QUANTUM-RESISTANT MULTIVARIATE QUADRATIC SYSTEMS AND DIGITAL SIGNATURES

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

BY

ESEN ALTUNDAĞ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
CRYPTOGRAPHY

JUNE 2019

Approval of the thesis:

QUANTUM-RESISTANT MULTIVARIATE QUADRATIC SYSTEMS AND DIGITAL SIGNATURES

submitted by **ESEN ALTUNDAĞ** in partial fulfillment of the requirements for the degree of **Master of Science in Cryptography Department, Middle East Technical University** by,

Prof. Dr. Ömür Uğur
Director, Graduate School of **Applied Mathematics** _____

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

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

Examining Committee Members:

Assoc. Prof. Dr. Ali Doğanaksoy
Mathematics Department, METU _____

Assoc. Prof. Dr. Murat Cenk
Cryptography Department, METU _____

Assist. Prof. Dr. Oğuz Yayla
Mathematics Department, Hacettepe University _____

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: ESEN ALTUNDAĞ

Signature :

ABSTRACT

QUANTUM-RESISTANT MULTIVARIATE QUADRATIC SYSTEMS AND DIGITAL SIGNATURES

Altundağ, Esen

M.S., Department of Cryptography

Supervisor : Assoc. Prof. Dr. Murat Cenk

June 2019, 80 pages

In the light of technological advances, scientists expect that quantum computers will be generated and substitute with classical ones, then all symmetric and asymmetric (public-key) cryptosystems will be invalid in the near future. This causes the need for quantum-resistant algorithms all araund the world. That's why, we have focused on multivariate public-key cryptosystems as a kind of post-quantum cryptography. In order to explain the root idea behind this kind of cryptosystems, as a starting point, the Matsumoto-Imai cryptosystem has been scrutinised together with its linearization equations attack. After that, we have constructed our own specific toy example for illustrating the construction of both the single-branch Matsumoto-Imai cryptosystem and its linearization equations attack. As well as these, Matsumoto-Imai variants which were developed with the aim of increasing the security of original one, have been examined. Then, it has been passed on to our main aim which is the analysis of the Multivariate Quadratic Digital Signature Scheme which comes from the family of multivariate public-key cryptosystems. In this process, its structural tools, security sources, parameter sets, general description, detailed description and security analysis have been studied. As a consequence of all these, we have realized that the security of Multivariate Quadratic Digital Signature Scheme against both classical and quantum computers is based on the intractability of the multivariate quadratic problem, the hardness of the commitment schemes which are the structural tools of this algorithm, the splitting idea of the secret-key that comes from the Sakumoto-Shirai-Hiwatari 5-pass Identification Scheme is a special kind of canonical $2n+1$-

pass identification schemes, and the Fiat-Shamir transform which maintains the security in the process of obtaining a signature scheme from an identification scheme. That is, it is possible to generate more secure and effective cryptographic protocols by improving the combination of these tools and ideas with the optimized parameter sets.

Keywords: quantum computers, post-quantum algorithms, random oracle model, quantum random oracle model, multivariate public-key cryptography, multivariate quadratic problem, Matsumoto-Imai cryptosystems, linearization equations, commitment schemes, 5-pass identification schemes, Fiat-Shamir transform, Multivariate Quadratic Digital Signature Scheme.

ÖZ

KUANTUM-DAYANIKLI ÇOK DEĞİŞKENLİ İKİ BİLİNMEYENLİ SİSTEMLER VE SAYISAL İMZALAR

Altundağ, Esen
Yüksek Lisans, Kriptografi Bölümü
Tez Yöneticisi : Doç. Dr. Murat Cenk

Haziran 2019, 80 sayfa

Teknolojik gelişmeler ışığında bilim insanları, yakın gelecekte kuantum bilgisayarların üretilip klasik olanların yerini almasını, müteakiben tüm simetrik ve asimetrik (açık anahtarlı) şifreleme sistemlerinin geçersiz olmasını bekliyorlar. Bu da kuantum-dayanıklı algoritmalara dünya çapında bir ihtiyaç doğuruyor. Bu sebeple, kuantum sonrası şifrelemenin bir türü olan çok değişkenli açık anahtarlı şifreleme sistemlerine odaklandık. Bu tür şifreleme sistemleri ardındaki ana fikri açıklamak için, başlangıç noktası olarak, Matsumoto-Imai şifreleme sistemi, doğrusallaştırma denklemleri saldırısı ile birlikte detaylı bir biçimde incelenmiştir. Akabinde, hem tek dallı Matsumoto-Imai şifreleme sisteminin hem de onun doğrusallaştırma denklemleri saldırısının kurgusunu göstermek için kendi örneğimizi oluşturduk. Bunların yanı sıra, orijinal olanın güvenliğini artırmak amacıyla geliştirilen Matsumoto-Imai varyantları çalışılmıştır. Ardından, çok değişkenli açık anahtarlı şifreleme sistemleri ailesinden gelen ve asıl amacımız olan Çok Değişkenli İkinci Dereceden Sayısal İmza Şemasının analizine geçilmiştir. Bu süreçte onun yapısal araçları, güvenlik kaynakları, parametre setleri, genel tanımlaması, ayrıntılı tanımlaması ve güvenlik analizleri çalışılmıştır. Tüm bunların bir sonucu olarak farkettik ki, Çok Değişkenli İkinci Dereceden Sayısal İmza Şemasının, hem klasik hem de kuantum bilgisayarlara karşı güvenliği, çok değişkenli ikinci derecede problemin zorluğuna, bu algoritmanın yapısal araçlarından olan bağlılık şemalarının güçlüğüne, standart $2n+1$-geçişli kimlik saptama şemalarının özel bir türü olan ve Sakumoto-Shirai-Hiwatari 5-geçişli kimlik saptama şemasından gelen gizli anahtarı ikiye ayırma fikrine ve kimlik saptama şemasından

imza şeması elde etme sürecinde güvenliği sağlayan Fiat-Shamir dönüşümüne dayalıdır. Yani, bu araçların ve fikirlerin optimize edilmiş parametre setleriyle kombinasyonunu geliştirerek daha güvenli ve etkili şifreleme protokolleri oluşturmak mümkündür.

Anahtar Kelimeler: kuantum bilgisayarlar, kuantum sonrası algoritmalar, rassal kahin modeli, kuantum rassal kahin modeli, çok değişkenli açık anahtarlı şifreleme, çok değişkenli iki bilinmeyenli problem, Matsumoto-Imai şifreleme sistemleri, doğrusallaştırma denklemleri, bağlılık şemaları, 5-geçişli kimlik saptama şemaları, Fiat-Shamir dönüşümü, Çok Değişkenli İki Bilinmeyenli Sayısal İmza Şeması

To my dear family ...

ACKNOWLEDGMENTS

Throughout the writing of this thesis, I have received a great deal of support and assistance from my thesis supervisor Assoc. Prof. Dr. Murat Cenk. So, I would first like to express my special thanks of gratitude to him for inspiring, motivating and encouraging me invaluably. Thanks to his precious advices and support, I could complete my thesis on time as I hoped and desired.

Besides my advisor, I would like to sincerely thank the rest of my thesis committee members and invaluable professors Assoc. Prof. Dr. Ali Doğanaksoy, and Assist. Prof. Dr. Oğuz Yayla for their insightful comments, discussions and teachings.

My very profound appreciation also goes to my dear mother Leyla Altundağ, father İsmail Hakkı Altundağ and sister Asya Eda Altundağ for providing me with unfailing support, sacrifice, enthusiastic encouragement and endless patience throughout my years of study and my life in general. I'm so glad to have you in my life.

Finally, I am extremely grateful to my dear parents and friends for their love, prayers, caring and understanding to be able to complete this thesis successfully. This accomplishment would not have been possible without them. Thank you.

TABLE OF CONTENTS

ABSTRACT vii

ÖZ ix

ACKNOWLEDGMENTS xiii

TABLE OF CONTENTS xv

LIST OF TABLES xxi

LIST OF ABBREVIATIONS xxii

CHAPTERS

1 INTRODUCTION 1

2 PRELIMINARIES 3

2.1 RANDOM ORACLE MODEL 3

2.1.1 PROPERTIES OF IDEAL CRYPTOGRAPHIC HASH FUNCTIONS 4

2.1.1.1 Preimage Resistance 4

2.1.1.2 Second Preimage Resistance 5

2.1.1.3 Collision Resistance 5

2.2 QUANTUM RANDOM ORACLE MODEL 6

3 MULTIVARIATE PUBLIC-KEY CRYPTOSYSTEMS 9

3.1 MATSUMOTO-IMAI (MI) CRYPTOSYSTEM 9

3.2 LINEARIZATION EQUATIONS ATTACK 11

3.2.1 LINEARIZATION EQUATIONS ATTACK FOR THE SINGLE-BRANCH MI CRYPTOSYSTEM 12

3.2.1.1 CONSTRUCTION OF THE LINEARIZATION EQUATIONS 13

3.2.1.2 DIMENSION OF THE LINEARIZATION EQUATIONS SPACE FOR THE SINGLE-BRANCH MI CRYPTOSYSTEM 16

3.2.1.3 TOY EXAMPLE OF THE LINEARIZATION EQUATIONS ATTACK FOR THE SINGLE-BRANCH MI CRYPTOSYSTEM 17

3.2.2 LINEARIZATION EQUATIONS ATTACK FOR THE MULTIPLE-BRANCH MI CRYPTOSYSTEM 21

3.3 MATSUMOTO-IMAI VARIANTS 21

3.3.1 THE MINUS METHOD 22

3.3.2 FLASH AND SFLASH 24

3.3.3 THE PLUS METHOD 27

3.3.4 THE MATSUMOTO-IMAI-PLUS-MINUS PUBLIC-KEY CRYPTOSYSTEM 27

4 CRYPTOGRAPHIC PROTOCOLS 31

4.1 IDENTIFICATION SCHEMES 31

4.1.1 SECURITY PROPERTIES 32

4.1.1.1 Security Against Impersonation Under Passive Attacks (IMP-PA Security) 32

4.1.1.2 Key Relation 33

4.1.1.3 Key-One-Wayness (KOW) 33

4.1.1.4 Soundness (with soundness error κ) 33

4.1.1.5 Computational Honest Verifier Zero Knowledge (HVZK) 33

4.2 DIGITAL SIGNATURE SCHEMES 33

4.2.1 SECURITY PROPERTIES 35

4.2.1.1 Existential Unforgeability Under Adaptive Chosen Message Attacks (EU-CMA Security) 35

4.2.1.2 Key Only Attacks (KOA Security) 36

4.3 COMMITMENT SCHEMES 36

4.3.1 SECURITY PROPERTIES 37

4.3.1.1 Binding Property 37

4.3.1.2 Hiding Property 37

5 STRUCTURAL TOOLS OF THE MULTIVARIATE QUADRATIC DIGITAL SIGNATURE SCHEME (MQDSS) 39

5.1 MULTIVARIATE QUADRATIC PROBLEM 39

5.2 CANONICAL $2n + 1$-PASS IDENTIFICATION SCHEMES 41

5.2.1 CANONICAL 3-PASS IDENTIFICATION SCHEME . . 42

5.2.2 CANONICAL 5-PASS IDENTIFICATION SCHEME . . 43

5.2.3 SECURITY PROPERTIES OF $2n + 1$-PASS IDENTIFICATION SCHEMES 43

5.2.3.1 Special n-soundness 44

5.2.3.2 Special Soundness 44

5.2.3.3 $q2$-extractor 45

5.3 THE SAKUMOTO-SHIRAI-HIWATARI (SSH) 3-PASS AND 5-PASS IDENTIFICATION SCHEMES 45

5.4 THE FIAT-SHAMIR TRANSFORM 50

5.4.1 THE FIAT-SHAMIR TRANSFORM OF CANONICAL 3-PASS IDS 50

5.4.2 THE FIAT-SHAMIR TRANSFORM OF CANONICAL 5-PASS IDS 51

5.4.3 SECURITY ANALYSIS 53

6 MULTIVARIATE QUADRATIC DIGITAL SIGNATURE SCHEME (MQDSS) 55

6.1 MQDSS PRELIMINARY PARAMETERS AND FUNCTIONS . . . 55

6.2 GENERAL DESCRIPTION OF THE MQDSS 56

6.3 OPTIMIZED PARAMETER SETS 58

6.4 DETAILED DESCRIPTION OF THE MQDSS 60

6.4.1 PRELIMINARY FUNCTIONS [5] 60

6.4.2 DESCRIPTION OF THE ALGORITHMS: KGen, Sign, Vf 67

6.5 CLASSICAL AND QUANTUM ALGORITHMS FOR SOLVING THE MQ PROBLEM 69

6.5.1 CLASSICAL ALGORITHMS FOR SOLVING THE MQ PROBLEM 70

6.5.1.1 Exhaustive Search 70

6.5.1.2 The HybridF5 Algorithm 70

6.5.1.3 The BooleanSolve Algorithm 70

6.5.1.4 The Crossbread Algorithm 71

6.5.2 QUANTUM ALGORITHMS FOR SOLVING THE MQ PROBLEM 71

6.5.2.1 The MQ Oracle 72

6.5.2.2 The BooleanSolve Oracle 72

6.5.2.3 The HybridF5 Oracle 72

6.5.2.4 The Crossbread Oracle 72

6.6 SECURITY ANALYSIS 73

7 CONCLUSION 75

REFERENCES 77

LIST OF TABLES

TABLES

Table 5.1 Canonical 3-pass IDS [5] 42

Table 5.2 Canonical 5-pass IDS [5] 43

Table 5.3 The SSH 3-pass IDS [35] 48

Table 5.4 The SSH 5-pass IDS [35] 49

LIST OF ABBREVIATIONS

DDS	Digital Signature Scheme
EU-CMA	Existential Unforgeability Under Adaptive Chosen Message Attacks
HVZK	Honest Verifier Zero Knowledge
IDS	Identification Scheme
IMP-PA	Impersonation Under Passive Attacks
IP	Isomorphism of Polynomials
KOA	Key Only Attacks
KOW	Key-One-Wayness
MI	Matsumoto-Imai
MQ	Multivariate Quadratic
MQDSS	Multivariate Quadratic Digital Signature Scheme
NESSIE	New European Schemes for Signatures, Integrity, and Encryption
NIST	National Institute of Standards, and Technology
QROM	Quantum Random Oracle Model
ROM	Random Oracle Model
SSH	Sakumoto-Shirai-Hiwatari

CHAPTER 1

INTRODUCTION

At the present time, one of the most essential needs of the human being is the topic of security and this concept is becoming increasingly important together with the technological advances. That is, many areas of life such as communication, online shopping, Internet banking and health care systems require high level of security. The main tool so as to satisfy this is the existence of secure cryptographic protocols like identification schemes and digital signature schemes. Let's pay attention the **lock icon** on any browser address bar that indicates a secure connection. If that connection can not be trusted, anything that relies on Internet security will be rendered useless. This scenario is possible in the near future since a developing technology which is said to be quantum computing has a potential to crack the encryption which is the basis for secure Internet communications. Today secure communications rely on exchange of keys or secret codes to ensure that the parties are who they say are and to exchange messages that can not read by others. The same type of mechanisms are also used to guarantee that application patches and code updates are coming from legitimate sources.

In 1994 a mathematician named **Peter Shor** developed an algorithm which can break the security of key exchanges and digital signatures [38]. By using this algorithm, a quantum computer would be able to crack today's most sophisticated encryption in minutes [39]. On the other hand, all the normal computers in the world working together would take longer than the universe has existed, because quantum and traditional computers operate differently from each other. That is, quantum ones work at atomic level and are not affected from the physical restrictions over traditional computer chips. Furthermore, any quantum computer used qubits instead of bits of a qubit has ability to represent 0 and 1 simultaneously and very few qubits can speed up certain types of computation by an enormous amount. However, this new technology can break the encryption on which the world relies. That is, all current symmetric and asymmetric (public-key) cryptosystems which based on the hardness of the mathematical problems like factorization and discrete logarithm can be invalid.

As soon as quantum computers are able to break the encryption which is estimated to be less than 10 years away, the threats will be innumerable. For instance, a hacker could break a software update key and send fake updates or malware to your computer or store private encrypted information could esaily be read by anyone with access to a cloud quantum computer. As well as this, if we follow the advances in the technology, we can realize the competition

among the famous firms such as IBM, Microsoft, Intel and Google for generating quantum computers [2]. At this juncture, the need for the quantum resistant algorithms which is called as **post quantum cryptography** becomes inevitable. In return for this, National Institute of Standards and Technology (NIST) organized a competition for the quantum-resistant algorithms on 20th December, 2016 [25]. Then, the round 2 candidates were announced by NIST on 30th January, 2019 [3]. If we focus on the submissions in the portal of the NIST, we can learn the well known quantum-resistant cryptographic algorithms which consist of multivariate public-key, lattice-based, hashed-based, isogeny-based and code-based cryptosystems [25].

In this thesis, we analyze an algotihm whose name is the **Multivariate Quadratic Digital Signature Scheme** (**MQDSS**) which comes from the family of **multivariate public-key cryptosystems** and is one of the round 2 candidates of NIST [3] since the systems which rely on multivariate public-key polynomials are resistant against quantum computers [14]. Now the security is based on the intractability of the multivariate quadratic (MQ) problem and the hardness of the commitment schemes instead of the classical problems like factorization and discrete logarithm which can be solved in polynomial time with fast computing capability of the quantum computers.

The rest of the thesis is organized as follows. To begin with, the preliminary section which includes the Random Oracle Model (ROM) and the Quantum Random Oracle Model (QROM) that are used in the security proofs of cryptographic algorithms are described. Then we focus on what are the multivariate public-key cryptosystems more detailed. In this process, firstly Matsumoto-Imai cryptosystem and it's linearization equations attack for single and multiple-branch cases are studied. Secondly, our own specific toy example for illustrating the construction of both the single-branch Matsumoto-Imai cryptosystem and it's linearization equations attack are generated. Finally, Matsumoto-Imai variants which were developed so as to increase the security of the original MI cryptosystem are investigated. After that, cryptographic protocols which are identification schemes, digital signature schemes and commithment schemes are analyzed in general terms associated with their security properties.

