting to be used in building cryptosystems. Specially, systems of multivariate quadratic polynomials are used, as the addition of more variables to the monomials does not increase the hardness of the problem.

These systems can be seen as maps. For instance, a system $\mathcal{P}$, with n variables and m equations, defines a map $\mathcal{P} : \mathbb{F}^n \to \mathbb{F}^m$. Applying this map over a vector of variables consists of substituting these variables into the equations and taking their results as the output of the map.

Definition 6. *A system $\mathcal{P}$ of multivariate quadratic polynomials over a finite field $\mathbb{F}$ is defined as:*

$$\begin{aligned}
p^{(1)}(x_1,\ldots,x_n) &= \sum_{i=1}^{n}\sum_{j=i}^{n}\alpha_{ij}^{(1)}x_ix_j + \sum_{i=1}^{n}\beta_i^{(1)}x_i + \gamma^{(1)},\\
p^{(2)}(x_1,\ldots,x_n) &= \sum_{i=1}^{n}\sum_{j=i}^{n}\alpha_{ij}^{(2)}x_ix_j + \sum_{i=1}^{n}\beta_i^{(2)}x_i + \gamma^{(2)},\\
&\vdots\\
p^{(m)}(x_1,\ldots,x_n) &= \sum_{i=1}^{n}\sum_{j=i}^{n}\alpha_{ij}^{(m)}x_ix_j + \sum_{i=1}^{n}\beta_i^{(m)}x_i + \gamma^{(m)},
\end{aligned} \tag{3.2}$$

where all $x, \alpha, \beta, \gamma \in \mathbb{F}$.

Each equation of the system can be represented as an upper triangular square matrix of order $n+1$, where the element on the i-th line and the j-th column represents the constant that multiplies the monomial x_ix_j. The last column is used to represent the linear terms and the constant term. The k-th polynomial of the system is represented by a matrix of the form:

$$A^{(k)} = \begin{pmatrix}
\alpha_{11}^{(k)} & \alpha_{12}^{(k)} & \alpha_{13}^{(k)} & \cdots & \alpha_{1n}^{(k)} & \beta_1^{(k)}\\
0 & \alpha_{22}^{(k)} & \alpha_{23}^{(k)} & \cdots & \alpha_{2n}^{(k)} & \beta_2^{(k)}\\
0 & 0 & \alpha_{33}^{(k)} & \cdots & \alpha_{3n}^{(k)} & \beta_3^{(k)}\\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots\\
0 & 0 & 0 & \cdots & \alpha_{nn}^{(k)} & \beta_n^{(k)}\\
0 & 0 & 0 & \cdots & 0 & \gamma^{(k)}
\end{pmatrix}. \tag{3.3}$$

Furthermore, $p^{(k)}$ may be written as:

$$p^{(k)}(x_1,\ldots,x_n) = (x_1,\ldots,x_n,1)\cdot A^{(k)}\cdot(x_1,\ldots,x_n,1)^T. \tag{3.4}$$

Systems of multivariate quadratic equations can be represented and stored with ease, as shown above. It is worth recalling that the coefficients of those equations are elements in a small finite field, thus operating with them is computationally efficient. Although easy to manipulate, storing these matrices is not space efficient. A notable effort resulted in various works that reduce the public map, e.g. (PETZOLDT; BULYGIN; BUCHMANN, 2010), or the private map, e.g. (YASUDA; SAKURAI; TAKAGI, 2012), but none of them reduced the key pair simultaneously.

With the keys represented as matrices, some works introduce special structures into these matrices, in such a way that representing them requires less space. For instance, the series of works by Petzoldt et al., introduce a framework that enables the public key to be partially selected. Such selection is done in a way that some special structure is introduced into the matrices, hence reducing their space requirements. Notably, CyclicRainbow (PETZOLDT; BULYGIN; BUCHMANN, 2010) uses a cyclic structure in the matrix representation of the public key.

3.2 BIPOLAR CONSTRUCTION

The basic construction used in multivariate cryptosystems is based on the composition of multiple transformations. As described above, a multivariate system, as per Definition 6, may be used as a map $\mathcal{F} : \mathbb{F}^n \to \mathbb{F}^m$. The central transformation of this construction is a system of such kind, that contains some specific structure such that, one can find preimages. In this case, $\mathcal{F}$ remains secret, and with the combination of one or more affine transformations, a public system of equations, with no apparent structure, is generated.

As shown in Figure 1, the bipolar construction consists of three secret maps, and a public map that is derived from these three. Let, $\mathcal{P}$ and $\mathcal{F}$ be multivariate systems, and $\mathcal{S}$ and $\mathcal{T}$ be random invertible affine maps. Affine maps are simply linear maps that do not preserve the origin, that is, they may translate the object. In the signing procedure, $\mathcal{F}^{-1}$ means finding one of possibly many preimages, and $\mathcal{F}$ introduces some structure that allows one to do such. The affine maps take care of hiding this structure by actually scrambling the variables of the system. When generating the keys, one calculates $\mathcal{P}$ composing the secret maps $\mathcal{P} = \mathcal{S} \circ \mathcal{F} \circ \mathcal{T}$.

Note that $n \geq m > 0$ should hold when using the construction for digital signature schemes. This makes the public map $\mathcal{P}$ surjective, and ensures that for every hash $h \in \mathbb{F}^m$ there exists a signature $\sigma \in \mathbb{F}^n$. Encryption schemes can be constructed too, in this case $n \leq m$ should hold, thus making the map injective and ensuring that the decryption process results in only one

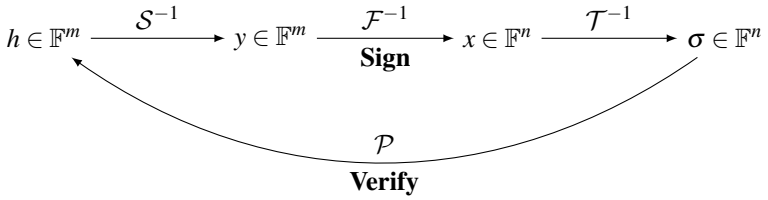


Figure 1 – Flow of the bipolar construction

plain text.

With a hash function $\mathcal{H} : \{0,1\}^* \to \mathbb{F}^m$ one can sign a document d by calculating $h = \mathcal{H}(d)$. As shown in Figure 1, recursively compute the signature $\sigma = \mathcal{T}^{-1}(\mathcal{F}^{-1}(\mathcal{S}^{-1}(h)))$. This is possible due to the fact that those maps are constructed such that they can be inverted. Actually, $\mathcal{F}$ can't be inverted, but preimages can be found. To verify the signature σ for document d one can simply check if $h = \mathcal{P}(\sigma)$ holds.

3.3 UNDERLYING COMPUTATIONAL PROBLEMS

This section describes the problems in which multivariate cryptosystems security rely on.

3.3.1 Polynomial system solving

Solving systems of polynomials is the basic underlying problem, since, solving the public system is sufficient to forge new signatures.

Definition 7. *Polynomial System Solving problem: given a system $\mathcal{P}$ as per Equation 3.2, find a vector $x' = (x'_1, \cdots, x'_n)$ such that $p^{(1)}(x') = p^{(2)}(x') = \cdots = p^{(m)}(x') = 0$.*

This problem was proven to be NP-Hard (GAREY; JOHNSON, 1979, Appendix A7.2) even for the simplest case of quadratic polynomials over $\mathbb{F}_2$. The special case of this problem where the polynomials have degree 2 is called **MQ-Problem**. The NP-Hardness of this problem is important due to the fact that, it is not feasible for an attacker to solve the public map $\mathcal{P}$ directly, hence not feasible to forge signatures this way.

3.3.2 Isomorphism of polynomials

The security of multivariate cryptosystems, due to their construction, usually does not rely exclusively on the MQ-Problem. An attacker knows that the public map is constructed by the composition of the private maps, thus one can try to decompose the map $\mathcal{P}$ into three maps, isomorphic to the private ones, and forge new signatures. The problem of finding such isomorphic maps is the Extended Isomorphism of Polynomials problem.

Definition 8. *Extended Isomorphism of Polynomials problem: given a nonlinear multivariate system $\mathcal{P} = \mathcal{S} \circ \mathcal{F} \circ \mathcal{T}$ with $\mathcal{F}$ belonging to some class $\mathcal{C}$ of special nonlinear systems that can be inverted, find $\tilde{\mathcal{S}}$, $\tilde{\mathcal{F}}$ and $\tilde{\mathcal{T}}$ such that $\mathcal{P} = \tilde{\mathcal{S}} \circ \tilde{\mathcal{F}} \circ \tilde{\mathcal{T}}$ and $\tilde{\mathcal{F}} \in \mathcal{C}$.*

With $\tilde{\mathcal{S}}$, $\tilde{\mathcal{F}}$ and $\tilde{\mathcal{T}}$ one may forge a signature for a document d by computing $\sigma = \tilde{\mathcal{T}}^{-1}(\tilde{\mathcal{F}}^{-1}(\tilde{\mathcal{S}}^{-1}(\mathcal{H}(d))))$ and publish it as a valid signature for the public key $\mathcal{P}$. Recall that $\tilde{\mathcal{F}} \in \mathcal{C}$ so it can be inverted.

Some variations of the problem with only one affine map or others that involve finding the original central map $\mathcal{F}$ exist, but the extended variation is the most generic one. In fact, solving the problem as stated above, in polynomial time, is enough to break the Rainbow Signature Scheme (DING; SCHMIDT, 2005) submitted to the NIST Post-Quantum Cryptography Standardization Process (DING et al., 2017).

Opposed to the MQ-Problem, the hardness of this problem is not well established. Actually, on the balanced Oil and Vinegar scheme (PATARIN, 1997) decomposing the public map was done efficiently by (KIPNIS; SHAMIR, 1998). While this remains an open problem, security proofs for multivariate schemes based on the bipolar construction will be absent. Still, MQDSS (CHEN et al., 2016) is a provably secure multivariate cryptosystem, however it is based on a totally different construction.

3.4 MULTIVARIATE DIGITAL SIGNATURE SCHEMES

This section contains a description of the main schemes that are based on the bipolar construction.

3.4.1 Oil and Vinegar

The original Oil and Vinegar signature scheme was presented by (PATARIN, 1997). It introduces a trapdoor that is based upon the idea of having

two sets of variables in the central map, called oil and vinegar. The central map is constructed in such a way that the polynomials are linear in the oil variables, that is, there are no monomials with two oil variables. With this fact in mind, when generating a signature, one may actually linearize the central system and solve for oil variables. A detailed description of the Oil and Vinegar signature scheme follows.

3.4.1.1 Key generation

Let K be a small finite field, e.g. $\mathbb{F}_2$. Let o and v be integers, such that o is the number of oil variables and v the number of vinegar variables. Let O be the set of oil variables and V the set of vinegar ones. Let $\mathcal{H} : \{0,1\}^* \to K^o$ be a cryptographic hash function.

The private key consists of two maps $\mathcal{T}$ and $\mathcal{F}$. Let $\mathcal{T} : K^{o+v} \to K^{o+v}$ be a random *invertible* affine transformation, its effect is to rewrite every variable as a linear combination of all other variables. This map is used to hide the structure of the central map. Let $\mathcal{F} : K^{o+v} \to K^o$ be the central map, composed of o equations of the form:

$$y^{(k)} = \underbrace{\sum_{i=1}^{v}\sum_{j=i}^{v}\alpha_{ij}^{(k)}x_i x_j}_{V\times V} + \underbrace{\sum_{i=v+1}^{v+o}\sum_{j=1}^{v}\beta_{ij}^{(k)}x_i x_j}_{O\times V} + \underbrace{\sum_{i=1}^{o+v}\gamma_i^{(k)}x_i}_{\text{linear}} + \underbrace{\delta^{(k)}}_{\text{constant}}, \tag{3.5}$$

where $x_1,\ldots,x_v \in V$ are the "vinegar" variables and $x_{v+1},\ldots,x_{v+o} \in O$ are the "oil" variables. Note that the oil variables are not multiplied between themselves, this is important to find preimages for this map when signing a message.

The public map $\mathcal{P} : K^{o+v} \to K^o$ is a simple composition of the secret maps, $\mathcal{P} = \mathcal{F} \circ \mathcal{T}$. Both $\mathcal{F}$ and $\mathcal{P}$ are multivariate systems, and $\mathcal{F}$ has the special structure mentioned above, nevertheless $\mathcal{P}$ looks randomly built. With these maps described, let the public/private key pair be $(\mathcal{P},(\mathcal{F},\mathcal{T}))$.

3.4.1.2 Signature generation

To sign a document d, compute its hash value $h = \mathcal{H}(d)$ and find a preimage $(x_1,\ldots,x_{o+v})$ for the map $\mathcal{F}$ such that $(y^{(1)},\ldots,y^{(o)}) = (h_1,\ldots,h_o)$. Observe that the structure present in $\mathcal{F}$ implies that, setting random values to the vinegar variables makes the system linear. Hence, finding a preimage

consists of selecting vinegar variables at random, substituting them into $\mathcal{F}$, and finding the oil variables through Gaussian elimination. If the system cannot be solved, new vinegar variables need to be chosen. The vector of variables x are found such that $\mathcal{F}(x) = h$. Next, using the map $\mathcal{T}$ the signature itself may be computed. Recall that $\mathcal{T}$ is invertible, thus $\mathcal{T}^{-1}$ can be obtained. With both maps inverted, the signature σ of d is published as $\sigma = \mathcal{T}^{-1}(x)$.

3.4.1.3 Signature verification

To verify a signature σ of a document d, one can simply check if $\mathcal{P}(\sigma) = \mathcal{H}(d)$ holds, therefore the signature is valid, otherwise it is invalid. The equality does actually hold for valid signatures, as $\mathcal{P}$ is a composition of the maps that were inverted in the signing procedure. Remember that $\mathcal{P}$ is a system of multivariate equations, hence hard to solve.

