EFFICIENT MULTIVARIATE-BASED RING SIGNATURE SCHEMES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF APPLIED MATHEMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MURAT DEMİRCİOĞLU

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
CRYPTOGRAPHY

AUGUST 2022

Approval of the thesis:

EFFICIENT MULTIVARIATE-BASED RING SIGNATURE SCHEMES

submitted by **MURAT DEMİRCİOĞLU** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Cryptography Department, Middle East Technical University** by,

Prof. Dr. A. Sevtap Selçuk Kestel _____
Dean, Graduate School of **Applied Mathematics**

Prof. Dr. Ferruh Özbudak _____
Head of Department, **Cryptography**

Prof. Dr. Murat Cenk _____
Supervisor, **Cryptography, METU**

Assoc. Prof. Dr. Sedat Akleylek _____
Co-supervisor, **Computer Engineering Dept., Ondokuz Mayıs Uni.**

Examining Committee Members:

Prof. Dr. Ersan Akyıldız
Mathematics, METU _____

Prof. Dr. Murat Cenk
Cryptography, METU _____

Prof. Dr. Zülfükar Saygı
Mathematics, METU _____

Assoc. Prof. Dr. Ali Doğanaksoy
Mathematics, TOBB ETU _____

Assist. Prof. Dr. Eda Tekin
Mathematics, Karabük Uni. _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: MURAT DEMİRCİOĞLU

Signature :

ABSTRACT

EFFICIENT MULTIVARIATE-BASED RING SIGNATURE SCHEMES

Demircioğlu, Murat
Ph.D., Department of Cryptography
Supervisor : Prof. Dr. Murat Cenk
Co-Supervisor : Assoc. Prof. Dr. Sedat Akleylek

August 2022, 53 pages

The ring signature scheme has a wide range of usage areas in public-key cryptography. One is leaking information within a group without exposing the signer's identity. The majority of the ring signature techniques in use, on the other hand, rely on classical crypto-systems such as RSA and ECDH, which are known to be vulnerable to Shor's algorithm on a large-scale quantum computer. In this thesis, we propose efficient quantum-resistant ring signature schemes based on GeMSS and Gui signature algorithms. Gui was a candidate in Round 1, and GeMSS was one of two multivariate-based signature algorithms along with Rainbow in Round 3 of the Post-Quantum Cryptography Standardization Project initiated by NIST in 2016. When we compare our proposed scheme with a Rainbow-based ring signature scheme, the experimental results show that we achieve 300 times faster signature verification and almost 50 times faster signature generation as the number of users in the group increases to 50. Moreover, the proposed scheme provides at least 20% smaller signature sizes. Therefore, our scheme is verified to be more effective to be used.

Keywords: cryptography, post-quantum, multivariate, GeMSS, Gui, ring signature

ÖZ

ÇOK DEĞİŞKENLİ TABANLI ETKİN HALKA İMZA ŞEMALARI

Demircioğlu, Murat
Doktora, Kriptografi Bölümü
Tez Yöneticisi : Prof. Dr. Murat Cenk
Ortak Tez Yöneticisi : Doç. Dr. Sedat Akleylek

Ağustos 2022, 53 sayfa

Halka imza şeması, açık anahtarlı kriptografide geniş bir kullanım alanına sahiptir. İmzalayanın kimliğini ifşa etmeden bir grup içinde bilgi sızdırma senaryosu bunlar içerisinden bir örnek olarak verilebilir. Öte yandan, kullanılan halka imza tekniklerinin çoğu, büyük ölçekli bir kuantum bilgisayarda Shor algoritmasına karşı savunmasız olduğu bilinen RSA ve ECDH gibi klasik kripto sistemlerine dayanmaktadır. Bu tezde, çok değişkenli imzalama algoritmaları olan GeMSS ve Gui algoritmalarına dayalı verimli ve kuantum dirençli halka imza şemaları önermekteyiz. GeMSS ve Gui, 2016 yılında NIST tarafından başlatılan Kuantum Sonrası Kriptografi Standardizasyon Projesi'nde yer almıştır. Projenin 1. turu sonrasında elenen Gui algoritmasının ardından 3. turuna GeMSS ile diğer çok değişken tabanlı imza algoritması Rainbow devam etmiştir. Önerilen GeMSS tabanlı halka imza şemamızı Rainbow tabanlı başka bir halka imza şemasıyla karşılaştırdığımızda, deneysel sonuçlar gösteriyor ki; gruptaki kullanıcı sayısı 50'ye yükseldikçe 300 kat daha hızlı imza doğrulama ve neredeyse 50 kat daha hızlı imza oluşturma süreleri elde etmekteyiz. Ayrıca, önerilen şema en az %20 daha küçük imza boyutu sağlamaktadır. Bu sayede, önerdiğimiz şemanın kullanılmak üzere daha etkili olduğu doğrulanmıştır.

Anahtar Kelimeler: kriptografi, kuantum sonrası, çok değişkenli, GeMSS, Gui, halka imza

To my lovely wife and sons
Kibariye, Ali Kemal and Erdem

ACKNOWLEDGMENTS

First of all, I would like to express my sincere gratitudes to my thesis supervisor Murat CENK, and co-supervisor Sedat AKLEYLEK for their patient support and guidance during my Ph.D. studies. They were always there for me when my motivation was low.

Besides my advisors, I would like to thank my thesis committee members Ersan AKYILDIZ, Zülfükar SAYGI, Ali DOĞANAKSOY and Eda TEKİN for their review and suggestions for my thesis.

I would like to thank for his valuable friendship to Halil Kemal TAŞKIN, with whom I have worked with many years throughout my academic and professional life.

I am also grateful to all my friends from Cryptography, especially, Ahmet SINAK and Fuad HAMİDLİ for their closed friendship.

I greatly thank my father and my mother for their support and encouragement to take the best education during my life.

Lastly, I greatly appreciate my wife, Kibariye DEMİRCİOĞLU, for her endless support during my long journey on Ph.D. studies. Without her support, this thesis would not be possible. Moreover, I would like to thank my sons, Ali Kemal DEMİRCİOĞLU and Erdem DEMİRCİOĞLU for their endless love and being the light of my life.

TABLE OF CONTENTS

ABSTRACT vii

ÖZ ix

ACKNOWLEDGMENTS xi

TABLE OF CONTENTS xiii

LIST OF TABLES xv

LIST OF FIGURES xvii

LIST OF ABBREVIATIONS xviii

CHAPTERS

1 INTRODUCTION 1

2 PRELIMINARIES 5

2.1 Public Key Cryptography 5

2.2 Post-Quantum Cryptography 6

2.3 Multivariate Public Key Cryptography 8

2.4 Ring Signature Schemes 10

2.5 Multivariate-Based Ring Signature Schemes 12

3 GUI BASED RING SIGNATURE SCHEME 15

3.1 Preliminaries 15

3.2 Protocol Description 17

3.3 Performance Analysis 19

4 GEMSS BASED RING SIGNATURE SCHEME 23

4.1 Preliminaries 23

4.2 Protocol Description 28

4.3 Security Analysis 30

4.4 Performance Analysis 33

4.5 Comparison 40

5 CONCLUSION 43

REFERENCES 45

CURRICULUM VITAE 51

LIST OF TABLES

Table 2.1 First set of quantum-resistant algorithms chosen by NIST to be standardized 7

Table 2.2 The 4^{th} round candidates in NIST Post-Quantum standardization project 8

Table 3.1 Parameter sets of Gui for the Security Levels 1, 3, and 5, respectively. 19

Table 3.2 Performance of Gui signature algorithm under different parameter sets 20

Table 3.3 Theoretical performance analysis of Gui-based ring signature scheme in three different security levels 21

Table 3.4 Signature verification time and signature size comparison of Gui-based and Rainbow-based ring signature schemes 21

Table 4.1 Parameter sets for GeMSS for the Security Levels 1, 3, and 5, respectively. 34

Table 4.2 Performance of GeMSS signature algorithm under different parameter sets 35

Table 4.3 Theoretical results of GeMSS-based ring signature schemes for all GeMSS parameter sets in Security Level 1 36

Table 4.4 Theoretical results of GeMSS-based ring signature schemes for all GeMSS parameter sets in Security Level 3 37

Table 4.5 Theoretical results of GeMSS-based ring signature schemes for all GeMSS parameter sets in Security Level 5 38

Table 4.6 Implementation results of our RedGeMSS based ring signature scheme for Security Level 1 parameter set 39

Table 4.7 Comparison of 3 multivariate signature algorithms in the same security level 40

Table 4.8 Comparison of experimental results of our proposed schemes with Rainbow-based scheme for Security Level 1 41

Table 4.9 Comparison of 3 multivariate signature algorithms after the offered solution of Rainbow team against security flaw 41

LIST OF FIGURES

Figure 2.1 Signature generation and verification in Multivariate Public Key Systems 10

Figure 3.1 Signature generation and verification in Gui signature algorithm . . 17

LIST OF ABBREVIATIONS

DSA	Digital Signature Algorithm
ECC	Elliptic Curve Cryptography
KEM	Key Encapsulation Mechanism
MPKC	Multivariate Public Key Cryptosystem
NIST	The National Institute of Standards and Technology
PGP	Pretty Good Privacy
PQC	Post Quantum Cryptography
RSA	Rivest–Shamir–Adleman
SSH	Secure Socket Shell
TLS	Transport Layer Security
UOV	Unbalanced Oil and Vinegar

CHAPTER 1

INTRODUCTION

Ring signature schemes are practical when it comes to privacy. It can be used for leaking secrets, electronic voting, and electronic money, among many other things. For instance, in order to use the government's black budget, one of the authorized managers may need to sign the payment request anonymously. As a result, anyone can check to see if a member of this group signed the request by using the public keys of the authorized users to build a ring signature.

The idea of the ring signature was firstly introduced by Rivest et al. [37] in 2001, and they used the RSA algorithm [36] as the underlying signature algorithm. Since then, many other ring signature techniques based on standard asymmetric algorithms have been proposed. One example is the identity-based proxy ring signature scheme from RSA proposed by Asaar et al. [1]. Later, the design and security aspects of ring signature techniques were examined by Bender et al. [3]. Ring signatures, unlike group signatures [10], have no central authority, setup, or revocation procedures. Anyone can choose a group of possible signers, including himself, and create a ring signature anonymously by using his secret and other users' public keys without their knowledge. Nobody, including the group members, can identify the signer where all possible signers are associated with a public key.

Ring signature schemes, which are based on the standard asymmetric cryptosystems such as RSA, DSA [26], and ECC, will be subjected to a critical security risk as large-scale quantum computers are built. Two most famous quantum algorithms that may cause a problem in classical cryptosystems are; *Shor's algorithm* [39] that solves integer factorization problem on quantum computers in polynomial time, and *Grover's*

algorithm [24] which is a quantum search algorithm for searching an unsorted data set with n entries in $\mathcal{O}(\sqrt{n})$ time and using $\mathcal{O}(n \log n)$ storage. Therefore, there arises a need for alternative cryptosystems that will work on classical computers securely against quantum computer-related attacks. This new study area is called Post-Quantum Cryptography [4]. In order to choose and standardize one or more quantum-resistant public-key cryptosystems, NIST initiated a process to evaluate candidate algorithms in 2016. Five alternative approaches; lattice-based, hash-based, code-based, multivariate-based, and supersingular elliptic curve isogeny, are the main focus of the researchers.

Among these approaches, multivariate cryptosystems [16] are extremely fast and just demand a small amount of processing power. There exists various multivariate-based signature algorithms; such as GeMSS [8], Gui [35], Rainbow [17], and UOV [25]. The MQ Problem of solving systems of multivariate quadratic polynomials over a finite field provides the main security of multivariate cryptosystems. Due to their huge key sizes with respect to other Post-Quantum approaches, multivariate cryptosystems are not suitable as general-purpose signature algorithms to be used in daily operations. On the other hand, there are applications that do not need to send keys very often. Moreover, there are not many quantum-resistant signature schemes with unique features like ring signatures. As the threat of quantum computers increases, many ring signature schemes based on multivariate cryptography have been proposed [29, 34, 41, 40, 43]. One is a Rainbow-based approach proposed by Mohamed et al. [29].

When we compare Rainbow and Gui signature algorithms, both Round 1 candidates in NIST's project, we see that Gui offers a smaller signature size with a faster verification time. Starting from this point of view, we proposed a new Gui-based ring signature scheme [12]. However, the Gui signature algorithm was eliminated when the candidates for Round 2 of NIST's project were announced. At this point, we decided to use the GeMSS signature algorithm, which was one of two multivariate signature algorithms in Round 3. The other one is the Rainbow algorithm which has a critical security flaw recently published by Ward Beullens [5]. He proposed a practical key-recovery attack for Level 1 parameter set of Rainbow. After this, the Rainbow team proposed NIST to replace the Level 1 parameters with Level 3 and

Level 3 parameters with Level 5. This solution may enhance Rainbow's security at the cost of an increase in the key sizes and computation times.