From now on, we examine the structural tools of the MQDSS which are composed of the MQ problem, is the main security source of the MQDSS, canonical $2n+1$-pass identification schemes, Sakumato-Shirai-Hiwatari (SSH) 3-pass and 5-pass identification schemes and Fiat-Shamir transform in common with their security properties. Finally, the detailed analysis of the MQDSS algoritm is made. In order to do this, preliminary parameters and functions, general description, optimized parameter sets, the detailed description and the the classical and the quantum algorithms for solving the MQ problem of the MQDSS are scrutinised. Then by taking all these points into consideration, we pass on to the security analysis of the MQDSS.

CHAPTER 2

PRELIMINARIES

Since the aim of the thesis is the analysis of the Multivariate Quadratic Digital Signature Scheme which belongs to the family of multivariate public-key cryptosystems, in the ongoing process, some examination tools whose names are the random oracle model and the quantum random oracle model, are needed. They are utilized both as a design strategy and as a proof strategy. Because, at the end of the thesis, the thing which is wanted to be said is that MQDSS is secure both in the random and the quantum random oracle model, these two concepts should be known at the beginning as a preliminary part of the thesis.

2.1 RANDOM ORACLE MODEL

Random oracle model (ROM) is a mathematical model which requires to have an ideal cryptographic hash function that match any input to that hash function with a uniform distribution. This model was introduced by Bellare and Rogaway.

According to the above definition, lets assume we have an ideal cryptographic hash function $h : \mathcal{X} \longrightarrow \mathcal{Y}$ and we don't have any access to a formula or an algorithm, which determines the values of h. That's why, it is possible to think h as a some kind of **black box**. As we come to the application of ROM, there exists one thing which we need to assume: **honest players** who are the players act upon defined cryptographic protocols. In the process of constructing a digital signature scheme, generators prefer to sign the hash of message rather than the message itself. Also adversary is interested in the hash of his or her guesses. Although the system enables everyone to compute the hash, hashing is done by a new magical party called the "**oracle**" whose operation can be regarded as a **black box**.

h behaves like an oracle that is we can query h and it will give the answer back and that answer will be correct. However, if we keep on querying h, we will not be able to get any pattern, and no matter how many times we have queried h before, the next query will behave randomly in other words, the next answer to the next query will seem to be coming from the space Y equiprobably. If we assume this property on h, then this means that we are following the ROM.

More formally, suppose that $h \in F^{X,Y}$ is chosen randomly, satisfies ROM and we have picked up certain points $\{x_1, x_2, \ldots, x_Q\}$ from X where Q is the number of queries that we have made. Let the set of those points be $X_0 = \{x_1, x_2, \ldots, x_Q\}$ such that $X_0 \subseteq X$. Assume that we have queried the oracle for h at those points. Thus, we will have the several accesses to several valid pairs like this: $\{(x_1, h(x_1)), (x_2, h(x_2)), \ldots, (x_Q, h(x_Q))\}$.

Now if X has ROM, then for any $x \in X \setminus X_0, \quad \Pr[h(x) = y] = \frac{1}{M}, \quad |Y| = M$ for all $y \in Y$ even after having the knowledge of these valid pairs. That's to say the hash function doesn't leak any information.

Furthermore, in the process of explaining ROM, to describe what is random oracle has an important role for understanding the whole concept. When any domain set $\mathcal{D}$ and the range set $\mathcal{R}$ are given, a random oracle is a randomly chosen function among all functions go from $\mathcal{D}$ to $\mathcal{R}$ and it reminds us a look up table whose first column represents the input and the second represents the corresponding output. The reason for this is that whenever the same input is given to the random oracle, the same output has to be returned. The expectation from this look up table is including all input and corresponding output sets that random oracle needs to compute. However, so as to store this kind of look up table, we need exponential space. Therefore, rather than considering a full table from the beginning, let's assume that we have a completely empty table and whenever an input value needs to be calculated by the random oracle, this calculation is done after then the input and output values are entered to the table. However, for each question asked to the oracle, firstly whether this question has been asked before is checked by looking at the previous lines of the table and later on new entries are made to the table only for previously unasked values. Thus, the problem of exponential space will disappear. In this manner, random oracle can be utilized both as a **design strategy** and as a **proof strategy** in the course of creating a scheme protocol.

Let's focus on the word of "**random**". In the real world, it is impossible to construct a genuine random function, so we are able to have only access to the pseudo random functions. For this, ideal cryptographic hash functions which have to satisfy three properties: preimage resistance, second preimage resistance and collision resistance are required to use in ROM.

2.1.1 PROPERTIES OF IDEAL CRYPTOGRAPHIC HASH FUNCTIONS

2.1.1.1 Preimage Resistance

When we analyze results of the algorithm FIND-PREIMAGE(h, y, Q), if for all pre-specified outputs, it is computationally infeasible to return something out of failure, then our hash function h will be called as **preimage resistant**.

Algorithm 1 FIND-PREIMAGE(h, y, Q) [34]

choose any $X_0 \subseteq X, \quad |X_0| = Q$
for each $x \in X_0$ **do**
    if $h(x) = y$ **then return** x
    end if
end for
return failure

2.1.1.2 Second Preimage Resistance

Algorithm 2 FIND-SECOND-PREIMAGE(h, x, Q) [34]

$y \leftarrow h(x)$
choose any $X_0 \subseteq X \setminus \{x\}, \quad |X_0| = Q - 1$
for each $x_0 \in X_0$ **do**
    if $h(x_0) = y$ **then return** x_0
    end if
end for
return failure

When we analyze results of the algorithm FIND-SECOND-PREIMAGE (h, x, Q), if for any specified input i.e., given x, it is computationally infeasible to return something out of failure, then our hash function h will be called as **second preimage resistant**.

2.1.1.3 Collision Resistance

Algorithm 3 FIND-COLLISION(h, Q) [34]

choose any $X_0 \subseteq X, \quad |X_0| = Q$
for each $x \in X_0$ **do**
    $y_x \leftarrow h(x)$
    if $y_x = y_{x'}$ for some $x' \neq x$ **then return** (x, x')
    end if
end for
return failure

When we analyze results of the algorithm FIND-COLLISION (h, Q), if for any specified input i.e., given x, it is computationally infeasible to return something out of failure, then our hash function h will be called as **collision resistant**.

2.2 QUANTUM RANDOM ORACLE MODEL

Quantum random oracle model (QROM) is a random oracle model in the quantum world. That's why first let us know the quantum world and quantum cryptography. In classical cryptography, the thing which makes the codes practically unbreakable is the tremendous amount of time and computing power. Therefore, if all the world's personal computers were working nonstop to try and break the code that keeps your information safe, it would take them several times of the universe's age. On the other hand, the modern computer may soon be replaced by the **quantum computer** and what takes a modern computer billions of years could be done by a quantum computer in days or even hours. Since researchers are ready for this improvement, they've come up with a new type of cryptography that's not just hard to break but impossible to break. It's called **quantum cryptography** and what makes it so powerful is that instead of math it relies on the laws of physiscs. The one we have to pay particular attention among these laws is the **Heisenberg Uncertainty Principle** which says that there is no chance to know absolutely everything about the state of a quantum particle because nature keeps some things hidden.

In the process of sending an encrypted message to a receiver over a secure line by using quantum cryptography, the key is a stream of light particles or photons whose property called spin can be changed when it passes through any one of the four kind of filters: vertical, horizontal and two diagonals. As we need to explain a little bit more, let's lump the filters into two groups: the diagonal scheme and the rectilinear scheme. Then to translate a photon spin into a key, we have a matching such that a photon with vertical or bottom-left to top-right diagonal spin means one, and horizantal or bottom-right to top-left diagonal spin means zero. Over the course of sending photons to the receiver, the sender stands to switch between filters at random. Now here's where the Heisenberg uncertainty princible becomes significant since the only way an adversary can measure a photon's spin is by passing it through a filter. If the adversary measures a photon has a specific shape with its right filter, then the guess will be correct and according to the shape the attacker notes down its corresponding bit value 1 or 0. Otherwise, the photon spin will be altered as it passes through and the attacker incorrectly reads the corresponding bit value. Unless the adversary knows beforehand which filter to use, then he or she runs a pretty big risk of changing the spin. Since the sender is switching between filters at random, the attacker will get it wrong about half of the time. For this reason, the laws of quantum physics prevent the adversary from knowing the key.

If we go one step further, the thing which we have to focus on is the quantum adversary who is an adversary with a quantum computer. As we recall from the previous section, one of the reasons for using ROM is providing security proofs in the field of practical cryptographic schemes. However, when it comes to quantum adversaries who have the ability for obtaining the outputs of hash functions over many inputs' superpositions, ROM can be no longer effective in security proofs. That's why a new model whose name is **quantum random oracle model** was developed and in QROM, hash functions are built as a quantumly accessible random oracle. [22] That is, a quantum adversary whose computational resources are really

different from those belong to classical adversary is able to make a quantum circuit which enables to analyze the hash oracle in superposition. This quantum circuit is a unitary mapping U_H such that $U_H : |x\rangle |y\rangle \longmapsto |x\rangle |y \oplus H(x)\rangle$ and it is possible to extend U_H for states' superpositions. Therefore, if we want to model abilities of a quantum adversary who implements any hash function, we should first tap into the random oracle by using U_H then enable to quantum access to U_H [16].

CHAPTER 3

MULTIVARIATE PUBLIC-KEY CRYPTOSYSTEMS

In multivariate public-key cryptosystems, post-quantum identification and signature algorithms rely on the hardness of large systems of the multivariate quadratic equations so called **multivariate quadratic (MQ) problem** to provide resistance against attacks. The first occurrence of these algorithms started with the aim of finding a new alternative to current standard schemes (e.g., RSA) which use the hardness of the factoring or discrete logarithm problem for public-key identifications and digital signatures.

Now the question is: **What is an MQ problem?**

MQ problem is a problem which requires to solve a multivariate quadratic equation system over a finite field. To be able to explain the root idea behind the MQ problem, the **Matsumoto-Imai cryptosystem** which was proposed by Matsumoto and Imai in 1988, is described. After that, linearization equations attack which broke the Matsumoto-Imai cryptosystem is identified [28]. Then, our own specific toy example for illustrating the construction of both the single-branch Matsumoto-Imai cryptosystem and its linearization equations attack, is illustrated. Finally, Matsumoto-Imai variants which were developed so as to increase the security of the original MI cryptosystem, are given.

3.1 MATSUMOTO-IMAI (MI) CRYPTOSYSTEM

The main idea of the Matsumoto-Imai cryptosystem is using both the vector space and the hidden field structure of k^n, k is a finite field, in the same time [24]. In other words, Matsumoto and Imai search invertable maps on K, a degree n field extension of k instead of looking for the maps over k^n (vector space) directly. [28] Let's define the construction of this cryptosystem.

KEY GENERATION

Let

- $k = \mathcal{F}_q$, where $q = 2^n \quad (k \to \text{finite field, char}\,(k) = 2)$,
- $g\,(x) \in k\,[x]$ be an irreducible polynomial whose degree n,
- $K = k\,[x]\,/g\,(x)$ be a field which is a degree n extension of k and
- $\phi : K \to k^n$ be the standard k-linear isomorphism between K and k^n given by:

$$\phi\left(a_0 + a_1 x + \cdots + a_{n-1} x^{n-1}\right) = (a_0, a_1, \ldots, a_{n-1})\,. \tag{3.1}$$

Suppose that ϕ is a k-linear map and k is a subfield of K. Select θ such that $0 < \theta < n$ and $\gcd\left(q^\theta + 1, q^n - 1\right) = 1$. Let $\tilde{F}$ be a map over K such that $\tilde{F}\,(X) = X^{1+q^\theta}$. If θ is chosen above, then MI Cryptosystem guarantees that $\tilde{F}$ will be an invertable map and $\tilde{F}^{-1}\,(X) = X^t$, because $t\left(1 + q^\theta\right) \equiv 1 \mod (q^n - 1)$, t is an integer. Let F be the map over k^n such that:

$$F\,(x_1, \ldots, x_n) = \phi \circ \tilde{F} \circ \phi^{-1}\,(x_1, \ldots, x_n) = (f_1, \ldots, f_n)\,, \tag{3.2}$$

where $f_1, \ldots, f_n \in k\,[x_1, \ldots, x_n]$. Now assume that L_1 and L_2 are two invertable affine transformations over k^n. Finally, let's define the map $\bar{F}$ over k^n such that:

$$\bar{F}\,(x_1, \ldots, x_n) = L_1 \circ F \circ L_2\,(x_1, \ldots, x_n) = \left(\bar{f}_1, \ldots, \bar{f}_n\right), \tag{3.3}$$

where $\bar{f}_1, \ldots, \bar{f}_n \in k\,[x_1, \ldots, x_n]$.

The Public-Key

The public key of the single-branch MI cryptosystem includes:

1. The field: k , the addition and multiplication tables of it,
2. The set of n polynomials: $\bar{f}_1, \ldots, \bar{f}_n \in k\,[x_1, \ldots, x_n]$.

The Private-Key

The private key of the single-branch MI cryptosystem includes:

1. The two invertible affine transformations: L_1 and L_2,
2. Parameter: θ.

ENCRYPTION

Given a plaintext message $(x'_1, \ldots, x'_n)$, the corresponding ciphertext: $(y'_1, \ldots, y'_n)$ where $y'_i = \bar{f}_i(x'_1, \ldots, x'_n)$ for $i = 1, \ldots, n$.

DECRYPTION

Given a ciphertext $(y'_1, \ldots, y'_n)$, the decryption process is the following computation:

$$\begin{aligned} &\bar{F}^{-1}(y'_1, \ldots, y'_n) = L_2^{-1} \circ F^{-1} \circ L_1^{-1}(y'_1, \ldots, y'_n) = \cdots \\ &\cdots = L_2^{-1} \circ \phi \circ \tilde{F}^{-1} \circ \phi^{-1} \circ L_1^{-1}(y'_1, \ldots, y'_n). \end{aligned} \tag{3.4}$$

We need to know that due to the construction of the system , $\bar{F}^{-1}$'s components will have very high degree. That's why, the decryption process will be seperated into 3 parts:

$$1.\ (z'_1, \ldots, z'_n) = L_1^{-1}(y'_1, \ldots, y'_n), \tag{3.5}$$

$$2.\ (\bar{z}_1, \ldots, \bar{z}_n) = \phi \circ \tilde{F}^{-1} \circ \phi^{-1}(z'_1, \ldots, z'_n), \tag{3.6}$$

$$3.\ (x'_1, \ldots, x'_n) = L_2^{-1}(\bar{z}_1, \ldots, \bar{z}_n). \tag{3.7}$$

The description above only belongs to the **single-branch MI**, but in practical application, the composition of several single-branch crytosystems, so called **multiple-branch MI**, is preferred. In the application process of the multiple-branch MI, the input which will be encrypted is partitioned first, then each part is ciphered by using the single-branch MI. After that the corresponding parts are combined to create a single ciphertext. Since it is necessary to hide the branches before and after the ciphering, invertable affine transformations are applied to the input and output [12]. This alternative cryptosystem was a candidate for the Japanese Government's security standards [32]. However, in 1995, Jacques Patarin broke the system by means of an algebraic attack whose name is **linearization equations attack**.

3.2 LINEARIZATION EQUATIONS ATTACK

Let's start with the description of the linearization equations attack for the single-branch MI cryptosystem. After that, its generalization to multiple branch case is identified. [28] Since the main tool of the linearization equations attack is the linearization equation, at the beginning, it's definition is given.

Definition 1. *Let $\mathcal{P} = \{p_1, \ldots, p_m\}$ be any set of m polynomials in $k[x_1, \ldots, x_n]$. A **linearization equation** for $\mathcal{P}$ is any polynomial in $k[x_1, \ldots, x_n, y_1, \ldots, y_m]$ of the form*

$$\sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij}x_i y_j + \sum_{i=1}^{n} b_i x_i + \sum_{j=1}^{m} c_j y_j + d, \tag{3.8}$$

*so that it gives the zero function in $k[x_1, \ldots, x_n]$ if we substitute in p_j for y_j, for $j = 1, \ldots, m$. In other words, a **linearization equation** is any equation in $k[x_1, \ldots, x_n]$ of the form*

$$\sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij}x_i p_j(x_1, \ldots, x_n) + \sum_{i=1}^{n} b_i x_i + \sum_{j=1}^{m} c_j p_j(x_1, \ldots, x_n) + d = 0 \tag{3.9}$$

such that it is satisfied for all $(x'_1, \ldots, x'_n) \in k^n$.

Let's move on to the definition of **linearization equation space** for a given $\mathcal{P}$.

Definition 2. *Assume that* $\mathcal{P} = \{p_1, \ldots, p_m\}$ *which was defined in 1, is given. Its set of all linearization equations that generates a* k*-vector space, is called as a* ***linearization equations space****.*

According to Patarin, so as to attack the Matsumoto-Imai cryptosystems, the thing which will be needed is finding the linearization equations space for the components of $\bar{F}$ which are the public-keys of this cryptosystem. Since $\bar{F} = \left\{\bar{f}_1, \ldots, \bar{f}_n\right\}$ was defined in the previous section, it is possible to explain the linearization equations attack over $\bar{F}$. That's why, assume that for the components of $\bar{F}$, there exists a linearization equation whose form is suitable to (3.8). Then, so as to attack the system, when a ciphertext $(y'_1, \ldots, y'_n)$ is given, y'_i should be written instead of $\bar{f}_i$ inside of this linearization equation. Thus, the corresponding linear equation which only consists of the variables $x_1, \ldots, x_n$, can be obtained. At this point, the important thing is the existence of the plaintext among its solution set.

Now, the aim is to derive sufficient number of linear equations from the existing linearization equations so that the unique solution of the system becomes the plaintext. In accordance with the linearization equations attack, in the case of not finding the plaintext directly from the obtained linear equations, they should be substituted into the public key equations $(f_i(x_1, \ldots, x_n) = y'_i \quad \forall i = 1, \ldots, n)$ for decreasing the number of unknown variables. In this process, for the existence of a feasible attack against the plaintext, there should be an adequate number of linearly independent linear equations. Thus, the system can be either solved directly or made easier to solve. In the attack, deciding the number of linearly independent linear equations is quite necessary for the feasibility analysis.