3.4.2 Unbalanced Oil and Vinegar

The original Oil and Vinegar explained in Section 3.4.1 is actually insecure due to the fact that $o = v$. It is called Balanced Oil and Vinegar due the same amount of oil and vinegar variables. This aspect of the cryptosystem was exploited by (KIPNIS; SHAMIR, 1998), where a method was introduced to efficiently forge new signatures when $o = v$. Subsequently, a new scheme was proposed, namely Unbalanced Oil and Vinegar (KIPNIS; PATARIN; GOUBIN, 1999). This work proposes new parameters for the original OV. The most common instantiation of this new scheme is the $v = 2o$ case.

3.4.3 Rainbow

The Rainbow signature scheme was proposed in (DING; SCHMIDT, 2005). It can be described simply as a multilayered UOV. Actually, it is a generalization of the UOV scheme, in other words, UOV is a single layer Rainbow instantiation. This newly proposed scheme greatly improves space requirements in comparison to UOV. Both public and private key sizes, as well as signatures size, are reduced for equivalent security levels. A detailed description of the Rainbow cryptosystem follows.

Each layer of this Rainbow has its own polynomials, as well as its own set of oil and vinegar variables. These polynomials have the same structure as OV polynomials, but the layers are intrinsically connected when construc-

ted. Namely, V_l and O_l are respectively the set of vinegar variables and the set of oil variables of the l-th layer. Let u be the number of layers. Let $v_1, v_2, \ldots, v_{u+1}$ be the number of vinegar variables in each layer, such that $0 < v_1 < v_2 < \cdots < v_{u+1} = n$. It may be observed that the layers actually grow bigger, this is due to the fact that each set of vinegar variables contains the vinegar variables from the previous layer, that is $V_1 \subset V_2 \subset \cdots \subset V_u$. Deeper into the layers, the oil variables become vinegar variables, *i.e.* $V_{l+1} = V_l \cup O_l$. Keep in mind that $|V_l| = v_l$ and $|O_l| = o_l = v_{l+1} - v_l$.

The connection between variables of the layers exist in such a way that the layers have to be inverted one after the other for the complete inversion of the central map. To start this process, one randomly selects the first set of vinegar variables and substitutes them, making the first layer linear. Notice that the oil variables of some layer are part of the vinegar variables of the subsequent layer. When the first layer is solved, the set O_1 can be used to construct V_2 in its entirety. With known values for V_2 one can linearize the second layer and solve it. Repeating this process for every layer, inverts the whole central map. It may happen that some layer is not be solvable, in this case, a new set V_1 is chosen and the inversion process starts over. This happens with very small probability as shown in Section 4.2.

3.4.3.1 Key generation

Let the sets $V_l = \{x_1, \ldots, x_{v_l}\}$ be the vinegar variables of the l-th layer, and $O_l = V_{l+1} - V_l$ the respective oil variables. Observe that, each layer l has $o_l = v_{l+1} - v_l$ equations, and thus o_l equations are needed to solve for O_l variables. For each layer l, construct a system $\mathcal{F}_l$ composed of o_l polynomials of the form:

$$y^{(k)} = \sum_{i=1}^{v_l} \sum_{j=i}^{v_l} \alpha_{ij}^{(k)} x_i x_j + \sum_{i=v_l+1}^{v_{l+1}} \sum_{j=1}^{v_l} \beta_{ij}^{(k)} x_i x_j + \sum_{i=1}^{v_{l+1}} \gamma_i^{(k)} x_i + \delta^{(k)}, \tag{3.6}$$

for $k = v_l, \ldots, v_{l+1} - 1$ and $l = 1, \ldots, u$. Observe that the polynomials are Oil and Vinegar polynomials, just like the ones in Equation 3.5, and they can be solved for O_l when V_l has known values. Let the central map $\mathcal{F} : K^n \to K^{n-v_1}$ be the union of the u layers. To hide the central map, two random invertible affine maps are used. Let $\mathcal{S} : K^{n-v_1} \to K^{n-v_1}$ and $\mathcal{T} : K^n \to K^n$ be part of the private key. Then, $\mathcal{P} = \mathcal{S} \circ \mathcal{F} \circ \mathcal{T}$ is published as the public key.

Security level (bits)	**Parameters** (K, o_1, v_1, v_2)	**Private key size (bytes)**	**Public key size (bytes)**
80	$(\mathbb{F}_{256}, 17, 17, 9)$	19208	25740
100	$(\mathbb{F}_{256}, 26, 22, 11)$	45450	60390
128	$(\mathbb{F}_{256}, 36, 28, 15)$	103704	139320
192	$(\mathbb{F}_{256}, 63, 46, 22)$	440638	596904
256	$(\mathbb{F}_{256}, 85, 63, 30)$	1086971	1498230

Table 3 – Rainbow key sizes for parameters proposed in (PETZOLDT, 2013)

3.4.3.2 Signature generation

Let $\mathcal{H} : \{0,1\}^* \to K^{n-v_1}$ be a cryptographic hash function. To sign a document d, compute $h = \mathcal{H}(d)$. Compute $y = (y_1, \ldots, y_{n-v_1}) = \mathcal{S}^{-1}(h)$. All layers of $\mathcal{F}$ are solved using the process described above, in order to find $x = (x_1, \cdots, x_n)$ such that $\mathcal{F}(x) = y$. Finally, $\sigma = \mathcal{T}^{-1}(x)$ is published as the signature for d.

3.4.3.3 Signature verification

Verifying a signature σ for a document d is as simple as checking if $\mathcal{P}(\sigma) = \mathcal{H}(d)$ holds. Remark that, again, $\mathcal{P}$ is a multivariate system that appears to be randomly built, thus hard to solve directly.

3.4.3.4 Key sizes

For illustration, Table 3 shows a comparison between the key sizes of some Rainbow instances proposed in (PETZOLDT, 2013, Chapter 6). Let $m = n - v_1$ be the number of equations in Rainbow central and public maps. The private and public key sizes of Rainbow-like signature schemes are given, respectively, by the formulas:

$$K_{pr} = \underbrace{m^2 + m}_{\mathcal{S}} + \underbrace{n^2 + n}_{\mathcal{T}} + \underbrace{\sum_{l=1}^{u} o_l \left(\frac{v_l(v_l+1)}{2} + v_l o_l + v_{l+1} + 1 \right)}_{\mathcal{F}}, \qquad (3.7)$$

$$K_{pu} = m\left(\frac{n(n+1)}{2} + n + 1\right) = m\frac{(n+1)(n+2)}{2}. \tag{3.8}$$

These can be easily checked by looking at Equation 3.6 and the size of the affine maps $\mathcal{S}$ and $\mathcal{T}$. Also, the public key size is simply the size of a multivariate system with n variables and m equations. The public key size formula can be checked by observing Equation 3.3, as the public map is composed of m matrices of such kind. With $u = 1$, OV and UOV key sizes can be calculated using K_{pr} and K_{pu} also, as Rainbow is a generalization of the Oil and Vinegar schemes.

3.5 RAINBOW VARIANTS

The cryptosystems described above have a notable space requirement. For instance, the Rainbow public key can get up to 1.6 MB for secure parameters (DING et al., 2017, Table 2) in comparison to ECDSA's 64 bytes public keys. This section presents some variants in the literature that reduce Rainbow key sizes. It is shown that, all of them reduce either the public map or the private central map, but not both.

3.5.1 Establishing a linear relation between public and private maps

In (PETZOLDT; BULYGIN; BUCHMANN, 2010) an approach is used in such a way that one can partially select the public key. It relies on an important aspect of the schemes previously discussed. In the UOV case, it is demonstrated that the secret map $\mathcal{S}$ actually establishes a linear relation between the private and public coefficients of the equations. This makes it possible to select $\mathcal{P}$ and $\mathcal{S}$ in a clever manner, such that their representations are smaller, introducing some kind of structure like a circulant matrix in the case of CyclicRainbow.

Think of the UOV construction of the maps $\mathcal{P} = \mathcal{F} \circ \mathcal{T}$. To better understand this linear relation, the equations can be simplified, without its linear and constant terms. Let $n = o + v$, and the k-th private polynomial of the system, without its linear and constant terms, be denoted as:

$$y^{(k)} = \sum_{r=1}^{n}\sum_{s=r}^{n}\left(\alpha_{rs}^{(k)} x_r x_s\right). \tag{3.9}$$

Let $\mathcal{T} \in K^{n\times n}$ be the matrix that describes the secret affine map. The

k-th public polynomial can be written as:

$$p^{(k)} = \sum_{r=1}^{n}\sum_{s=r}^{n}\left[\alpha_{rs}^{k}\sum_{i=1}^{n}(t_{ir}x_i)\sum_{j=1}^{n}(t_{js}x_j)\right], \tag{3.10}$$

where t_{ij} is the element of $\mathcal{T}$ in the i-th line and the j-th column. The inner summations come from the matrix multiplication operation, where the vector of x variables is multiplied with $\mathcal{T}$ when the map is applied, giving a "new" vector of x variables, that are actually just linear combinations of the old ones. Let the public polynomial be denoted similarly to the private one:

$$p^{(k)} = \sum_{r=1}^{n}\sum_{s=r}^{n}\left(\rho_{rs}^{(k)}x_r x_s\right). \tag{3.11}$$

To establish a relation between α_{rs} and ρ_{rs} observe the structure of Equation 3.10 where α_{rs} is multiplying two polynomials. Expanding the inner summations gives:

$$(t_{1r}x_1 + t_{2r}x_2 + \cdots + t_{nr}x_n)(t_{1s}x_1 + t_{2s}x_2 + \cdots + t_{nr}x_n), \tag{3.12}$$

which can be described as a new polynomial with quadratic terms, applying distributive multiplication:

$$\sum_{i=1}^{n}\sum_{j=i}^{n}\left(\tau_{ij}^{rs}x_i x_j\right) \quad \text{where :} \quad \tau_{ij}^{rs} = \begin{cases} t_{ir}t_{is} & \text{if } i = j, \\ t_{ir}t_{js} + t_{jr}t_{is} & \text{otherwise.} \end{cases} \tag{3.13}$$

Substituting Equation 3.13 into Equation 3.10 yields:

$$p^{(k)} = \sum_{r=1}^{n}\sum_{s=r}^{n}\left[\alpha_{rs}^{k}\sum_{i=1}^{n}\sum_{j=i}^{n}\left(\tau_{ij}^{rs}x_i x_j\right)\right], \tag{3.14}$$

therefore, a relation between private and public coefficients can be written as:

$$\rho_{ij}^{(k)} = \sum_{r=1}^{n}\sum_{s=r}^{n}\left(\tau_{ij}^{rs}\alpha_{rs}^{k}\right). \tag{3.15}$$

After randomly choosing $\mathcal{S}$, Equation 3.15 depicts the linear relation between the public (ρ) and private (α) coefficients. Hence one can choose and fix ρ and find α using this equation. Observe that not all public coefficients can be chosen, otherwise no structure would be present in $\mathcal{F}$ and it could not be inverted. This could also be used to recalculate $\mathcal{F}$ from $\mathcal{P}$ and $\mathcal{S}$. The idea

behind this relation is further expanded to Rainbow key pairs, and is described in detail by (PETZOLDT et al., 2011). This relation was extensively used to reduce either public or private keys, but not both. Reducing key sizes with this strategy consists of introducing some structure into one part of the key, such that its representation becomes smaller, and calculating the other part using the equations above. Notable key optimizations are briefly discussed hereafter.

3.5.2 CyclicRainbow

CyclicRainbow (PETZOLDT; BULYGIN; BUCHMANN, 2010) uses the relations explained in Section 3.5.1 to structure the public key. In the key generation step, parts of the public key matrix representation are selected such that they form a circulant matrix.

Definition 9. *A square circulant matrix of order n is of the form:*

$$A = \begin{pmatrix} a_1 & a_2 & \cdots & a_{n-1} & a_n \\ a_n & a_1 & a_2 & \cdots & a_{n-1} \\ a_{n-1} & a_n & a_1 & & a_{n-2} \\ \vdots & & \ddots & \ddots & \vdots \\ a_2 & \cdots & a_{n-1} & a_n & a_1 \end{pmatrix}. \tag{3.16}$$

Such matrices allow for a compact representation. Storing the vector $(a_1, a_2, \ldots, a_n)$ is enough to represent the matrix completely, as its elements form a structured repetition. Introducing this structure in parts of the public key, when possible, reduces the representation of the public system significantly. In fact this method allows a public key reduction factor of up to 2.9, for secure parameters. This repetition of elements also leads up to a repetition of operations when verifying a signature. Exploiting this common field multiplications leads to a reduction factor of up to 2.4 in the number of operations needed to evaluate the public map, *i.e.* to verify a signature (PETZOLDT, 2013, Table 10.3).

3.5.3 RainbowLRS2

RainbowLRS2 (PETZOLDT, 2013, Section 9.2) introduces matrices generated by Linear Recurring Sequences into the public key, just like in CyclicRainbow. To generate an $m \times n$ matrix of this type, a vector $a = (a_1, a_2, \ldots, a_m) \in \mathbb{F}^m$ of distinct elements is selected. The i-th row of this

matrix is:

$$A[i] = (a_i^0, a_i^1, a_i^2, \ldots, a_i^{n-1}) \qquad i = 1, \ldots, m. \tag{3.17}$$

Storing the vector a is sufficient to represent the matrix A, since every row can be calculated using Equation 3.17. Using matrices of this fashion in the public map representation, allows for a reduction factor of up to 3.1. Veritably, this method is similar to the one used in CyclicRainbow, thus provides an akin improvement in space requirements. The main difference is that CyclicRainbow uses rotations to generate each row. In RainbowLRS2 the rows are defined by Equation 3.17. It is shown in (PETZOLDT, 2013) that field multiplications also repeat in the verification process of RainbowLRS2, allowing a speed up factor of 2.2 (PETZOLDT, 2013, Table 10.3).