As we showed in [13], the initial theoretical results for our new GeMSS-based ring signature scheme are better than both the Rainbow-based ring signature algorithm [29] and our previous Gui-based proposal [12]. In addition, the implementation results show that our proposed method provides significantly faster signature generation and verification time with respect to the Rainbow-based scheme as the number of group members increases. Our scheme will provide up to 300 times faster verification and 50 times faster signature generation if the group consists of 50 members.

This thesis is organized as follows. In Chapter 2, we summarize the concept of public key cryptography and post-quantum cryptography. The definitions for ring signature schemes and multivariate cryptosystems are also given in this section. We introduce our Gui-based ring signature scheme and the expected theoretical efficiency results in Chapter 3. After that, we present our GeMSS-based ring signature technique in Chapter 4 along with security proofs. In this section, the proposed scheme's public key, signature sizes, and computation time are compared with others. We conclude the thesis in Section 5.

CHAPTER 2

PRELIMINARIES

In this chapter, we present the necessary background and definitions used in this thesis. This chapter mainly consists of five sections: Public Key Cryptography, Post-Quantum Cryptography, Multivariate Public Key Cryptography, Ring Signature Schemes, and Multivariate-based Ring Signature Schemes.

2.1 Public Key Cryptography

In information security, public-key cryptography plays an essential role in confidentiality, authenticity, and non-reputability for digital communication and data storage. We can see it in many standard protocols, such as TLS, SSH, and PGP.

Unlike symmetric-key cryptography, public-key cryptography is a cryptographic system that uses pair of keys, which are a public key and a private key. It requires keeping the private key secret and distributing the public key without compromising security. Public-key cryptographic algorithms are based on a **trapdoor one-way function**, which is a function that is easy to compute for any input but hard to invert for a given output unless some secret information is known. One of the most famous trapdoor functions is the integer factorization problem which is used in RSA algorithm [36].

The idea of public-key cryptography emerged to solve two of the most difficult problems associated with symmetric-key cryptography; key distribution and digital signatures.

1. Key distribution under symmetric-key encryption requires either the communicating parties already have a shared key or use a key distribution center. Since public-key cryptography relies on a public key for encryption and a secret key for decryption, anyone can encrypt a message or a session key by using the receiver's public key, and the encrypted data can only be decrypted by the receiver's private key. In the case of sharing a symmetric encryption key by using a public-key cryptosystem, two parties can safely continue to communicate over a secure channel by using the symmetric key encryption, which is faster with respect to public-key cryptography.

2. With the use of public-key cryptography, a sender can create a short signature for the message by using his secret key. Anyone can easily verify if the message belongs to the sender by using the sender's public key.

While Diffie–Hellman key exchange [14] provides key distribution, and Digital Signature Algorithm [26] provides digital signature, RSA algorithm [36] provides both. However, public-key cryptography is too slow with respect to symmetric cryptography. At this point, modern cryptosystems choose to use a hybrid model in which both public-key cryptography and symmetric cryptography are used together; by using the public-key algorithm to share a session key securely and then using it for symmetric encryption.

2.2 Post-Quantum Cryptography

The development of quantum computers has advanced steadily over the past few years. If large-scale quantum computers are built, several widely used standard asymmetric cryptosystems will be in danger of losing their security. The discrete logarithm problem, the integer factorization problem, and the elliptic-curve discrete logarithm problem are the three hard mathematical problems that are especially in the scope of this inevitable danger of quantum computing. Shor's algorithm [39] can easily solve all of these problems in polynomial time on a powerful enough quantum computer. Hash functions and other symmetric cryptographic primitives, on the other hand, would not be as significantly affected. Although Grover's search algorithm [24]

speeds up attacks on symmetric ciphers, these attacks may be successfully prevented by increasing the key size. As a result, the current state of symmetric cryptography can continue to be used in post-quantum symmetric cryptography.

It is not known when a large-scale quantum computer will be built. Eventually, it will happen, and at that time, information security systems have to be ready against quantum attacks. Finding new public-key cryptosystems that are compatible with classical computer architectures and secure against attacks related to quantum computing has become a new focus of research area as a result. This important new domain of research is known as post-quantum cryptography (PQC) [4]. Current post-quantum researchers mostly focused on lattice-based cryptography, multivariate cryptography, hash-based cryptography, code-based cryptography, and supersingular elliptic curve isogeny cryptography.

The NIST Post-Quantum Cryptography Standardization Process

In December 2016, the National Institute of Standards and Technology (NIST) launched a public project to choose asymmetric cryptography algorithms that are resistant to quantum computing. As a result of this project, some of the candidate post-quantum algorithms have been chosen as the new standard post-quantum algorithms after three rounds of examination and analysis in July 2022. The selected algorithms to be standardized are listed in Table 2.1. Both CRYSTALS-KYBER, CRYSTALS-Dilithium, and FALCON are lattice-based algorithms. Besides these, SPHICNS$^+$ is a hash-based algorithm.

Table 2.1: First set of quantum-resistant algorithms chosen by NIST to be standardized

Public-Key Encryption/KEMs	**Digital Signatures**
	CRYSTALS–Dilithium
CRYSTALS–KYBER	FALCON
	SPHINCS$^+$

The evaluation process continues with new candidates in Round 4. These are listed in Table 2.2. Among these candidates, SIKE is a supersingular isogeny-based algorithm, and the others are code-based algorithms.

NIST also announced that they are planning to launch a new Call for Proposals for

Table 2.2: The 4^{th} round candidates in NIST Post-Quantum standardization project

Public-Key Encryption/KEMs	Digital Signatures
BIKE	
Classic McEliece	
HQC	
SIKE	

quantum-resistant public-key digital signature algorithms by the end of the summer of 2022. According to NIST, their main goal is to diversify their signature portfolio; hence they are most interested in signature schemes that are not based on structured lattices. NIST is looking for contributions of signature algorithms with fast verification time and short signature size. At this point, multivariate algorithms will have an advantage as they provide the required short signatures and fast verification times.

2.3 Multivariate Public Key Cryptography

Let $\mathbb{F}$ be a finite field. The main idea of multivariate cryptography is to choose the central map $\mathcal{F} : \mathbb{F}^n \to \mathbb{F}^m$, which is a multivariate system of easily invertible quadratic polynomials. After the choice of $\mathcal{F}$, two affine linear invertible maps $\mathcal{S} : \mathbb{F}^m \to \mathbb{F}^m$ and $\mathcal{T} : \mathbb{F}^n \to \mathbb{F}^n$ are chosen to hide the structure of the central map. Therefore, public-key is $\mathcal{P} = \mathcal{S} \circ \mathcal{F} \circ \mathcal{T} : \mathbb{F}^n \to \mathbb{F}^m$, and private key is $(\mathcal{S}, \mathcal{F}, \mathcal{T})$.

Multivariate quadratic equations over finite fields are the basic components of multivariate cryptography.

$$\begin{aligned} f^{(1)}(x_1, \ldots, x_n) &= \sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij}^{(1)} \cdot x_i x_j + \sum_{i=1}^{n} f_i^{(1)} \cdot x_i + f_0^{(1)} \\ f^{(2)}(x_1, \ldots, x_n) &= \sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij}^{(2)} \cdot x_i x_j + \sum_{i=1}^{n} f_i^{(2)} \cdot x_i + f_0^{(2)} \\ &\vdots \\ f^{(m)}(x_1, \ldots, x_n) &= \sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij}^{(m)} \cdot x_i x_j + \sum_{i=1}^{n} f_i^{(m)} \cdot x_i + f_0^{(m)} \end{aligned} \tag{2.1}$$

The security of multivariate cryptosystems is mainly based on the difficulty of two following problems; multivariate quadratic polynomial problem (*MQ*), and MinRank problem.

Problem 1 (MQ Problem): For a given finite field $\mathbb{F}$ and set of m multivariate quadratic polynomials $f^{(1)}, \ldots, f^{(m)}$ in n variables $(x_1, \ldots, x_n)$ as stated in Equation (2.1), *MQ* problem is to determine if there exists a solution $\overline{x} \in \mathbb{F}^n$ such that $f^{(1)}(\overline{x}) = \cdots = f^{(m)}(\overline{x}) = 0$. This decisional problem was shown to be NP-hard in all fields in [33]. It is proven to be NP-hard (for $m \approx n$) even for quadratic polynomials over $\mathbb{F}_2$ [22].

Problem 2 (MinRank Problem): Let m, n, r, k be positive integers and let $\mathbb{F}$ be a finite field. With a given finite field $\mathbb{F}$, k matrices M_i of size $m \times n$ with entries in $\mathbb{F}$, and a rank bound r, the objective of this decisional problem is to check if there are values of $\alpha_i \in \mathbb{F}$ that satisfy the following equation:

$$rank(\sum_{i=1}^{k} \alpha_i M_i) \leq r. \tag{2.2}$$

According to [7], the MinRank problem is an NP-hard problem.

Gröbner basis algorithms like F4/F5 (see [19, 20]) and linearization algorithms like XL (see [11]) are two of the most efficient generic known attacks on the MQ problem. The efficiency of the MinRank attacks may differ depending on the size and quantity of matrices as well as the target rank. Two primary techniques are the support minors approach (see [2]) and the combinatorial search approach, which was presented in [23].

A generic multivariate signature algorithm consists of key generation, signature generation, and signature verification functions which can be summarized as follows:

- **Key Generation:** This functions uses a security parameter λ to generate a key pair $(sk, pk) = ((\mathcal{S}, \mathcal{F}, \mathcal{T}), \mathcal{P})$.
- **Signature Generation:** In order to sign a message M, the signer uses a hash function $\mathcal{H} : \{0,1\}^* \rightarrow \mathbb{F}^m$ to compute $h = \mathcal{H}(M) \in \mathbb{F}^m$. Then he calculates recursively $x = \mathcal{S}^{-1}(h) \in \mathbb{F}^m$, $y = \mathcal{F}^{-1}(x) \in \mathbb{F}^n$ and $z = \mathcal{S}^{-1}(y) \in \mathbb{F}^n$. At the end, the signature of the message M is $\sigma = \mathcal{T}^{-1}(\mathcal{F}^{-1}(\mathcal{S}^{-1}(h)))$.
- **Signature Verification:** In order to check if the signature σ is valid for the message M, the verifier computes $h = \mathcal{H}(M)$ and $h' = \mathcal{P}(\sigma)$. If the results are the same, then the signature is valid; otherwise not.

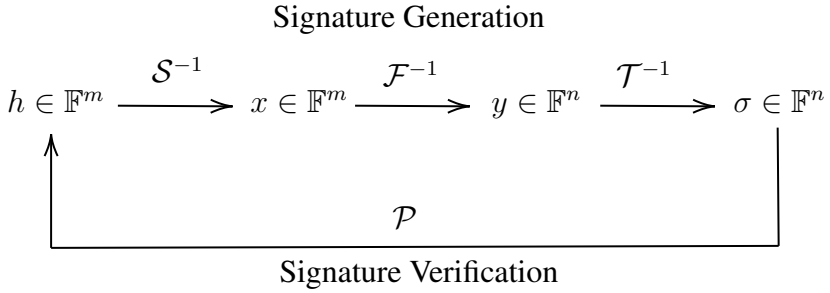

Figure 2.1: Signature generation and verification in Multivariate Public Key Systems

Due to their large public key size and slow signature generation time, multivariate signature algorithms are not suitable for daily use. On the other hand, the fast verification times and small signature sizes make them better candidates for signature schemes with special properties, such as ring signature, group signature, and threshold signature schemes.

2.4 Ring Signature Schemes

In a group $\mathcal{R} = \{u_1, \ldots, u_t\}$ consisting of t-many possible signers, a user $u_i \in \mathcal{R}$ can create a ring signature of a message anonymously on behalf of the group $\mathcal{R}$. Anyone, who wants to verify the authenticity of the message, can easily validate if the signature of the message is created by a user from this group. Nobody, not even the members of the belonging group, is able to disclose the true identity of the actual user who forged the ring signature.