3.2.1 LINEARIZATION EQUATIONS ATTACK FOR THE SINGLE-BRANCH MI CRYPTOSYSTEM

In the case of a single branch MI, the plaintext is encrypted directly without partitioning or permutating. That's why, in this situation, it is possible to directly focus on the linearly independent linear equations which are needed for the lineariazation equations attack against the single-branch MI cryptosystem. Since deciding the number of them is necessary, [28] let's give a theorem which is related with the lower bound of this number.

Theorem 1. *Assume that there is a single-branch MI cryptosystem whose public-key is the following set:* $\bar{F} = \left\{\bar{f}_1, \ldots, \bar{f}_n\right\}$*. Let* $\bar{\mathcal{L}}$ *be the linearization equations space of this set. If for a given ciphertext* $Y' = (y'_1, \ldots, y'_n) \in k^n$*,* $\bar{\mathcal{L}}_{Y'}$ *is obtained as a linearization equations space by putting* y'_i *instead of* y_i *for all* $i = 1, \ldots, n$*, in each equation of* $\bar{\mathcal{L}}$*, then the lower bound of the linearly independent linear equations in the space* $\bar{\mathcal{L}}_{Y'}$ *is the following:*

$$n - \gcd(n, \theta) \geq \frac{2n}{3}. \tag{3.10}$$

Only in the case of the existence of trivial equations, there will be an exceptional situation such that $L^{-1}(Y') = (0, \ldots, 0)$.

There exist two lemmas which are needed to see the correctness of the theorem (1). Let's give them:

Lemma 1. *Assume that there is a single-branch MI cryptosystem whose public-key is derived from the following construction:* $\bar{F} = L_1 \circ F \circ L_2$. *Let* $\mathcal{L}$ *and* $\bar{\mathcal{L}}$ *be the linearization equations spaces for the sets* $\{f_1, \ldots, f_n\}$ *and* $\left\{\bar{f}_1, \ldots, \bar{f}_n\right\}$ *respectively. Then the dimensions of these* k*-vector spaces will be the same and it's denoted by:*

$$\dim_k \mathcal{L} = \dim_k \bar{\mathcal{L}}. \tag{3.11}$$

Notice that: So as to show the equality for the dimensions of the k-vector spaces in the lemma (1), the technique is associated with the finding a bijection between these spaces. That's why, first suppose the transformation L_2 as an identity, then suppose the transformation L_1 as an identity. Since they are invertable affine transformations, the one which is not assumed as an identity, can be defined together with the help of unknown coefficients and variables. After that combine these two cases and get the answer. In this process, the thing which is needed to do, is starting from the linearization equation for one of the sets $\{f_1, \ldots, f_n\}$ or $\left\{\bar{f}_1, \ldots, \bar{f}_n\right\}$, with the form of (3.8) and substituting the defined transformation inside of this equation, then trying to pass on to the linearization equation of the other set via the arrangements over this linearization equation. For the two-sidedness, make this operation for both of the sets. Thus, the bijection between the linearization equations can be obtained, which means the equality of their dimensions. .

Lemma 2. *Let* $\mathcal{L}$ *and* $\bar{\mathcal{L}}$ *be the linearization equations spaces for the sets* $\{f_1, \ldots, f_n\}$ *and* $\left\{\bar{f}_1, \ldots, \bar{f}_n\right\}$ *respectively, then let* $Z = L_1^{-1}(Y') = (z_1, \ldots, z_n)$. *If for a given ciphertext* $Y' = (y'_1, \ldots, y'_n)$, $\bar{\mathcal{L}}_{Y'}$ *is obtained as a linearization equations space by putting* y'_i *instead of* y_i *in each equation of* $\mathcal{L}$ *and* $\mathcal{L}_Z$ *is obtained as a linearization equations space by putting* z_i *instead of* y_i *in each equation of* $\bar{\mathcal{L}}$ *for all* $i = 1, \ldots, n$, *then the dimensions of these* k*-vector spaces will be the same and it's denoted by:*

$$\dim_k \mathcal{L}_Z = \dim_k \bar{\mathcal{L}}_{Y'}. \tag{3.12}$$

As such in the lemma (1), there is also a bijection between the two linearization equations spaces in the lemma (2) with the same reason.

3.2.1.1 CONSTRUCTION OF THE LINEARIZATION EQUATIONS

Before explaining the two main approaches which are benefited for generating linearization equations, let's give the logic behind the construction of these equations from the Patarin's

point of view. So, let K be as described in the section (3.1) and $X, Y \in K$ such that $Y = \tilde{F}(X) = X^{q^\theta+1}$. Then take the $\left\{q^\theta - 1\right\}^{th}$ power of the both sides and get:

$$Y^{q^\theta-1} = (X^{q^\theta+1})^{q^\theta-1} = X^{(q^\theta+1)(q^\theta-1)} = X^{q^{2\theta}-1}. \tag{3.13}$$

Now mutiply both sides with XY and get:

$$XY^{q^\theta} = X^{q^{2\theta}}Y \quad \Rightarrow \quad XY^{q^\theta} - X^{q^{2\theta}}Y = 0. \tag{3.14}$$

Define $\tilde{R}(X, Y) \in K[X, Y]$ such that

$$\tilde{R}(X, Y) = XY^{q^\theta} - X^{q^{2\theta}}Y. \tag{3.15}$$

After that, as in the case with $F = \phi \circ \tilde{F} \circ \phi^{-1}$, define R such that

$$R = \phi \circ \tilde{R} \circ (\phi^{-1} \times \phi^{-1}). \tag{3.16}$$

Thus, it will be possible to derive n linearization equations for the vectors $f_1, \ldots, f_n$ with the form of (3.8) from $R(x_1, \ldots, x_n, y_1, \ldots, y_n)$. According to the definition (1), if f_i is written instead of y_i for $i = 1, \ldots, n$, in each linearization equation, then the zero polynomial will be obtained in $k[x_1, \ldots, x_n]$. Let's focus on the number of linearly independent linear equations that will be derived from R for a given particular ciphertext $Y' = (y'_1, \ldots, y'_n)$. Let $(x'_1, \ldots, x'_n) = F^{-1}(y'_1, \ldots, y'_n) \in k^n$ be the corresponding plaintext, $Y' = \phi^{-1}(y'_1, \ldots, y'_n)$ and $X' = \phi^{-1}(x'_1, \ldots, x'_n)$. At this point, in the case of $Y' \neq 0$, one of the solutions for the equation $X^{q^{2\theta}}Y' = X(Y')^{q^\theta}$ has to be X'. Via dividing both sides by XY', the following equation:

$$X^{q^{2\theta}-1} = (Y')^{q^\theta-1} \tag{3.17}$$

will be attained. The number of solutions in the field K, for the equation (3.17) is at most $\gcd(q^{2\theta} - 1, q^n - 1)$. Since $\gcd(q^\theta + 1, q^n - 1) = 1$, according to the section (3.1),

$$\gcd(q^{2\theta} - 1, q^n - 1) = \gcd(q^\theta - 1, q^n - 1). \tag{3.18}$$

That's why, together with the trivial solution, the upper bound for the total number of solutions of the equation (3.17) is $\gcd(q^\theta - 1, q^n - 1) + 1$. In order to find a more precise result, it can be benefited from the following lemma.

Lemma 3. *For any given $a, b \in \mathbb{N}^+$,*

$$\gcd(q^a - 1, q^b - 1) = q^{\gcd(a,b)} - 1. \tag{3.19}$$

In the case of applying the lemma (3) to the above result, the upper bound for the total number of linear equations will be the following:

$$\gcd(q^\theta - 1, q^n - 1) + 1 = q^{\gcd(\theta,n)} - 1 + 1 = q^{\gcd(\theta,n)}. \tag{3.20}$$

Now let λ be the number of linearly independent ones among all linear equations. So it can be assumed that the number of linear equations of the corresponding system is equal to $q^{n-\lambda}$.

Since the upper bound for the total number of all these equations is $q^{\gcd(\theta,n)}$, the following inequality $q^{n-\lambda} \leq q^{\gcd(\theta,n)}$ and thereupon the consequent condition $\lambda \geq n - \gcd(\theta, n)$ will be obtained. Then, let's try to find the largest value of $\gcd(\theta, n)$. With regard to the definition of the greatest common divisor, $\gcd(\theta, n)$ can not be larger than the following three values: n, $\frac{n}{2}$ and $\frac{n}{3}$. When these values are analyzed, $\gcd(\theta, n)$ can not be equal to the first two values. For the former case, since $0 < \theta < n$, $\gcd(\theta, n) \neq n$. For the latter case, if $\gcd(\theta, n)$ is assumed to be $\frac{n}{2}$, then θ has to be also $\frac{n}{2}$ because $\theta < n$. In this case,

$$\gcd(q^{\theta} + 1, q^n - 1) = \gcd(q^{\frac{n}{2}} + 1, q^n - 1) = q^{\frac{n}{2}} + 1 > 1 \tag{3.21}$$

creates a contradiction with the condition $\gcd(q^{\theta} + 1, q^n - 1) = 1$ which is required for the invertability of $\tilde{F}$ over the field K, in the section (3.1). Therefore, the value of $\gcd(\theta, n)$ can not be larger than $\frac{n}{3}$. Since $\lambda \geq n - \gcd(\theta, n)$ and $\gcd(\theta, n) \leq \frac{n}{3}$, the lower bound for the number of linearly independent linear equations will be at least:

$$\lambda = n - \gcd(\theta, n) \geq n - \frac{n}{3} = \frac{2n}{3}. \tag{3.22}$$

Thus, the proof of the following theorem was done. Let's give the theorem:

Theorem 2. *Let $\mathcal{L}$ be the linearization equations space for the vectors $f_1, \ldots, f_n$. If for a given ciphertext $Y' = (y'_1, \ldots, y'_n) \in k^n$, $\mathcal{L}_{Y'}$ is obtained as a linearization equations space by putting y'_i instead of y_i for all $i = 1, \ldots, n$ in each equation of $\mathcal{L}$, then the lower bound for $\dim_k \mathcal{L}_{Y'}$ is at least:*

$$n - \gcd(\theta, n) \geq \frac{2n}{3} \tag{3.23}$$

except when $Y' = (0, \ldots, 0)$.

Furthermore, if we combine the theorem (2) and the lemma (2), then this combination gives the proof of the theorem (1). However, in the theorem (1), there is an exception which is $\mathcal{L}_1^{-1}(Y') = (0, \ldots, 0)$. As well as this, it is possible to obtain trivial linear equations, $0 = 0$ from the linearization equations. Since the number of linearly independent linear equations is at least $n - \gcd(\theta, n)$, there will be an enough amount of linear equations for cracking the system in the case of $\gcd(\theta, n) = 1$.

Another important point is the level of security for the single-branch MI cryptosystem due to the lower bound for the number of linearly independent linear equations which will be able to obtained. Since this lower bound is equal to $\frac{2n}{3}$, this means that at least $\frac{2n}{3}$ linear equations which are satisfied by the plaintext for a given ciphertext, can be found, and thus $\frac{2}{3}$ of the information for this system can be attained. In the same time, when $\frac{2n}{3}$ linear equations are acquired, they can be utilized for eliminating $\frac{2}{3}$ of the variables of the public equations which are generated from the ciphertext and the public key. By doing this, the system will become much more solvable.

Now let's focus on the two different approaches which are ground on plaintext-ciphertext pairs and the structure of polynomial functions respectively, so as to generate linearization equations.

1) Plaintext-Ciphertext Pairs

Since the main tool of this approach is the public key, by using it, lots of plaintext-ciphertext pairs can be obtained. Then, the thing that is needed to do, for any plaintext-ciphertext pair which is given by $\bar{F}(x'_1, \ldots, x'_n) = (y'_1, \ldots, y'_n)$, is putting x'_i and y'_i instead of x_i and y_i respectively in the generic linearization equation with the form of (3.8). By doing this, for each pair, one linear equation which has $(n+1)^2$ unknowns, will be generated. Due to this number, approximately $(n+1)^2$ plaintext-ciphertext pairs should be chosen for solving the resulting system.

At this point, let's analyze the total cost. To begin with, it is possible to divide this cost into two parts. The former part is the computation of the $(n+1)^2$ plaintext-ciphertext pairs, and its complexity is equal to $O(n^4)$. The latter part is solving the set of $(n+1)^2$ linear equations each of which contains $(n+1)^2$ unkonwns, and its complexity is equal to $O(n^6)$.

2) Structure of Polynomial Functions

In this approach, the starting point is the generic linearization equation for the vectors $\bar{f}_1, \ldots, \bar{f}_n$ in the following form:

$$\sum\sum a_{ij}x_iy_j + \sum b_ix_i + \sum c_jy_j + d = 0, \tag{3.24}$$

where the coefficients $a_{ij}, b_i, c_j, d \in k$. Then, for an arrangement, the quadratic public equations which correspond to the vectors $\bar{f}_1, \ldots, \bar{f}_n$, should be placed into the equation (3.24) instead of $y_1, \ldots, y_n$. Then, the resulting equation will have the following form:

$$\sum\sum \alpha_{ijl}x_ix_jx_l + \sum \beta_{ij}x_ix_j + \sum \gamma_ix_i + \delta = 0, \tag{3.25}$$

whose coefficients $\alpha_{ijl}, \beta_{ij}, \gamma_i, \delta$ are linear functions which are derived from the coefficients a_{ij}, b_i, c_j, d. After that each of $\alpha_{ijl}, \beta_{ij}, \gamma_i, \delta$ will be equalized to zero. Thus, $\frac{(n+1)(n+2)(n+3)}{6}$ linear equations with the $(n+1)^2$ unknown coefficients a_{ij}, b_i, c_j, d will be obtained for $q > 2$. Thus, the problem is transformed into the problem of solving the set of $\frac{(n+1)(n+2)(n+3)}{6}$ linear equations with $(n+1)^2$ unknown variables. However, there is no need to use all of these linear equations. Since the number of unknown parameters is equal to $(n+1)^2$, using approximately $(n+1)^2$ linear equations will be enough. In this process, knowing the dimension of the linearization equations space can be quite beneficial. Because, according to this dimension, the decision of adding equations to the system can be made for finding the right solution space. The total cost of this approach is derived from solving the set of $(n+1)^2$ linear equations with $(n+1)^2$ unknowns, and it's complexity is equal to $O(n^6)$.

3.2.1.2 DIMENSION OF THE LINEARIZATION EQUATIONS SPACE FOR THE SINGLE-BRANCH MI CRYPTOSYSTEM

In the case of a single-branch MI cryptosystem, so as to calculate the dimension of a linearization equations space, the following theorem can be used [11, 28].

Theorem 3. *Let $\mathcal{L}$ be the linearization equations space for the vectors $\bar{f}_1, \ldots, \bar{f}_n$ which are the components of the Matsumoto-Imai map $\bar{F}$. Since the map $\bar{F}$ is invertable, it is possible to assume $\theta \neq n/2$.*

If $q > 2$, then

$$\dim_k \mathcal{L} = \begin{cases} 2n/3, & \text{if } \theta = n/3,\ 2n/3; \\ n, & otherwise. \end{cases} \tag{3.26}$$

If $q = 2$ and $\theta = n/3,\ 2n/3$, then

$$\dim_k \mathcal{L} = \begin{cases} 7, & \text{if } n = 6,\ \theta = 2,\ 4; \\ 8, & \text{if } n = 3,\ \theta = 1,\ 2; \\ 2n/3, & otherwise. \end{cases} \tag{3.27}$$

If $q = 2$ and $\theta \neq n/3,\ 2n/3$, then

$$\dim_k \mathcal{L} = \begin{cases} 10, & \text{if } n = 4,\ \theta = 1,\ 3; \\ 2n, & \text{if } \theta = 1,\ n-1,\ (n \pm 1)/2; \\ 3n/2, & \text{if } \theta = (n \pm 2)/2; \\ n, & otherwise \end{cases} \tag{3.28}$$

3.2.1.3 TOY EXAMPLE OF THE LINEARIZATION EQUATIONS ATTACK FOR THE SINGLE-BRANCH MI CRYPTOSYSTEM

Let $k = GF(2^2)$ be a finite field which has $q = 2^2 = 4$ factors. Through the field element α that satisfies the equation $\alpha^2 + \alpha + 1 = 0$, the multiplicative group for the nonzero factors of k can be produced. These elements are represented with the set $\{0, 1, \alpha, \alpha^2\}$ whose construction is defined with the following addition and multiplication tables.