3.5.4 Circulant Rainbow

Circulant Rainbow (PENG; TANG, 2017) introduces circulant matrices in the central private map, in contrast to CyclicRainbow that does it on the public map. These circulant relations are present in very specific parts of the central polynomials, precisely in the terms that have oil variables. Throughout the signature generation, after inserting values into the vinegar variables, one has to solve a linear system as shown in Section 3.4.3. Due to the structure introduced in the coefficients of the central map, this system is represented by a circulant matrix. Solving such systems can be done more efficiently using a method described in (PENG; TANG, 2017). Indeed, the private key needs less space to be represented. Circulant Rainbow has a private key 1.8 times smaller than the original Rainbow, and the signature generation can be 2.9 times faster if the optimized method for solving the special linear systems is used instead of the Gaussian elimination. Notably, (HASHIMOTO, 2018) discourages the use of Circulant Rainbow. It is shown that the Kipnis-Shamir's attack (KIPNIS; SHAMIR, 1998) can be used to recover equivalent Circulant Rainbow private keys in polynomial time.

3.5.5 NC-Rainbow

NC-Rainbow (YASUDA; SAKURAI; TAKAGI, 2012) proposes the use of non-commutative rings instead of a finite field in the central map. Further, isomorphisms between the rings and the fields are used in the composition of the public map. Through this isomorphism it can be shown that a NC-Rainbow central map is equivalent to the original Rainbow central map.

In fact, it is stated that an element α of a non-commutative ring R can be represented by r elements in the finite field K. This smaller representation of the elements in R is what leads to a more compact representation of the central map. Indeed, it is shown that the use of this technique can reduce the private key by a factor of 4 and sign documents with 1.6 times less field multiplications with the proposed parameters for the new scheme.

In (THOMAE, 2012), it is shown that NC-Rainbow is just a special case of introducing structures into the private central map. Additionally, it is demonstrated that, the reduction of NC-Rainbow to the original Rainbow is not enough to prove security. The reduction in the opposite direction is needed and is absent in the original work. The existent reduction just provides an upper bound for the security of the new scheme. Actually, known attacks on Rainbow were sensibly improved for the NC-Rainbow case. The optimizations reduce the MinRank attack complexity from 2^{288} to 2^{192} and for the HighRank, the reduction is from 2^{128} to 2^{96}. Hence, the usage of non-commutative rings in the central map is not recommended, as it greatly decreases the security of the original Rainbow.

3.6 EQUIVALENT KEYS IN MULTIVARIATE CRYPTOSYSTEMS

In (WOLF; PRENEEL, 2005) is introduced the idea of equivalent keys for Multivariate Cryptosystems. Equivalent keys are different private keys that lead to the same public key. They can be found using the idea of "sustainers". Sustainers are transformations that do not alter the property present in the central map that allow its inversion. For instance, a sustainer to UOV is a transformation that, when applied to the central map, does not change the fact that oil variables do not multiply themselves. By letting Δ, Γ be sustainers, an equivalent key can be found using the equation:

$$\mathcal{P} = \underbrace{\mathcal{S} \circ \Delta^{-1}}_{\mathcal{S}'} \circ \underbrace{\Delta \circ \mathcal{F} \circ \Gamma}_{\mathcal{F}'} \circ \underbrace{\Gamma^{-1} \circ \mathcal{T}}_{\mathcal{T}'}. \tag{3.18}$$

The maps $\mathcal{S}', \mathcal{F}', \mathcal{T}'$ are equivalent keys for $\mathcal{P}$. If for every $\mathcal{S}$ and $\mathcal{T}$, Δ and Γ can be found such that $\mathcal{S}'$ and $\mathcal{T}'$ contain some special structure, the private key space can be reduced.

For instance, if this idea is applied to UOV, additive sustainers can be used to reduce the storage requirements of $\mathcal{T}$. An additive sustainer only adds a term to the constant term. This sustainer clearly does not break the Oil and Vinegar property of the central map, and it can be used to subtract the constant terms in $\mathcal{T}$. Let $\mathcal{T}$ be a random affine map, then Γ^{-1} can be chosen such that it subtracts all its constant terms. So, for every $\mathcal{T}$, there exists an

equivalent map $\mathcal{T}'$ that has no constants. Using this fact, $\mathcal{T}$ can be chosen without its constants already.

Other sustainers are presented in (WOLF; PRENEEL, 2011). For instance, Gauss sustainers can be used to further reduce the private key space. These sustainers are transformations that permute or add rows and columns of a matrix and they can be described by invertible matrices. Again, let Γ be a sustainer for UOV:

$$\Gamma(x) = \begin{pmatrix} A & 0 \\ B & C \end{pmatrix} \cdot x, \tag{3.19}$$

such that $A \in K_q^{v \times v}$, $B \in K_q^{o \times v}$, $C \in K_q^{o \times o}$ and A, C are invertible. Observe that x is the column vector of variables in the central equations. The transformation Γ only shuffles the vinegar variables within themselves, so it is in fact a sustainer. The construction of Equation 3.18 can be applied as:

$$\mathcal{P} = \mathcal{F} \circ \begin{pmatrix} A & 0 \\ B & C \end{pmatrix} \circ \begin{pmatrix} A^{-1} & 0 \\ -C^{-1} \cdot B \cdot A^{-1} & C^{-1} \end{pmatrix} \circ \mathcal{T}, \tag{3.20}$$

Now an structure can be enforced in $\mathcal{T}'$. Let I_n be identity matrix of size n:

$$\begin{aligned} \mathcal{T} &= \Gamma \circ \mathcal{T}' \\ \begin{pmatrix} T_1 & T_2 \\ T_3 & T_4 \end{pmatrix} &= \begin{pmatrix} T_1 & 0 \\ T_3 & T_4 - T_3 \cdot T_1^{-1} \cdot T_2 \end{pmatrix} \circ \begin{pmatrix} I_v & T_1^{-1} \cdot T_2 \\ 0 & I_o \end{pmatrix}. \end{aligned} \tag{3.21}$$

For every $\mathcal{T}$, with T_1 being invertible, there exists an equivalent map $\mathcal{T}'$, thus $\mathcal{T}$ can be chosen with the shape of $\mathcal{T}'$. This greatly reduces the space requirements, as only a matrix of size $v \times o$, *i.e.* the upper right portion of $\mathcal{T}'$, is sufficient to represent $\mathcal{T}$.

The idea of equivalent keys and the use of Gauss sustainers to reduce the outer maps of the private keys, is extended in (PETZOLDT, 2013, Chapter 3.5) to be used in Rainbow. The multilayer construction allows to build Δ sustainers that mix layer equations within their own layer. The improvement is actually used in the Rainbow submission (DING et al., 2019) to round 2 of NIST's standardization process. The proposal consists of using almost upper triangular matrices, with the main diagonal only composed by 1's, in the outer maps. Matrices of such kind are always invertible due to the fact that their determinant is always equal to 1. This reduction in the outer maps key space can be used along with the Rainbow variants mentioned above, nonetheless it only reduces the maps $\mathcal{S}$ and $\mathcal{T}$ which do not make part of the majority of the private key size, as it can be observed in Equation 3.7.

4 REDUCING PRIVATE KEYS BY REUSING VINEGAR VARIABLES

In Section 3.5, some examples of Rainbow variants that introduce structure in either the public key or the central map $\mathcal{F}$ are introduced. As stated above, none of these strategies could be used simultaneously to achieve smaller key pairs. In this section, a novel modification to the original Rainbow scheme is presented. It can reduce up to 84% of the private key. Additionally, this optimization can be used along with variants that introduce structure into the public key, reducing the whole key pair by up to 71% (ZAMBONIN; BITTENCOURT; CUSTÓDIO, 2019). The pitfalls of using such optimization are discussed as well.

4.1 PROPOSAL

It can be noted that introducing structure into the central map is generally not well succeeded. Such schemes usually offer some vulnerability due to the special structure in $\mathcal{F}$. With this in mind, a new general framework to reduce private keys in Rainbow-like signature schemes is proposed. The modification, namely Rainbow-η, consists in tweaking the key generation and the signature generation step, rather than introducing structures in the central system of equations. Remind that the first set of vinegar variables is chosen at random for every new signature. If this set of vinegar variables could be reused across multiple signatures, a reduced version of the central map, with the first set of vinegar variables substituted into the equations could be stored, in contrast to the whole map with all variables.

Generating new keys now includes an additional step. After building $\mathcal{F}$, the set V_1 is chosen randomly and then substituted into the equations. Notice that the first layer polynomials are now linearized. In this way V_1 can be substituted in the subsequent layers too. These layers remain quadratic but with far less monomials, because the monomials that contain variables from V_1 are simplified. This linearized central map $\mathcal{F}'$ is stored along with the chosen V_1 for use in the signature generation step. The rest of the key generation step remains the same. Henceforth, $\mathcal{P}$ is calculated using the whole $\mathcal{F}$, but only $\mathcal{F}'$ needs to be stored. The private key is now the tuple $(\mathcal{S}, \mathcal{F}', \mathcal{T}, V_1)$.

To further generate new signatures, the first step of the signing procedure, that is, choose and substitute V_1 into $\mathcal{F}$, is not needed anymore, as this is done in the key generation step. The layers are inverted as described in Section 3.4.3. After the preimage of $\mathcal{F}$ is found, the set V_1, selected in the

key generation step, is used along with the rest of the variables to compute the signature. It may happen that some layer is not solvable for a given signature. Originally, when this happens, a new set V_1 is chosen and the inversion process is repeated. In Rainbow-η this cannot be done, due to the fact that the central map $\mathcal{F}'$ is already simplified with the previously chosen V_1, and $\mathcal{F}$ is not available. With this issue in mind, three alternatives are proposed to viabilize Rainbow-η.

4.1.1 Rainbow-η_1

In order to regenerate the central map $\mathcal{F}$, a PRNG (Pseudorandom Number Generator) seeded by S can be used to generate $\mathcal{F}$ in the key generation step. Storing S with the rest of the private key, enables the signer to regenerate $\mathcal{F}$ in the case of the some layer not being solvable for some signature. Therefore, in the signature procedure, if some layer fails to be solved, $\mathcal{F}$ is regenerated completely and a new set $\tilde{V}_1$ is chosen until the first layer becomes solvable. After $\tilde{V}_1$ is substituted into $\mathcal{F}$, only this new simplified version of $\mathcal{F}$ needs to be kept for subsequent signatures.

This alternative is efficient and adds only the cost of regenerating $\mathcal{F}$ every time that some layer is not solvable, which occurs with small probability, as it is shown in Section 4.2. Also, the small cost of storing S along with the private key is added.

4.1.2 Rainbow-η_2

It is show in Section 3.5.1 that a linear relation between the public and private maps exist. This relation can be used to generate $\mathcal{F}$ from $\mathcal{P}$, $\mathcal{S}$ and $\mathcal{T}$. Therefore, in the case of the signer being unable to find a preimage for the central map, $\mathcal{F}$ is regenerated and the signing procedure occurs just like in Rainbow-η_1. With this method, no additional data is needed in the private key, however, regenerating the central map by this manner is less efficient than through the PRNG solution.

4.1.3 Rainbow-η_3

If the central map layers could be solved for different values, there would be no need for new vinegar variables to be plugged in $\mathcal{F}$. This could be achieved using a random nonce in the generation of h. When creating a new

signature, a random salt r is chosen and h is given by $h = \mathcal{H}(\mathcal{H}(d)\|r)$. Note that changing r changes h completely. Observe that the image y is calculated by $y = \mathcal{S}^{-1}(h)$, hence changing r completely alters y. With this modified hashing procedure, in the case of an unsolvable layer, r is changed and the hashing repeated, getting new values for y. This is done until all layers are solvable and the signature step can finish properly. The verification process is done by checking if $\mathcal{P}(\sigma) = \mathcal{H}(\mathcal{H}(d)\|r)$ holds. Note that now, r is part of the signature.

Actually, this method is used in the Rainbow proposal to NIST's standardization process (DING et al., 2017). This technique is used to achieve EUF-CMA security and to avoid the need of choosing new vinegar variables in the case of some layer being unsolvable. Rainbow-η_3 is clearly more efficient than the other variants, as it avoids the regeneration of $\mathcal{F}$. Since this method is more efficient, and it is already used in Rainbow implementations, the use of Rainbow-η_3 is recommended.

4.2 INVERTIBILITY OF THE CENTRAL MAP

As described in Section 3.4.3, to sign a document, $\mathcal{F}$ needs to be inverted[1]. To invert $\mathcal{F}$, all layers need to be inverted individually. Inverting each layer, after the values for the vinegar variables are substituted, consists in solving systems of linear equations. These systems may be solvable or not. In the case of some of these systems not being solvable, the inversion of the central map needs to be restarted. Specially, in Rainbow-η restarting this process adds some cost to the signature generation. In this section, it is shown that this happens with small probability, hence the average cost introduced by the η variants is small.

A linear system of equations can be represented by a matrix of its coefficients. Each row of this matrix represents one equation. For instance, the following system with n equation and n variables:

$$\begin{aligned} y^{(1)} &= \alpha_1^{(1)} x_1 + \alpha_2^{(1)} x_2 + \cdots + \alpha_n^{(1)} x_n + \beta^{(1)}, \\ y^{(2)} &= \alpha_1^{(2)} x_1 + \alpha_2^{(2)} x_2 + \cdots + \alpha_n^{(2)} x_n + \beta^{(2)}, \\ &\vdots \\ y^{(n)} &= \alpha_1^{(n)} x_1 + \alpha_2^{(n)} x_2 + \cdots + \alpha_n^{(n)} x_n + \beta^{(n)}, \end{aligned} \tag{4.1}$$

[1] Note that $\mathcal{F}$ is surjective, hence there is no inverse of this map. This inversion is actually a "pseudo-inversion" and it consists in finding a preimage for $\mathcal{F}$.

with all $x, y, \alpha \in \mathbb{F}_q$, can be rewritten as:

$$y = \begin{pmatrix} \alpha_1^{(1)} & \alpha_2^{(1)} & \cdots & \alpha_n^{(1)} \\ \alpha_1^{(2)} & \alpha_2^{(2)} & \cdots & \alpha_n^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{(n)} & \alpha_2^{(n)} & \cdots & \alpha_n^{(n)} \end{pmatrix} \cdot x + \begin{pmatrix} \beta_n^{(1)} \\ \beta_n^{(2)} \\ \vdots \\ \beta_n^{(n)} \end{pmatrix}, \tag{4.2}$$

with x and y being the column of x and y variables. Let A be the coefficient matrix of Equation 4.2 and B the column of constant terms. The system for a given vector y can be solved by calculating:

$$x = A^{-1} \cdot (y - B). \tag{4.3}$$

Actually, the system is solvable if and only if A is invertible. Therefore, observing the probability of a randomly built matrix being invertible shows the probability of random linear systems being solvable. From now on, it is assumed that the linear systems generated in the signature step, are randomly built.