A standard ring signature scheme requires following operations for key generation, ring signature generation, and ring signature verification operations:

- `KeyGen(`1^λ`)` is a probabilistic algorithm that generates a public and private key pair (sk, pk) by taking a security parameter λ as an input. Each user $u_i \in \mathcal{R}$ generates their key pairs and announces only the public keys.
- `RingSign(`$M,\ \ sk_i,\ \ \{pk_1, \ldots, pk_t\}$`)` is a probabilistic algorithm in which user $u_i \in \mathcal{R}$ signs the message M on behalf of the group $\mathcal{R}$. The output of this operation is the ring signature σ.
- `RingVerify(`$(M, \sigma), \{pk_1, \ldots, pk_t\}$`)` is a deterministic algorithm that will

only return true if the ring signature is valid.

A ring signature scheme must be complete, and it has to achieve two essential security requirements, which are anonymity and unforgeability. The definitions are given below, and their proofs are stated in Section 4.3.

Definition 1 [Completeness]: A ring signature is assumed to be complete if the following equation

$$RingVerify((M, RingSign(M, sk_i, \{pk_1, \ldots, pk_t\})), \{pk_1, \ldots, pk_t\}) = TRUE \quad (2.3)$$

holds for all $i \in \{1, \ldots, t\}$. This ensures that a ring signature of a message created by any member of the group will always be verified.

Definition 2 [Anonymity]: The signer of the message should remain anonymous, and the verifier should not have any clue about the identity of the signer. For a clear understanding, we can define an anonymity game consisting of the following steps:

1. Each user uses *KeyGen* algorithm to generate their own key pairs (sk_i, pk_i), where $i \in \{1, \ldots, t\}$. All of the public keys $\{pk_1, \ldots, pk_t\}$ will be shared with the adversary $\mathcal{A}$.
2. A signing oracle $\mathcal{OS}(i, M)$ is available for the adversary $\mathcal{A}$. Index $i \in \{1, \ldots, t\}$ stands for the group member id, and M is the message that will be signed. The oracle uses the private key sk_i of the user u_i to calculate a legitimate ring signature σ of the given message in the name of the target group $\mathcal{R} = \{u_1, \ldots, u_t\}$.
3. A message M^* and two different indices $i_0, i_1 \in \{1, \ldots, t\}$ are chosen by the adversary $\mathcal{A}$. A signature $\sigma = RingSign(M^*, sk_{i_b}, \{pk_1, \ldots, pk_t\})$ will be computed and shared with the adversary $\mathcal{A}$ where the value $b \in \{0, 1\}$ is chosen randomly.
4. The adversary $\mathcal{A}$ chooses $b^* \in \{0, 1\}$. He wins the game if $b = b^*$ holds.

The ring signature scheme provides anonymity if the probability of being successful in guessing b^* is $1/2$. In other words, $(1 - 2 \cdot Pr[b = b^*])$ is negligible for every PPT adversary $\mathcal{A}$.

Definition 3 [Unforgeability]: An adversary, who is not a member of the group $\mathcal{R}$, cannot create a legitimate ring signature in the name of the group $\mathcal{R}$. An unforgeability game consisting of the following steps will be used to demonstrate this security property:

1. Each user uses *KeyGen* algorithm to generate their own key pairs (sk_i, pk_i), where $i \in \{1, \ldots, t\}$. Public keys $\{pk_1, \ldots, pk_t\}$ will be shared with the adversary $\mathcal{A}$.

2. There is a signing oracle $\mathcal{OS}(M)$ which takes a message M as input. On behalf of the target group $\mathcal{R} = \{u_1, \ldots, u_t\}$, it outputs a valid ring signature σ of the provided message. This signing oracle is accessible to the adversary $\mathcal{A}$.

3. A challenge message M^* is given to $\mathcal{A}$. He wins the game if he can create a valid ring signature σ^* for the provided message in the name of the group $\mathcal{R}$. At this step, the adversary is not permitted to use the signing oracle for the given message M^*.

The probability of success $Pr_{\mathcal{A}}[success]$ can be described as

$$Pr[\textit{RingVerify}((M^*, \sigma^*), \{pk_1, \ldots, pk_t\}) = TRUE].$$

If it is negligible for every PPT adversary $\mathcal{A}$, then the ring signature scheme provides the unforgeability criteria.

2.5 Multivariate-Based Ring Signature Schemes

This section provides brief information about previous ring signature schemes based on multivariate signature algorithms.

In 2011, Wang et al. presented a general multivariate ring signature system based on multivariate public-key cryptosystems [41]. In their security analysis, the proofs of completeness and anonymity were given. Additionally, they claimed that the underlying multivariate signature algorithm had to be secure against known attacks on MPKCs, such as algebraic attacks, and rank attacks, in order for their proposed ring signature scheme to be secure against these kind of attacks.

Threshold ring signatures were introduced in 2002 by Bresson et al. in [6]. In a $(t, \mathcal{R})$-threshold ring signature, the verifier will be convinced that t users from the group $\mathcal{R}$ have signed the message without revealing the members of this subgroup. In 2012, Petzoldt et al. suggested a multivariate-based threshold ring signature scheme [34]. They extended the identification scheme of [38] to a threshold ring identification scheme. After that, they transformed the Fiat-Shamir heuristic (see [21]) into a threshold ring signature scheme. Its shorter signature is the main advantage of this scheme with respect to code-based and lattice-based alternatives. They used the same parameters in [38] to evaluate the performance of their threshold ring signature scheme. When they compared the results with the alternative Post-Quantum threshold ring signature schemes (see [28, 9]), their scheme offered smaller size of signatures with respect to the code-based and the lattice-based schemes with a cost of extra rounds.

In 2013, Wang presented a ring signature scheme based on MPKC by transforming the identification scheme of [38] and gave its security analysis in [40]. This approach, however, is declared unsafe in [29] because it only uses one round of the identification process, giving an attacker a $2/3$ chance to forge a valid signature.

In [43], Zhang and Zhao presented a Γ-protocol based on the MQ problem and converted it to a threshold ring signature scheme by using the Γ-transformation [42]. In comparison to the Fiat-Shamir heuristic, Γ-transformation preserves all of its benefits, has a stronger level of provable security, and solves some significant drawbacks, including rigid implementation in interactive protocols and public/private storage restrictions. When they compared their scheme with [34], they achieved 29% reduced rounds and 21% smaller signatures for 80-bit security, 33% reduced rounds and 25% smaller signatures for 100-bit security.

In [29], Mohamed and Petzoldt presented a multivariate-based ring signature scheme in 2017. The underlying algorithm in their proposal was the Rainbow signature algorithm [17]. Their proposed scheme requires $k-1$ evaluations of signature verification operation and 1 evaluation of signature generation operation. Although Rainbow was one of two multivariate signature candidates along with GeMSS in Round 3 of NIST Post-Quantum Standardization Project, a new practical key recovery attack was intro-

duced by Beullens [5]. For the Security Level 1 parameters of Rainbow, this attack returns the corresponding secret key after 53 hours of computation time on a standard computer. In order to enhance security, the Rainbow team proposes to use Level 3 parameters instead of Level 1 parameters and Level 5 parameters instead of Level 3. As a result, the Rainbow-based ring signature scheme is not providing the defined security anymore as given in [29].

Demircioglu et al. proposed a Gui-based ring signature in 2018 (see [12]), and a GeMSS-based ring signature scheme in 2020 (see [13]). These new schemes provide smaller signature sizes and faster verification times with respect to the Rainbow-based ring signature scheme in [29]. The details are given in Chapter 3 and Chapter 4.

CHAPTER 3

GUI BASED RING SIGNATURE SCHEME

In this chapter, we propose a ring signature scheme based on the Gui signature algorithm [35], which was one of the Round 1 candidates in the NIST Post-Quantum Standardization Project. In Section 3.1, we give the necessary background about Gui and then show how to use this signature algorithm in our proposed ring signature scheme in Section 3.2. At the end of this chapter, we give theoretical results for different security levels of Gui.

This research was presented in CECC'18 conference (*Central European Conference on Cryptology 2018*) [12].

3.1 Preliminaries

The HFEv-signature system, which was presented by Patarin [32], is the basis for the Gui signature algorithm. Gui signature increases the number of minus equations and vinegar variables while using less HFE polynomial, which speeds up the signature generation process without compromising security. The short signatures, low processing requirements, and efficiency of the Gui signature algorithm are its key benefits. Furthermore, Gui's private key is considerably smaller, making it possible to be stored on compact data storage devices like smart cards. On the other side, Gui's biggest drawback is the size of its public key, which ranges from 400 KB to 5 MB.

The Gui parameters to be used are as follows:

- $\mathbb{F} = \mathbb{F}_q$: finite field with q elements, $q = 2^e$

- $\mathbb{E} = \mathbb{F}_{q^n}$: degree n extension field of $\mathbb{F}$
- $\phi : \mathbb{F}^n \to \mathbb{E}$: isomorphism between the vector space $\mathbb{F}^n$ and the extension field $\mathbb{E}$
- D: degree of the HFE polynomial, set $r = \lfloor \log_q(D-1) \rfloor + 1$
- a: number of minus equations
- v: number of vinegar variables
- k: repetition factor
- $\bar{l}$: length of the random salt to achieve EUF-CMA security

Gui Signature Operations

Three Gui functions are used in this proposed ring signature scheme; `GuiKeyGen`, `GuiSign`, and `GuiVer`. The flow chart of signature generation and signature verification of Gui are shown in Figure 3.1.

1. Let `GuiKeyGen` be the function to generate Gui keypair (pk, sk) (see Sec. 1.2 in [15]).
   - **Input:** Gui parameters (q, n, D, a, v), isomorphism ϕ, and $\bar{l}$
   - **Output:** Gui keypair $(sk, pk) = ((\mathcal{S}, \mathcal{F}, \mathcal{T}), \mathcal{P})$
     - $\mathcal{S} : \mathbb{F}^n \to \mathbb{F}^{n-a}$: affine transformation of maximal rank
     - $\mathcal{T} : \mathbb{F}^{n+v} \to \mathbb{F}^{n+v}$: invertible affine transformation
     - $\mathcal{F} : \mathbb{E} \times \mathbb{F}^v \to \mathbb{E}$: central map
     - $\mathcal{P} = \mathcal{S} \circ \phi^{-1} \circ \mathcal{F} \circ (\phi \circ id_v) \circ \mathcal{T} : \mathbb{F}^{n+v} \to \mathbb{F}^{n-a}$: public key
2. Let `GuiSign` be the function to generate a Gui signature σ for a given message d (see Sec. 1.3 in [15]).
   - **Input:** Gui private key $sk = (\mathcal{S}, \mathcal{F}, \mathcal{T})$, message d, repetition factor k
   - **Output:** Signature $\sigma \in \mathbb{F}^{(n-a)+k\cdot(a+v)} \times \{0,1\}^{\bar{l}}$
3. Let `GuiVer` be the function to verify if the given Gui signature is valid (see Sec. 1.4 in [15]).

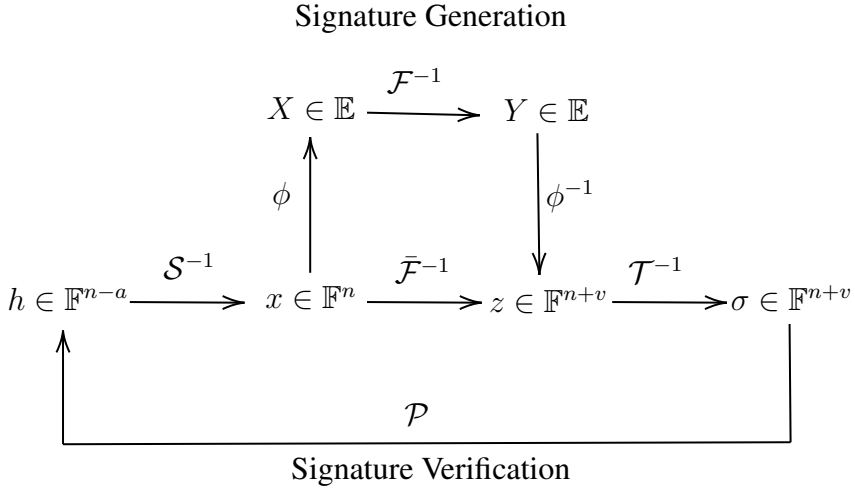

Figure 3.1: Signature generation and verification in Gui signature algorithm

- **Input:** Signature σ, isomorphism ϕ, Gui public key pk, message d, repetition factor k
- **Output:** $\omega \in \mathbb{F}^{n-a}$. If it is equal to zero, then the signature is valid. Otherwise, it is not valid.

3.2 Protocol Description

In this section, we propose a multivariate-based ring signature scheme, where the underlying signature algorithm is Gui.