$+$	0	1	α	α^2
0	0	1	α	α^2
1	1	0	α^2	α
α	α	α^2	0	1
α^2	α^2	α	1	0

$*$	0	1	α	α^2
0	0	0	0	0
1	0	1	α	α^2
α	0	α	α^2	1
α^2	0	α^2	1	α

Next, we select $n = 3$ and $g(x) = x^3 + x + 1$ which is an irreducible polynomial in $k[x]$. Set $K = k[x]/(x^3 + x + 1)$. Since $0 < \theta < n$, then $\theta = 1$ or $\theta = 2$. In this example we will use $\theta = 2$. The map $\tilde{F}$ and its inverse is the following:

$$\tilde{F}(X) = X^{1+4^2}, \quad \tilde{F}^{-1}(X) = X^{26}. \tag{3.29}$$

We choose the invertible affine transformations: L_1, L_2:

$$L_1(x_1, x_2, x_3) = \begin{pmatrix} \alpha^2 & \alpha & \alpha \\ \alpha & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \\ \alpha \end{pmatrix},$$

$$L_2(x_1, x_2, x_3) = \begin{pmatrix} 1 & 0 & \alpha \\ 0 & 1 & \alpha \\ 1 & \alpha & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} \alpha \\ \alpha^2 \\ \alpha^2 \end{pmatrix}.$$

Now we will find the public key components in terms of the following variables x_1, x_2, x_3 by using:

$$\bar{F}(x_1, x_2, x_3) = L_1 \circ \phi \circ \tilde{F} \circ \phi^{-1} \circ L_2(x_1, x_2, x_3). \tag{3.30}$$

So first we need to compute:

$$\phi^{-1} \circ L_2(x_1, x_2, x_3) = (\alpha + x_1 + \alpha x_3) + (\alpha^2 + x_2 + \alpha x_3)x + (\alpha^2 + x_1 + \alpha x_2)x^2. \tag{3.31}$$

If the above polynomial is denoted by X, then $\tilde{F}(X) = X^{1+4^2} = X.X^{16}$. Since the finite field k which we are working on, has the characteristic 2, there are no degrees higher than 2 in $\tilde{F}(X)$. After the polynomial multiplication, x will be reduced according to the irreducible polynomial $g(x)$, and α, x_1, x_2, x_3 will be reduced according to the tables. After the reductions, $\tilde{F}(X)$ is found to be:

$$\begin{aligned} &1 + \alpha^2 x_1 + \alpha x_2 + x_3 + x_1 x_2 + \alpha x_1 x_3 + \alpha^2 x_2 x_3 \\ &\quad + (\alpha + \alpha x_1 + x_2 + \alpha^2 x_3 + x_1^2 + \alpha^2 x_1 x_2 + x_2^2 + x_2 x_3)x + (\alpha^2 + \alpha^2 x_1 \\ &\quad\quad + \alpha x_2 + \alpha x_3 + x_1^2 + x_1 x_2 + \alpha x_1 x_3 + \alpha^2 x_2^2 + \alpha x_2 x_3 + \alpha^2 x_3^2)x^2. \end{aligned} \tag{3.32}$$

Finally we compute $L_1 \circ \phi(\tilde{F}(X))$ to obtain the public key components:

$$\bar{f}_1(x_1, x_2, x_3) = 1 + x_3 + \alpha x_1 x_3 + \alpha^2 x_2^2 + \alpha^2 x_2 x_3 + x_3^2, \tag{3.33}$$

$$\bar{f}_2(x_1, x_2, x_3) = 1 + \alpha^2 x_1 + \alpha x_2 + x_3 + x_1^2 + x_1 x_2 + \alpha^2 x_1 x_3 + x_2^2, \tag{3.34}$$

$$\bar{f}_3(x_1, x_2, x_3) = \alpha^2 x_3 + x_1^2 + \alpha^2 x_2^2 + x_2 x_3 + \alpha^2 x_3^2. \tag{3.35}$$

Now with these we can demonstrate the usage of linearization equations. For the given plaintext $(x_1, x_2, x_3) \in k^3$, where $n = 3$, we add a new value $x_0 = 1$. Thus, we can represent the public key with a more compact form and it will be possible to write it by the summation of quadratic terms. For our row vector $x = (x_0, x_1, x_2, x_3)$ the public key is given by:

$$y_1 \quad = \quad x \begin{pmatrix} 1 & 0 & 0 & 1 \\ & 0 & 0 & \alpha \\ & & \alpha^2 & \alpha^2 \\ & & & 1 \end{pmatrix} x^T, \tag{3.36}$$

$$y_2 = x \begin{pmatrix} 1 & \alpha^2 & \alpha & 1 \\ & 1 & 1 & \alpha^2 \\ & & 1 & 0 \\ & & & 0 \end{pmatrix} x^T, \tag{3.37}$$

$$y_3 = x \begin{pmatrix} 0 & 0 & 0 & \alpha^2 \\ & 1 & 0 & 0 \\ & & \alpha^2 & 1 \\ & & & \alpha^2 \end{pmatrix} x^T. \tag{3.38}$$

The entries left blank of the above matrices are equal to 0. Assume that a plaintext produced the ciphertext $(1, 0, 1)$. We will use the linearization equation attack to recover the plaintext.

First we need to introduce $y_0 = 1$, such that the public-key will be represented with $y = (y_0, y_1, y_2, y_3)$. We use $m = n = 3$ in this example, so it is possible to write linearization equations in the matrix form:

$$xAy^T = 0. \tag{3.39}$$

where A is a 4×4 matrix with unknown coefficients $A_{i,j}$, $i, j = 0, 1, 2, 3$. Now we will represent these $(m+1)(n+1)$ unknowns with a one dimensional array. There is a commonly utilized notation in programming language:

$$A_{i,j} \Longleftrightarrow A\left[(m+1)\, i + j\right] = 0. \tag{3.40}$$

By putting the public key into the equation (3.39), it can be generated a homogeneous polynomial such that they are cubic in x_i where $i = 0, 1, 2, 3$. Then a homogenous linear equations system which has 16 unknowns from $A[0]$ to $A[15]$, is generated by gathering the coefficients of 20 distinct terms. Thus, the resulting matrix's rank will be 14. Therefore, linearization equations' dimension will be equal to $16 - 14 = 2$, in compliance with the theorem (3). As a result of a reduction operation over this matrix with the aim of generating row echelon form, the following equations will be attained:

$$\begin{aligned}
A[0] &= \alpha^2 A[2] + A[6], & A[9] &= A[2] + \alpha^2 A[6], \\
A[1] &= \alpha A[2] + A[6], & A[10] &= \alpha A[2] + \alpha^2 A[6], \\
A[3] &= A[2] + \alpha^2 A[6], & A[11] &= \alpha A[2] + \alpha A[6], \\
A[4] &= \alpha^2 A[6], & A[12] &= A[2] + \alpha A[6], \\
A[5] &= \alpha^2 A[2] + \alpha A[6], & A[13] &= \alpha A[6], \\
A[7] &= A[6], & A[14] &= A[2] + \alpha^2 A[6], \\
A[8] &= A[2], & A[15] &= \alpha^2 A[6],
\end{aligned}$$

where $A[2]$ and $A[6]$ are free parameters. These values and the given ciphertext:

$$y = (1, y_1', y_2', y_3') = (1, 1, 0, 1) \tag{3.41}$$

are now substituted back into (3.39) to obtain:

$$xAy^T = \begin{pmatrix} 1 & x_1 & x_2 & x_3 \end{pmatrix} \begin{pmatrix} A[0] & A[1] & A[2] & A[3] \\ A[4] & A[5] & A[6] & A[7] \\ A[8] & A[9] & A[10] & A[11] \\ A[12] & A[13] & A[14] & A[15] \end{pmatrix} \begin{pmatrix} 1 \\ y_1' \\ y_2' \\ y_3' \end{pmatrix}$$

$$= \begin{pmatrix} 1 & x_1 & x_2 & x_3 \end{pmatrix} \begin{pmatrix} \alpha^2 A[2] + A[6] & \alpha A[2] + A[6] & A[2] & A[2] + \alpha^2 A[6] \\ \alpha^2 A[6] & \alpha^2 A[2] + \alpha A[6] & A[6] & A[6] \\ A[2] & A[2] + \alpha^2 A[6] & \alpha A[2] + \alpha^2 A[6] & \alpha A[2] + \alpha A[6] \\ A[2] + \alpha A[6] & \alpha A[6] & A[2] + \alpha^2 A[6] & \alpha^2 A[6] \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & x_1 & x_2 & x_3 \end{pmatrix} \begin{pmatrix} \alpha^2 A[6] \\ \alpha^2 A[2] \\ \alpha A[2] + A[6] \\ A[2] + \alpha^2 A[6] \end{pmatrix} = 0$$

$xAy^T = (\alpha^2 x_1 + \alpha x_2 + x_3)A[2] + (\alpha^2 + x_2 + \alpha^2 x_3)A[6] = 0.$

By equalizing the coefficients of $A[2]$ and $A[6]$ to 0, the following equations are obtained:

$$\alpha^2 x_1 + \alpha x_2 + x_3 = 0, \tag{3.42}$$

$$\alpha^2 + x_2 + \alpha^2 x_3 = 0. \tag{3.43}$$

The solution of the above system is the following:

$$x_1 = \alpha, \tag{3.44}$$

$$x_2 = \alpha^2 + \alpha^2 x_3. \tag{3.45}$$

Ultimately, there are two ways for finding the plaintext:

As the first way, $\forall x_3 \in k$, we check whether or not obtained plaintexts correspond to given ciphertext. With each possible value of x_3 in the solutions of above equations and the public keys in (3.36), (3.37), and (3.38), the following possibilities are attained:

plaintext		ciphertext
$(\alpha, \alpha^2, 0)$	$\Longrightarrow$	$(0, \alpha, \alpha)$
$(\alpha, 0, 1)$	$\Longrightarrow$	$(\alpha, \alpha^2, \alpha^2)$
(α, α, α)	$\Longrightarrow$	$(1, 0, 1)$
$(\alpha, 1, \alpha^2)$	$\Longrightarrow$	$(\alpha^2, \alpha, 0)$.

Merely the third case generates the desired ciphertext. That's why, the original plaintext is found as (α, α, α).

Another method is to substitute the linear equations $x_1 = \alpha$ and $x_2 = \alpha^2 + \alpha^2 x_3$ into the

public keys (3.36), (3.37), (3.38), and set it equal to the given ciphertext, that is

$$y_1 = 1, \tag{3.46}$$

$$y_2 = 0, \tag{3.47}$$

$$y_3 = 1. \tag{3.48}$$

From here we obtain three quadratic equations, which the free parameter needs to satisfy. Solving those equations yields $x_3^2 = \alpha^2$. From this we conclude that $x_3 = \alpha$. We already know that $x_1 = \alpha$, so $x_2 = \alpha^2 + \alpha^2 x_3 = \alpha^2 + \alpha^2\alpha = \alpha^2 + 1 = \alpha$. Hence the plaintext is (α, α, α) [28].

3.2.2 LINEARIZATION EQUATIONS ATTACK FOR THE MULTIPLE-BRANCH MI CRYPTOSYSTEM

As mentioned in the section (3.1), a multiple-branch MI is a generalized form of a single-branch MI. In the case of a multiple-branch MI, the following theorem is used in order to give the lower bound for the number of linearly independent linear equations which are derived from the linearization equations space (the generalized form of the theorem (1) in the section (3.1)).

Theorem 4. *Assume that there is a multiple-branch MI cryptosystem whose public key is the following set:* $\bar{F} = \{\bar{f}_1, \ldots, \bar{f}_n\}$. *Let* $\bar{\mathcal{L}}$ *be the linearization equations space of this set. If for a given ciphertext* $Y' = (y'_1, \ldots, y'_n) \in k^n$, $\bar{\mathcal{L}}_{Y'}$ *is obtained as a linearization equations space by putting* y'_i *instead of* y_i *for all* $i = 1, \ldots, n$, *in each equation of* $\bar{\mathcal{L}}$, *then the lower bound for* $\dim_k \bar{\mathcal{L}}_{Y'}$ *is at least:*

$$n - \sum_{i=1}^{b} \gcd(n_i, \theta_i) \geq \frac{2n}{3} \tag{3.49}$$

with the following probability:

$$\frac{(q^{n_1} - 1)(q^{n_2} - 1) \cdots (q^{n_b} - 1)}{q^n}. \tag{3.50}$$

Consequently, for each type of Matsumoto-Imai cryptosystems, linearization equations attack is applicable. Furthermore, Patarin improved this algorithm by seperating the branches before the attack [28, 30].

3.3 MATSUMOTO-IMAI VARIANTS

After Patarin proposed the linearization equations attack which broke the Matsumoto-Imai cryptosystem in 1995, lots of new variants of this system were developed with the aim of increasing it's security. These new variants can be considered as kinds of two main methods

whose names are the "**Minus**" method and the "**Plus**" method. While the aim of the former was improving the security of the original MI cryptosystem, the latter one was developed for making a cipher injective, because an injective mapping is crucial so as to decrypt the cipher-text correctly. In this section, the Minus method, Flash and Sflash which are the significant variants of the Minus method, the Plus method, and the Matsumoto-Imai-Plus-Minus public key cryptosystem will be described respectively. Among all the Matsumoto-Imai variants, **Sflash**^{v2} can be acknowledged as the most successful and widely used one. Let's start with the description of the Minus method [28].

3.3.1 THE MINUS METHOD

To begin with, Shamir proposed the Minus method in 1993 [37] and after that it was explored by Matsumoto and Patarin independently from Shamir [32]. According to the Minus method, some of the components of the public-key are deleted. Let's say the number of deleted ones is r. Depending on the selection of r, resistance against linearization equations attack can be given to the MI cryptosystem. In other words, r shouldn't be selected too small if secure systems are desired. Now let $\bar{F} : k^n \to k^m$ be the set of public key components which are the multivariate quadratic polynomials: $\bar{f}_1, \ldots, \bar{f}_m \in k[x_1, \ldots, x_n]$. In general, m is assumed to be equal to n. However, this case doesn't matter for the Minus method since it works in other cases as well. Once the Minus method is applied to $\bar{F}$, the resulting map $\bar{F}^- : k^n \to k^{m-r}$ such that

$$\bar{F}^-(x_1, \ldots, x_n) = (\bar{f}_1, \ldots, \bar{f}_{m-r}) \tag{3.51}$$

is obtained. Furthermore the Minus method has the following structure.

The Public-Key

The public key of the Minus method includes:

1. The field: k , the addition and multiplication tables of it,
2. The set of $m - r$ polynomials: $(\bar{f}_1, \ldots, \bar{f}_{m-r}) \in k[x_1, \ldots, x_n]$.

The Private-Key

The private-key of the Minus method is the same as in the original MI cryptosystem:

1. The two invertible affine transformations: L_1 and L_2,
2. Parameter: θ.

SIGNING PROCESS

Because it's not bijective, this method is only useful for signing a document and not for an encryption. Suppose the document to be signed, or its hash value, is $Y'^- = (y'_1, \ldots, y'_{n-r})$ be

a vector in k^{n-r}. A person who wants to sign this document first selects r random elements $y'_{n-r+1}, \ldots, y'_n$ in k and appends these values to Y'^{-} in order to obtain $Y' = (y'_1, \ldots, y'_n)$ in k^n. Then the signing is done in the same way as the decryption in the original MI cryptosystem;

$$X' = (x'_1, \ldots, x'_n) = \bar{F}^{-1}(Y')$$

Lastly, X' is the signature of Y'^{-} and (X', Y'^{-}) is sent as the signature and the message.

VERIFYING PROCESS

The person who gets the document Y'^{-} and its signature X' controls whether or not

$$(\bar{f}_1(X'), \ldots, \bar{f}_{l-r}(X')) = Y'^{-} \tag{3.52}$$

by using the public key polynomials. If the equality is satisfied, the signature can be accepted as valid, otherwise it is understood that the signature is invalid and therefore rejected.

Here, it is so significant that the appended values $y'_{n-r+1}, \ldots, y'_n$ are kept secret. If they are known publicly, an attacker can use these values to recover the missing polynomials and forge a signature [27].

The Minus method is especially beneficial so as to transform an ecryption scheme into a signature scheme, since there is no more a necessity for the injectivity. The security of this kind of signature schemes comes from the fact that solving a set of $l - r$ nonlinear equations in n variables is considered quite hard.

Now it will be illustrated how this signature scheme works. For that, the toy example that was utilized to demonstrate how linearization equation attacks work, will be given. In that example, there were 5 public-key polynomials; y_1, y_2, y_3, y_4, and y_5. This once only the first 4 of them are made public and y_5 is kept secret and is not a part of the public-key. As well as this, anyone which is required to sign the document has the secret-key, and the invertible affine transformations L_1 and L_2, or their inverses:

$$L_1^{-1}(y_1, y_2, y_3, y_4, y_5) = \begin{pmatrix} \alpha^2 & \alpha & \alpha^2 & 1 & 1 \\ 0 & 0 & \alpha^2 & \alpha^2 & 1 \\ \alpha & \alpha & 1 & \alpha^2 & 1 \\ \alpha & \alpha^2 & 0 & \alpha & 1 \\ 0 & 0 & \alpha & \alpha & 1 \end{pmatrix} \begin{pmatrix} y_1 - \alpha^2 \\ y_2 - \alpha^2 \\ y_3 - 0 \\ y_4 - 1 \\ y_5 - 0 \end{pmatrix},$$

$$L_2^{-1}(y_1, y_2, y_3, y_4, y_5) = \begin{pmatrix} \alpha^2 & 1 & \alpha^2 & 0 & 1 \\ \alpha & 1 & \alpha & 1 & \alpha \\ 0 & \alpha^2 & \alpha & \alpha & 0 \\ \alpha & 1 & 1 & \alpha & \alpha \\ 0 & 1 & 0 & \alpha & \alpha^2 \end{pmatrix} \begin{pmatrix} y_1 - 1 \\ y_2 - 0 \\ y_3 - \alpha^2 \\ y_4 - \alpha^2 \\ y_5 - \alpha^2 \end{pmatrix}.$$

θ is also known, and in this example $\theta = 3$. This gives $\tilde{F}^{-1}(X) = X^{362}$, and the irreducible polynomial $g(x) = x^5 + x^3 + x + \alpha^2$.

Let's assume the plaintext is $Y'^{-} = (\alpha^2, \alpha, \alpha^2, 0)$. Since $n = 5$, there will be 1 extra value y'_5, and as mentioned before this value should be selected randomly. In this toy example y'_5 can be one of four possible values, and they are all displayed below, along with the signatures they generate.

$$\begin{array}{rcl} Y'\text{(Document)} & & X'\text{(Signature)} \\ (\alpha^2, \alpha, \alpha^2, 0, 0) & \Rightarrow & (0, \alpha, \alpha, 0, \alpha^2) \\ (\alpha^2, \alpha, \alpha^2, 0, 1) & \Rightarrow & (1, 1, \alpha, \alpha, \alpha) \\ (\alpha^2, \alpha, \alpha^2, 0, \alpha) & \Rightarrow & (1, 0, 1, 1, \alpha) \\ (\alpha^2, \alpha, \alpha^2, 0, \alpha^2) & \Rightarrow & (\alpha^2, 1, 1, 0, \alpha^2) \end{array}$$

Any of these four signatures, when used with the public-keys, y_1, y_2, y_3, y_4 will give

$$(y_1, y_2, y_3, y_4) = (\alpha^2, \alpha, \alpha^2, 0), \tag{3.53}$$

hence any of them will be accepted as valid.

If the four components of the public-key were to be utilized in a linearization equation attack, the adversary would find that $\dim_k \mathcal{L}_{Y'} = 1$, and therefore would just attain one linearization equation

$$x_1 = \alpha^2 x_2 + \alpha x_3 + \alpha^2 x_4 + \alpha x_5 + \alpha^2. \tag{3.54}$$

The above equation (3.54) holds for the four possible signatures, hence it is not enough to forge a signature. Broadly, in the case of larger r, the linearization equations are removed entirely [28].