Definition 10. *Let I_n be the identity matrix of size n. A matrix $A \in \mathbb{F}_q^{n \times n}$ is invertible if and only if there exists another matrix $B \in \mathbb{F}_q^{n \times n}$ such that $AB = BA = I_n$. The inverse of A is denoted as A^{-1}.*

It can be shown that all rows of an invertible matrix are linearly independent. Let A be an invertible matrix. By contradiction, assume that some row of A is linearly dependent of one or more other lines. So, there exists a nonzero column vector x such that $Ax = 0$. If A is invertible, then $A^{-1}Ax = 0$ can be calculated and the equality holds only if $x = 0$, contradicting the linear dependency.

With this fact in mind, invertible matrices can be constructed row by row, choosing a vector that is not linearly dependent of the previously selected ones. For the first row l_1 there are a total of $q^n - 1$ choices, as the zero vector cannot be chosen. The second row must satisfy the condition $l_2 \neq cl_1$, since q values for c can be selected, the second row has $q^n - q$ possibilities. Without loss of generality, the k-th row must satisfy $l_k \neq c_1 l_1 + c_2 l_2 + \cdots + c_{k-1} l_{k-1}$. As there are q^{k-1} possibilities for c constants, the k-th row has $q^n - q^{k-1}$ possibilities. Multiplying all these possibilities, the probability of a random matrix with elements in $\mathbb{F}_q$ being invertible is given by:

$$p(n,q) = \prod_{k=1}^{n} \frac{q^n - q^{k-1}}{q^n} = \prod_{k=1}^{n} \left(1 - q^{-k}\right). \tag{4.4}$$

Moreover, the probability of all layers being invertible for a given

Rainbow instance with parameters $(v_1, o_1, o_2, \ldots, o_u)$ is:

$$\tilde{p}(q, o_1, \ldots, o_u) = \prod_{l=1}^{u} \prod_{k=1}^{o_l} \left(1 - q^{-k}\right). \tag{4.5}$$

It can be observed that smaller matrices with bigger fields are more probable to be invertible. For instance, with the parameters proposed in (PETZOLDT, 2013, Chapter 6), using the finite field $\mathbb{F}_{16}$ and a Rainbow instance that achieves 256-bit security level, the probability of all layers being invertible is $\tilde{p}(16, 69, 59) \approx 87.15\%$. However using another proposed instance, with $\mathbb{F}_{256}$ and for the same security level, $\tilde{p}(256, 63, 30) \approx 99.21\%$. So, only in 0.79% of the cases some layer is not solvable. In Rainbow-η, this probability is even smaller, as the first layer is fixed throughout multiple signatures and it only has a chance of not being invertible when a new set V_1 is chosen. Specially, in Rainbow-η_3, as no new set V_1 is chosen, only the probability of the subsequent layers need to be taken into account. It can be noted that the third proposed variant does not add any cost to the implementation submitted in (DING et al., 2017).

4.3 ATTACKING MULTIPLE SIGNATURES

Rainbow-η fixes part of the preimage of $\mathcal{F}$, which may raise a concern. Multiple signatures, with the same vinegar variables, can leak information about the private keys or even the variables found in the signature process.

It should be noted that the attacker has no information about $\mathcal{T}$, so if one would try to find and invert this first transformation that hides the central map $\mathcal{F}$, a quadratic polynomial system would need to be resolved. That is, one would need to resolve the equation $\sigma = \mathcal{T}^{-1}(x)$, and with x and $\mathcal{T}$ being unknowns, it actually forms a quadratic polynomial system. If part of x is fixed, this system remains quadratic, still with more variables than equations. Suppose that even though it is unfeasible to solve the above equation and to find the original $\mathcal{T}$, an attacker finds it efficiently via some other method. The possession of $\mathcal{T}$ would be of no use because $\mathcal{P}$ is also hidden by $\mathcal{S}$.

4.4 APPLICATION OF KNOWN ATTACKS

There are many known attacks on Rainbow and UOV that could also be applied to Rainbow-η. In the previous sections, some problems and possibly security flaws are analysed. In this section, the applicability of existing

attacks is discussed.

4.4.1 Direct Attack

A direct attack consists on solving the public system directly, *i.e.* solve $\mathcal{P}(x) = h$, without any information of the private key or signatures. This can be done via methods like the one described in (BETTALE; FAUGERE; PERRET, 2009). This is the most generic attack to multivariate systems, as it tries to tackle the base MQ-Problem which, as described in Section 3.3.1, is NP-Hard. Clearly this attack is not facilitated by Rainbow-η, as the proposed scheme does not alter the public key in any form, and the attack does not make use of multiple signatures.

4.4.2 Rank Attacks

The attack based on the MinRank problem was exposed in (BILLET; GILBERT, 2006). The attack consists in finding a linear combination of all public matrices, that represent the equations, with very low rank. This rank threshold depends on the parameters of Rainbow. Finding this combination allows an attacker to extract the first layer of the central polynomials, and part of the outer maps. Repeating the process, the others layers can recovered as well, and with negligible effort, due to the partial knowledge of the private maps.

HighRank can be seen as the opposite of MinRank. It tries to find linear combinations between the public matrices and inverts Rainbow layers from last to first. An improved version of the HighRank attack for Rainbow is proposed by (DING et al., 2008). Both the MinRank and the HighRank attacks try to recover the private map from the public one. The attacks cannot be optimized for Rainbow-η as, again, it does not modify the public key generation and structure.

4.4.3 Rainbow-Band-Separation Attack

The Rainbow-Band-Separation attack was proposed in (DING et al., 2008) as an extension of the UOV-Reconciliation attack. Basically the attack tries to find an equivalent private key to forge new valid signatures. Again, it attacks the public key directly, hence the complexity of the attack for Rainbow-η is the same.

Security level (bits)	**Parameters** (K, o_1, v_1, v_2)	K_{pr} **(bytes)**	K_{pr}^{η} **(bytes)**	**Difference**
80	$(\mathbb{F}_{256}, 17, 17, 9)$	19208	5914	-69.21%
100	$(\mathbb{F}_{256}, 26, 22, 11)$	45450	11013	-75.77%
128	$(\mathbb{F}_{256}, 36, 28, 15)$	103704	22110	-78.68%
192	$(\mathbb{F}_{256}, 63, 46, 22)$	440638	71773	-83.71%
256	$(\mathbb{F}_{256}, 85, 63, 30)$	1086971	164721	-84.85%

Table 4 – Rainbow-η improvements

4.4.4 Side-Channel Attacks

As described in Sections 3.4.3 and 4.1, when some Rainbow layer is not solvable, a new set of first layer vinegar variables should be chosen. This adds a considerable computation time to the signature generation step when some layer is not solvable. Specially, in Rainbow-η_2 this difference can be huge, as the signer recalculates the central map. In a chosen message attack, an attacker can observe the time taken to generate signatures and know the messages that cause a non-invertibility of some layer.

(DING et al., 2017) claims that their implementation is side-channel resistant as *"... all key dependent operations are performed in a time-constant manner."*. However, when some layer is not solvable, the signature generation takes longer, as described in Section 4.1.3. There are no known attacks that make use of such information, but this possibility is not discarded, and the submitted implementation may be vulnerable to such timing attack.

4.5 IMPROVEMENT ON RAINBOW INSTANCES

The new private key size for Rainbow-η, without the additional data required for Rainbow-η_1, can be calculated by:

$$K_{pr}^{\eta} = m^2 + m + n^2 + n + v_1 + \sum_{l=1}^{u} o_l \left(\frac{(v_l - v_1)(v_l - v_1 + 1)}{2} + (v_l - v_1)o_l + (v_{l+1} - v_1) + 1 \right). \quad (4.6)$$

Table 4 shows the reduction of the private key achieved by using Rainbow-η on those parameters. This reduction can be combined with some

variant that reduces public keys, like the ones described in Section 3.5, to achieve smaller key pairs. Also, the use of the limited key spaces, described in Section 3.6, to reduce the outer maps of the private key can be used along with the η variant. So this method is a general framework that can be applied to all Rainbow variants that do not change the private key, and it can be joined with other methods to achieve smaller key pairs.

4.6 IMPLEMENTATION

A proof of concept implementation[2] was made to validate the proposed optimization. The repository contains a modified version of the Rainbow implementation submitted to NIST's standardization process. In this implementation, the first set of vinegar variables is only substituted in the first layer, so the full potential of the technique is not exploited.

[2] `https://github.com/matheuspb/rainbow-eta`

5 CONCLUSION

Rainbow has a space requirement problem as its keys are orders of magnitude bigger than the classic digital signature algorithms. In order to viabilize the usage of Rainbow in a post-quantum scenario, this issue is being extensively addressed in the literature. A series of modifications to the original Rainbow scheme presented in the literature were studied in order to comprehend the techniques used to tackle the key size problem. It is observed that the introduction of structures into the private key, in general, makes the schemes less secure. Finally, a novel modification was proposed.

5.1 CONTRIBUTIONS

In this work a detailed description and comprehension of the Rainbow digital signature scheme was given, along with all the basic foundations needed to understand it. Also, a new method to reduce Rainbow private keys was proposed. This method does not involve a direct modification of the central map, thus can be unified with other methods to achieve smaller Rainbow key pairs. The use of such method can reduce up to 84% of the private key.

5.2 FUTURE WORKS

The security analysis of Rainbow-η could be further expanded, specially its sensitivity to side-channel attacks, as this could be an issue of the original Rainbow too. Also, cryptanalytic attacks were not discarded, as an attacker with access to multiple signatures could potentially derive some information of the central map due to the reuse of the vinegar variables.

APÊNDICE A – ARTIGO DO TCC
Publicado no evento Africacrypt 2019

Handling Vinegar Variables to Shorten Rainbow Key Pairs

Gustavo Zambonin(✉), Matheus S. P. Bittencourt, and Ricardo Custódio

Departamento de Informática e Estatística
Universidade Federal de Santa Catarina
88040-900, Florianópolis, Brazil
gustavo.zambonin@posgrad.ufsc.br, matheus.spb@grad.ufsc.br, ricardo.custodio@ufsc.br

Abstract. Multivariate quadratic equations are the basis of one of the main mathematical techniques for the creation of digital signatures that are quantum-resistant. In these schemes, the creation and verification of signatures is highly efficient. However, key sizes are quite impractical and orders of magnitude greater than conventional schemes. One of the best-known signature schemes built upon multivariate equations is called Rainbow, which is based on the Oil-Vinegar principle. We observe that the reuse of vinegar variables in the signature generation step of the Rainbow scheme leads to a shorter representation of its central map, and thus, of the entire private key. We analyse the security implications of this strategy and present a modification to the Rainbow scheme, enabling a private key size reduction of up to 85% with secure parameters. Additionally, this framework can be applied on top of already existing schemes that shorten either private or public keys, spawning derivatives that reduce the total key pair size by a factor of 3.5.

Keywords: multivariate cryptography · digital signatures · Rainbow

1 Introduction

Secure exchange of messages is nowadays treated as a requirement in digital systems, instead of a privilege. It is often mandatory that data is not altered in transit, that its sender is uniquely identifiable and that it cannot deny having sent the message. These notions, known

as integrity, authenticity and non-repudiation, are achieved through the use of cryptographic foundations known as digital signatures. Data protected with such a method is adequate to prevent forgery and ensure confidentiality, according to Goldreich [13].

Conventional digital signature schemes are predominantly bound to one of two mathematical problems, namely integer factorisation and discrete logarithm. The most common examples are the RSA and ECDSA signature schemes [12], respectively. Nonetheless, in the wake of possible quantum adversaries, these problems are provably solvable in polynomial time, due to Shor's algorithm [26]. Ergo, the design of quantum-resistant, or post-quantum digital signature schemes, is indispensable to preserve secure communications in a scenario with quantum computers.

The creation of post-quantum digital signatures can be achieved through several approaches, one of which is based on systems of multivariate quadratic equations. Due to this fact, it is named multivariate cryptography, and schemes derived from this mathematical foundation are based on problems not known to be more efficiently solved by quantum computers [2]. Moreover, their signature generation and verification procedures are extremely efficient [7], since most computations rely only on simple finite field arithmetic.

It is known that multivariate cryptography hosts distinct schemes with several combinations of security parameters, signature and key pair lengths, as summarised by the authors of [8]. A balanced choice lies in the Rainbow signature scheme [9], itself a generalisation of the classic Unbalanced Oil and Vinegar (UOV) scheme [16]. It is a popular scheme, with several improvements featured in the literature, and multiple hardware implementations, *e.g.* [27,5,34]. Furthermore, it is currently featured in the second round of the standardisation process organised by the National Institute of Standards and Technology (NIST) [1].

One major drawback of multivariate cryptography, including Rainbow, is the size of private and public keys. While conventional signature schemes have key sizes that are a few bytes long, schemes based on multivariate equations feature keys that are dozens of kilobytes long. Hence, it is desired to reduce these by means of novel mathematical strategies, without decreasing the security of the scheme. Various strategies are applied to shorten keys, such as generating systems of equations represented by sparse matrices, or elements produced by cyclic recurrences. However, the security implications of such modifications are often obscure and possibly harmful.