Let $\mathcal{R} = \{u_1, \ldots, u_t\}$ be a group consisting of t members. Our aim is to create a ring signature scheme that allows a member of the group to create a signature of a message without exposing the signer's identity. In this scheme, we have two assumptions for the setup:

- All members of the group $\mathcal{R}$ have already generated their public key pairs accordingly and made their public keys available for everyone.
- All possible verifiers have already obtained the public keys of the group $\mathcal{R}$ so that they do not have to download these keys with each ring signature again and again.

In this scheme, *key generation* is a one-time operation for each user. After each member generates their key pairs, this operation will not be necessary again. The protocol mainly consists of two operations; *signature generation*, and *signature verification*. In the scenario where the signer wants to leak a secret by using our proposed scheme, the signer will do the signature generation operation one time; hence the time and computational power required for this operation may not be so important. On the other hand, this ring signature may be verified countless times, which makes the verification operation more important in terms of time and power. Moreover, It is also very important that the signature size is small to avoid unnecessary network traffic.

Key Generation

Each user $u_i \in \mathcal{R}$ generates a Gui keypair $(sk_i, pk_i) = ((\mathcal{S}_i, \mathcal{F}_i, \mathcal{T}_i), \mathcal{P}_i)$ according to the given parameter set by using the function `GuiKeyGen`. For all members of the group, the public key $\mathcal{P}_i$ is published, while $\mathcal{S}_i, \mathcal{F}_i, \mathcal{T}_i$ are kept secret.

Signature Generation

The user $u_i \in \mathcal{R}$ will sign the message M on behalf of the group $\mathcal{R}$ consisting of t users by following the steps:

1. Compute the hash of the message M and take the first $n - a$ bits of the result

$$h = \mathcal{H}(M) \in \mathbb{F}^{n-a}, \tag{3.1}$$

   where $\mathcal{H}$ is a hash function. The proper hash functions to be used are described in the Gui proposal [15]. They are using SHA-2 hash function family members as follows:

   - SHA256 for Gui-184,
   - SHA384 for Gui-312, and
   - SHA512 for Gui-448.

   In our proposed ring signature scheme, choosing a secure hash algorithm that produces an output of length more than $n - a$ bits would be sufficient. It is not mandatory to use the hash algorithms suggested by the Gui.

2. Choose random vectors $\sigma_1, \ldots, \sigma_{i-1}, \sigma_{i+1}, \ldots, \sigma_t \in \mathbb{F}^{(n-a)+k\cdot(a+v)} \times \{0,1\}^{\bar{l}}$,

and then compute

$$\bar{h} = h - \sum_{j=1, j\neq i}^{t} \texttt{GuiVer}(M, pk_j, k, \sigma_j) \in \mathbb{F}^{n-a} \tag{3.2}$$

3. Use the secret key sk_i to find a vector σ_i such that $\texttt{GuiVer}(M, pk_i, k, \sigma_i) = \bar{h}$.

4. The ring signature for the message M is $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_t) \in \mathbb{F}^{t\cdot[(n-a)+k\cdot(a+v)]} \times \{0,1\}^{\bar{l}\cdot t}$.

Signature Verification

The verifier will check if the given signature σ of the message M is created by a member of the group $\mathcal{R}$ by following the steps:

1. Compute the hash of the message M and take the first $n-a$ bits of the result

$$h = \mathcal{H}(M) \in \mathbb{F}^{n-a}, \tag{3.3}$$

where $\mathcal{H}$ is a hash function.

2. Uses the public keys $\mathcal{P}_1, \ldots, \mathcal{P}_t$ of the group members to compute

$$\bar{h} = \sum_{j=1}^{t} \texttt{GuiVer}(M, pk_j, k, \sigma_j). \tag{3.4}$$

If $\overline{h} = h$ holds, then the ring signature is valid.

3.3 Performance Analysis

Based on different security levels, there are three sets of parameters of Gui; Gui-184, Gui-312, and Gui-448, as shown in Table 3.1.

Table 3.1: Parameter sets of Gui for the Security Levels 1, 3, and 5, respectively.

Security Level	scheme	parameters (n, D, a, v, k)
I	Gui-184	(184, 33, 16, 16, 2)
III	Gui-312	(312, 129, 24, 20, 2)
V	Gui-448	(448, 513, 32, 28, 2)

Table 3.2 shows the key and signature sizes (*128-bit salt included in signature*) of Gui instances that is proposed for the NIST Post-Quantum Standardization Project. In this table, the performance results of Gui on the NIST Reference Platform for signature generation, signature verification, and key generation processes are also listed, where MC (resp. KC) stands for Mega (resp. Kilo) Cycles. Performance results on other platforms are also given (see Sec. 4.2 in [15]), and they provide better results. However, we prefer to continue with the NIST Reference Platform results in order to make the comparison fair with other signature algorithms.

Table 3.2: Performance of Gui signature algorithm under different parameter sets

	|pk| (KB)	|sk| (KB)	sign (bit)	key gen. (ms)	sign (ms)	verify (ms)	key gen. (MC)	sign (MC)	verify (KC)
Gui-184	416.3	19.1	360	213	10.4	0.05	2,408	1,910	152
Gui-312	1,955.1	59.3	504	13,227	7,707	0.26	43,817	25,436	846
Gui-448	5,789.2	155.9	664	71,485	264,530	0.54	239,502	872,949	1,787

As the security level increases, Table 3.2 shows that the computation times of key generation and signature generation processes increase excessively. Moreover, the public key sizes may cause a problem if they are supposed to be distributed with each signed message. However, in a ring signature scheme, the size of the signature and the speed of verification is much more important. That is why we choose Gui in our ring signature scheme since it provides small signatures with very fast signature verification times.

In order to generate a ring signature on behalf of a group $\mathcal{R}$ of users $\{u_1, u_2, \ldots, u_t\}$, a user u_i will perform $t-1$ evaluations of `GuiVer`, and 1 evaluation of `GuiSign` functions. The verification can be done by t evaluations of `GuiVer` function. Based on the performance on the NIST Reference Platform stated in Gui proposal [15], the theoretical time estimations for our proposed Gui-based ring signature scheme in $\mathbb{F}_2$ are given in Table 3.3.

Table 3.3: Theoretical performance analysis of Gui-based ring signature scheme in three different security levels

Parameters (n, D, a, v, k)		5 users	10 users	20 users	50 users
Gui-184 (184,33,16,16,2)	PK Size (kB)	2,081.5	4,163	8,326	20,815
	Sign. size (bit)	1,968	3,768	7,368	18,168
	Sign. gen. (ms)	10.604	10.859	11.396	12.899
	Sign. ver. (ms)	0.255	0.51	1.02	2.55
Gui-312 (312,129,24,20,2)	PK size (kB)	9,775.5	19,551	39,102	97,755
	Sign. size (bit)	2,808	5,328	10,368	25,488
	Sign. gen. (ms)	7,708.024	7,709.304	7,711.864	7,719.544
	Sign. ver. (ms)	1.28	2.56	5.12	12.8
Gui-448 (448,513,32,28,2)	PK size (kB)	28,946	57,892	115,784	289,460
	Sign. size (bit)	3,736	7,056	13,696	33,616
	Sign. gen. (ms)	264,532.168	264,534.878	264,540.298	264,556.558
	Sign. ver. (ms)	2.71	5.42	10.84	27.1

We compared the theoretical results of our proposed Gui-based ring signature scheme given in Table 3.3 with the running times and signature sizes of the Rainbow-based ring signature scheme (see Table 4 in [29]). Table 3.4 shows that the Gui-based ring signature scheme provides better results with respect to the alternative one.

Table 3.4: Signature verification time and signature size comparison of Gui-based and Rainbow-based ring signature schemes

Sec. Level			5 users	10 users	20 users	50 users
I	Gui	Sign. size (bit)	1,968	3,768	7,368	18,168
		Sign. ver. (ms)	0.255	0.51	1.02	2.55
	Rainbow	Sign. size (bit)	1,920	4,240	10,240	48,800
		Sign. ver. (ms)	5.08-10.81	13.54-26.23	27.42-50.51	703-1,200
III	Gui	Sign. size (bit)	2,808	5,328	10,368	25,488
		Sign. ver. (ms)	1.28	2.56	5.12	12.8
	Rainbow	Sign. size (bit)	2,600	5,680	12,960	54,000
		Sign. ver. (ms)	5.57-12.75	16.38-35.55	57.35-115.97	859-1,547
V	Gui	Sign. size (bit)	3,736	7,056	13,696	33,616
		Sign. ver. (ms)	2.71	5.42	10.84	27.1
	Rainbow	Sign. size (bit)	3,160	6,640	15,040	60,800
		Sign. ver. (ms)	7.70-20.03	20.45-50.43	72.24-163.22	1,610-3,110

Although we achieved efficient results with the Gui signature algorithm in our ring signature scheme, we were supposed to change the underlying signature algorithm. This was because Gui was not announced within the Round 2 candidates of the NIST

Post-Quantum Standardization Project. When we checked the possible candidates to replace Gui, we decided to continue with the GeMSS signature algorithm. The details of the new GeMSS-based ring signature scheme are given in the next chapter of the thesis.

CHAPTER 4

GEMSS BASED RING SIGNATURE SCHEME

In this chapter, we propose another ring signature scheme based on the GeMSS signature algorithm, which is one of Round 3 candidates in NIST Post-Quantum Standardization Project. In Section 4.1, we give necessary background about GeMSS algorithm, and then show how to use it in our proposed signature scheme in Section 4.2. The security proofs are given in Section 4.3. After that, in Section 4.4, we give both theoretical results for all three security levels and implementation results for only Security Level 1. At the end of this chapter, we compare our proposed scheme results with both the Gui-based (see Chapter 3) and the Rainbow-based ring signature schemes (see [29]).

This new GeMSS-based ring signature scheme and the theoretical results was presented in *2 IWCA 19* (*Second International Workshop on Cryptography and Its Applications*) conference, and then published in [13].

4.1 Preliminaries

GeMSS [8] (Great Multivariate Short Signature) is a multivariate signature algorithm with a short signature size, fast verification, and a medium/large public key size. As well as being in direct descendant from QUARTZ [32], GeMSS also adopts some parts of the Gui signature algorithm [18]. It utilizes Feistel-Patarin iterations and adheres to the hash-and-sign paradigm. The trapdoor function in GeMSS is based on Hidden Field Equation with Vinegar Variables and the Minus Modifier (HFEv-).

It is a member of the multivariate cryptosystems' big field family. In order to describe the multivariate trapdoor function stated in terms of the small field $GF(q)$ as a univariate function over the big field, $GF(q^n)$, these schemes use a bijective mapping between $GF(q^n)$ and $GF(q)^n$. Thus, this makes it possible to invert the function effectively. In order to generate the public key, the function is composed of linear mappings over the small field, which is thought to hide the structure. After the original big field scheme of Matsumoto and Imai [27] was broken by [30], the HFE cryptosystem was created [31]. HFE, however, has very slow signing when using secure parameters. By [32], the Vinegar and Minus modifiers were added in an effort to boost security with minimum sacrifice to performance.

The main parameters of GeMSS are:

- D, a positive integer that is the degree of a secret polynomial such that $D = 2^i$ for $i \geq 0$, or $D = 2^i + 2^j$ for $i \neq j$, and $i, j \geq 0$,
- K, the output size in bits of the hash function,
- λ, the security level of GeMSS,
- m, number of equations in the public-key,
- $nb\_ite > 0$, number of iterations in the public-key,
- n, the degree of a field extension of $\mathbb{F}_2$,
- v, the number of vinegar variables,
- Δ, the number of minus, where $m = n - \Delta$.

GeMMS Signature Operations

The **secret-key** is composed of a couple of invertible matrices $(\mathcal{S}, \mathcal{T}) \in \mathsf{GL}_{n+v}(\mathbb{F}_2) \times \mathsf{GL}_n(\mathbb{F}_2)$ and a polynomial $\mathcal{F} \in \mathbb{F}_{2^n}[X, v_1, \ldots, v_v]$ with a specific structure detailed in [8]. The procedure of generating a signature is mainly based on finding the roots of $\mathcal{F}$. The **public-key** is a set $\mathcal{P} = (p_1, \ldots, p_m) \in \mathbb{F}_2[x_1, \ldots, x_{n+v}]^m$ of m quadratic equations in $n + v$ variables. These equations are derived from a multivariate polynomial $\mathcal{F}$.

There are three main algorithms of GeMSS; key generation, signing process, and verification process. Let $\mathbb{F} = \mathbb{F}_2$ and choose $nb\_ite = 4$ as stated in [8].