3.3.2 FLASH AND SFLASH

In 2000, The New European Schemes for Signatures, Integrity, and Encryption Project (NESSIE) was started, and in 2004 final selections were made. Sflash^{v2} which is a fast multivariate signature scheme was elected by NESSIE so as to utilize in low-cost smart cards [1]. Sflash^{v2} is said to be Flash by NESSIE. Since there was a flaw in the initial submission Sflash^{v1}, it was broken in 2002. This flaw was because of the preference of $GF(2)$ for the field elements [18]. But actually this preference was conscious in order to keep the signature size and the public key size small. However it was not an irreversible flaw as it could easefully be corrected by selecting the field as $GF(2^7)$ in Sflash^{v2} [31]. For this new version, lengthes of the signature and the public key are 259 bits and 15 KBytes respectively.

The creators of Sflash^{v2} asserted that it is the fastest digital signature scheme (DSS) in the earth, and is the sole DSS that can be utilized in smart cards. Afterwards, because of further security concerns, the producers suggested another version, Sflash^{v3} which has a longer

signature whose length is 469 bits and a public key of 112 KBytes [26]. But they then realized that their concerns weren't necessary, and therefore Sflash^{v2} is suggested again. Today, Sflash^{v2} has been thought as a still secure DSS [8].

For the illustration of the signature scheme Sflash^{v2}, single-branch map $\bar{F}$ with $\theta = 11$ will be used. Moreover, Sflash utilizes $n = 37$ and $r = 11$. So that gives $\bar{F}^{-} : k^{37} \rightarrow k^{26}$ such that

$$\bar{F}^{-}(x_1, \ldots, x_{37}) = (\bar{f}_1, \ldots, \bar{f}_{37-11}) \tag{3.55}$$

where $\bar{f}_1, \ldots, \bar{f}_{26} \in k[x_1, \ldots, x_{37}]$. [28] The structure of the Sflash signature scheme includes:

The Public-Key

1. The field: $k = GF(2^7) = GF(2)[x]/(x^7 + x + 1)$ with its additive and multiplicative structure,

2. The set of 26 quadratic polynomials: $(\bar{f}_1, \ldots, \bar{f}_{26}) \in k[x_1, \ldots, x_{37}]$.

The Private-Key

1. A randomly selected secret key whose length is 80-bit : Δ,

2. The two invertible affine transformations: L_1 and L_2.

THE SIGNATURE GENERATION

Assume $\psi : k \rightarrow GF(2)^7$ is a vector space isomorphism. The subscripts which will be utilized below corresponds to the location in the bit string, and $||$ is the concatenation of bit strings. Let's assume the message to be signed is M. The following processes are fulfilled for signing the message

1. Calculate $M1 =$ SHA-1(M) and $M2 =$ SHA-1$(M1)$. $M1$ and $M2$ are the strings whose lengthes are 160-bit.

2. Let

$$V = M1||(M2_1, \ldots, M2_{22}) = (V_1, \ldots, V_{182})$$
$$W = SHA(V||\Delta) = (W_1, \ldots, W_{77}).$$

3. Let

$$\begin{aligned} M'_1 &= \psi^{-1}(V_1, \ldots, V_7) \\ M'_2 &= \psi^{-1}(V_8, \ldots, V_{14}) \\ &\vdots \\ M'_{26} &= \psi^{-1}(V_{176}, \ldots, V_{182}) \end{aligned}$$

$$\begin{aligned} M'_{27} &= \psi^{-1}(W_1, \ldots, W_7) \\ M'_{28} &= \psi^{-1}(W_8, \ldots, W_{14}) \\ &\vdots \\ M'_{37} &= \psi^{-1}(W_{71}, \ldots, W_{77}). \end{aligned}$$

   Lastly, let $M' = (M'_1, \ldots, M'_{37})$.

4. Compute the signature S of message M by:

$$\begin{aligned} S &= \bar{F}^{-1}(M') \\ &= L_2^{-1} \circ F^{-1} \circ L_1^{-1}(M') \\ &= L_2^{-1} \circ \phi \circ \tilde{F}^{-1} \circ \phi^{-1} \circ L_1^{-1}(M'). \end{aligned}$$

   The pair (S, M), the message M with its signature S, is sent.

SIGNATURE VERIFICATION

Once the message-signature pair (S, M) is received, the verification process is very similar to the signing process for the most part.

1. Calculate

$$\begin{aligned} M1 &= \text{SHA-1}(M) \\ M2 &= \text{SHA-1}(M1) \\ V &= M1||(M2_1, \ldots, M2_{22}) = (V_1, \ldots, V_{182}) \end{aligned}$$

   After that since Δ is the secret key, W can't be computed in the verification. Instead

2. Let

$$\begin{aligned} N'_1 &= \psi^{-1}(V_1, \ldots, V_7) \\ N'_2 &= \psi^{-1}(V_8, \ldots, V_{14}) \\ &\vdots \\ N'_{26} &= \psi^{-1}(V_{176}, \ldots, V_{182}) \end{aligned}$$

   and $N' = (N'_1, \ldots, N'_{26})$.

3. If $N' = \bar{F}^{-}(S)$, then the signature is accepted, otherwise it is rejected.

Here it can be easily seen that so as to forge a signature for the message M, an attacker is required to be able to attain a unique pre-image of N' under $\bar{F}^{-}$, meaning that they should obtain one solution to a system of 26 equations in 37 variables.

The secret key Δ is so significant for satisfying security, and it is especially important that it stays secret [27]. If this secret key is made known to the public, an attacker can utilize it to reach the missing Minus polynomials, and therefore break the system and forge a signature easily.

As it can be recalled from the previous parts, while being secure, the Minus method only works for signatures, not for encryption. Because in signature schemes, the only need is to have one pre-image, while for encryption it is required to find a unique pre-image. That's why, the "Plus" method can be considered as a technique in order to alter the Minus method to work for encryptions [28].

3.3.3 THE PLUS METHOD

The Plus method consists of adding a few, for example s, randomly selected polynomials to a given multivariate public key, and after that mixing them with an invertible affine transformation.

Suppose $\bar{F} : k^n \to k^m$ is a multivariate public key cryptosystem with polynomial components $\bar{f}_1, \ldots, \bar{f}_m \in k[x_1, \ldots, x_n]$. The randomly selected polynomials $p_1, \ldots, p_s$ are appended to generate the following map: $\bar{F}^{+} : k^n \to k^{m+s}$ such that

$$\bar{F}^{+}(x_1, \ldots, x_n) = L_3 \circ (\bar{f}_1, \ldots, \bar{f}_m, p_1, \ldots, p_s). \tag{3.56}$$

Here L_3 is an invertible affine transformation from k^{m+s} to k^{m+s}. To begin with, the Plus method hasn't been proposed to enhance the security of the cryptosystem. Unlike in the Minus method the linearization equations are still there. The main purpose of the Plus method is to take the map $\bar{F}$, which is not injective, and turn it into an injective map, thus enabling us to use it for encryption. In other words, if $\bar{F}^{-1}$ has multiple elements, then the Plus method can reduce the number of pre-images to only one and therefore helps to figure out the real plaintext amongst many possible candidate plaintexts, but for this reduction, the number s should be big enough [28].

3.3.4 THE MATSUMOTO-IMAI-PLUS-MINUS PUBLIC-KEY CRYPTOSYSTEM

As mentioned above, the Plus method doesn't increase the security of Matsumoto-Imai cryptosystem when it is implemented alone. So instead we use the Plus method and the Minus method together to form the Matsumoto-Imai-Plus-Minus public-key cryptosystem.

Suppose $\bar{F} : k^n \to k^n$ is a multivariate public key cryptosystem with polynomial components $\bar{f}_1, \ldots, \bar{f}_n \in k[x_1, \ldots, x_n]$. In order to apply the two methods together, first remove the last r polynomials, then append s randomly selected quadratic polynomials $p_1, \ldots, p_s \in k[x_1, \ldots, x_n]$, and lastly mix these with the system through an invertible affine transformation. The new map $\bar{F}^{\pm} : k^n \to k^m$ is defined by

$$\bar{F}^{\pm} = L_3 \circ (\bar{f}_1, \ldots, \bar{f}_{n-r}, p_1, \ldots, p_s) = (\bar{f}_1^{\pm}, \ldots, \bar{f}_m^{\pm})$$

in which $m = n - r + s$ and $L_3 : k^m \to k^m$ is an invertible affine transformation. Also $r \leq s$, since if $r > s$, this new map would not be injective and would be only useful for signatures.

Let's give the structure [28]:

The Public-Key

1. The field: k, the addition and multiplication tables of it,
2. m quadratic polynomials: $\bar{f}_1^{\pm}, \ldots, \bar{f}_m^{\pm} \in k[x_1, \ldots, x_n]$.

The Private-Key

1. The quadratic Plus polynomials: $p_1, \ldots, p_s \in k[x_1, \ldots, x_n]$,
2. The invertible affine transformations: L_1, L_2, L_3.

ENCRYPTION

Assume the plaintext to be encrypted is $(x'_1, \ldots, x'_n) \in k^n$. Ciphertext $(y'_1, \ldots, y'_m) \in k^m$ will be calculated with the following polynomials of $\bar{F}^{\pm}$:

$$(y'_1, \ldots, y'_m) = \bar{F}^{\pm}(x'_1, \ldots, x'_n). \tag{3.57}$$

DECRYPTION

For the decryption, the following processes will be executed:

1. Compute $(z_1, \ldots, z_{n-r+s}) = L_3^{-1}(y'_1, \ldots, y'_{n-r+s})$
2. $\forall w = (w_1, \ldots, w_r) \in k^r$, calculate
$$t_w = (t_1, \ldots, t_n) = \bar{F}^{-1}(z_1, \ldots, z_{n-r}, w_1, \ldots, w_r),$$
and let $T = \{(w, t_w) | w \in k^r\}$.
3. $\forall (w, t_w) \in T$, control whether or not
$$p_i(t_w) = z_{n-r+i} \tag{3.58}$$

holds $\forall i = 1, 2, \ldots, s$. Keep all t_w which satisfy the equation (3.58), and remove the others. In the case of sufficiently large s, there should remain solely one element and it's said to be the plaintext $(x'_1 \ldots, x'_n)$.

Consequently, the Plus method also helps increase the security, because the map L_3 mixes the random polynomials into the system and an attacker can't identify which ones are the original polynomials and which ones are Plus polynomials. Thus, any method which can be utilized on the Minus method will be too hard to use for Plus-Minus Cryptosystem [28].

CHAPTER 4

CRYPTOGRAPHIC PROTOCOLS

In our current world, technological improvements which include the creation of the quantum computers, affect and threaten the Internet security. The main reason of these is losing the validity of the cryptographic protocols in conjunction with the existence of the quantum computers. In this section of the thesis, the explanation of the three main cryptographic protocols which consist of the identification schemes, the digital signature schemes and the commitment schemes, will be given together with their security properties.

4.1 IDENTIFICATION SCHEMES

An **identification scheme** is a protocol that is run in the public-key setting just like public-key encryption and digital signatures. Here there exist two entities. One of which, so called **prover** locally generates a pair of public and private keys then publicizes the public-key and makes it widely available. The second entity, so called **verifier** is assumed to be able to obtain an authentic copy of the prover's public-key. Now the goal of the identification scheme is to allow the prover to convince the verifier that the prover is who he or she claims. Therefore, the prover needs to be able to interact with the verifier and via this interaction convinces the verifier of his or her identity.

In the construction which will be defined, $IDS\left(1^k\right)$ means an identification scheme with security parameter k. To describe this frame, we need a triplet of probabilistic polynomial time algorithms: $KGen$, $\mathcal{P}$, $\mathcal{V}$ such that $KGen$ represents the key generation algorithm, $\mathcal{P}$ and $\mathcal{V}$ indicate the prover and the verifier respectively. As we need to explain more detaild, firstly $KGen$ algorithm gives us a key pair (sk, pk), then the algorithms $\mathcal{P}$ and $\mathcal{V}$ interact with each other. In other words, they execute a common protocol, denoted by $\langle P\left(sk\right), V\left(pk\right)\rangle$ in which the prover's input is the secret key sk and the verifier's input is the public-key pk. At the end of this protocol, the verifier's output will be a bit, say b. According to the identification framework, if $b = 1$, then it indicates "**accept**" otherwise indicates "**reject**".

[5] Let's define some concepts related with the identification schemes. The first one which is a property of the $IDS\left(1^k\right)$ is the notion of "**perfectly correct**". To be able to say $IDS\left(1^k\right)$ is perfectly correct, the probability that $\langle P\left(sk\right), V\left(pk\right)\rangle$ equals to 1 , denoted by

$P\left[\langle P\left(sk\right), V\left(pk\right)\rangle\right] = 1$, has to be 1. Secondly, we will define a **transcript** of an identification scheme's execution. In the process of prover and verifier's interaction, they exchange messages between each other. If we store all these messages which are created during an interaction, the set of all these messages will give us the transcript of this execution that is denoted by $trans\left(\langle P\left(sk\right), V\left(pk\right)\rangle\right)$.

4.1.1 SECURITY PROPERTIES

[5] In this step, we analyze the security properties which an identification scheme needs to satisfy.

4.1.1.1 Security Against Impersonation Under Passive Attacks (IMP-PA Security)

The relatively weak notion of security is the security against passive eavesdropping attacks. In this model, an attacker eavesdrops multiple on honest executions between the prover and the verifier, which the attacker has access to valid transcripts of honest executions and also has access to public-key. Now the idea about the security notion is that even after eavesdropping on these multiple honest executions, the attacker should not be able to convince a verifier falsely that the attacker is the prover. That is, the attacker shouldn't have the ability to succeed in carrying out an execution of the identification protocol with a verifier when the real prover is not around.

Let's define IMP-PA security with mathematical terms:

Experiment: $Exp^{imp-pa}_{IDS\left(1^k\right)}\left(\mathcal{I}\right)$

1. $(sk, pk) \leftarrow KGen\left(\right)$
2. $(\text{state}, \text{com}) \leftarrow \mathcal{I}^{Trans(pk,sk,\cdot)}(pk)$
3. For every $i \in \{1, \ldots, n\}$
    1. $\text{ch}_i \leftarrow_R \text{ChS}_i(1^k)$
    2. $(\text{state}, \text{resp}_i) \leftarrow \mathcal{I}^{Trans(pk,sk,\cdot)}(pk)$
4. Return $1 \Leftrightarrow Vf\left(pk, \text{com}, \text{ch}_1, \text{resp}_1, \ldots, \text{ch}_n, \text{resp}_n\right) = 1$.

Definition 3. *Let $k \in \mathbb{N}$ and $IDS\left(1^k\right)$ be an identification scheme with a security parameter k. $IDS\left(1^k\right)$ is called as IMP-PA secure if $\forall Q_t$, $t = poly\left(k\right)$, the success probability*

$$Succ^{imp-pa}_{IDS\left(1^k\right)}\left(\mathcal{I}\right) = negl\left(k\right) \tag{4.1}$$

for any impersonator $\mathcal{I}$ running in time $\leq t$, where Q_t is the upper bound of the number of queries which $\mathcal{I}$ makes to $Trans$ in $Exp^{imp-pa}_{IDS\left(1^k\right)}\left(\mathcal{I}\right)$ experiment.

Note that: A function f is said to be $negl\left(k\right)$ which means negligible in k if $\forall$ positive polynomial g and sufficiently large k, $f\left(k\right) < \frac{1}{g(k)}$.

4.1.1.2 Key Relation

Definition 4. *Let $k \in \mathbb{N}$ and $IDS\left(1^k\right)$ be an identification scheme with a security parameter k and R be a relation. If $\forall\,(pk, sk) \leftarrow KGen\,() : (pk, sk) \in R$, then $IDS\left(1^k\right)$ has the key relation R.*

4.1.1.3 Key-One-Wayness (KOW)

Definition 5. *Let $k \in \mathbb{N}$ and $IDS\left(1^k\right)$ be an identification scheme with a security parameter k. Assume that $IDS\left(1^k\right)$ has a key relation R. If $\forall$ polynomial time algorithm A,*

$$\begin{aligned} &Succ^{pq-kow}_{IDS\left(1^k\right)}(A) = \cdots \\ &\cdots = \Pr\left[(pk, sk) \leftarrow KGen\,(), sk' \leftarrow A\,(pk) : \left(pk, sk'\right) \in R\right] = negl\,(k)\,, \end{aligned} \tag{4.2}$$

then $IDS\left(1^k\right)$ is said to be key-one-way (KOW) according to R.

4.1.1.4 Soundness (with soundness error κ)

Definition 6. *Let $k \in \mathbb{N}$ and $IDS\left(1^k\right)$ be an identification scheme with a security parameter k. If $\forall$ probabilistic polynomial time algorithm A,*

$$\Pr\left[(pk, sk) \leftarrow KGen\,() : \left\langle A\left(1^k, pk\right), \mathcal{V}\,(pk)\right\rangle = 1\right] \leq \kappa + negl\,(k) \tag{4.3}$$

where A indicates the adversary, then $IDS\left(1^k\right)$ is said to be a sound, with soundness error κ.

4.1.1.5 Computational Honest Verifier Zero Knowledge (HVZK)

Definition 7. *Let $k \in \mathbb{N}$ and $IDS\left(1^k\right)$ be an identification scheme with a security parameter k. If $\exists$ a probabilistic polynomial time algorithm S, say simulator such that for any $(pk, sk) \leftarrow KGen\,()$ and polynomial time algorithm A (the adversary):*

$$\begin{aligned} &Succ^{pq-hvzk}_{IDS\left(1^k\right)}(A) = \cdots \\ &\left|\Pr\left[1 \leftarrow A\left(sk, pk, trans\left(\langle P\,(sk), V\,(pk)\rangle\right)\right)\right] - \Pr\left[1 \leftarrow A\left(sk, pk, S\,(pk)\right)\right]\right| \\ &\cdots = negl\,(k)\,, \end{aligned} \tag{4.4}$$

then $IDS\left(1^k\right)$ is said to be computational HVZK.