Our contributions. We present a general framework that can be applied to any Rainbow-like signature scheme, with the final intent of reducing private key sizes. It manipulates vinegar variables that are originally chosen randomly to successfully invert the central map. These variables are now locked into the private key, thus reducing the degree of all monomials that feature such variables, lowering the total number of field elements used to represent it by up to a factor of 6.25. To sustain our proposal, we analyse the relation between signatures and the choice of vinegar variables, security implications of this strategy and experiment on known Rainbow variants. To the best of our knowledge, Rainbow variants proposed in the literature allow the reduction of private or public keys, but not both simultaneously. We show that our proposal allows for a shorter private key without preventing modifications to the public key. Thus, by making use of known proposals to reduce public keys, we create the first Rainbow variants that reduce the total size of the key pair.

Notation. We will use the following symbols throughout this work. The symbol $\xleftarrow{\$}$ is read as "chosen randomly from", and $\approx_\varepsilon$ means that two numbers are equal within a precision of ε. A finite field $\mathbb{F}$ with order q and elements as vectors of length n is represented as $\mathbb{F}_q^n$, with q and n omitted for brevity if appropriate. The cardinality of a set S is given by $|S|$. This notation may also be used as the absolute value of an integer, if applicable. The usual function composition is given by the symbol $\circ$, and the inverse of a function f is given by f^{-1}. The usual standard deviation and mean functions for a set of elements S are respectively given by $\sigma(S)$ and $\mu(S)$.

Organisation. The next sections are organised as follows. Section 2 succinctly describes the theoretical background needed to assimilate our proposal, with a definition of the Rainbow signature scheme in Subsection 2.1 and a review of works that already reduce keys for this scheme in Subsection 2.2. Section 3 presents the rationale for our proposal and a formal description, alongside a security analysis. Section 4 shows the impact of our proposal when applied to the original Rainbow and variants. Finally, Section 5 offers our final considerations.

2 Preliminaries

2.1 Original Rainbow signature scheme

We will present below a description of the Rainbow signature scheme, a generalised version of the UOV scheme that reduces the length of keys and signatures. It consists of several "oil and vinegar" layers, that are combined to create a "rainbow". Consider a finite field $\mathbb{F}_q$ and $u, n \in \mathbb{N}$ where $u \leq n$. Choose a sequence of integers $v_1, \ldots, v_u$ such that $0 = v_0 < v_1 < \cdots < v_u < v_{u+1} = n$. Take the usual set $V = \{1, \ldots, n\}$ and define the vinegar variables as $V_l = \{1, \ldots, v_l\}$ for all $l \in \{1, \ldots, u\}$. Observe that $v_l = |V_l|$ and $V_1 \subset \cdots \subset V_u = V$. Oil variables are given by $O_l = \{v_l + 1, \ldots, v_{l+1}\}$. Note that $o_l = |O_l|$ and $O_l = V_{l+1} - V_l$. Let $m = n - v_1$. Now, we define vector spaces spanned by quadratic Oil-Vinegar polynomials of the form

$$P_l = \sum_{i,j \in V_l} \alpha_{ij} \cdot x_i \cdot x_j + \sum_{i \in V_l, j \in O_l} \beta_{ij} \cdot x_i \cdot x_j + \sum_{i \in V_l \cup O_l} \gamma_i \cdot x_i + \delta. \quad (1)$$

Key generation. The central map of Rainbow is defined as $\mathcal{F} : \mathbb{F}^n \longrightarrow \mathbb{F}^m$, with the following construction: for each layer l, $F_l = (F_l^1, \ldots, F_l^{o_l}) \xleftarrow{\$} P_l$, and $\mathcal{F} = (F_1, \ldots, F_l)$. Since each sequence of vinegar variables in a layer contains all variables from the previous layer, this allows for the inversion of this map. Further, let $\mathcal{S} : \mathbb{F}^m \longrightarrow \mathbb{F}^m$ and $\mathcal{T} : \mathbb{F}^n \longrightarrow \mathbb{F}^n$ be two affine invertible maps, used as the trapdoor to this construction. Let $\mathcal{P} : \mathbb{F}^n \longrightarrow \mathbb{F}^m$ as $\mathcal{P} = \mathcal{S} \circ \mathcal{F} \circ \mathcal{T}$. Coefficients $\alpha_{ij}, \beta_{ij}, \gamma_i, \delta \in \mathbb{F}$ are chosen randomly. The private key is the triple $(\mathcal{S}, \mathcal{F}, \mathcal{T})$ and the public key is the map $\mathcal{P}$.

Signature generation. To sign a message M, consider a cryptographic hash function $\mathcal{H} : \{0,1\}^* \longrightarrow \mathbb{F}^m$, and obtain the message digest $d = \mathcal{H}(M)$. The signature will be the set of variables which yield the solution to the equation $\mathcal{P}(x_1, \ldots, x_n) = d$. Compute $x = \mathcal{S}^{-1}(d)$. To generate $y = \mathcal{F}^{-1}(x)$, every layer must be inverted recursively. Start by randomly choosing values for $x_1, \ldots, x_{v_1}$ and inserting them into the first layer. This will bring forth a system of o_1 linear equations in $x_{v_1+1}, \ldots, x_{v_2}$. It can be solved with an algorithm such as Gaussian elimination. If the system does not have a solution, new vinegar variables have to be chosen. These solutions can then be substituted into the next layer, which will create a system of o_2 linear equations, that can be solved analogously. This procedure is repeated until all layers are solved. Finally, we compute $\sigma = \mathcal{T}^{-1}(y)$.

Signature verification. To verify a signature, compute $d' = \mathcal{P}(\sigma)$. If $d = d'$, then the signature is valid, and invalid otherwise.

Finally, denote an instance of the scheme by Rainbow$(\mathbb{F}_q, v_1, o_1, \ldots, o_u)$. Note that when $u = 1$, we get the UOV scheme. Measured in field elements, the size of a private key is

$$|\mathcal{K}_{Pr}| = m^2 + m + n^2 + n + \sum_{k=1}^{u} o_k \cdot \left(\frac{v_k \cdot (v_k + 1)}{2} + v_k \cdot o_k + v_{k+1} + 1 \right), \quad (2)$$

whereas the size of a public key is

$$|\mathcal{K}_{Pu}| = m \cdot \frac{(n+1) \cdot (n+2)}{2}. \quad (3)$$

Further details on the construction of Rainbow may be found on [7, Section 3.3].

2.2 Related works

Schemes based on multivariate cryptography with modifications that enable the reduction of private key sizes have been suggested even before Rainbow was created. Tame transformation schemes, such as the ones listed by Wolf and Preneel in [29], feature sparseness in their maps, a common strategy used to shorten private keys. However, these schemes were either broken, as summarised by the authors in [10], or in the case of Enhanced TTS, new parameters were suggested, and it was subsequently found to be a special case of Rainbow [28].

Additionally, there have been several published variations of Rainbow with the same goal, making use of distinct approaches. A scheme called Lite-Rainbow-0 [25] makes use of a small pseudorandom number generator (PRNG) seed to replace the private key entirely. This shortens the private key by a factor of approximately 99.8%, but greatly increases the cost for signature generation. NC-Rainbow was proposed in [31] with a novel strategy based in non-commutative rings to reduce a private key by up to 75%. However, it was shown by independent researchers to be insecure [28,14]. Other variants called MB-Rainbow [30] and NT-Rainbow [33] employ sparseness of maps to reduce the number of terms in the private key by up to 40%.

The authors merged MB- and NC-Rainbow into a single scheme called MNT-Rainbow [32], shortening private keys by up to 76%. Nevertheless, the original schemes were deemed insecure and new parameters

were suggested in [18]. It also proposes a new scheme called Circulant Rainbow, which reduces the private key by up to 45% due to the concept of rotating relations. Yet, it was broken shortly after [15].

It is also relevant to cite the approach by the authors of [21], which is, to the best of our knowledge, the main method for public key reduction without compromises to the signature size. It is summarised in several publications [22,24,20]. However, these cannot be combined with the private key improvements previously cited. Furthermore, it appears that the introduction of structures in the private key is highly threatening to the overall security of a Rainbow scheme. We will subsequently present a novel approach to these issues.

3 Our proposal

We will describe our improvement to Rainbow-like signature schemes below, as well as supporting research on its soundness. Subsections 3.1 and 3.2 give a formal description of our modifications. In Subsection 3.3, we look into the probability of matrices with elements in finite fields being invertible. In Subsection 3.4, we present a statistical analysis of the structure of signatures created by our method, and finish with a security overview in Subsection 3.5.

3.1 Modification to the original scheme

Our approach consists of modifications to the key and signature generation steps of Rainbow-like signature schemes. We propose to reuse the first set of vinegar variables for several signatures and replace these only when necessary, *i.e.* situations where the central map cannot be inverted and creating a signature would fail. By locking such variables and substituting them on the central map $\mathcal{F}$ early in the key generation algorithm, we create a $\mathcal{F}'$ linear in v_1, thus reducing storage requirements. This approach does not modify the underlying structure of the private key, but rather of the central map preimages.

To induce lower storage requirements for key pairs of Rainbow-like schemes, we explore constructions given in the literature and suggest general alterations to use our proposal. As per Subsection 2.2, most variants that shorten private keys are structural in nature, that is, the key space is limited by some heuristic with the intent of producing a compact private key. Moreover, the main approach to reduce public keys [19] prevents alterations to the private key, since it indirectly generates $\mathcal{F}$ from a partial public key through linear relations between the maps.

This division of improvements is blurred by our proposal. We present general methods based on different techniques that shorten private keys in all Rainbow-like schemes. We collectively denote these by Rainbow-η and use the same definitions as in Subsection 2.1, further denoting the vinegar variables for the first layer as $\widetilde{V}_1 = (x_1, \ldots, x_{v_1})$.

Rainbow-η_1 key generation. We use the fact that a PRNG has the ability to regenerate the same sequence of numbers given a seed. The choice of such a generator is outside the scope of our work, and we assume that a cryptographically secure PRNG is chosen. This approach is similar to Lite-Rainbow-0, but it is not as costly, since the private key does not need to be regenerated before every signature generation. It is best suited to environments in which an efficient generator is previously supplied.

We bound the creation of the key pair to a seed $\mathbf{S}$. We are not aware of any Rainbow variants that disallow this practice. Thus, $\mathcal{S}$, $\mathcal{F}$ and $\mathcal{T}$, as well as the public key $\mathcal{P} = \mathcal{S} \circ \mathcal{F} \circ \mathcal{T}$ are generated through the target scheme key generation algorithm, seeded by $\mathbf{S}$. We set $\widetilde{V}_1 \xleftarrow{\$} \mathbb{F}$, and substitute these into $\mathcal{F}$, giving $\mathcal{F}'$. According to Subsection 3.3, in the rare case that a failure occurs in the central map inversion algorithm, we use $\mathbf{S}$ to regenerate $\mathcal{F}$, choose other values for $\widetilde{V}_1$ and create a different $\mathcal{F}'$. The private key of Rainbow-η_1 is $(\mathbf{S}, \mathcal{S}, \mathcal{F}', \mathcal{T})$ and the public key is $\mathcal{P}$.

Rainbow-η_2 key generation. This approach is based on the fact that a private key owner is able to recover the original $\mathcal{F}$ through the possession of all other private maps and the public key. We make use of the linear relations given by the authors of [21] and applied in the definition of the well-known CyclicRainbow scheme. A short explanation is given below, with the full rationale available in [19, Chapter 7].

Consider the public key $\mathcal{P} = \mathcal{S} \circ \mathcal{F} \circ \mathcal{T}$ and let $\mathcal{Q} = \mathcal{F} \circ \mathcal{T}$. Denote $\widetilde{Q}$ as a matrix containing only coefficients of the quadratic monomials from $\mathcal{Q}$, and define $\widetilde{F}$ and $\widetilde{P}$ similarly. Further let $\widetilde{T}$ be the matrix representation of $\mathcal{T}$, with its coefficients $t_{ij}, i, j \in \{1, \ldots, n\}$, and define $\widetilde{S}$ analogously. By fixing t_{ij}, the composition of $\mathcal{P}$ actually represents a linear relation between coefficients q_{ij}^k, f_{ij}^k of the monomial $x_i \cdot x_j$ in

the k-th component of, respectively, $\mathcal{Q}$ and $\mathcal{F}$, with the form

$$q_{ij}^k = \sum_{r=1}^{n}\sum_{s=r}^{n} \alpha_{ij}^{rs} \cdot f_{rs}^k, \quad \alpha_{ij}^{rs} = \begin{cases} t_{ri} \cdot t_{si} & \text{if } i = j, \\ t_{ri} \cdot t_{sj} + t_{rj} \cdot t_{si} & \text{otherwise,} \end{cases} \quad k \in \{v_1 + 1, \ldots, n\}. \tag{4}$$

This can be simplified, since $\mathcal{F}$ does not allow quadratic monomials with only oil variables, and results in

$$q_{ij}^k = \sum_{r=1}^{v_l}\sum_{s=r}^{v_{l+1}} \alpha_{ij}^{rs} \cdot f_{rs}^k, \quad k \in O_l, \quad l \in \{1, \ldots, u\}. \tag{5}$$

A square matrix of order $\frac{n^2+n}{2}$ is created to further streamline the previous equations. Given a particular monomial ordering, let $A = (\alpha_{ij}^{rs})$ such that $i, j, r, s \in \{1, \ldots, n\}$, where $i \leq j$ and $r \leq s$ denote row and column indices, respectively. Thus, we have that $\widetilde{P} = \widetilde{S} \cdot \widetilde{Q}$ and $\widetilde{Q} = \widetilde{F} \cdot A^{\mathrm{T}}$. We note that the performance of this method is lower than that of Rainbow-η_1. However, it is a general technique that works on all Rainbow-like schemes.