1. Let `GKeyGen` be the function to generate GeMSS keypair (pk, sk) (see Sec. 2.2 in [8]).

   - **Input:** GeMSS parameters (λ, D, n, v, m)
   - **Output:** GeMSS keypair $(sk, pk) = ((\mathcal{F}, \mathcal{S}, \mathcal{T}), \mathcal{P})$

2. Let `GSign` be the function to generate a GeMSS signature σ for a given message M (see Sec. 2.3 in [8]). Algorithm 1 shows the steps of this function.

   - **Input:** GeMSS private-key $sk = (\mathcal{F}, \mathcal{S}, \mathcal{T})$, message M, repetition factor nb_ite
   - **Output:** Signature $\sigma = (S_{nb\_ite}, X_{nb\_ite}, \ldots, X_1) \in \mathbb{F}^{m+nb\_ite(n+v-m)}$

Algorithm 1 GeMSS signing process

Input: $sk = (\mathcal{F}, \mathcal{S}, \mathcal{T})$, $M \in \{0,1\}^*$, nb_ite

Output: $\sigma = (S_{nb\_ite}, X_{nb\_ite}, \ldots, X_1) \in \mathbb{F}^{n+nb\_ite(n+v-m)}$

1: $H \leftarrow \text{SHA3}(M)$
2: $S_0 \leftarrow 0 \in \mathbb{F}_2^m$
3: **for** $i \leftarrow 1$ `to` nb_ite **do**
4: $\quad D_i \leftarrow$ first m bits of H
5: $\quad (S_i, X_i) \leftarrow GeMSS.Inv_p(D_i \oplus S_{i-1})$ $\triangleright$ $S_i \in \mathbb{F}$, $X_i \in \mathbb{F}^{n+v-m}$, and $\oplus$ is the component-wise XOR
6: $\quad H \leftarrow \text{SHA3}(H)$
7: **end for**
8: **return** $(S_{nb\_ite}, X_{nb\_ite}, \ldots, X_1)$

3. Let `GVer` be the function to verify if the given GeMSS signature is valid (see Sec. 2.4 in [8]). Algorithm 2 shows the steps of this function.

   - **Input:** Signature σ, GeMSS public key pk, message M, repetition factor nb_ite
   - **Output:** Returns TRUE only if the signature is valid.

Algorithm 2 GeMSS verification process

Input: $\sigma \in \mathbb{F}^{n+nb\_ite(n+v-m)}$, $pk = \mathcal{P}$, $M \in \{0,1\}^*$, nb_ite

Output: $True$ or $False$

1: $H \leftarrow$ SHA3(M)
2: $(S_{nb\_ite}, X_{nb\_ite}, \ldots, X_1) \leftarrow \sigma$
3: **for** $i \leftarrow nb\_ite$ `to` 1 **do**
4: $\quad D_i \leftarrow$ first m bits of H
5: $\quad H \leftarrow$ SHA3(H)
6: **end for**
7: **for** $i \leftarrow (nb\_ite - 1)$ `to` 0 **do**
8: $\quad S_i \leftarrow \mathcal{P}(S_{i+1}, X_{i+1}) \oplus Di + 1$
9: **end for**
10: **return** VALID if $S_0 = 0$ and INVALID otherwise.

In the signing process function `GSign`, an $S_0 \in \mathbb{F}^m$ vector is initialized to zero, and after that the calculations are completed as stated in (Algorithm 4, [8]). In order to check if the given signature is valid, the verification process `GVer` does the calculations as stated in (Algorithm 5, [8]), and at the end, it checks if the S_0 vector equals to zero. However, we have to generate a signature with a given non-zero S_0 value in our proposed ring signature scheme. This means that, during the signature generation process, S_0 will be taken as an input, and the verification process will calculate the final S_0 value as an output. So we need to modify the signature generation and verification functions. Let's rename them as $GSignM$ and $GVerM$ to imply that they are modified.

1. Let `GSignM` be the modified `GSign` function. In the original signature process defined in Algorithm 4 in [8], $S_0 \in \mathbb{F}^m$ value is initialized to all-zero vector. However, we need to use a non-zero S_0 vector as input as stated in Algorithm 3.

Algorithm 3 Modified GeMSS signing process

Input: $sk = (\mathcal{F}, \mathcal{S}, \mathcal{T})$, $M \in \{0,1\}^*$, nb_ite, $S_0 \in \mathbb{F}^m$

Output: $\sigma = (S_{nb\_ite}, X_{nb\_ite}, \ldots, X_1) \in \mathbb{F}^{n+nb\_ite(n+v-m)}$

1: $H \leftarrow$ SHA3(M)
2: **for** $i \leftarrow 1$ `to` nb_ite **do**
3: $D_i \leftarrow$ first m bits of H
4: $(S_i, X_i) \leftarrow GeMSS.Inv_p(D_i \oplus S_{i-1})$ ▷ $S_i \in \mathbb{F}$, $X_i \in \mathbb{F}^{n+v-m}$, and $\oplus$ is the component-wise XOR
5: $H \leftarrow$ SHA3(H)
6: **end for**
7: **return** $(S_{nb\_ite}, X_{nb\_ite}, \ldots, X_1)$

2. Let `GVerM` be the modified `GVer` function which is defined in Algorithm 5 in [8]. Instead of checking if S_0 is all-zero, the new function will return its value as shown in Algorithm 4.

Algorithm 4 Modified GeMSS verification process

Input: $\sigma \in \mathbb{F}^{n+nb\_ite(n+v-m)}$, $pk = \mathcal{P}$, $M \in \{0,1\}^*$, nb_ite

Output: $S_0 \in \mathbb{F}^m$

1: $H \leftarrow$ SHA3(M)
2: $(S_{nb\_ite}, X_{nb\_ite}, \ldots, X_1) \leftarrow \sigma$
3: **for** $i \leftarrow nb\_ite$ `to` 1 **do**
4: $D_i \leftarrow$ first m bits of H
5: $H \leftarrow$ SHA3(H)
6: **end for**
7: **for** $i \leftarrow (nb\_ite - 1)$ `to` 0 **do**
8: $S_i \leftarrow \mathcal{P}(S_{i+1}, X_{i+1}) \oplus Di + 1$
9: **end for**
10: **return** S_0

Please note that, instead of using an initialized S_0, the new signature function will take S_0 as an input. Additionally, instead of returning TRUE or FALSE, the new verification function will output the final value of S_0. Therefore, the extra computation cost of these modifications will be negligible.

4.2 Protocol Description

In this section, we propose a new multivariate ring signature scheme based on the GeMSS signature algorithm. Since our proposed scheme is mainly based on the verification process, GeMSS is a perfect choice with its fast verification time and small signature size.

Let $\mathcal{R} = \{u_1, \ldots, u_t\}$ be a group consisting of t members. We aim to create a ring signature scheme that allows a group member to create a signature of a message without exposing the signer's identity. In this ring signature scheme, there are two initial assumptions:

- All members of the group $\mathcal{R}$ have already generated their public key pairs accordingly and made their public keys available for everyone.

- All possible verifiers have already got the public keys of the group $\mathcal{R}$ so that they do not have to download these keys with each ring signature again and again.

In our proposed scheme, *key generation* is a one-time operation for any member of the group $\mathcal{R}$. After each member generates their key pairs, this operation will not be necessary again. The protocol mainly consists of two operations; *signature generation*, and *signature verification*. The signer will do the signature operation one time; hence the time and computational power required for this operation may not be so important. On the other hand, this ring signature may be verified countless times, which makes the verification operation more important in terms of time and power. Moreover, It is also very important that the signature size is small to avoid unnecessary network traffic.

Key Generation

Each member $u_i \in \mathcal{R}$ generates their own key pairs $(sk_i, pk_i) = ((\mathcal{F}_i, \mathcal{S}_i, \mathcal{T}_i), \mathcal{P}_i)$ by

using the key generation function `GKeyGen` of GeMSS, where

$$\begin{aligned}(\mathcal{S}_i, \mathcal{T}_i) &\in \mathsf{GL}_{n+v}(\mathbb{F}_2) \times \mathsf{GL}_n(\mathbb{F}_2) \\ \mathcal{F}_i &\in \mathbb{F}_{2^n}[X, v_1, \ldots, v_v] \\ \mathcal{P}_i &\in \mathbb{F}_2[x_1, \ldots, x_{n+v}]^m\end{aligned} \tag{4.1}$$

The group public key is the list of the public keys of all users, i.e. $\mathcal{PK} = \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_t\}$. Each user u_i keeps their private key $sk_i = (\mathcal{F}_i, \mathcal{S}_i, \mathcal{T}_i)$ as secret.

Signature Generation

A user u_i should follow the steps below to sign a message M on behalf of the group $\mathcal{R}$:

1. Compute the hash of the message M and take the first m-bits:

$$h = \mathcal{H}(M) \in \mathbb{F}_2^m \tag{4.2}$$

   In our proposed ring signature scheme, choosing a secure hash algorithm that produces an output of length more than m-bits would be sufficient.

2. Choose random vectors $\sigma_1, \ldots, \sigma_{i-1}, \sigma_{i+1}, \ldots, \sigma_t \in \mathbb{F}_2^{m+nb\_ite(n+v-m)}$ as fake signatures, and then calculate

$$\overline{h} = h - \sum_{j=1, j\neq i}^{t} \texttt{GVerM}(\sigma_j, pk_j, M, nb\_ite) \tag{4.3}$$

3. Compute a vector σ_i such that $\texttt{GVerM}(\sigma_i, pk_i, M, nb\_ite) = \overline{h}$. In order to that, the signer will calculate $\texttt{GSignM}(sk_i, M, nb\_ite, \overline{h}) = \sigma_i$ by using his private key sk_i

The ring signature for the message M is $\sigma = (\sigma_1, \ldots, \sigma_t) \in \mathbb{F}_2^{[m+nb\_ite(n+v-m)]t}$.

Signature Verification

In order to check if the given ring signature σ is valid for the message M, the verifier follows the following steps:

1. Compute the hash of the message M and take the first m-bits:

$$h = \mathcal{H}(M) \in \mathbb{F}_2^m \tag{4.4}$$

2. Use the group public key $\mathcal{PK}$ and compute

$$\overline{h} = \sum_{j=1}^{t} \texttt{GVerM}(\sigma_j, pk_j, M, nb\_ite) \tag{4.5}$$

If $\overline{h} = h$ holds, then the ring signature is valid. Otherwise, the given ring signature for the message M is not valid.

4.3 Security Analysis

Instead of analyzing the security of the underlying GeMSS signature algorithm, we will be concentrated on the security of our construction. Therefore, the anonymity and unforgeability properties of the proposed ring signature scheme have to be analyzed.

Theorem 1 (Completeness)**.** *Our proposed ring signature scheme provides completeness. In other words, if a message is signed by a group member by using our ring signature method, then it will always be verified to be true by any verifier.*

Proof. The verifier gets a ring signature $\sigma = (\sigma_1, \ldots, \sigma_t)$ of a message M with respective to the group $\mathcal{R}$ consisting of $t-$users. We assume that the signature is generated according to our proposed ring signature scheme, and there isn't any data loss during the transmission of the data. We know that $t-1$ components of σ were randomly generated during the ring signature generation process. These $(t-1)$ fake signatures will always be summed up to the same result

$$C = \sum_{j=1, j\neq i}^{t} \texttt{GVerM}(\sigma_j, pk_j, M, nb\_ite). \tag{4.6}$$

Regardless of which user in the group creates the ring signature scheme, the signer will use his private key to calculate σ_i such that $\texttt{GVerM}(\sigma_i, pk_i, M, nb\_ite) = h - C$, where h is the first $m-$bits of the hash of the message M. This can be done simply by calculating $\sigma_i \leftarrow \texttt{GSignM}(sk_i, M, nb\_ite, h - C)$.