4.2 DIGITAL SIGNATURE SCHEMES

Firstly, let's focus on the role of digital signature schemes in modern cryptography. The conventional signature on any document is utilized to verify or certify which the signer has the

responsibility for the document's content. Signing a document involves allocating a unique sequence of characaters to represent your name for effectively binding your identity to that document. Essentially the signing is a physical piece of the document and while forgery is probable, it is hard to make this so cogently. A digital signature can be considered as the electronic analog of a physical signature. It is possible to achieve the same properties discussed in the physical one meaning that there is a method of digital signing that is functionally equipotent to a physical signature and very resistant to forgery. Schemes that enable this functionality are said to be **digital signature schemes**.

There are two components of digital signature schemes. The former is the private signing algorithm that allows a utilizer to securely sign a plaintext. The latter is the public verification algorithm that allows any person to verify the authenticity of the signatures. The expectation from the signing algorithm is making a connection between a message and a signature so that the signature is not be able to utilized for signing another document. In practical terms, both algorithms shoud be relatively fast and their computational complexities should be in enough amount.

Let's broadly illustrate how a digital signature works. Assume Alice wants to digitally sign a document and send the corresponding signature to Bob for the verification. In the scheme, Alice is first going to generate her two keys simultaneously: the private signing key and the public verification key. These keys have a mathematical relationship, but most importantly it should be hard to come up with the signing key given only the verification key. Therefore, Alice is going to digitally sign a message M by applying a mathematical transformation to her message M and her signing key sk. Then the output of the transformation given these two inputs will be a special sequence of numbers or her digital signature. It should be noted that the only person who possesses the signing key can generate this type of output.

The verification process that goes hand in hand with the signing process and involves the public verification key has three different inputs: Alice's message M, signature on message S_M and Alice's public verification key pk. For Bob, the goal of the mathematical transformation is to check the signature he received from Alice was produced by her private signing key sk. It is significant to know that Bob can accomplish to answer the question of whether or not he should accept the signature as valid without having the access to Alice's signing key. At this point, Alice's public-key serves as an identifier to her just like a handwritten signature would.

Note that the transformation takes the message M as one of its inputs. In other words, changing the message would yield a different output unlike a handwritten signature which doesn't change depending on what you're signing. Furthermore, digital signature schemes are often associated with cryptographic hash functions. Before Alice signs a message, she needs to apply a cryptographic hash function to her message and receive a shorter output, called the digest of the hash function. Then, Alice should sign the message digest instead of the original message itself. This process is necessary in digital signing, because dealing with a fixed length input instead with an input of arbitrary length is much more feasible in practical application. All the process which was defined above is based on the assumption that it is very

difficult to find two messages that map to the same output when the hash funciton is applied.

In the construction which will be defined, $DDS\left(1^k\right)$ means a digital signature scheme with security parameter k. To describe this frame, we need a triplet of polynomial time algorithms: $KGen$, $Sign$, Vf such that $KGen$ represents a probabilistic key generation algorithm, $Sign$ indicates a possibly probabilistic signature generation algorithm and finally Vf represents a deterministic verification algorithm. More specifically, $KGen$ algorithm gives us a key pair (sk, pk), $Sign$ algorithm whose inputs are a secret key sk and a message M gives us a signature σ and Vf algorithm whose inputs are a public-key pk, a message M and a signature σ gives us a bit b such that if $b = 1$, then it indicates "**accept**" otherwise indicates "**reject**"[5].

4.2.1 SECURITY PROPERTIES

In this step, we will analyze the security properties which a digital signature scheme needs to satisfy. At this point, notice that identification schemes have only limited applicability on their own. But they're extremely important as a building block for digital signature schemes. That's why, in the process of constructing digital signatures from identification schemes, identification schemes have to satisfy some security properties so that corresponding signature schemes can satisfy security properties which are defined below. Actually there are two kinds of attacks against which it is wanted to resist: message attacks and key only attacks. Therefore, the following security properties will be associated with these attacks respectively [5].

4.2.1.1 Existential Unforgeability Under Adaptive Chosen Message Attacks (EU-CMA Security)

In adaptive chosen message attack which is a kind of message attacks, an adversary has the ability to examine some signatures correspond to his or her chosen messages. If we assume A as a signer, then the attacker can utilize A as an oracle, meaning that the attacker is able to have an access to signatures of messages which depend on both the public-key of A and previously obtained signatures.

Existential forgery means producing a new signature for at least one message. However, the adversary doesn't have any control over the message whose signature he or she has. That's why existential unforgeability can be thought as a standard security notion for DSS. Let's describe the experiment $Exp^{eu-cma}_{DSS\left(1^k\right)}\left(\mathcal{A}\right)$ to be able to define EU-CMA security [19].

Experiment: $Exp^{eu-cma}_{DSS\left(1^k\right)}\left(\mathcal{A}\right)$

1. $(sk, pk) \leftarrow KGen\left(\right)$
2. $(M^*, \sigma^*) \leftarrow A^{Sign(sk,\cdot)}\left(pk\right)$
3. Let $\{(M_i)\}_1^{Q_s}$ be the queries to $Sign\left(sk, \cdot\right)$.
4. Return $1 \Leftrightarrow Vf\left(pk, M^*, \sigma^*\right) = 1$ and $M^* \notin \{M_i\}_1^{Q_s}$.

Definition 8. *Let $k \in \mathbb{N}$ and $DSS\left(1^k\right)$ be a digital signature scheme with a security parameter k. If $\forall Q_s$, $t = poly\,(k)$, the success probability*

$$Succ^{eu-cma}_{DSS\left(1^k\right)}(A) = \Pr\left[Exp^{eu-cma}_{DSS\left(1^k\right)}(\mathcal{A}) = 1\right] = negl\,(k) \tag{4.5}$$

for any probabilistic polynomial time algorithm A running in time $\leq t$, where Q_s is the upper bound of the number of queries which will be signed in $Exp^{eu-cma}_{DSS\left(1^k\right)}(\mathcal{A})$, then $DSS\left(1^k\right)$ is said to be EU-CMA secure.

4.2.1.2 Key Only Attacks (KOA Security)

Experiment: $Exp^{koa}_{DSS\left(1^k\right)}(\mathcal{A})$

1. $(sk, pk) \leftarrow KGen\,()$
2. $(M^*, \sigma^*) \leftarrow \mathcal{A}\,(pk)$
3. Return $1 \Leftrightarrow Vf\,(pk, M^*, \sigma^*) = 1$.

Definition 9. *Let $k \in \mathbb{N}$ and $DSS\left(1^k\right)$ be a digital signature scheme with a security parameter k. If $\forall t = poly\,(k)$, the success probability*

$$Succ^{koa}_{DSS\left(1^k\right)}(A) = \Pr\left[Exp^{koa}_{DSS\left(1^k\right)}(\mathcal{A}) = 1\right] = negl\,(k) \tag{4.6}$$

for any probabilistic polynomial time algorithm A running in time $\leq t$, then $DSS\left(1^k\right)$ is said to be KOA secure.

Notice that: If we compare the two security properties of digital signature schemes, they are similar, but KOA security is weaker than EU-CMA security, because KOA security doesn't enable to any access to the signing oracle.

4.3 COMMITMENT SCHEMES

The concept of commitment has a really crucial role in both identification schemes and digital signature schemes in terms of security. In any commitment scheme, there exist two parties: prover $\mathcal{P}$ and verifier $\mathcal{V}$. It is possible to think $\mathcal{P}$ and $\mathcal{V}$ as players of a game. In the process of making a commitment, the prover should select a value from a given finite set, then this player needs to commit his or her chosen value not to change his or her choice. In addition to this, $\mathcal{P}$ can not display this value even if $\mathcal{P}$ wants to do it. Let's illustrate a commitment scheme with an informal example.

Assume that $\mathcal{P}$ and $\mathcal{V}$ will play a game. The first step of this game starts with $\mathcal{P}$'s commitment meaning that $\mathcal{P}$ desires to commit a message m. That's why, $\mathcal{P}$ writes the message on a paper then puts it in a box which is locked with the help of a padlock. The second step is transferring

the locked box of $\mathcal{P}$ to the $\mathcal{V}$. In the final step, the key of the padlock is important, because if the prover wants to open the commitment, the thing which $\mathcal{P}$ needs to do is giving this key to the $\mathcal{V}$.

In practical application, the locked box can be thought as a mathematical function more specifically as one-way-functions. Thus, a commitment scheme can be secure. In crpytography, collision-intractable hash functions are used as a one-way-function. Inputs of these collision-intractable hash functions will be a message m and a string r (which is the public key of the protocol). The output which is the locked box can be thought as the committed message, say $com\,(r, m)$, where com is the one-way-function (collision-intractable hash function). As a result of all these, we have the following theorem [10].

Theorem 5. *If collision-intractable hash functions exist, then commitment schemes with security properties which are computationally binding and hiding, exist.*

4.3.1 SECURITY PROPERTIES

A secure commitment scheme has to satisfy: binding and hiding properties. Now we will analyze them [5].

4.3.1.1 Binding Property

If we continue from above example, let's focus on the locked box. After $\mathcal{P}$ gave this box to $\mathcal{V}$, $\mathcal{P}$ loses the opportunity of changing the message which is inside of the box. Therefore, when the lock is opened, we can be sure that $\mathcal{P}$'s original selection and the message which is revealed from the box are the same. This property is said to be computationally binding property.

Definition 10. *[10] Let $k \in \mathbb{N}$ and $Com\left(1^k\right)$ be a commitment scheme with a security parameter k. If for any polynomial time algorithm $\mathcal{A}$, random string r, random string r', message m and message m',*

$$\begin{aligned} &\Pr\left[Com\,(r,m) = Com\left(r',m'\right) \wedge m \neq m' : \left(r,m,r',m'\right) \leftarrow \mathcal{A}\left(1^k\right)\right] = \cdots \\ &\cdots = negl\,(k)\,, \end{aligned} \tag{4.7}$$

then $Com\left(1^k\right)$ is said to be computationally binding.

4.3.1.2 Hiding Property

The same example is also valid for defining the hiding property. As we remember, $\mathcal{P}$ gave the locked box to the $\mathcal{V}$. The thing which we have to know is that $\mathcal{V}$ needs to know the key of the padlock so as to learn the message inside of the box and this key belongs to $\mathcal{P}$. That's why, $\mathcal{P}$ and $\mathcal{V}$ have to work together. This property is said to be computationally hiding property.

Definition 11. *[10] Let $k \in \mathbb{N}$ and $Com\left(1^k\right)$ be a commitment scheme with a security parameter k. If for any polynomial time algorithm $\mathcal{A}$, random string r, message m and message m',*

$$\left|\Pr\left[1 \leftarrow \mathcal{A}\left(Com\left(r, m\right)\right)\right] - \Pr\left[1 \leftarrow \mathcal{A}\left(Com\left(r, m'\right)\right)\right]\right| = negl\left(k\right), \tag{4.8}$$

then $Com\left(1^k\right)$ is said to be computationally hiding.

CHAPTER 5

STRUCTURAL TOOLS OF THE MULTIVARIATE QUADRATIC DIGITAL SIGNATURE SCHEME (MQDSS)

In the design process of the MQDSS, it can be benefited from the lots of tools which are composed of the multivariate quadratic problem, canonical $2n+1$-pass identification schemes together with its specific examples for $n = 1$ and $n = 2$ and its security properties, the Sakumoto-Shirai-Hiwatari (SSH) 3-pass and 5-pass identification schemes together with its security properties, the Fiat-Shamir transform together with its two specific examples for canonical 3-pass and 5-pass identification schemes and its security analysis. The Fiat-Shamir transform of the one particular SSH 5-pass identification scheme will be utilized in order to obtain the MQDSS. That's why, in this section of the thesis, these structural tools will be introduced.

5.1 MULTIVARIATE QUADRATIC PROBLEM

In this thesis, we focus on multivariate public-key cryptosystem which is a kind of post quantum cryptography as it was mentioned at the beginning. [2, 5] Since multivariate public key cryptosystem is based on multivariate quadratic problem, say MQ problem, the aim of this section is to explain what is MQ problem.

Definition 12. *Let $m, n, q \in \mathbb{N}$, $\mathbb{F}_q$ be a finite field with q elements where q is a prime power, $\mathbb{F}_q^n$ be an n-dimensional vector space over $\mathbb{F}_q$ and $MQ(n, m, \mathbb{F}_q)$ be a system of MQ polynomials with n variables and m equations such that $F : \mathbb{F}_q^n \to \mathbb{F}_q^m$ of degree 2 over $\mathbb{F}_q$:*

$$MQ(n, m, \mathbb{F}_q) = \{F(x) = (f_1(x), \ldots, f_m(x))\} \tag{5.1}$$

$$F = \begin{cases} f_1(x_1, \ldots, x_n) = \sum_i^n \sum_j^n a_{i,j}^{(1)} x_i x_j + \sum_i^n b_i^{(1)} x_i \\ f_2(x_1, \ldots, x_n) = \sum_i^n \sum_j^n a_{i,j}^{(2)} x_i x_j + \sum_i^n b_i^{(2)} x_i \\ \vdots \\ f_m(x_1, \ldots, x_n) = \sum_i^n \sum_j^n a_{i,j}^{(m)} x_i x_j + \sum_i^n b_i^{(m)} x_i \end{cases} \tag{5.2}$$

Briefly,

$$f_s(x) = \sum_{i,j} a_{i,j}^{(s)} x_i x_j + \sum_i b_i^{(s)} x_i, \quad s \in \{1, \ldots, m\} \tag{5.3}$$

where $a_{i,j}^{(s)},\ b_i^{(s)} \in \mathbb{F}_q$ *for* $1 \leq s \leq m$.

Let F be a system of MQ polynomials, say $F \in MQ(n, m, \mathbb{F}_q)$ (multivariate quadratic function). For any given $v \in \mathbb{F}_q^m$, $F(x) = v$ represents a system of m quadratic equations in n variables.

At this point, the thing which it should be known is that there are 2 versions of the MQ problem: search version and decision version. To describe any version of the MQ problem, the definition of the polar form and the definition of the MQ relation become necessary. Let's define respectively:

Definition 13. *Let* $F \in MQ(n, m, \mathbb{F}_q)$. *Then the polar form of the function* F, *say* $G(x, y)$ *is defined by:*

$$G(x, y) = F(x + y) - F(x) - F(y) \tag{5.4}$$

such that $G(x, y)$ *is bilinear, i.e.,* $\forall u_1, u_2, v \in \mathbb{F}_q^n$,

$$G(u_1 + u_2, v) = G(u_1, v) + G(u_2, v) \tag{5.5}$$

$$G(v, u_1 + u_2) = G(v, u_1) + G(v, u_2). \tag{5.6}$$

Definition 14. *Let* $F \in MQ(n, m, \mathbb{F}_q)$, $v \in \mathbb{F}_q^m$, $s \in \mathbb{F}_q^n$. *The MQ relation which is a binary relation is defined as follows:*

$$R_{MQ(n,m,\mathbb{F}_q)} \subseteq \left(MQ(n, m, \mathbb{F}_q) \times \mathbb{F}_q^m\right) \times \mathbb{F}_q^n : ((F, v), s) \in R_{MQ(n,m,\mathbb{F}_q)} \ \Leftrightarrow \ F(s) = v. \tag{5.7}$$

Since the polar form and MQ relation have been defined, now we can pass on to the definition of MQ problem.

Definition 15. *Assume that* $m, n, q \in \mathbb{N}$.

Given any $MQ(F, v)$, *the* ***MQ (search) problem*** *is defined as follows:*
1. Assume $F \in MQ(n, m, \mathbb{F}_q)$, $v \in \mathbb{F}_q^m$ *are given.*
2. Find, if any, $s \in \mathbb{F}_q^n$ *such that* $((F, v), s) \in R_{MQ(n,m,\mathbb{F}_q)}$.

Given any $MQ(F, v)$, *the* ***MQ (decision) problem*** *is defined as follows:*
1. Assume $F \in MQ(n, m, \mathbb{F}_q)$, $v \in \mathbb{F}_q^m$ *are given.*
2. Is there any $s \in \mathbb{F}_q^n$ *such that* $((F, v), s) \in R_{MQ(n,m,\mathbb{F}_q)}$.

Notice that: It is reasonable to think any decision problem as a problem whose solution is of type "yes" or "no".

In the following process, identification schemes and digital signature schemes which are based on the **MQ problem** will be described. That's why, it is crucial to know that if we want to construct secure identification and digital signature schemes, then MQ problem has to be intractable even for quantum computers. Let's give the assumption which is widely believed about the intractability of the MQ problem.

Assumption 1. *Let $m, n, q \in \mathbb{N}$. Assume that F is randomly chosen system of quadratic polynomials from $MQ(n, m, \mathbb{F}_q)$ and s is randomly chosen n dimensional vector from $\mathbb{F}_q^n$. $\forall$ polynomial time quantum algorithm A given F and $v = F(s)$, it is difficult to solve MQ problem over $\mathbb{F}_q^n$, meaning that, it is difficult to find a solution s' to the $MQ(F, v)$ problem. More mathematically,*

$$Pr\left[\left((F,v),s'\right) \in R_{MQ(n,m,\mathbb{F}_q)} : \begin{array}{l} F \leftarrow_R MQ(n,m,\mathbb{F}_q) \\ s \leftarrow_R \mathbb{F}_q^n \\ ((F,v),s) \in R_{MQ(n,m,\mathbb{F}_q)} \\ s' \leftarrow A\left(1^k, F, v\right) \end{array}\right] = negl(k). \tag{5.8}$$

If A is thought as a polynomial algorithm, then the above assumption means that there is no known A for solving MQ problem even for quantum computers. It shows that MQ problem which is **NP-complete** is an intractable problem not only for classical algorithms but also for quantum algorithms as we desired.