Observe that the central map may not feature any linear or constant terms, due to the use of the above relations. This does not lower the overall security of the scheme, due to the fact that they are not multiplied with quadratic terms. With this implication in mind, the usual key generation algorithm for the target scheme is employed, yielding $(\mathcal{S}, \mathcal{F}, \mathcal{T})$ and $\mathcal{P}$ at a marginally faster rate. Substitute the sequence $\widetilde{V}_1 \xleftarrow{\$} \mathbb{F}$ into $\mathcal{F}$, giving $\mathcal{F}'$. By the relations above, one is able to reconstruct $\mathcal{F}$ with no additional mechanisms if the central map inversion algorithm fails. The private key of Rainbow-η_2 is $(\mathcal{S}, \mathcal{F}', \mathcal{T})$ and the public key is $\mathcal{P}$.

Signature generation. A digest $d = \mathcal{H}(M)$ from a message M is signed with a similar procedure. Compute $x = \mathcal{S}^{-1}(d)$, and attempt to generate $y = \mathcal{F}'^{-1}(x)$ by inverting every layer recursively. The first layer already has $\widetilde{V}_1$ set, and the remaining linear system needs only to be solved by providing appropriate values of d. It will generate a new set of vinegar variables, that can be used on the next layer, until all layers are solved. If any of the transitory systems are not solvable, a new $\widetilde{V}_1$ is chosen and $\mathcal{F}'$ regenerated, according to one of the methods given above. We finish by computing $\sigma = \mathcal{T}^{-1}(y)$.

Signature verification. This step does not change. If $d = \mathcal{P}(\sigma)$, then the signature is valid, and invalid otherwise.

By making the first layer linear and substituting the remaining variables, the size of the private key is now

$$|\mathcal{K}^{\eta}_{Pr}| = m^2 + m + n^2 + n + |\widetilde{V}_1| + \sum_{k=1}^{u} o_k \cdot \left(\frac{(v_k - v_1)(v_k - v_1 + 1)}{2} + (v_k - v_1) \cdot o_k + (v_{k+1} - v_1) + 1 \right), \tag{6}$$

plus the additional size of $\mathbf{S}$ if the Rainbow-η_1 method is used. One needs to store $\widetilde{V}_1$, since it is part of the central map preimage, used on further map applications. The public key size does not change.

3.2 Application to the EF-CMA variant

The Rainbow submission to the NIST standardisation process [6] presents a scheme description that diverges from the original works. The authors introduce modifications that provide security against the existential forgery under chosen-message attack (EF-CMA) model, whereas the original scheme only offers security against universal forgery. These changes are built upon the introduction of a random salt. We will briefly describe this approach, with the intent of preventing the recalculation of $\mathcal{F}'$ in the case that $\widetilde{V}_1$ is not suitable. Let us denote this method as **Rainbow-η_3**.

Key generation. Consider $w \in \mathbb{N}$ as the length of the aforementioned salt. Generate private and public keys as per Subsection 3.1. The private key for this scheme is $(\mathcal{S}, \mathcal{F}', \mathcal{T}, w)$, with the addition of $\mathbf{S}$ in the case of Rainbow-η_1. The public key is $(\mathcal{P}, w)$.

Signature generation. Let $r \xleftarrow{\$} \{0,1\}^w$. The digest value is calculated as $\mathbf{d} = \mathcal{H}(\mathcal{H}(M) \mid\mid r)$, where M is the message. The value $x = \mathcal{S}^{-1}(\mathbf{d})$ is obtained as usual. In the rare case that the $y = \mathcal{F}'^{-1}(x)$ preimage calculation does not succeed, new variables in $\widetilde{V}_1$ are chosen. However, the addition of a random salt to the original message digest alters $\mathbf{d}$ completely, due to the cryptographic hash function application. Thus, it is only necessary to generate a new r and restart the signature generation process, such that $\widetilde{V}_1$, and consequently $\mathcal{F}'$, are not modified. Alternatively, if the preimage is generated successfully, we finish by letting $z = \mathcal{T}^{-1}(y)$ and $\sigma = (z, r)$.

Signature verification. Recalculate the digest value $\mathbf{d}$. If $\mathbf{d} = \mathcal{P}(z)$, the signature is valid, and invalid otherwise.

The size of the private and public keys increase in exactly one element due to the addition of w. Real implementations of Rainbow-η_3 are tested on Section 4.

3.3 Invertibility of $\mathcal{F}$

Recall that, to create a Rainbow signature, the central map $\mathcal{F}$ needs to be inverted. Random guessing of vinegar variables is done in order to create a solvable linear system. It is also known that the central map is expressed as multivariate systems of equations, which can be themselves interpreted as multidimensional matrices of coefficients. Observe that, to describe these in a clearer way, a given monomial ordering is used such that only usual matrices are needed. With this in mind, we first derive the probability that a random matrix with elements in $\mathbb{F}$ is invertible.

Assume a square matrix M of order n such that $m_{ij} \in \mathbb{F}_q, i, j \in \{1, \ldots, n\}$. For M to be invertible, it must be composed entirely of vectors, *i.e.* its rows $m_i \in \mathbb{F}^n$, that are linearly independent. The zero vector $(0, \ldots, 0) \in \mathbb{F}^n$ is linearly dependent of all other vectors. Thus, $m_1 \neq (0, \ldots, 0)$, with all other $q^n - 1$ possible vectors eligible. m_2 must not feature any of the q multiples of m_1, and $q^n - q$ vectors remain. Without loss of generality, $m_k \neq c_1 v_1 + c_2 v_2 + \cdots + c_{k-1} v_{k-1}, c_k \in \mathbb{F}$, and $q^n - q^{k-1}$ vectors can be selected. Then, the probability that all vectors chosen are linearly independent is

$$\begin{aligned} \Pi(q, n) &= \prod_{k=1}^{n} \frac{q^n - q^{k-1}}{q^{-n}} \\ &= \prod_{k=1}^{n} 1 - q^{-k}. \end{aligned} \tag{7}$$

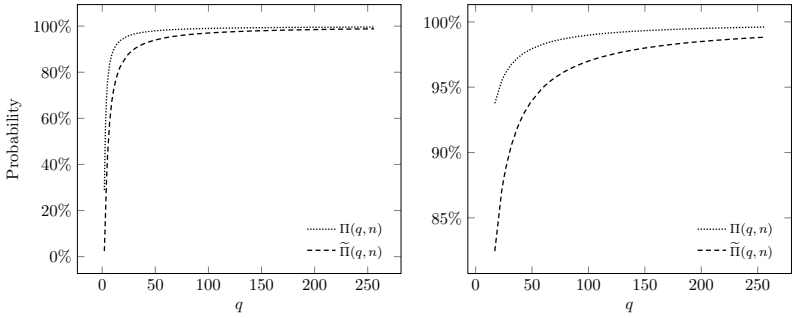
In the context of Rainbow, the number of layers directly influences $\Pi(q, n)$, since it dictates how many linear systems have to be solved. In other words, all square matrices of size $v_i, i \in \{1, \ldots, u\}$ need to be invertible to achieve a preimage under $\mathcal{F}$. Thus, the probability

$$\Pi(q, n, u) = \prod_{i=1}^{u} \prod_{k=1}^{v_{i+1}} 1 - q^{-k} \tag{8}$$

more accurately represents the upper bound for these chances. In the literature, the usual number of layers for a Rainbow instance is two,

and we will denote this common case as $\Pi(q, n, 2) = \widetilde{\Pi}(q, n)$. Note that $\Pi(q, n, 1) = \Pi(q, n)$. Hence, schemes with more layers have a slightly lower probability of success in the signature generation preimage step.

Parameters for Rainbow are selected according to a number of restrictions, imposed by attacks that may harm the security of the scheme. Furthermore, note that the central map can be represented as square matrices of order n. Hence, we choose $n \in \{56, \ldots, 90\}$ from [19, Tables 6.4, 6.8, 6.13] and calculate the probability that a random matrix is invertible in finite fields of typical orders. For instance, $\Pi(16, 90) \approx 93.3594\%$ and $\Pi(256, 56) \approx 99.6078\%$. Figure 1 depicts the lowest probabilities computed for the appropriate range. To simulate layering, we set $v_i = i \cdot \lceil \frac{n}{u} \rceil$ and approximate to n when needed.



(a) $\operatorname{argmin}_{56 \leq n \leq 90}$ of $\Pi(q, n)$ and $\widetilde{\Pi}(q, n)$.

(b) Figure 1a, with $q \geq 16$.

Fig. 1. Probability of obtaining an invertible matrix, populated with field elements where $q \in \{2, \ldots 256\}$ and q is a prime power, given the quantity of layers of Rainbow.

It is also useful to calculate $\lim_{n \to \infty} \Pi(q, n)$ to observe changes in the probability with the growth of m. Note that this is very similar to the Euler function $\phi(q)$. Ergo, we can use one of Euler's identities to redefine the above limit as

$$\Pi(q) = \sum_{k=-\infty}^{\infty} (-1)^k q^{\frac{-3 \cdot k^2 + k}{2}} \tag{9}$$

and obtain a fast approximation of the probability when n tends to infinity. We use the SageMath language arbitrary precision real numbers to obtain these values and find out that, when $n \geq 56$, $\Pi(q) \approx_{10^{-18}} \Pi(q, n)$ and $\widetilde{\Pi}(q) \approx_{10^{-8}} \widetilde{\Pi}(q, n)$. Thus, Figure 1 also accurately reflects the behaviour of $\Pi(q)$, *i.e.*, current values of n already reach effective upper bounds for this probability.

If we consider that the two-dimensional coefficient matrix of $\mathcal{F}$ has an effective size of $\frac{n^2+n}{2}$ due to the aforementioned monomial ordering strategy, we note that the inversion event happens almost surely. This evidence shows that computing a preimage in order to sign a message happens at the first try with high probability in a wide range of Rainbow configurations. Therefore, the cost of a central map reconfiguration, in the case that chosen vinegar variables do not lead to an invertible central map, is amortised by the overwhelming probability that a signature is successfully generated.

3.4 Similarity of multiple signatures

Vinegar variables chosen to invert the central map are an integral part of the preimage $y = \mathcal{F}^{-1}(x)$. For instance, in the case $u = 2$, these make roughly a third of the output, considering common parameters for Rainbow. Further, recall that there are approximately q^v possibilities for y. Our proposal eliminates this choice by locking vinegar variables into the private key. Hence, it is essential to know if such variables create patterns in which private information may leak through a multi-target attack. We use the SageMath PRNG, which implements a front-end to the `/dev/urandom` Linux kernel space generator.

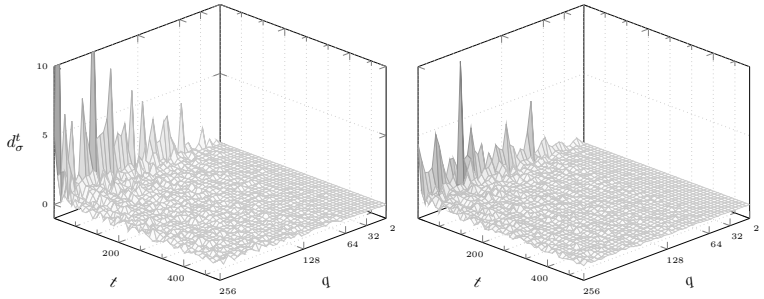
Recall that a message digest d is signed instead of the entire document. Evidently, a secure cryptographic hash function shall produce an output that appears to be random. The application $x = \mathcal{S}^{-1}(d)$ does not affect this behaviour, since the map is also random. Hence, we need not simulate this calculation in this analysis. According to Subsection 3.3, the inversion $y = \mathcal{F}^{-1}(x)$ creates a valid preimage with overwhelming probability, where the first v_1 elements of any y will be the same.

We observe the distribution of field elements in vectors after the final function application, that is, $z = \mathcal{T}^{-1}(y)$. Let $Z'_t = (z_1, \ldots, z_t) \xleftarrow{\$} \mathbb{F}^n, t \in \mathbb{N}$ be a t-uple of "signatures". We build the sequence $Z_t = (z_1^1, z_1^2, \ldots, z_1^n, z_2^1, \ldots, z_t^{n-1}, z_t^n)$. When part of the vector y is fixed, we will instead denote these by $\widetilde{Z}'_t$ and $\widetilde{Z}_t$. Our hypothesis is that Z_t and $\widetilde{Z}_t$ will behave similarly to observations sampled from the discrete uniform

distribution $\mathcal{U}\{0, q-1\}$. It is known that its standard deviation, where r values are observed in an equally likely manner, is equal to $\sqrt{\frac{r^2-1}{12}}$. For a finite field $\mathbb{F}$, we set $r = q$ and obtain the desired value. It is expected that

$$\lim_{t\to\infty} \sigma(\widetilde{Z}_t) = \sqrt{\frac{q^2-1}{12}}, \tag{10}$$

suggesting that greater values of n and t approximate faster to the theorised standard deviation.



(a) d_σ^t for $n = 42, v_1 = 17$. (b) d_σ^t for $n = 90, v_1 = 35$.

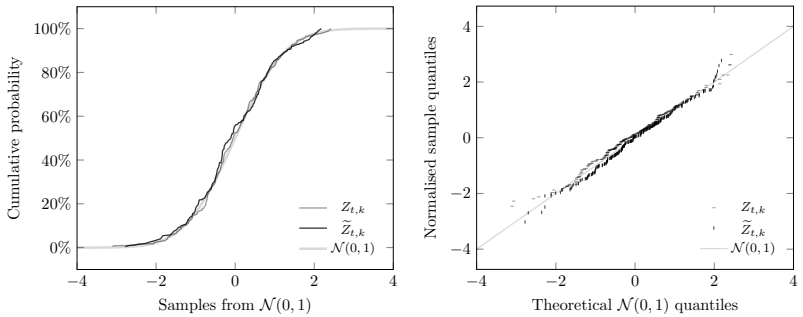
Fig. 2. Difference of standard deviations when $t \in \{1, \ldots, 1024\}$, and $q \in \{2, \ldots, 256\}$ with q as a prime power.