The verifier then uses GVerM function to check if the given ring signature is valid or

not.

$$\begin{aligned}\sum_{j=1}^{t} \texttt{GVerM}(\sigma_j, pk_j, M, nb\_ite) = \\ \sum_{j=1, j\neq i}^{t} \texttt{GVerM}(\sigma_j, pk_j, M, nb\_ite) \\ +\texttt{GVerM}(\sigma_i, pk_i, M, nb\_ite) \\ = C + (h - C) = h\end{aligned} \tag{4.7}$$

Regardless of the signer, the result of the ring signature verification will always be the hash of the message if the message is signed by a group member. Therefore, this concludes the completeness of our proposed scheme. □

Theorem 2 (Anonymity)**.** *Full anonymity is provided with our proposed ring signature scheme. This means that the calculated ring signature does not contain any information about the actual identity of the signer. The verifier will only know that the signer is a member of the group.*

Proof. Assume that the adversary $\mathcal{A}$ has access to a signing oracle $\mathcal{OS}(i, M)$ which returns a valid ring signature in the name of the group $\mathcal{R}$ by using the secret key sk_i of the user $u_i \in \mathcal{R}$. As defined in the anonymity game, adversary $\mathcal{A}$ chooses a message M along with two separate indices $i_0, i_1 \in \{1, \ldots, t\}$. After that, $b \in \{0, 1\}$ is chosen randomly, and then the ring signature σ is generated by using the secret key sk_{i_b} which belongs to user $u_{i_b} \in \mathcal{R}$. Then the ring signature $\sigma = (\sigma_1, \ldots, \sigma_t)$ and the entire secret keys $\{sk_1, \ldots, sk_t\}$ are given to the adversary $\mathcal{A}$. At this point, adversary $\mathcal{A}$ is not allowed to use the signing oracle to query for i_0 and i_1.

Next, we will analyze the distribution of the ring signature σ. During the ring signature generation process, all σ_i's, where $i \neq i_b$, are randomly chosen, and σ_{i_b} is calculated by using the secret key sk_{i_b} that will be resulted as one of $2^{m+nb\_ite(n+v-m)}$ possible ring signatures for the message M. Because of this reason, we may conclude that $\sigma = (\sigma_1, \ldots, \sigma_t)$ is uniformly distributed. Even if the adversary has access to all of the group members' private keys, his chances of guessing the actual signer's identity between sk_{i_0} and sk_{i_1} is not greater than $1/2$. In other words, we have $Pr[sk_{i_0}\ is\ used\ to\ generate\ \sigma] = Pr[sk_{i_1}\ is\ used\ to\ generate\ \sigma] \approx 1/2$. Therefore, adversary $\mathcal{A}$ doesn't have any advantage in this game, and he can only

make a guess whether σ was computed with sk_{i_0} or sk_{i_1}. In conclusion, our proposed scheme satisfies the anonymity criteria. □

Theorem 3 (Unforgeability)**.** *Our proposed scheme provides unforgeability; thus, no one out of the group can create a valid ring signature on behalf of the target group by using our method.*

Proof. If an adversary $\mathcal{A}$ wants to forge a valid ring signature on behalf of a group $\mathcal{R} = \{u_1, \ldots, u_t\}$, then he may try to follow the steps of creating a ring signature as a member of the target group by simply

- Choosing random vectors $\sigma_1, \ldots, \sigma_{t-1} \in \mathbb{F}_2^{m+nb\_ite(n+v-m)}$,
- Computing $\overline{h} = h - \sum_{j=1}^{t-1} \texttt{GVerM}(\sigma_j, pk_j, M, nb\_ite)$,
- Trying to find a σ_t solution such that $\texttt{GVerM}(\sigma_t, pk_t, M, nb\_ite) = \overline{h}$.

The last step requires the secret key sk_t of the group's t^{th} user. Without knowing the secret key, finding a solution for that equation means the underlying signature algorithm GeMSS is broken.

The public-key of GeMSS is a set $\mathcal{P} = (p_1, \ldots, p_m) \in \mathbb{F}_2[x_1, \ldots, x_{n+v}]^m$ of m non-linear equations in $n+v$ variables. For a given hash $h = (h_1, \ldots, h_m) \in \mathcal{F}_2^m$, forging a signature is equivalent to solve the following system of non-linear equations:

$$\begin{aligned} p_1(x_1, \ldots, x_{n+v}) - h_1 &= 0 \\ p_2(x_1, \ldots, x_{n+v}) - h_2 &= 0 \\ &\vdots \\ p_m(x_1, \ldots, x_{n+v}) - h_m &= 0 \end{aligned} \tag{4.8}$$

Since the parameters are chosen as $n + v > m$, this system is underdetermined. In [8], the number of binary operations needed for exhaustive search is calculated as $4\ log_2(m)2^m$. For the security Levels 1, 3, and 5, the values of m are 162, 243, and 324, respectively. These parameter choices result in $2^{166.87}, 2^{247.98}, 2^{329.98}$ binary operations in the given order. Another proposed method in [8] is to perform a fast quantum exhaustive search by using Grover's algorithm [24]. Again, the cost for this operation is not feasible.

The security of GeMSS is studied deeply in [8] by the authors. It is also one of the signature algorithm candidates in NIST's project since 2016, and that makes it a target for cryptanalysis researchers. Until now, there is not any published article concerning about practical security flaws of GeMSS. If we follow the recommendations to choose the parameters for the GeMSS signature algorithm as stated in their paper, our proposed ring signature scheme will also be secure, thus unforgeable. □

4.4 Performance Analysis

This section explains why the GeMSS signature algorithm is a better choice for our ring signature scheme with respect to other alternatives. We give both theoretical and implementation results for our proposed scheme. In order to create a ring signature on behalf of a group $\mathcal{R} = \{u_1, \ldots, u_t\}$ consisting of t members, a signer has to perform one signature generation operation and $t - 1$ signature verification operations. We can conclude that the performance of the overall scheme mainly relies on how fast the verification process is. If we assume that the public keys of the group members have already been published, then the verifier only needs to download the signature along with the signed document. This is why the signature size of the chosen signature algorithm is also important.

There are six sets of parameters for GeMSS, as show in Table 4.1. These are GeMSS, BlueGeMSS, RedGeMSS, WhiteGeMSS, CyanGeMSS and MagentaGeMSS.

Table 4.1: Parameter sets for GeMSS for the Security Levels 1, 3, and 5, respectively.

Security Level	Scheme	(λ, D, n, Δ, v, nb_ite)
I	GeMSS128	(128, 513, 174, 12, 12, 4)
	BlueGeMSS128	(128, 129, 175, 13, 14, 4)
	RedGeMSS128	(128, 17, 177, 15, 15, 4)
	WhiteGeMSS128	(128, 513, 175, 12, 12, 3)
	CyanGeMSS128	(128, 129, 177, 14, 13, 3)
	MagentaGeMSS128	(128, 17, 178, 15, 15, 3)
III	GeMSS192	(192, 513, 265, 22, 20, 4)
	BlueGeMSS192	(192, 129, 265, 22, 23, 4)
	RedGeMSS192	(192, 17, 266, 23, 25, 4)
	WhiteGeMSS192	(192, 513, 268, 21, 21, 3)
	CyanGeMSS192	(192, 129, 270, 23, 22, 3)
	MagentaGeMSS192	(192, 17, 271, 24, 24, 3)
V	GeMSS256	(256, 513, 354, 30, 33, 4)
	BlueGeMSS256	(256, 129, 358, 34, 32, 4)
	RedGeMSS256	(256, 17, 358, 34, 35, 4)
	WhiteGeMSS256	(256, 513, 364, 31, 29, 3)
	CyanGeMSS256	(256, 129, 364, 31, 32, 3)
	MagentaGeMSS256	(256, 17, 366, 33, 33, 3)

Table 4.2 shows public and secret key sizes, signature size, and performance results of signature generation and verification processes where MC (resp. KC) stands for Mega (resp. Kilo) Cycles. Even though the MagentaGeMSS parameter set does not provide the fastest verification, it has the fastest signature time with respect to other parameter sets. Although the signature generation function is used only once during ring signature generation, it greatly impacts overall computation time. Unless the scheme is used by a group consisting of thousands of members, it is better to choose the parameter set with a faster signature generation time. On the other hand, RedGeMSS provides similar performance results with smaller public key sizes with respect to Magenta GeMSS. Our proposed scheme works on any parameter sets of GeMSS. Our aim is to show that the scheme works fluently. Because of this reason, we choose to use the RedGeMSS parameter set in our implementation.

Table 4.2: Performance of GeMSS signature algorithm under different parameter sets

Security Level	Scheme	key gen. (MC)	sign (MC)	verify (KC)	|pk| (KB)	|sk| (bits)	sign (bits)
I	*GeMSS128*	19.6	608	106	352.19	128	258
	BlueGeMSS128	18.4	67.2	134	363.61	128	270
	RedGeMSS128	16.3	2.05	141	375.21	128	282
	WhiteGeMSS128	20	436	91.7	358.17	128	235
	CyanGeMSS128	18.5	49.8	91	369.72	128	244
	MagentaGeMSS128	16.7	1.82	101	381.46	128	253
III	*GeMSS192*	69.4	1,760	304	1,237.96	192	411
	BlueGeMSS192	65	173	325	1,264.12	192	423
	RedGeMSS192	57.1	5.55	335	1,290.54	192	435
	WhiteGeMSS192	73.1	1,330	263	1,293.85	192	373
	CyanGeMSS192	68.2	131	269	1,320.8	192	382
	MagentaGeMSS192	60.3	4.53	274	1,348.03	192	391
V	*GeMSS256*	158	2,490	665	3,040.7	256	576
	BlueGeMSS256	152	248	680	3,087.96	256	588
	RedGeMSS256	143	8.76	709	3,135.59	256	600
	WhiteGeMSS256	163	1,920	516	3,222.69	256	513
	CyanGeMSS256	159	190	535	3,272.02	256	522
	MagentaGeMSS256	148	7.61	535	3,321.72	256	531

Theoretical Performance Results

Based on the expectations from our proposed scheme in terms of fast signature generation, fast verification, or small key and signature sizes, a signer will choose the parameter set to be used. We use Table 4.2 to calculate the theoretical performance of our scheme for a group consisting of 5, 10, 20, and 50 users, respectively, and show the results for all GeMSS parameter sets on 3 different security levels in Tables 4.3, 4.4, and 4.5. By checking these tables, a signer in a group consisting of 50 users may choose WhiteGeMSS for shorter signatures, MagentaGeMSS for faster signature generation, CyanGeMSS for faster verification, etc. There will be a trade-off between these properties while choosing the appropriate set of parameters.

Table 4.3: Theoretical results of GeMSS-based ring signature schemes for all GeMSS parameter sets in Security Level 1

		5 users	10 users	20 users	50 users
GeMSS128	PK size (kB)	1,761	3,522	7,044	17,610
	Sign. size (bit)	1,290	2,580	5,160	12,900
	Sign. Gen (MC)	608	609	610	613
	Sign Ver. (KC)	530	1,060	2,120	5,300
BlueGeMSS128	PK size (kB)	1,818	3,636	7,272	18,181
	Sign. size (bit)	1,350	2,700	5,400	13,500
	Sign. Gen (MC)	67.74	68.41	69.75	73.77
	Sign Ver. (KC)	670	1,340	2,680	6,700
RedGeMSS128	PK size (kB)	1,876	3,752	7,504	18,761
	Sign. size (bit)	1,410	2,820	5,640	14,100
	Sign. Gen (MC)	2.61	3.32	4.73	8.96
	Sign Ver. (KC)	705	1,410	2,820	7,050
WhiteGeMSS128	PK size (kB)	1,791	3,582	7,163	17,909
	Sign. size (bit)	1,175	2,350	4,700	11,750
	Sign. Gen (MC)	436	437	438	440
	Sign Ver. (KC)	459	917	1,834	4,585
CyanGeMSS128	PK size (kB)	1,849	3,697	7,394	18,486
	Sign. size (bit)	1,220	2,440	4,880	12,200
	Sign. Gen (MC)	50.16	50.62	51.53	54.26
	Sign Ver. (KC)	455	910	1,820	4,550
MagentaGeMSS128	PK size (kB)	1,907	3,815	7,629	19,073
	Sign. size (bit)	1,265	2,530	5,060	12,650
	Sign. Gen (MC)	2.22	2.73	3.74	6.77
	Sign Ver. (KC)	505	1,010	2,020	5,050

Table 4.4: Theoretical results of GeMSS-based ring signature schemes for all GeMSS parameter sets in Security Level 3