5.2 CANONICAL $2n+1$-PASS IDENTIFICATION SCHEMES

[5] Let $IDS\left(1^k\right) = (KGen, \mathcal{P}, \mathcal{V})$ be an identification scheme with a security parameter k and $C_1, \ldots, C_n$ be n-challenge spaces of $IDS\left(1^k\right)$. $IDS\left(1^k\right)$ is called as $2n+1$-**pass identification scheme** if the prover and the verifier seperate their processes into $n+1$ groups: $\mathcal{P} = (P_0, P_1, \ldots, P_n)$ and $\mathcal{V} = (ChS_1, \ldots, ChS_n, Vf)$ such that:

- $P_0, P_1, \ldots, P_n$ are iterative algorithms meaning that the output of P_i is the input of P_{i+1} for $0 \leq i \leq n-1$. The input of P_0 is the secret key sk which is obtained from $KGen$ algorithm, say $(sk, pk) \leftarrow KGen$.
- The outputs of $P_0(sk)$ are the initial commitment "com" which is the first message will be sended to verifier $\mathcal{V}$ and a state "state" which is one of the inputs of P_1
- ChS_i generates the i^{th} challenge message ch_i by selecting randomly from challenge space C_i, i.e., $ch_i \leftarrow_R C_i$.
- $P_i(state, ch_i) \rightarrow (resp_i, state)$ where $resp_i$ is the i^{th} response of the prover $\mathcal{P}$ and the state $state$ is updated in each iteration so as to supply a state as an input for the next iteration for $1 \leq i \leq n$.

- $Vf\left(pk, com, ch_1, resp_1, \ldots, ch_n, resp_n\right) \rightarrow b$ where pk is the public-key which is produced by $KGen$ algorithm $((sk, pk) \leftarrow KGen)$ and b is a bit which defines verifier $\mathcal{V}$'s final decision will be "accept" or "reject".

- The whole transcript of this protocol, denoted by $trans\left(\langle P\left(sk\right), V\left(pk\right)\rangle\right)$ is the following set: $\{com, ch_1, resp_1, \ldots, ch_n, resp_n\}$.
  In other words, $\{com, ch_1, resp_1, \ldots, ch_n, resp_n\}$ is the set of all messages exchanged between $\mathcal{P}$ and $\mathcal{V}$.

Two specific choices exist for canonical $2n+1$-pass identification schemes: canonical 3-pass IDS and canonical 5-pass IDS where $n = 1$ and $n = 2$ respectively.
Canonical 3-pass IDS $(n = 1)$ can be thought as a standard choice of both canonical $2n+1$-pass IDS and its corresponding signature scheme. Canonical 5-pass IDS $(n = 2)$ becomes fundamental for us, because in the following process, our aim is to analyze the construction of the Multivariate Quadratic Digital Signature Scheme, say "MQDSS" which based on the Fiat-Shamir transform of a special kind of canonical 5-pass IDS whose name is the Sakumoto-Shirai-Hiwatari (SSH) 5-pass IDS.

5.2.1 CANONICAL 3-PASS IDENTIFICATION SCHEME

In the construction of canonical 3-pass IDS, the most essential point which should be known is the number of messages that are exchanged between the prover and the verifier. As it can be understood from the name of the IDS, this number equals to 3. Since $n = 1$, $\exists$ one challenge space C_1 in this framework, besides prover and verifier split into $(n+1)$ subroutines such that $\mathcal{P} = (P_0, P_1)$ and $\mathcal{V} = (ChS_1, Vf)$.

The whole transcript of the scheme $Trans\left(\langle P\left(sk\right), V\left(pk\right)\rangle\right)$ is the following set: $\{com, ch_1, resp_1\}$. Therefore, as a final step, verifier should check the IDS by using the public-key pk and the above transcript i.e., $\mathcal{V}$ checks whether $Vf\left(pk, com, ch_1, resp_1\right) = b$ or not and the bit b gives the result: "accept" or "reject". Let's define more mathematically:

Table 5.1: Canonical 3-pass IDS [5]

$\mathcal{P}$		$\mathcal{V}$
$(state, com) \longleftarrow P_0\left(sk\right)$		
	$\xrightarrow{com}$	
		$ch_1 \longleftarrow_R ChS_1\left(1^k\right)$
	$\xleftarrow{ch_1}$	
$resp_1 \longleftarrow P_1\left(state, ch_1\right)$		
	$\xrightarrow{resp_1}$	
		$b \longleftarrow Vf\left(pk, com, ch_1, resp_1\right)$

5.2.2 CANONICAL 5-PASS IDENTIFICATION SCHEME

In the construction of canonical 5-pass IDS, the most essential point which should be known is the number of messages that are exchanged between the prover and the verifier. As it can be understood from the name of the IDS, this number equals to 5. Since $n = 2$, $\exists\, 2$ challenge spaces C_1 and C_2 in this framework, besides prover and verifier split into $3 = (n+1)$ subroutines such that $\mathcal{P} = (P_0, P_1, P_2)$ and $\mathcal{V} = (ChS_1, ChS_2, Vf)$.

The whole transcript of the scheme $Trans\left(\langle P\left(sk\right), V\left(pk\right)\rangle\right)$ is the following set: $\{com, ch_1, ch_2, resp_1, resp_2\}$. Therefore, as a final step, verifier should check the IDS by using the public-key pk and the above transcript i.e., $\mathcal{V}$ checks whether or not $Vf\left(pk, com, ch_1, ch_2, resp_1, resp_2\right) = b$ and the bit b gives the result: "accept" or "reject". Let's define more mathematically:

Table 5.2: Canonical 5-pass IDS [5]

$\mathcal{P}$		$\mathcal{V}$
$(state, com) \longleftarrow P_0\left(sk\right)$		
	$\xrightarrow{com}$	
		$ch_1 \longleftarrow_R ChS_1\left(1^k\right)$
	$\xleftarrow{ch_1}$	
$(state, resp_1) \longleftarrow P_1\left(state, ch_1\right)$		
	$\xrightarrow{resp_1}$	
		$ch_2 \longleftarrow_R ChS_2\left(1^k\right)$
	$\xleftarrow{ch_2}$	
$resp_2 \longleftarrow P_2\left(state, ch_2\right)$		
	$\xrightarrow{resp_2}$	
		$b \longleftarrow Vf\left(pk, com, ch_1, ch_2, resp_1, resp_2\right)$

Now we will define a special kind of 5-pass identification scheme whose name is $q2$ **Identification Scheme**.

Definition 16. *[5] Let $IDS\left(1^k\right) = (KGen, \mathcal{P}, \mathcal{V})$ be a canonical 5-pass identification scheme with a security parameter k. Since $n = 2$, $\exists\, 2$ challenge spaces C_1 and C_2 in this framework. If $\|C_1\| = q$ and $\|C_2\| = 2$, then $IDS\left(1^k\right)$ is said to be $q2$-**identification scheme**.*

5.2.3 SECURITY PROPERTIES OF $2n+1$-PASS IDENTIFICATION SCHEMES

In the process of defining identification schemes, the security properties of IDS have been explained, but these properties are not inclusive of special n-soundness, special soundness $(n = 1)$ and $q2$-extractor properties which are only special to canonical $2n + 1$-pass IDS.

Since we have described canonical $2n+1$-pass IDS, now we can pass on to the definitions of these properties.

5.2.3.1 Special n-soundness

The security property of special n-soundness is an important concept for the construction of secure digital signature schemes, because if we transform an identification scheme which fulfills the properties of honest verifier zero knowledge and special soundness to a digital signature scheme, then this signature scheme will be EU-CMA secure which is a security property was defined before. The property of special soundness is only related with canonical 3-pass IDS, but special n-soundness embraces all canonical $2n+1$-pass IDS. That's why, special n-soundness is a generalized form of the special soundness $(n=1)$ to all $n \in \mathbb{N}$. Let's define the special n-soundness.

Definition 17. *[6] Let $IDS\left(1^k\right) = (KGen, \mathcal{P}, \mathcal{V})$ be a canonical $2n+1$-pass identification scheme with security parameter k. If $\exists$ a probabilistic polynomial time algorithm ϵ, the extractor that upon input of two valid transcripts:*

$$trans = (com, ch_1, resp_1, \ldots, ch_{n-1}, resp_{n-1}, ch_n, resp_n) \tag{5.9}$$

$$trans' = \left(com, ch_1, resp_1, \ldots, ch_{n-1}, resp_{n-1}, ch'_n, resp'_n\right), \tag{5.10}$$

*where $ch_n \neq ch'_n$ and the corresponding public key pk, outputs a matching secret key sk for pk with non-negligible success probability, then $IDS\left(1^k\right)$ is said to satisfy the property of **special n-soundness**.*

5.2.3.2 Special Soundness

For the property of special soundness, the thing which we need to focus on is the value of n. Since this property is special to canonical 3-pass IDS, n equals to 1.

Definition 18. *[5] Let $IDS\left(1^k\right) = (KGen, \mathcal{P}, \mathcal{V})$ be a canonical 3-pass identification scheme with a security parameter k and a key relation R. Assume that for $IDS\left(1^k\right)$, A is a polynomial time algorithm whose inputs are the security parameter k and the public-key pk, outputs are 2 valid transcripts which correspond to pk with non-negligible probability:*

$$trans = (com, ch_1, resp_1) \tag{5.11}$$

$$trans' = \left(com, ch'_1, resp'_1\right), \tag{5.12}$$

*where $ch_1 \neq ch'_1$. If $\exists$ a polynomial time algorithm ϵ, the extractor, whose inputs are pk and A, output is a secret key sk so that $(pk, sk) \in R$ with non-negligible success probability in k, then $IDS\left(1^k\right)$ is said to satisfy the property of **special soundness**.*

5.2.3.3 $q2$-extractor

The property of $q2$-extractor is a specific kind of special n-soundness such that $IDS\left(1^k\right)$ is a $q2$-Identification scheme that is a variant of canonical 5-pass IDS which was defined before. That's why, in this case, n equals to 2.

Definition 19. *[5] Let $IDS\left(1^k\right) = (KGen, \mathcal{P}, \mathcal{V})$ be a $q2$-Identification scheme with a security parameter k and a key relation R. Assume that for $IDS\left(1^k\right)$, A is a polynomial time algorithm whose inputs are the security parameter k and the public-key pk, outputs are 4 valid transcripts which correspond to pk with non-negligible probability:*

$$trans^{(1)} = (com, ch_1, resp_1, ch_2, resp_2)\,, \tag{5.13}$$

$$trans^{(2)} = \left(com, ch_1, resp_1, ch_2', resp_2'\right), \tag{5.14}$$

$$trans^{(3)} = \left(com, ch_1', resp_1', ch_2, resp_2\right), \tag{5.15}$$

$$trans^{(4)} = \left(com, ch_1', resp_1', ch_2', resp_2'\right), \tag{5.16}$$

*where $ch_1 \neq ch_1'$ and $ch_2 \neq ch_2'$. If $\exists$ a polynomial time algorithm ϵ, the extractor, whose inputs are pk and A, output is a secret key sk so that $(pk, sk) \in R$ with non-negligible success probability in k, then $IDS\left(1^k\right)$ is said to satisfy the property of $q2$-**extractor**.*

5.3 THE SAKUMOTO-SHIRAI-HIWATARI (SSH) 3-PASS AND 5-PASS IDENTIFICATION SCHEMES

Up to this point, we have examined the construction of the MQ problem and canonical $2n+1$-pass identification schemes. Now let's analyze their one of the applications. Sakumoto, Shirai and Hiwatari proposed a specific application for both canonical 3-pass and 5-pass identification schemes which use the hardness of the MQ problem [35]. Their public-key identification schemes base on multivariate quadratic polynomials, are called as SSH 3-pass and SSH 5-pass IDS.

When we investigate the previous public-key identification schemes, their security depends on some problems in multivariate cryptography such as **MinRank** [7], **Isomorphism of Polynomials (IP)** [29], **Extended IP** [13] or **IP with partial knowledge** [33] problems. On the other hand, many schemes have been broken since they rely on the hardness of problems like MinRank or IP instead of relying on the intractability of the MQ problems. As the examples of these broken problems, **Oil-and-Vinegar**, **SFLASH**, **MQQ-Sig**, **Enhanced TTS** and **Enhanced STS** can be given [35]. The significance of SSH 5-pass IDS is its dependence for the hardness of the MQ problem and the security level of the non-interactive commitment scheme which is statistically hiding and computationally binding.

Notice that SSH 3-pass IDS has the property of **statistical zero knowledge** and SSH 5-pass IDS has the property of **argument of knowledge (soundness)**. As we stated before, there is no known polynomial time quantum algorithm for solving any random instance of the MQ

problem which contains multivariate quadratic polynomials. That's why, it is possible to accept MQ function as a one-way function whose input and output sizes are short. When we analyze its complexity, for the generic attacks which make use of the Gröbner basis, it is exponential in time and space. If we consider the best known attack against the MQ function over $\mathbb{F}_2$ whose input and output sizes are 84-bit and 80-bit respectively, then its complexity requires $2^{88.7}$ bit operations.

After the description of the SSH 3-pass and 5-pass IDS, we will pass on to the Fiat-Shamir transform which is needed to transform these identification schemes to digital signature schemes. The technique for this transformation is to get sequential or parallel composition of the identification schemes by using optimal number of rounds. If the structural tools SSH 3-pass and 5-pass IDS to construct DDS satisfy statistical zero knowledge and argument of knowledge respectively and the commitment schemes which are fundamental parts of these IDS fulfill statistically hiding and computationally binding properties, then the corresponding digital signature schemes will be secure against impersonation under passive attacks. Since we know that SSH 3-pass and 5-pass IDS and commitment schemes satisfy the desired properties, we expect from the corresponding DDS to have the security properties that were mentioned above. Yet, the security under active attacks of the acquired digital signature schemes is not overtly known.

Let's describe the framework and the key idea of the SSH 3-pass and 5-pass IDS. In the process of constructing SSH 3-pass and 5-pass IDS, Sakumoto, Shirai and Hiwatari use the idea of [35] splitting the secret key sk and benefit from the polar form $G(x,y)$ of $F(x)$ $(F \in MQ(n,m,\mathbb{F}_q))$. Let $(sk,pk) \leftarrow KGen$. Since these schemes are based on the MQ problem, the secret key sk and the public-key pk will be s and $v = F(s)$ respectively. According to this idea, they first seperate the secret key s into 2 parts:

$$s = r_0 + r_1. \tag{5.17}$$

Then, the public key $v = F(s)$ can be written in terms of polar form $G(x,y)$ of $F(x)$ such that

$$v = F(s) = F(r_0 + r_1) = F(r_0) + F(r_1) + G(r_0, r_1) \tag{5.18}$$

since

$$G(x,y) = F(x+y) - F(x) - F(y). \tag{5.19}$$

However v includes the term $G(r_0, r_1)$ whose inputs are both r_0 and r_1, meaning that the polar form depends on both parts of the secret key s. That's why, they further split r_0 and $F(r_0)$ such that

$$\alpha r_0 = t_0 + t_1, \tag{5.20}$$

$$\alpha F(r_0) = e_0 + e_1. \tag{5.21}$$

Then,

$$\alpha v = \underbrace{\alpha F(r_0)}_{=e_0+e_1} + \alpha F(r_1) + \underbrace{\alpha G(r_0, r_1)}_{G(\alpha r_0, r_1)}, \tag{5.22}$$

where

$$G(\underbrace{\alpha r_0}_{t_0+t_1}, r_1) = G(t_0 + t_1, r_1) = G(t_0, r_1) + G(t_1, r_1) \tag{5.23}$$

Since the polar form is bilinear, $G(t_0 + t_1, r_1)$ can be represented as: $G(t_0, r_1) + G(t_1, r_1)$. If this representation is substituted into the right hand side of the equation (5.22), then the following equation is obtained:

$$\begin{aligned} \alpha v &= e_0 + e_1 + \alpha F(r_1) + G(t_0, r_1) + G(t_1, r_1) \\ &= \underbrace{(e_1 + \alpha F(r_1) + G(t_1, r_1))}_{(r_1, t_1, e_1)} + \underbrace{(e_0 + G(t_0, r_1))}_{(r_1, t_0, e_0)}. \end{aligned} \tag{5.24}$$

When the summation is seperated into two parts as it can be seen in the above formula, inputs of the first summand will be (r_1, t_1, e_1) and inputs of the second summand will be (r_1, t_0, e_0). Hence, in the case of an access to any of these two summands, the secret key s will still be secret. Since $(pk, sk) \in ((F, v), s)$, this key pair has to satisfy MQ key relation i.e., $((F, v), s) \in R_{MQ}$. Now let's define the commitment scheme, com which is a tool of SSH 3-pass and 5-pass IDS. There exist two phases for the com which satisfies the properties of statistically hiding and computationally binding. Firstly, prover $\mathcal{P}$ computes a value c which will be sended to the verifier $\mathcal{V}$ by using the function com such that $c \leftarrow com(s, r)$ where s is a string and r is a random string. Secondly, $\mathcal{P}$ sends (s, r) to $\mathcal{V}$ so that $\mathcal{V}$ verifies $c = com(s, r)$. Because of the security properties of com, it is not possible to distinguish two commitment values which are generated from two distinct strings s_1 and s_2, from each other. Furthermore, after $\mathcal{P}$ sends the commitment value c to $\mathcal{V}$, $\mathcal{P}$ can not change his or her committed string s in polynomial time due to the same reason. Since cryptographic hash functions are used in the construction of the commitment schemes, then we expect from com to satisfy these two security properties [5, 35].