Let us denote the absolute difference between standard deviations for a value t as $d_\sigma^t = |\sigma(Z_t) - \sigma(\widetilde{Z}_t)|$. Figure 2 shows the amplitude of such values for various values of q and t. We note that the largest values of d_σ^t occur for finite fields of higher orders and lower t. For instance, given the finite field $\mathbb{F}_{223}^{42}$, we have $d_\sigma^1 \approx 8.25$, and for a slightly higher t, we obtain a much lower value $d_\sigma^{11} \approx 0.24$. This behaviour is also observed within absolute differences of means, defined analogously as d_μ^t. The field $\mathbb{F}_{191}^{42}$ gives the values $d_\mu^1 \approx 5.31$ and, comparatively, $d_\mu^9 \approx 1.14$.

The comparison of expected and obtained standard deviations and means in our experiments, gives positive results and confirms the law of large numbers. Still, it is interesting to look at the diffusion of values within $\widetilde{Z}_t$ and infer that it does not simply simulate the mean and standard deviation for a known discrete uniform. We count the amount

of values for each class $k \in \{0, \ldots, q-1\}$ and refer to them by $Z_{t,k}$ and $\widetilde{Z}_{t,k}$. By the central limit theorem, these counts should be normally distributed.

Figure 3 shows the cumulative distribution function (CDF) plot and the Q-Q (quantile-quantile) plot for such samples. The expected CDF, as well as examples for Z_t and $\widetilde{Z}_t$, show that all values are fairly distributed, with small variations due to the random generation of field elements. However, we note that this is due to the low number of classes, *i.e.* the order of the finite field, and experimentally confirm that such discrepancies are largely reduced with $q = 2^{10}$. This is further confirmed by the Q-Q plot created with rankits, where the points are sufficiently close to the $y = x$ expected line.



(a) CDF plot. (b) Q-Q plot.

Fig. 3. Distribution of counts of elements in Z_t and $\widetilde{Z}_t$ such that $t = 2^{16}$ for $\mathbb{F}_{256}^{90}$.

Our argument indicates that, even if part of the preimage created by the central map is fixed, the remaining affine map application disrupts this pattern with high efficacy. Hence, an attacker with possession of multiple signatures created by our method would not be more capable of forging a new signature or deducing private information.

3.5 Security analysis

A variety of attacks currently thwart the security of Rainbow-like signature schemes if parameters are not chosen carefully. We will briefly

state each of those, along with their estimated complexities [23], and argue that our methods do not facilitate such attacks.

Direct attack. An attacker with possession of a digest d and the public key $\mathcal{P}$ tries to solve $\mathcal{P}(x) = d$. This is done by fixing some of the variables and applying an algorithm built upon the theory of Gröbner basis, such as the Hybrid approach [3]. While it is hard to pinpoint the exact running time of such methods, the authors give an estimation of its asymptotic complexity in Equation 5 of the aforementioned work.

UOV attack. The multi-layer approach of Rainbow does not hinder attacks that also work on the UOV signature scheme. This attack was originally created by Kipnis and Shamir [17] to break the Balanced Oil-Vinegar scheme. The objective of this attack is to obtain an equivalent private key by means of finding the preimage of a specific oil subspace under the map $\mathcal{T}$. The complexity of the generalised attack for unbalanced schemes [16] is $o_u^4 \cdot q^{n-1-2\cdot o_u}$ field multiplications.

MinRank attack. All systems of polynomials in the public key $\mathcal{P}$ may be individually represented as matrices. This attack consists in finding linear combinations of these, such that they have a lesser rank than v_2, in the case of Rainbow. This allows an attacker to isolate the central map polynomials from the first layer of Rainbow, and analogously recover the remaining layers with a much lower effort. In the context of Rainbow [4], its complexity is $q^{v_1+1} \cdot m \cdot (\frac{n^2}{2} - \frac{m^2}{6})$ field multiplications.

HighRank attack. In a similar way to MinRank, linear combinations of public key matrices are used to find the variables which appear the lowest number of times in the central map. This is used to identify the last Rainbow layer, and obtain the previous layers similarly. The complexity of the improved attack [11] is $q^{o_u} \cdot \frac{n^3}{6}$ field multiplications.

Rainbow-Band-Separation attack. An extension of the UOV - Reconciliation attack by the same authors [11] that targets Rainbow, with the intent of producing an equivalent private key. It explores the fact that the central map matrix representation is composed of zeroes on its lower right corner. These yield quadratic equations which, if solved, lead to an alternative private key. The complexity of this attack is given by the hardness of solving a large system of equations, as seen above, is hard to estimate.

Side-channel attacks. It may be observed that none of the proposed Rainbow variants, as well as the original scheme, present constant time signature generation algorithms. Particularly, in Rainbow-η_2, a considerable amount of computation is added to the signature algorithm when one of the systems is not solvable. In a chosen message attack, one may observe the time spent on multiple signature generation steps and easily check if the linear systems are solvable, thus obtaining information about the central map. Although there are no known attacks that make use of this technique, it is possible that there may exist information leaks when applying our methods to Rainbow-like schemes.

We do not discard the possibility that specialised attacks exist, particularly ones that take in account multiple signatures, due to our fixing of vinegar variables. However, we have seen in Subsection 3.4 that signatures generated by our method are comparably random with respect to conventional Rainbow signatures. Furthermore, we note that most attacks look for special structures within the private key. While our methods indeed modify the private key representation, it is still present in its entirety on the public key composition, which is the only information available to malicious entities that can be possibly used to forge signatures. We thus suggest that the right choice of parameters is made whenever our methods are applied, *e.g.* according to [23], to protect the scheme instance against these attacks.

4 Enhancement of existing schemes

Our method does not depend on special structures inserted on the private key. Consequently, it can be applied to all known Rainbow-like schemes. We experiment with several sets of parameters and observe the reduction of private keys. It is known that there are various limitations for the choice of parameters that lead to secure instances of Rainbow [23]. We implement several known guidelines and confirm that our proposal does indeed work for a large range of parameters. However, we only show results for known secure parameter sets to prevent accidental endorsement of untested, and possibly insecure, instances.

We show results for the application of our method in Table 1, considering the following Rainbow instances. Conservative choices were made by the Rainbow submission authors [6] to fit security categories as requested by NIST. We apply our method to these recent proposals, and additionally choose parameters from Petzoldt [19, Table 6.12] for further comparison. The latter are named P-ℓ, where ℓ is the security

Table 1. Reduction of Rainbow key sizes, in bytes, for various instances of the scheme.

Instance	Parameters	n	m	$\lvert\mathcal{K}_{Pr}\rvert$	$\lvert\mathcal{K}_{Pr}^{\eta}\rvert$	Difference
I-a	$(\mathbb{F}_{16}, 32, 32, 32)$	96	64	100208	33152	-66.92%
I-b	$(\mathbb{F}_{31}, 36, 28, 28)$	92	56	114308	31676	-72.29%
I-c	$(\mathbb{F}_{256}, 40, 24, 24)$	88	48	143384	33024	-76.97%
III-b	$(\mathbb{F}_{31}, 64, 32, 48)$	144	80	409463	87628	-78.60%
III-c	$(\mathbb{F}_{256}, 68, 36, 36)$	140	72	537780	99656	-81.47%
IV-a	$(\mathbb{F}_{16}, 56, 48, 48)$	152	96	376140	103336	-72.53%
V-c	$(\mathbb{F}_{256}, 92, 48, 48)$	188	96	1274316	218984	-82.82%
VI-a	$(\mathbb{F}_{16}, 76, 64, 64)$	204	128	892078	233044	-73.88%
VI-b	$(\mathbb{F}_{31}, 84, 56, 56)$	196	112	1016868	217244	-78.64%
P-080	$(\mathbb{F}_{256}, 17, 17, 9)$	43	26	19208	5914	-69.21%
P-100	$(\mathbb{F}_{256}, 26, 22, 21)$	69	43	75440	23193	-69.26%
P-128	$(\mathbb{F}_{256}, 36, 28, 15)$	79	43	103704	22110	-78.68%
P-192	$(\mathbb{F}_{256}, 63, 46, 22)$	131	68	440638	71773	-83.71%
P-256	$(\mathbb{F}_{256}, 85, 63, 30)$	178	93	1086971	164721	-84.85%

level in bits. Indeed, the choice of v_1 remarkably affects the results. Moreover, a minimal value of o_u is also known to further reduce the private key size. Indeed, we suggest that $v_1 \geq o_u$ as much as possible to maximise the results of our method. However, we remark that one must set sufficient parameters for o_i such that the scheme still resists direct and UOV attacks.

The case of Rainbow variants is slightly more convoluted. Schemes claim optimisations of the private key often through the inclusion of inner structuring. To measure the impact of our method within the context of these schemes, it is imperative to understand such structures. For instance, it may be the case that a method introduces sparseness related to specific vinegar variables. Thus, the reduction would not be equally distributed over the private key elements and, as such, our method would have its efficiency reduced.

To the best of our knowledge, the schemes presented in Subsection 2.2 feature changes that target the whole private key evenly. Hence, our method would yield similar results to those in Table 1 if this assumption is true. However, it is also the case that some variants were subsequently broken or new parameters were suggested. We will thus

consider only schemes that reduce the public key size, *i.e.* CyclicRainbow [22] and RainbowLRS2 [19, Section 9.2].

Table 2. Total reduction of Rainbow key pairs, in bytes, for variants of the scheme.

Instance	Parameters	Variant	$\lvert\mathcal{K}_{Pr}\rvert$	$\lvert\mathcal{K}^{\eta}_{Pr}\rvert$	$\lvert\mathcal{K}_{Pu}\rvert$	Difference
P-080	$(\mathbb{F}_{256}, 17, 13, 13)$	Classic	19546	6524	25740	−28.76%
		Cyclic			10618	−62.15%
		LRS2			9789	−63.98%
P-100	$(\mathbb{F}_{256}, 26, 16, 17)$	Classic	46131	12474	60390	−31.60%
		Cyclic			22246	−67.41%
		LRS2			20662	−68.89%
P-128	$(\mathbb{F}_{256}, 36, 21, 22)$	Classic	105006	24924	139320	−32.78%
		Cyclic			48411	−69.98%
		LRS2			45547	−71.16%

We compare the total key pair sizes $|\mathcal{K}_{Pr}| + |\mathcal{K}_{Pu}|$ when our method is used alongside Rainbow variants that reduce the public key size. Table 2 shows the quantity of field elements for sets of parameters from Petzoldt [19, Table 9.8]. We calculate $|\mathcal{K}_{Pu}|$ for the variants according to Equations 9.2 and 9.4 of the same work, and as per its Remark 9.1, note that $q = 16$ and $q = 31$ are not considered due to a security restriction of RainbowLRS2. We obtain positive results, with key pair size reductions of up to factors of 3 and no security harm to the resulting scheme.

The use of CyclicRainbow or RainbowLRS2 with the Rainbow-η_2 method is recommended. These variants are based on the linear relations described in Subsection 3.1, and resulting implementations may be effortlessly modified to use our proposal. Moreover, in the case that higher parameters are needed, *e.g.* a security level of 256 bits, we note that the key pair will be reduced more aggressively. Thus, our results reflect changes over a wide variety of platforms and possible Rainbow deployments that benefit from lower storage requirements.

We also briefly discuss the effect of these changes on the signature generation step overall performance. In the case of Rainbow-η_1, it does not vary greatly due to the fast regeneration of the central map elements from a given PRNG and $\mathbf{S}$. On the other hand, Rainbow-η_2 uses

elaborate techniques to reconstruct the central map if vinegar variables are not suitable. This process is not without cost, and it may negatively affect the average signature generation time. Still, by making use of Rainbow-η_3, these computations are entirely avoided by choosing a new salt instead of new vinegar variables, reducing the inherent overhead.

5 Conclusion

Throughout this work, we have proposed general methods to lower private key sizes that can be applied to all known Rainbow variants. We suggest fixing the first sequence of vinegar variables and reuse it on the creation of signatures, reducing the static central map storage requirements, and thus obtaining a smaller private key. Our security analysis shows that this modification creates orderly signatures and does not harm the target scheme. Furthermore, we have also addressed the problem in which no scheme could reduce both keys in the key pair, by applying our proposal to known variants that reduce the public key size. We obtain gains of up to 85% on the private key size and 71% on the total key pair size.

We propose some topics to extend this work. Evidently, it is crucial for the security of our proposal that multiple signatures do not leak information for the chosen vinegar variables. Thus, we point out that further security analysis on multi-target and side-channel attacks is desirable. We also observe that our methods directly affect the signature generation performance, since the first layer computations are moved to the key generation step. As such, we suggest that measurements are made considering the average time for signature generation, in the case that the private key has to be recomputed due to a new choice of vinegar variables.

Acknowledgements

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. Additionally, we thank the anonymous referees for their suggestions.