		5 users	10 users	20 users	50 users
GeMSS192	PK size (kB)	6,190	12,380	24,759	61,898
	Sign. size (bit)	2,055	4,110	8,220	20,550
	Sign. Gen (MC)	1,761	1,763	1,766	1,775
	Sign Ver. (KC)	1,520	3,040	6,080	15,200
BlueGeMSS192	PK size (kB)	6,321	12,641	25,282	63,206
	Sign. size (bit)	2,115	4,230	8,460	21,150
	Sign. Gen (MC)	174.30	175.93	179.18	188.93
	Sign Ver. (KC)	1,625	3,250	6,500	16,250
RedGeMSS192	PK size (kB)	6,453	12,905	25,811	64,527
	Sign. size (bit)	2,175	4,350	8,700	21,750
	Sign. Gen (MC)	6.89	8.57	11.92	21.97
	Sign Ver. (KC)	1,675	3,350	6,700	16,750
WhiteGeMSS192	PK size (kB)	6,469	12,939	25,877	64,693
	Sign. size (bit)	1,865	3,730	7,460	18,650
	Sign. Gen (MC)	1,331	1,332	1,335	1,343
	Sign Ver. (KC)	1,315	3,350	5,260	13,150
CyanGeMSS192	PK size (kB)	6,604	13,208	26,416	66,040
	Sign. size (bit)	1,910	3,820	7,640	19,100
	Sign. Gen (MC)	132.08	133.42	136.11	144.18
	Sign Ver. (KC)	1,345	2,690	5,380	13,450
MagentaGeMSS192	PK size (kB)	6,740	13,480	26,961	67,402
	Sign. size (bit)	1,955	3,910	7,820	19,550
	Sign. Gen (MC)	5.63	7	9.74	17.96
	Sign Ver. (KC)	1,370	2,740	5,480	13,700

Table 4.5: Theoretical results of GeMSS-based ring signature schemes for all GeMSS parameter sets in Security Level 5

		5 users	10 users	20 users	50 users
GeMSS256	PK size (kB)	15,204	30,407	60,814	152,035
	Sign. size (bit)	2,880	5,760	11,520	28,800
	Sign. Gen (MC)	2,493	2,496	2,503	2,523
	Sign Ver. (KC)	3,325	6,650	13,300	33,250
BlueGeMSS256	PK size (kB)	15,440	30,880	61,759	154,398
	Sign. size (bit)	2,940	5,880	11,760	29,400
	Sign. Gen (MC)	250.72	254.12	260.92	281.32
	Sign Ver. (KC)	3,400	6,800	13,600	34,000
RedGeMSS256	PK size (kB)	15,678	31,356	62,712	156,780
	Sign. size (bit)	3,000	6,000	12,000	30,000
	Sign. Gen (MC)	11.60	15.14	22.23	43.50
	Sign Ver. (KC)	3,545	7,090	14,180	35,450
WhiteGeMSS256	PK size (kB)	16,113	32,227	64,454	161,135
	Sign. size (bit)	2,565	5,130	10,260	25,650
	Sign. Gen (MC)	1,922	1,925	1,930	1,945
	Sign Ver. (KC)	2,580	5,160	10,320	25,800
CyanGeMSS256	PK size (kB)	16,360	32,720	65,440	163,601
	Sign. size (bit)	2,610	5,220	10,440	26,100
	Sign. Gen (MC)	192.14	194.82	200.17	216.22
	Sign Ver. (KC)	2,675	5,350	10,700	26,750
MagentaGeMSS256	PK size (kB)	16,609	33,217	66,434	166,086
	Sign. size (bit)	2,655	5,310	10,620	26,550
	Sign. Gen (MC)	9.75	12.43	17.78	33.83
	Sign Ver. (KC)	2,675	5,350	10,700	26,750

Implementation Results

In order to show that our proposed GeMSS-based ring signature scheme works efficiently without any problem, we implemented it in the C programming language by using the reference implementation of the GeMSS project. The source code of the reference implementation, which was submitted to the second round of the NIST Post-Quantum standardization project, is available on the GeMSS project website [1]. This reference implementation includes only GeMSS, BlueGeMSS, and RedGeMSS parameter sets. Due to its faster verification performance, we decided to continue with GeMSS128 parameter set for the proof of concept implementation. We had to make some modification in the code to use the modified signature generation `GSignM` and

[1] See https://www-polsys.lip6.fr/Links/NIST/GeMSS.html for the reference implementation.

verification `GVerM` functions as described in Section 4.1.

We run our code on a machine with the following specifications:

- **Processor:** 2.0 GHz quad-core Intel Core i7 processor with 6MB shared L3 cache
- **Clock Speed:** 2.0 GHz
- **Memory:** 16GB of 1600MHz DDR3L onboard memory
- **Operating System:** MacOS Big Sur Version 11.6.5, Apple clang version 13.0.0 (clang-1300.0.29.30)

Table 4.6 shows the computation time of our code after running it 1,000 times and taking the average times of both signature generation and verification functions.

Source code for our proposed GeMSS-based ring signature scheme with RedGeMSS128 parameter set can be accessed online [2].

Table 4.6: Implementation results of our RedGeMSS based ring signature scheme for Security Level 1 parameter set

	5 users	10 users	20 users	50 users
PK size (kB)	1,876.05	3,752.1	7,504.2	18,760.5
Signature size (bit)	1,440	2,880	5,760	14,400
Sign. Gen. (ms)	21.57	21.5	22.04	25.28
Sign. Ver. (ms)	0.368	0.77	1.54	4.06

A 2.0 gigahertz (GHz) processor means 2 billion cycles per second. In other words, the computer can achieve 2 megacycles (MC) in 1 millisecond. In Table 4.3, the theoretical verification time of the RedGeMSS128-based scheme for 50 users group is approximately 7 MC, and it will take 3.5 ms to be calculated if there do not exist other processes at that time. When we check the verification time in Table 4.6, we see that the verification time is 4.06 ms for 50 users case. Therefore, it can be said that the theoretical and practical results are close to each other.

[2] See https://github.com/ofLee34/GeMSS_Based_Signature_Scheme for the code.

4.5 Comparison

In Table 4.7, we compare three multivariate signature algorithms. Rainbow is used in [29] and Gui is used in our previous research [12]. RedGeMSS provides the fastest verification time and the smallest signature size with respect to others. Rainbow is slightly better than RedGeMSS in signature generation time. However, due to the slower verification time of Rainbow, the RedGeMMS-based ring signature scheme provides better overall time as the number of members increases in the group.

Table 4.7: Comparison of 3 multivariate signature algorithms in the same security level

Sec. Level	Scheme	public key (KB)	signature (bits)	sign (MC)	verify (KC)
I	Rainbow Ic	187.70	832	1.52	939
	Gui-184	416.30	360	1,910	152
	RedGeMSS128	375.21	282	2.05	141
III	Rainbow IIIc	703.90	1,248	4.05	2,974
	Gui-312	1,995.10	504	25,436	846
	RedGeMSS192	1,290.54	435	5.55	335
V	Rainbow Vc	1,683.30	1,632	8.69	6,174
	Gui-448	5,789.20	664	872,949	1,787
	RedGeMSS256	3,135.59	600	8.76	709

In Table 4.8, we compare our proposed Gui-based and GeMSS-based ring signature algorithms with the Rainbow-based ring signature algorithm in Security Level 1. This table contains experimental results of GeMSS-based and Rainbow-based ring signature schemes, and theoretical results for Gui-based ring signature scheme. Signature generation and signature verification time for the Rainbow-based scheme are given as the slowest and fastest execution time in [29]. That is why we use the same notation for the comparison. Compared to the Rainbow-based approach, the experimental results reveal that our suggested GeMSS-based ring signature technique offers 20% shorter signature sizes in the case of 5 users group, and we get better signature sizes with larger groups. As the number of users increases in the group, our scheme also provides better signature generation time. While the signature generation time is close to each other for ten users case, the GeMSS-based scheme provides up to 50 times faster ring signature generation time with respect to the Rainbow-based scheme as the

number of group members is just above 50 users. Moreover, we get up to 300 times faster verification time and 20% smaller signature size within a group of 50 users.

Table 4.8: Comparison of experimental results of our proposed schemes with Rainbow-based scheme for Security Level 1

# users		Rainbow [29]	Gui [12]	GeMSS [13]
5 users	PK size (kB)	191	2,081.5	1,876
	Sign. size (bit)	1,920	1,968	1,440
	Sign. gen. time (ms)	9.15 – 13.31	10.604	21.50
	Sign. ver. time (ms)	5.08 – 10.81	0.255	0.368
10 users	PK size (kB)	551	4,163	3,752.13
	Sign. size (bit)	4,240	3,768	2,880
	Sign. gen. time (ms)	17.31 – 28.73	10.859	21.57
	Sign. ver. time (ms)	13.54 – 26.23	0.51	0.770
20 users	PK size (kB)	2,095	8,326	7,504.26
	Sign. size (bit)	10,240	7,368	5,760
	Sign. gen. time (ms)	37.04 – 58.98	11.396	22.04
	Sign. ver. time (ms)	27.42 – 50.51	1.02	1.54
50 users	PK size (kB)	40,588	20,815	18,760.65
	Sign. size (bit)	48,800	18,168	14,400
	Sign. gen. time (ms)	738 – 1,225	12.899	25.28
	Sign. ver. time (ms)	703 - 1200	2.55	4.06

After a practical key-recovery attack for the Level 1 parameter set of Rainbow is published in [5], the Rainbow team proposes to replace Level 1 parameters with Level 3 parameters and Level 3 parameters with Level 5 parameters. After we redesign Table 4.7 with these new offered security parameters, we get Table 4.9.

Table 4.9: Comparison of 3 multivariate signature algorithms after the offered solution of Rainbow team against security flaw

Sec. Level	Scheme	public key (KB)	signature (bits)	sign (MC)	verify (KC)
I	Rainbow Ic	703.90	1,248	4.05	2,974
	Gui-184	416.30	360	1,910	152
	RedGeMSS128	375.21	282	2.05	141
III	Rainbow IIIc	1,683.30	1,632	8.69	6,174
	Gui-312	1,995.10	504	25,436	846
	RedGeMSS192	1,290.54	435	5.55	335
V	Rainbow Vc	N/A	N/A	N/A	N/A
	Gui-448	5,789.20	664	872,949	1,787
	RedGeMSS256	3,135.59	600	8.76	709

As a result, GeMSS becomes a much better choice with respect to Rainbow. At all security levels, GeMSS provides smaller public keys and signatures along with much faster signature generation and signature verification. This security update of parameters in Rainbow will also affect the efficiency of the Rainbow-based ring signature scheme in [29]. Their implementation results will be less efficient and less effective when they adopt the new suggested parameters.

CHAPTER 5

CONCLUSION

GeMSS, Gui, and Rainbow algorithms are both multivariate-based cryptosystems proposed in the NIST Post-Quantum Cryptography Standardization Project. If we compare their key and signature sizes and performances on their reference implementations under the same security levels, one can see that the GeMSS and Gui signature algorithms provide smaller signature sizes and faster verification times with respect to the Rainbow signature algorithm. Signature generation times for both GeMSS and Rainbow are close to each other. Since our proposed ring signature scheme is mainly based on a signature verification algorithm, using GeMSS as a signature algorithm in a ring signature scheme will result in a faster evaluation time and smaller signature sizes with respect to the ring signature scheme [29], which is based on the Rainbow algorithm.

In this thesis, we show how to use Gui and GeMSS multivariate signature algorithms in a ring signature scheme efficiently. Moreover, we give the theoretical computation times, signature, and key sizes for both Gui and GeMSS-based schemes under all three security levels with different parameter sets. We also implemented the RedGeMSS128 parameter set of the GeMSS signature algorithm to compare the experimental and theoretical results. Since the other security levels of RedGeMSS parameter set provides faster verification times and smaller signature sizes with respect to the Rainbow algorithm in the same security levels, the implementation results can be projected to provide better results. Finally, we give security necessary security proofs for a ring signature algorithm, which are completeness, anonymity, and unforgeability. In conclusion, we provide an efficient GeMSS-based ring signature algo-

rithm, which provides small signatures with fast calculation time in terms of signature generation, and signature verification.

REFERENCES

[1] M. R. Asaar, M. Salmasizadeh, and W. Susilo, A short identity-based proxy ring signature scheme from RSA, Comput. Stand. Interfaces, 38, pp. 144–151, 2015.

[2] M. Bardet, M. Bros, D. Cabarcas, P. Gaborit, R. A. Perlner, D. Smith-Tone, J. Tillich, and J. A. Verbel, Improvements of algebraic attacks for solving the rank decoding and minrank problems, in S. Moriai and H. Wang, editors, *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part I*, volume 12491 of *Lecture Notes in Computer Science*, pp. 507–536, Springer, 2020.

[3] A. Bender, J. Katz, and R. Morselli, Ring signatures: Stronger definitions, and constructions without random oracles, IACR Cryptol. ePrint Arch., p. 304, 2005.

[4] D. Bernstein, J. Buchmann, and E. Dahmen, *Post-Quantum Cryptography*, Springer Berlin Heidelberg, 2009, ISBN 9783540887027.

[5] W. Beullens, Breaking rainbow takes a weekend on a laptop, IACR Cryptol. ePrint Arch., p. 214, 2022.