References

1. Alagic, G., Alperin-Sheriff, J., Apon, D., Cooper, D., Dang, Q., Miller, C., Moody, D., Peralta, R., Perlner, R., Robinson, A., Smith-Tone, D., Liu, Y.: Status Report on the First Round of the NIST Post-Quantum Cryptography Standardization Process. Internal Report 8240, National Institute of Standards and Technology (NIST) (Jan 2019). https://doi.org/10.6028/NIST.IR.8240
2. Bernstein, D.J., Buchmann, J., Dahmen, E.: Post Quantum Cryptography. Springer, 1st edn. (2008)
3. Bettale, L., Faugère, J.C., Perret, L.: Solving Polynomial Systems over Finite Fields: Improved Analysis of the Hybrid Approach. In: Proceedings of the 37th International Symposium on Symbolic and Algebraic Computation. pp. 67–74 (Jul 2012). https://doi.org/10.1145/2442829.2442843
4. Billet, O., Gilbert, H.: Cryptanalysis of Rainbow. In: de Prisco, R., Yung, M. (eds.) Security and Cryptography for Networks. Lecture Notes in Computer Science, vol. 4116, pp. 336–347 (Sep 2006). https://doi.org/10.1007/11832072_23
5. Czypek, W.: Implementing Multivariate Quadratic Public Key Signature Schemes on Embedded Devices. Master's thesis, Ruhr-Universität Bochum (Apr 2012)
6. Ding, J., Chen, M.S., Petzoldt, A., Schmidt, D., Yang, B.Y.: Rainbow - Algorithm Specification and Documentation. Round 1 Submission, NIST Post-Quantum Cryptography Standardisation Process (Dec 2017)
7. Ding, J., Gower, J., Schmidt, D.: Multivariate Public Key Cryptosystems. Springer, 1st edn. (2006)
8. Ding, J., Petzoldt, A.: Current State of Multivariate Cryptography. IEEE Security & Privacy **15**(4), 28–36 (Jul 2017). https://doi.org/10.1109/MSP.2017.3151328
9. Ding, J., Schmidt, D.: Rainbow, a New Multivariable Polynomial Signature Scheme. In: Ioannidis, J., Keromytis, A., Yung, M. (eds.) Applied Cryptography and Network Security. Lecture Notes in Computer Science, vol. 3531, pp. 164–175 (Jun 2005). https://doi.org/10.1007/11496137_12
10. Ding, J., Schmidt, D., Yin, Z.: Cryptanalysis of the new TTS scheme in CHES 2004. International Journal of Information Security **5**(4), 231–240 (Apr 2006). https://doi.org/10.1007/s10207-006-0003-9
11. Ding, J., Yang, B.Y., Chen, C.H., Chen, M.S., Cheng, C.M.: New Differential-Algebraic Attacks and Reparametrization of Rainbow. In: Bellovin, S., Gennaro, R., Keromytis, A., Yung, M. (eds.) Applied Cryptography and Network Security. Lecture Notes in Computer Science, vol. 5037, pp. 242–257 (Jun 2008). https://doi.org/10.1007/978-3-540-68914-0_15
12. von zur Gathen, J.: CryptoSchool. Springer, 1st edn. (2015)
13. Goldreich, O.: Foundations of Cryptography: Volume 2, Basic Applications. Cambridge University Press, 1st edn. (2004)

14. Hashimoto, Y.: Cryptanalysis of the Quaternion Rainbow. In: Sakiyama, K., Terada, M. (eds.) Advances in Information and Computer Security. Lecture Notes in Computer Science, vol. 8231, pp. 244–257 (Feb 2013). https://doi.org/10.1007/978-3-642-41383-4_16
15. Hashimoto, Y.: On the security of Circulant UOV/Rainbow. Cryptology ePrint Archive, Report 2018/847 (Oct 2018), https://eprint.iacr.org/2018/947
16. Kipnis, A., Patarin, J., Goubin, L.: Unbalanced Oil and Vinegar Signature Schemes. In: Stern, J. (ed.) Advances in Cryptology – EUROCRYPT '99. Lecture Notes in Computer Science, vol. 1592, pp. 206–222 (Apr 1999). https://doi.org/10.1007/3-540-48910-X_15
17. Kipnis, A., Shamir, A.: Cryptanalysis of the Oil and Vinegar Signature Scheme. In: Krawczyk, H. (ed.) Advances in Cryptology – CRYPTO '98. Lecture Notes in Computer Science, vol. 1462, pp. 257–266 (Aug 1998). https://doi.org/10.1007/BFb0055733
18. Peng, Z., Tang, S.: Circulant Rainbow: A New Rainbow Variant With Shorter Private Key and Faster Signature Generation. IEEE Access **5**, 11877–11886 (Jun 2017). https://doi.org/10.1109/ACCESS.2017.2717279
19. Petzoldt, A.: Selecting and Reducing Key Sizes for Multivariate Cryptography. Ph.D. thesis, Technische Universität Darmstadt (Jul 2013)
20. Petzoldt, A., Bulygin, S.: Linear Recurring Sequences for the UOV Key Generation Revisited. In: Kwon, T., Lee, M.K., Kwon, D. (eds.) Information Security and Cryptology – ICISC 2012. Lecture Notes in Computer Science, vol. 7839, pp. 441–455 (Nov 2012). https://doi.org/10.1007/978-3-642-37682-5_31
21. Petzoldt, A., Bulygin, S., Buchmann, J.: A Multivariate Signature Scheme with a Partially Cyclic Public Key. In: Faugère, J.C., Cid, C. (eds.) International Conference on Symbolic Computation and Cryptography. pp. 229–235 (Jun 2010)
22. Petzoldt, A., Bulygin, S., Buchmann, J.: CyclicRainbow – A Multivariate Signature Scheme with a Partially Cyclic Public Key. In: Gong, G., Gupta, K.C. (eds.) Progress in Cryptology – INDOCRYPT 2010. Lecture Notes in Computer Science, vol. 6498, pp. 33–48 (Dec 2010). https://doi.org/10.1007/978-3-642-17401-8_4
23. Petzoldt, A., Bulygin, S., Buchmann, J.: Selecting Parameters for the Rainbow Signature Scheme. In: Sendrier, N. (ed.) Post-Quantum Cryptography. Lecture Notes in Computer Science, vol. 6061, pp. 218–240 (May 2010). https://doi.org/10.1007/978-3-642-12929-2_16
24. Petzoldt, A., Bulygin, S., Buchmann, J.: Linear Recurring Sequences for the UOV Key Generation. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) Public Key Cryptography – PKC 2011. Lecture Notes in Computer Science, vol. 6571, pp. 335–350 (Mar 2011). https://doi.org/10.1007/978-3-642-19379-8_21
25. Shim, K.A., Park, C.M., Baek, Y.J.: Lite-Rainbow: Lightweight Signature Schemes Based on Multivariate Quadratic Equations and Their

Secure Implementations. In: Biryukov, A., Goyal, V. (eds.) Progress in Cryptology – INDOCRYPT 2015. Lecture Notes in Computer Science, vol. 9462, pp. 45–63 (Dec 2015). https://doi.org/10.1007/978-3-319-26617-6_3
26. Shor, P.W.: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. SIAM Journal on Computing **26**(5), 1484–1509 (Oct 1997). https://doi.org/10.1137/S0097539795293172
27. Tang, S., Yi, H., Ding, J., Chen, H., Chen, G.: High-Speed Hardware Implementation of Rainbow Signature on FPGAs. In: Yang, B.Y. (ed.) Post-Quantum Cryptography. Lecture Notes in Computer Science, vol. 7071, pp. 228–243 (Nov 2011). https://doi.org/10.1007/978-3-642-25405-5_15
28. Thomae, E., Wolf, C.: Cryptanalysis of Enhanced TTS, STS and All Its Variants, or: Why Cross-Terms Are Important. In: Mitrokotsa, A., Vaudenay, S. (eds.) Progress in Cryptology – AFRICACRYPT 2012. Lecture Notes in Computer Science, vol. 7374, pp. 188–202 (Jul 2012). https://doi.org/10.1007/978-3-642-31410-0_12
29. Wolf, C., Preneel, B.: Taxonomy of Public Key Schemes based on the problem of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic equations. Cryptology ePrint Archive, Report 2005/077 (Mar 2005), `https://eprint.iacr.org/2005/077`
30. Yasuda, T., Ding, J., Takagi, T., Sakurai, K.: A Variant of Rainbow with Shorter Secret Key and Faster Signature Generation. In: Chen, K., Xie, Q., Qiu, W., Xu, S., Zhao, Y. (eds.) ACM Workshop on Asia Public-Key Cryptography. pp. 57–62 (May 2013). https://doi.org/10.1145/2484389.2484401
31. Yasuda, T., Sakurai, K., Takagi, T.: Reducing the Key Size of Rainbow Using Non-commutative Rings. In: Dunkelman, O. (ed.) Topics in Cryptology – CT-RSA 2012. Lecture Notes in Computer Science, vol. 7178, pp. 68–83 (Feb 2012). https://doi.org/10.1007/978-3-642-27954-6_5
32. Yasuda, T., Takagi, T., Sakurai, K.: Efficient variant of Rainbow using sparse secret keys. Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications **5**(3), 3–13 (Sep 2014). https://doi.org/10.22667/JOWUA.2014.09.31.003
33. Yasuda, T., Takagi, T., Sakurai, K.: Efficient Variant of Rainbow without Triangular Matrix Representation. In: Mahendra, M.S., Neuhold, E.J., Tjoa, M.A., You, I. (eds.) Information and Communication Technology. Lecture Notes in Computer Science, vol. 8407, pp. 532–541 (Apr 2014). https://doi.org/10.1007/978-3-642-55032-4_55
34. Yi, H., Tang, S.: Very Small FPGA Processor for Multivariate Signatures. The Computer Journal **59**(7), 1091–1101 (Jul 2016). https://doi.org/10.1093/comjnl/bxw008

REFERENCES

BETTALE, L.; FAUGERE, J.-C.; PERRET, L. Hybrid approach for solving multivariate systems over finite fields. **Journal of Mathematical Cryptology**, Walter de Gruyter GmbH & Co. KG, v. 3, n. 3, p. 177–197, 2009.

BILLET, O.; GILBERT, H. Cryptanalysis of Rainbow. In: SPRINGER. **International Conference on Security and Cryptography for Networks**. [S.l.], 2006. p. 336–347.

CHEN, M.-S. et al. From 5-Pass $\mathcal{MQ}$-Based Identification to $\mathcal{MQ}$-Based Signatures. In: SPRINGER. **International Conference on the Theory and Application of Cryptology and Information Security**. [S.l.], 2016. p. 135–165.

DIFFIE, W.; HELLMAN, M. New Directions in Cryptography. **IEEE transactions on Information Theory**, IEEE, v. 22, n. 6, p. 644–654, 1976.

DING, J. et al. **Rainbow - Algorithm Specification and Documentation**. 2017. Round 1 Submission, NIST Post-Quantum Cryptography Standardisation Process.

DING, J. et al. **Rainbow - Algorithm Specification and Documentation**. 2019. Round 2 Submission, NIST Post-Quantum Cryptography Standardisation Process.

DING, J.; SCHMIDT, D. Rainbow, a New Multivariable Polynomial Signature Scheme. In: SPRINGER. **International Conference on Applied Cryptography and Network Security**. [S.l.], 2005. p. 164–175.

DING, J. et al. New Differential-Algebraic Attacks and Reparametrization of Rainbow. In: SPRINGER. **International Conference on Applied Cryptography and Network Security**. [S.l.], 2008. p. 242–257.

GAREY, M. R.; JOHNSON, D. S. **Computers and Intractability; A Guide to the Theory of NP-Completeness**. New York, NY, USA: W. H. Freeman & Co., 1979. ISBN 0716710455.

HASHIMOTO, Y. **On the security of Circulant UOV/Rainbow**. [S.l.], 2018.

KIPNIS, A.; PATARIN, J.; GOUBIN, L. Unbalanced Oil and Vinegar Signature Schemes. In: SPRINGER. **International Conference on the Theory and Applications of Cryptographic Techniques**. [S.l.], 1999. p. 206–222.

KIPNIS, A.; SHAMIR, A. Cryptanalysis of the Oil and Vinegar Signature Scheme. In: SPRINGER. **Annual International Cryptology Conference**. [S.l.], 1998. p. 257–266.

LIDL, R.; NIEDERREITER, H. **Encyclopedia of mathematics and its applications vol 20**. [S.l.]: Cambridge, Cambridge university press, 1983.

PATARIN, J. **The Oil and Vinegar Algorithm for Signatures**. 1997. Dagstuhl Seminar 9739. Disponível em: <https://www.dagstuhl.de/9739>.

PENG, Z.; TANG, S. Circulant Rainbow: A New Rainbow Variant With Shorter Private Key and Faster Signature Generation. **IEEE Access**, IEEE, v. 5, p. 11877–11886, 2017.

PETZOLDT, A. **Selecting and Reducing Key Sizes for Multivariate Cryptography**. Tese (Doutorado) — tuprints, 2013.

PETZOLDT, A.; BULYGIN, S.; BUCHMANN, J. CyclicRainbow–A Multivariate Signature Scheme with a Partially Cyclic Public Key. In: SPRINGER. **International Conference on Cryptology in India**. [S.l.], 2010. p. 33–48.

PETZOLDT, A. et al. Small Public Keys and Fast Verification for Multivariate Quadratic Public Key Systems. In: SPRINGER. **International Workshop on Cryptographic Hardware and Embedded Systems**. [S.l.], 2011. p. 475–490.

RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. **On Digital Signatures and Public-Key Cryptosystems.** [S.l.], 1977.

SHOR, P. W. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. **SIAM review**, SIAM, v. 41, n. 2, p. 303–332, 1999.

STINSON, D. R. **Cryptography: Theory and Practice**. Third edition. [S.l.]: Chapman and Hall/CRC, 2006.

THOMAE, E. Quo Vadis Quaternion? Cryptanalysis of Rainbow over Non-Commutative Rings. In: SPRINGER. **International Conference on Security and Cryptography for Networks**. [S.l.], 2012. p. 361–373.

WOLF, C.; PRENEEL, B. Equivalent Keys in HFE, C*, and variations. In: SPRINGER. **International Conference on Cryptology in Malaysia**. [S.l.], 2005. p. 33–49.

WOLF, C.; PRENEEL, B. Equivalent Keys in Multivariate Quadratic Public Key Systems. **Journal of Mathematical Cryptology**, Walter de Gruyter GmbH & Co. KG, v. 4, n. 4, p. 375–415, 2011.

YASUDA, T.; SAKURAI, K.; TAKAGI, T. Reducing the Key Size of Rainbow Using Non-commutative Rings. In: SPRINGER. **Cryptographers' Track at the RSA Conference**. [S.l.], 2012. p. 68–83.

ZAMBONIN, G.; BITTENCOURT, M. S. P.; CUSTÓDIO, R. Handling Vinegar Variables to Shorten Rainbow Key Pairs. In: SPRINGER. **International Conference on Cryptology in Africa**. [S.l.], 2019. p. 391–408.