[6] E. Bresson, J. Stern, and M. Szydlo, Threshold ring signatures and applications to ad-hoc groups, in M. Yung, editor, *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*, volume 2442 of *Lecture Notes in Computer Science*, pp. 465–480, Springer, 2002.

[7] J. F. Buss, G. S. Frandsen, and J. O. Shallit, The computational complexity of some problems of linear algebra, Electron. Colloquium Comput. Complex., 33(9), 1997.

[8] A. Casanova, J. Faugère, G. Macario-Rat, J. Patarin, L. Perret, and J. Ryckeghem, Gemss: A great multivariate short signature, https://csrc.nist.gov/CSRC/media/Projects/post-quantum-cryptography/documents/round-3/submissions/GeMSS-Round3.zip, oct 2020, round 3 submission for NIST Post-Quantum Standardization Project.

[9] P. Cayrel, R. Lindner, M. Rückert, and R. Silva, A lattice-based threshold ring signature scheme, in M. Abdalla and P. S. L. M. Barreto, editors, *Progress in*

Cryptology - LATINCRYPT 2010, First International Conference on Cryptology and Information Security in Latin America, Puebla, Mexico, August 8-11, 2010, Proceedings, volume 6212 of *Lecture Notes in Computer Science*, pp. 255–272, Springer, 2010.

[10] D. Chaum and E. van Heyst, Group signatures, in D. W. Davies, editor, *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings*, volume 547 of *Lecture Notes in Computer Science*, pp. 257–265, Springer, 1991.

[11] N. Courtois, A. Klimov, J. Patarin, and A. Shamir, Efficient algorithms for solving overdefined systems of multivariate polynomial equations, in *IN ADVANCES IN CRYPTOLOGY, EUROCRYPT'2000, LNCS 1807*, pp. 392–407, Springer-Verlag, 2000.

[12] M. Demircioglu, S. Akleylek, and M. Cenk, Gui based ring signature scheme, Central European Conference on Cryptology 2018, Smolenica, Slovakia, jun 2018.

[13] M. Demircioglu, S. Akleylek, and M. Cenk, Efficient gemss based ring signature scheme, Malaysian Journal of Computing and Applied Mathematics, 3(1), pp. 35–39, jun 2020.

[14] W. Diffie and M. E. Hellman, New directions in cryptography, IEEE Trans. Inf. Theory, 22(6), pp. 644–654, 1976.

[15] J. Ding, Gui, https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/submissions/Gui.zip, dec 2017, round 1 submission for NIST Post-Quantum Standardization Project.

[16] J. Ding, J. E. Gower, and D. Schmidt, *Multivariate Public Key Cryptosystems*, volume 25 of *Advances in Information Security*, Springer, 2006, ISBN 978-0-387-32229-2.

[17] J. Ding and D. Schmidt, Rainbow, a new multivariable polynomial signature scheme, in J. Ioannidis, A. D. Keromytis, and M. Yung, editors, *Applied Cryptography and Network Security, Third International Conference, ACNS 2005, New York, NY, USA, June 7-10, 2005, Proceedings*, volume 3531 of *Lecture Notes in Computer Science*, pp. 164–175, 2005.

[18] J. Ding and B. Yang, Degree of regularity for hfev and hfev-, in P. Gaborit, editor, *Post-Quantum Cryptography - 5th International Workshop, PQCrypto 2013, Limoges, France, June 4-7, 2013. Proceedings*, volume 7932 of *Lecture Notes in Computer Science*, pp. 52–66, Springer, 2013.

[19] J.-C. Faugère, A new efficient algorithm for computing gröbner bases (f_4), in *Journal of Pure and Applied Algebra*, pp. 75–83, ACM Press, 1999.

[20] J.-C. Faugère, A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5), in *International Symposium on Symbolic and Algebraic Computation Symposium - ISSAC 2002*, pp. 75–83, ACM, Villeneuve d'Ascq, France, July 2002, colloque avec actes et comitè de lecture. internationale.

[21] A. Fiat and A. Shamir, How to prove yourself: Practical solutions to identification and signature problems, in A. M. Odlyzko, editor, *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, volume 263 of *Lecture Notes in Computer Science*, pp. 186–194, Springer, 1986.

[22] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, 1979, ISBN 0-7167-1044-7.

[23] L. Goubin, N. T. Courtois, and S. Cp, Cryptanalysis of the ttm cryptosystem, in *Advances of Cryptology, Asiacrypt'2000*, pp. 44–57, Springer, 2000.

[24] L. K. Grover, A fast quantum mechanical algorithm for database search, in G. L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pp. 212–219, ACM, 1996.

[25] A. Kipnis, J. Patarin, and L. Goubin, Unbalanced oil and vinegar signature schemes, in J. Stern, editor, *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, volume 1592 of *Lecture Notes in Computer Science*, pp. 206–222, Springer, 1999.

[26] D. W. Kravitz, Digital signature algorithm, US patent 5231668, jul 1991.

[27] T. Matsumoto and H. Imai, Public quadratic polynominal-tuples for efficient signature-verification and message-encryption, in C. G. Günther, editor, *Advances in Cryptology - EUROCRYPT '88, Workshop on the Theory and Application of of Cryptographic Techniques, Davos, Switzerland, May 25-27, 1988, Proceedings*, volume 330 of *Lecture Notes in Computer Science*, pp. 419–453, Springer, 1988.

[28] C. A. Melchor, P. Cayrel, P. Gaborit, and F. Laguillaumie, A new efficient threshold ring signature scheme based on coding theory, IEEE Trans. Inf. Theory, 57(7), pp. 4833–4842, 2011.

[29] M. S. E. Mohamed and A. Petzoldt, Ringrainbow - an efficient multivariate ring signature scheme, in M. Joye and A. Nitaj, editors, *Progress in Cryptology - AFRICACRYPT 2017 - 9th International Conference on Cryptology in Africa, Dakar, Senegal, May 24-26, 2017, Proceedings*, volume 10239 of *Lecture Notes in Computer Science*, pp. 3–20, 2017.

[30] J. Patarin, Cryptanalysis of the matsumoto and imai public key scheme of eurocrypt'88, in D. Coppersmith, editor, *Advances in Cryptology - CRYPTO '95, 15th Annual International Cryptology Conference, Santa Barbara, California, USA, August 27-31, 1995, Proceedings*, volume 963 of *Lecture Notes in Computer Science*, pp. 248–261, Springer, 1995.

[31] J. Patarin, Hidden fields equations (HFE) and isomorphisms of polynomials (IP): two new families of asymmetric algorithms, in U. M. Maurer, editor, *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, volume 1070 of *Lecture Notes in Computer Science*, pp. 33–48, Springer, 1996.

[32] J. Patarin, N. T. Courtois, and L. Goubin, Quartz, 128-bit long digital signatures, in D. Naccache, editor, *Topics in Cryptology - CT-Asymmetric cryptography, also known as public-key cryptography, 2001, The Cryptographer's Track at RSA Conference 2001, San Francisco, CA, USA, April 8-12, 2001, Proceedings*, volume 2020 of *Lecture Notes in Computer Science*, pp. 282–297, Springer, 2001.

[33] J. Patarin and L. Goubin, Trapdoor one-way permutations and multivariate polynominals, in Y. Han, T. Okamoto, and S. Qing, editors, *Information and Communication Security, First International Conference, ICICS'97, Beijing, China, November 11-14, 1997, Proceedings*, volume 1334 of *Lecture Notes in Computer Science*, pp. 356–368, Springer, 1997.

[34] A. Petzoldt, S. Bulygin, and J. Buchmann, A multivariate based threshold ring signature scheme, IACR Cryptol. ePrint Arch., p. 194, 2012.

[35] A. Petzoldt, M. Chen, B. Yang, C. Tao, and J. Ding, Design principles for hfev-based multivariate signature schemes, in T. Iwata and J. H. Cheon, editors, *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part I*, volume 9452 of *Lecture Notes in Computer Science*, pp. 311–334, Springer, 2015.

[36] R. L. Rivest, A. Shamir, and L. M. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Commun. ACM, 21(2), pp. 120–126, 1978.

[37] R. L. Rivest, A. Shamir, and Y. Tauman, How to leak a secret, in C. Boyd, editor, *Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings*, volume 2248 of *Lecture Notes in Computer Science*, pp. 552–565, Springer, 2001.

[38] K. Sakumoto, T. Shirai, and H. Hiwatari, Public-key identification schemes based on multivariate quadratic polynomials, in P. Rogaway, editor, *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, volume 6841 of *Lecture Notes in Computer Science*, pp. 706–723, Springer, 2011.

[39] P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM Rev., 41(2), pp. 303–332, 1999.

[40] L. L. Wang, A new multivariate-based ring signature scheme, in *Instruments, Measurement, Electronics and Information Engineering*, volume 347 of *Applied Mechanics and Materials*, pp. 2688–2692, Trans Tech Publications Ltd, 10 2013.

[41] S. Wang, R. Ma, Y. Zhang, and X. Wang, Ring signature scheme based on multivariate public key cryptosystems, Comput. Math. Appl., 62(10), pp. 3973–3979, 2011.

[42] A. C. Yao and Y. Zhao, Online/offline signatures for low-power devices, IEEE Trans. Inf. Forensics Secur., 8(2), pp. 283–294, 2013.

[43] J. Zhang and Y. Zhao, A new multivariate based threshold ring signature scheme, in M. H. Au, B. Carminati, and C. J. Kuo, editors, *Network and System Security - 8th International Conference, NSS 2014, Xi'an, China, October 15-17, 2014, Proceedings*, volume 8792 of *Lecture Notes in Computer Science*, pp. 526–533, Springer, 2014.

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Demircioğlu, Murat
Nationality: Turkish
Date and Place of Birth: 1986, İstanbul

EDUCATION

Degree	Institution	Year of Graduation
M.Sc., Cryptography	Middle East Technical University	2011
B.S., Mathematics	Middle East Technical University	2009
High School	Ümraniye Anatolian High School	2004

PUBLICATIONS

International Conference Publications

- Murat Demircioğlu, Sedat Akleylek, Murat Cenk, *Efficient GeMSS Based Ring Signature Scheme*, Malaysian Journal of Computing and Applied Mathematics, 3(1), pp. 35–39, jun 2020. DOI: https://doi.org/10.37231/myjcam.2020.3.1.41

- Cihangir Tezcan, Halil Kemal Taşkın and Murat Demircioğlu, *Improbable Differential Attacks on Serpent using Undisturbed Bits*, In Proceedings of the 7th International Conference on Security of Information and Networks (SIN '14). Glasgow, UK. 2014. DOI: https://doi.org/10.1145/2659651.2659660 .

Proceedings/Posters

- Murat Demircioğlu, Sedat Akleylek, Murat Cenk, *GUI Based Ring Signature Scheme*, CECC'18, Smolenice, Slovakia, June 2018.
- Halil Kemal Taşkın, Murat Demircioğlu and Salim Sarımurat, *End-to-end Encrypted Communication Between Multi-device Users*, Information Security and Cryptology Conference, İstanbul, Turkey, October 2014.
- Murat Demircioğlu, Halil Kemal Taşkın and Salim Sarımurat, *Security Analysis of the Encrypted Mobile Communication Applications*, Information Security and Cryptology Conference, İstanbul, Turkey, October 2014.
- Halil Kemal Taşkın and Murat Demircioğlu, *Off-the-Record Communication with Location Hiding*, Information Security and Cryptology Conference, Ankara, Turkey, September 2013.
- Halil Kemal Taşkın and Murat Demircioğlu, *Applications of Cryptology in Cyber security and End-to-end Encryption*, TBD 31st National IT Congress, Ankara, Turkey, November 2014.
- Murat Demircioğlu, *Effects of file and disk encryption on digital forensics*, International Symposium on Digital Forensics, Ankara, Turkey,May 2014. (Poster)

Invited Talks/Presentations

- *State of Art of Multivariate Public Key Cryptography*, 2nd Workshop on Post-Quantum Cryptography, TÜBİTAK BİLGEM, November 2020

Projects

- Student Researcher in TÜBİTAK 115R289 Project - *Development of Efficient Algorithms for Public Key Cryptography*, (2016 - 2018)
- Expert Researcher in TÜBİTAK 1130063 Project - *Design and Development of a Secure Communication System on Mobile Terminals by Using Cryptographic Methods*, (2014 - 2016)

- Student Researcher in TÜBİTAK 112T011 Project - *Mathematical View of Curve Based Cryptography*, (2012 - 2014)
- Student Researcher in TÜBİTAK 112E101 Project - *Improbable Cryptanalysis of Block Ciphers*, (2012 - 2013)