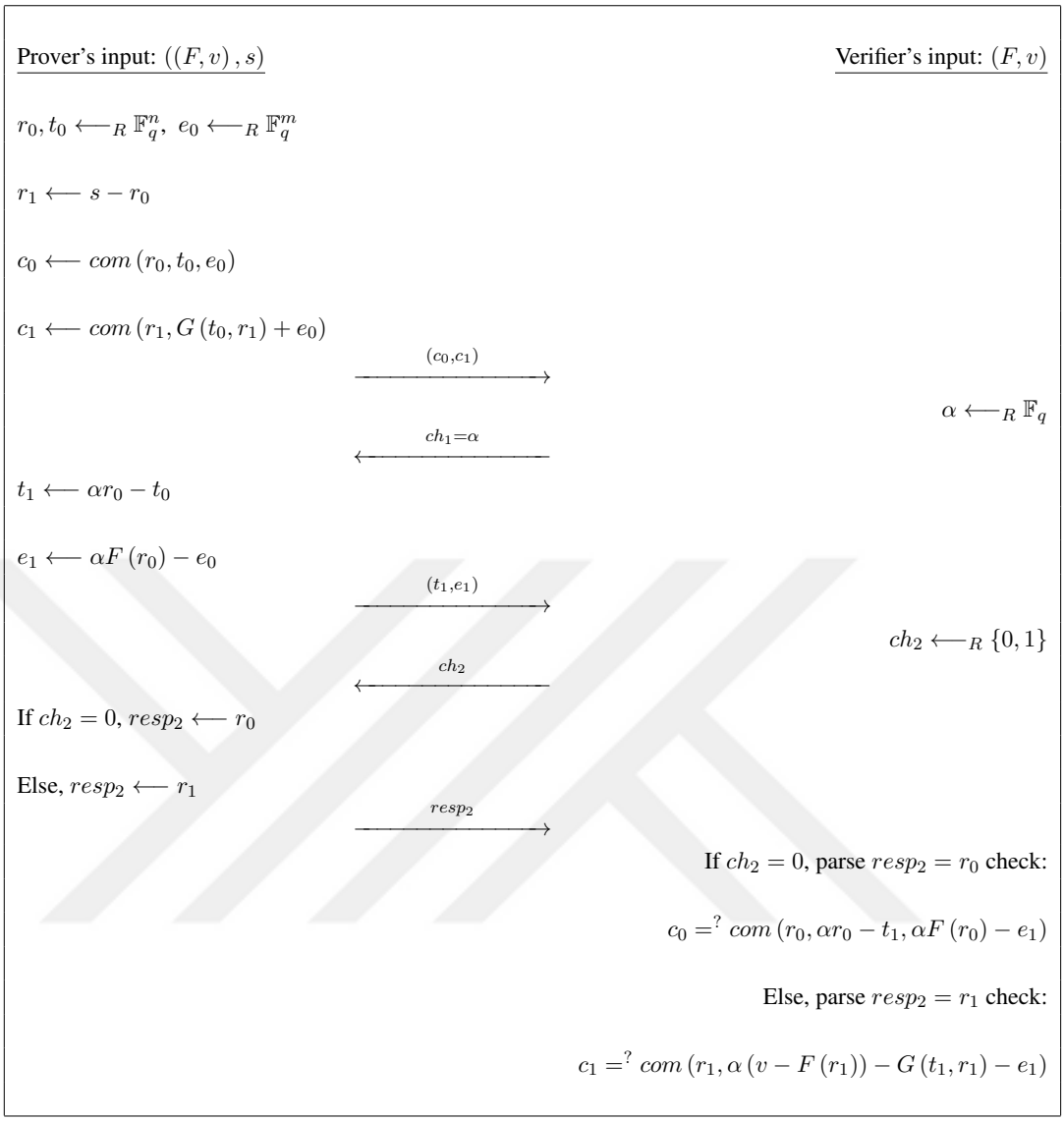
Note that: In the description of only SSH 5-pass IDS, the function com will get 3 inputs instead of 2 in the process of computing the first commitment value c_0.

Let's describe the SSH 3-pass and 5-pass IDS respectively:

Table 5.3: The SSH 3-pass IDS [35]

Prover's input: $((F, v), s)$		Verifier's input: (F, v)
$r_0, t_0 \longleftarrow_R \mathbb{F}_q^n,\ e_0 \longleftarrow_R \mathbb{F}_q^m$		
$r_1 \longleftarrow s - r_0,\ t_1 \longleftarrow r_0 - t_0$		
$e_1 \longleftarrow F(r_0) - e_0$		
$c_0 \longleftarrow com(r_1, G(t_0, r_1) + e_0)$		
$c_1 \longleftarrow com(t_0, e_0)$		
$c_2 \longleftarrow com(t_1, e_1)$		
	$\xrightarrow{(c_0, c_1, c_2)}$	
		$ch_1 \longleftarrow_R \{0, 1, 2\}$
	$\xleftarrow{ch_1}$	
If $ch_1 = 0$, $resp_1 \longleftarrow (r_0, t_1, e_1)$		
If $ch_1 = 1$, $resp_1 \longleftarrow (r_1, t_1, e_1)$		
If $ch_1 = 2$, $resp_1 \longleftarrow (r_1, t_0, e_0)$		
	$\xrightarrow{resp_1}$	
		If $ch_1 = 0$, parse $resp_1 = (r_0, t_1, e_1)$ check:
		$c_1 =^? com(r_0 - t_1, F(r_0) - e_1)$
		$c_2 =^? com(t_1, e_1)$
		If $ch_1 = 1$, parse $resp_1 = (r_1, t_1, e_1)$ check:
		$c_0 =^? com(r_1, v - F(r_1) - G(t_1, r_1) - e_1)$
		$c_2 =^? com(t_1, e_1)$
		If $ch_1 = 2$, parse $resp_1 = (r_1, t_0, e_0)$ check:
		$c_0 =^? com(r_1, G(t_0, r_1) + e_0)$
		$c_1 =^? com(t_0, e_0)$

Table 5.4: The SSH 5-pass IDS [35]



Theorem 6. *[5, 35] The security properties of the SSH* 5*-pass IDS are defined in the following:*

- *The SSH* 5*-pass IDS has a key relation* $R_{MQ(n,m,\mathbb{F}_q)}$.
- *If the MQ search problem is hard on average, then the SSH* 5*-pass IDS is key-one-wayness.*
- *The SSH* 5*-pass IDS is perfectly correct i.e.,*
  $\forall (pk, sk) \leftarrow KGen,\ P[\langle \mathcal{P}(sk), \mathcal{V}(pk) \rangle = 1] = 1.$
- *If the commitment scheme* com *is computationally hiding, then the SSH* 5*-pass IDS is computationally honest verifier zero knowledge.*
- *If the commitment scheme* com *is computationally binding, then the SSH* 5*-pass IDS is argument of knowledge (soundness) for* $R_{MQ(n,m,\mathbb{F}_q)}$ *with knowledge error* $\frac{1}{2} + \frac{1}{2q}$.
- *If the commitment scheme* com *is computationally binding, then the SSH* 5*-pass IDS is sound with soundness error* $\frac{1}{2} + \frac{1}{2q}$.
- *If the commitment scheme* com *is computationally binding, then the SSH* 5*-pass IDS has a* $q2$*-Extractor.*

5.4 THE FIAT-SHAMIR TRANSFORM

As we explained before, identification schemes can be thought as a structural tool to construct digital signature schemes. In this process, the thing which will be needed is the existence of a mechanism or an algorithm which transforms an identification scheme to a digital signature scheme. Furthermore, the obtained DSS needs to be secure in both ROM and QROM. The most common example of this transformation in which the acquired DSS satisfies the desired security properties, is applied to the canonical $2n+1$-pass identification schemes. Let's describe this transformation mechanism whose name is the **Fiat-Shamir transform**, through the canonical 3-pass and 5-pass IDS respectively since the desription of the Fiat-Shamir transform of $2n+1$-pass IDS is too complex and not common.

5.4.1 THE FIAT-SHAMIR TRANSFORM OF CANONICAL 3-PASS IDS

Definition 20. *[5] Assume $k \in \mathbb{N}$ and $IDS\left(1^k\right) = (KGen_{IDS}, \mathcal{P}, \mathcal{V})$ is a canonical 3-pass identification scheme with a security parameter k where $\mathcal{P} = (P_0, P_1)$ and $\mathcal{V} = (ChS_1, Vf_{IDS})$.*

1. *Suppose that $IDS\left(1^k\right)$ satisfies the security property of **soundness** with soundness error κ.*

2. *Then select the number of rounds, say r to create a $DSS\left(1^k\right)$ such that $\kappa^r = negl\,(k)$ and its challenge space C^r has exponential size in k.*

3. *Let's get the r rounds sequential or parallel composition of $IDS\left(1^k\right)$. Then, $IDS^{(r)}$ will be obtained.*

4. *Now select a cryptographic hash function H such that $H : \{0,1\}^* \to C^r$.*

5. *Then the corresponding $DSS\left(1^k\right)$ which is obtained from $IDS\left(1^k\right)$ is the following triplet: $(KGen, Sign, Vf)$. Let's define these 3 algorithms:*

Algorithm 4 Fiat-Shamir Key Generation

```
procedure KGEN( )
    (pk, sk) ← KGen_IDS
    return (pk, sk)
end procedure
```

Algorithm 5 Fiat-Shamir Signature Generation

procedure SIGN(sk,M)
  for $j \in \{1, \ldots, r\}$ **do**
    $\left(state^{(j)}, com^{(j)}\right) \leftarrow P_0^{(j)}\left(sk\right)$
  end for
  $state := \left(state^{(1)}, \ldots, state^{(r)}\right)$
  $com := \left(com^{(1)}, \ldots, com^{(r)}\right)$
  $\sigma_0 := com$
  $ch \leftarrow H\left(pk, M, \sigma_0\right)$
  Parse ch as $ch = \left(ch^{(1)}, ch^{(2)}, \ldots, ch^{(r)}\right)$, $ch^{(j)} \in C$
  for $j \in \{1, \ldots, r\}$ **do**
    $resp^{(j)} \leftarrow P_1^{(j)}\left(state^{(j)}, ch^{(j)}\right)$
  end for
  $resp := \left(resp^{(1)}, \ldots, resp^{(r)}\right)$
  $\sigma_1 := resp$
  return $\sigma = (\sigma_0, \sigma_1)$
end procedure

Algorithm 6 Fiat-Shamir Signature Verification

procedure VF(pk,σ,M)
  Parse $\sigma = (\sigma_0, \sigma_1)$
  Parse σ_0 as $\sigma_0 := \left(com^{(1)}, \ldots, com^{(r)}\right)$
  $ch \leftarrow H\left(pk, M, \sigma_0\right)$
  Parse ch as $ch = \left(ch^{(1)}, ch^{(2)}, \ldots, ch^{(r)}\right)$, $ch^{(j)} \in C$
  Parse σ_1 as $\sigma_1 = \left(resp^{(1)}, \ldots, resp^{(r)}\right)$
  for $j \in \{1, \ldots, r\}$ **do**
    $b^{(j)} \leftarrow Vf_{IDS}^{(j)}\left(pk, com^{(j)}, ch^{(j)}, resp^{(j)}\right)$
  end for
  $b \leftarrow b^{(1)} \wedge b^{(2)} \wedge \cdots \wedge b^{(r)}$
  return b
end procedure

5.4.2 THE FIAT-SHAMIR TRANSFORM OF CANONICAL 5-PASS IDS

Definition 21. *[5] Assume $k \in \mathbb{N}$ and $IDS\left(1^k\right) = (KGen_{IDS}, \mathcal{P}, \mathcal{V})$ is a canonical 5-pass identification scheme with a security parameter k in which $\mathcal{P} = (P_0, P_1, P_2)$ and $\mathcal{V} = (ChS_1, ChS_2, Vf_{IDS})$.*

1. *Suppose that $IDS\left(1^k\right)$ satisfies the security property of **soundness** with soundness error κ.*

2. *Then select the number of rounds, say r to create a $Dss\left(1^k\right)$ such that $\kappa^r = negl\left(k\right)$ and its challenge spaces C_1^r, C_2^r have exponential size in k.*

3. *Let's get the r rounds sequential or parallel composition of $IDS\left(1^k\right)$. Then, $IDS^{(r)}$ will be obtained.*

4. *Now select the cryptographic hash functions H_1 and H_2 such that $H_1 : \{0,1\}^* \rightarrow C_1^r$ and $H_2 : \{0,1\}^* \rightarrow C_2^r$.*

5. *Then the corresponding $Dss\left(1^k\right)$ which is obtained from $IDS\left(1^k\right)$ is the following triplet: $(KGen, Sign, Vf)$. Let's define these 3 algorithms:*

Algorithm 7 Fiat-Shamir Key Generation

procedure KGEN( )
  $(pk, sk) \leftarrow KGen_{IDS}$
  return (pk, sk)
end procedure

Algorithm 8 Fiat-Shamir Signature Generation

procedure SIGN(sk,M)
  for $j \in \{1, \ldots, r\}$ **do**
    $\left(state^{(j)}, com^{(j)}\right) \leftarrow P_0^{(j)}\left(sk\right)$
  end for
  $state := \left(state^{(1)}, \ldots, state^{(r)}\right)$
  $com := \left(com^{(1)}, \ldots, com^{(r)}\right)$
  $\sigma_0 := com$
  $h_1 \leftarrow H_1\left(pk, M, \sigma_0\right)$
  Parse h_1 as $h_1 = \left(ch_1^{(1)}, ch_1^{(2)}, \ldots, ch_1^{(r)}\right)$, $ch_1^{(j)} \in C_1$
  for $j \in \{1, \ldots, r\}$ **do**
    $\left(state^{(j)}, resp_1^{(j)}\right) \leftarrow P_1^{(j)}\left(state^{(j)}, ch_1^{(j)}\right)$
  end for
  $state := \left(state^{(1)}, \ldots, state^{(r)}\right)$
  $resp_1 := \left(resp_1^{(1)}, \ldots, resp_1^{(r)}\right)$
  $\sigma_1 := resp_1$
  $h_2 \leftarrow H_2\left(pk, M, \sigma_0, \sigma_1\right)$
  Parse h_2 as $h_2 = \left(ch_2^{(1)}, ch_2^{(2)}, \ldots, ch_2^{(r)}\right)$, $ch_2^{(j)} \in C_2$
  for $j \in \{1, \ldots, r\}$ **do**
    $resp_2^{(j)} \leftarrow P_2^{(j)}\left(state^{(j)}, ch_2^{(j)}\right)$
  end for
  $resp_2 := \left(resp_2^{(1)}, \ldots, resp_2^{(r)}\right)$
  $\sigma_2 := resp_2$
  return $\sigma = (\sigma_0, \sigma_1, \sigma_2)$
end procedure

Algorithm 9 Fiat-Shamir Signature Verification

procedure VF(pk,σ,M)
  Parse $\sigma = (\sigma_0, \sigma_1, \sigma_2)$
  Parse σ_0 as $\sigma_0 := \left(com^{(1)}, \ldots, com^{(r)}\right)$
  $h_1 \leftarrow H_1\left(pk, M, \sigma_0\right)$
  Parse h_1 as $h_1 = \left(ch_1^{(1)}, ch_1^{(2)}, \ldots, ch_1^{(r)}\right), ch_1^{(j)} \in C_1$
  Parse σ_1 as $\sigma_1 = \left(resp_1^{(1)}, \ldots, resp_1^{(r)}\right)$
  $h_2 \leftarrow H_2\left(pk, M, \sigma_0, \sigma_1\right)$
  Parse h_2 as $h_2 = \left(ch_2^{(1)}, ch_2^{(2)}, \ldots, ch_2^{(r)}\right), ch_2^{(j)} \in C_2$
  Parse σ_2 as $\sigma_2 = \left(resp_2^{(1)}, \ldots, resp_2^{(r)}\right)$
  for $j \in \{1, \ldots, r\}$ **do**
    $b^{(j)} \leftarrow Vf_{IDS}^{(j)}\left(pk, com^{(j)}, ch_1^{(j)}, resp_1^{(j)}, ch_2^{(j)}, resp_2^{(j)}\right)$
  end for
  $b \leftarrow b^{(1)} \wedge b^{(2)} \wedge \cdots \wedge b^{(r)}$
  return b
end procedure

5.4.3 SECURITY ANALYSIS

Let's analyze the security of the **Fiat-Shamir transform**. The most significant thing which we need to know related with the Fiat-Shamir transform is that the digital signature scheme which is constructed by using this transform, is provably secure in the ROM against classical adversaries. Furthermore, there exists a proof which uses the ROM and the forking lemma whose techniques consist of adaptively programming of the random oracle and rewinding of the attacker such that if we transform an IDS which is HVZK and satisfies the security property of special soundness, then the corresponding DSS will be EU-CMA secure.

As a different proof, Abdalla proposed that the identification scheme is IMP-PA secure if and only if the corresponding digital signature scheme that is obtained by the Fiat-Shamir transform, is EU-CMA secure. On the other hand, if we assume the existence of quantum adversaries, then they will query the random oracle in superposition. That's why, the security of the Fiat-Shamir transform in the QROM is not clear, meaning that it is not easy to generalize results for the ROM to the QROM.

Recently **Jelle Don, Serge Fehr, Christian Majenz and Christian Schaffner** proposed that if the IDS satisfies security properties of the standard soundness and the proof of knowledge, then the Fiat-Shamir transform preserves these properties even when enabling quantum attacks. In other words, for any IDS which is proof of knowledge and satisfies soundness, the corresponding Fiat-Shamir signature scheme is secure in the QROM. Also they showed the security of **Fish** that is the Fiat-Shamir variant of **Picnic** which is a NIST candidate, in the QROM [5, 9, 15].

Since the most popular canonical 5-pass IDS is the $q2$-IDS, let's give a theorem associated with the security of $q2$-signature schemes which are obtained by the Fiat-Shamir transform of $q2$-Identificatin schemes:

Theorem 7. *[5] Let $k \in \mathbb{N}$ and $IDS\left(^{k}\right)$ be a $q2$-Identification scheme with a security parameter k and a key relation R. If $IDS\left(1^{k}\right)$ has a $q2$-extractor ϵ, is KOW secure and HVZK, then the corresponding $q2$-signature scheme $\left(q2 - Dss\left(1^{k}\right)\right)$ which is attained by the Fiat-Shamir transform of the $IDS\left(1^{k}\right)$, will be EU-CMA secure.*

CHAPTER 6

MULTIVARIATE QUADRATIC DIGITAL SIGNATURE SCHEME (MQDSS)

In 2018, Ming-Shing Chen, Andreas Hülsing, Joost Rijneveld, Simona Samardjiska and Peter Schwabe proposed a new digital signature scheme whose name is **Multivariate Quadratic Digital Signature Scheme (MQDSS)**. Since specialists expect quantum computers in the near future, quantum resistant cryptographic algorithms started to have a great importance. That's why, NIST organized a competition for the post-quantum algorithms which contains multivariate public-key, lattice-based, hashed-based, isogeny-based and code-based cryptosystems. In this competiton, MQDSS which is a NIST candidate in the area of multivariate public key cryptosytems, passed to the second round.

The security of the MQDSS is based on the intractability of the MQ problem and the security level of the non-interactive commitment schemes. In the construction of the MQDSS, they benefited from the MQ Problem, non-interactive commitment schemes which are statistically hiding and computationally binding, SSH 5-pass identification scheme and the Fiat-Shamir transform. In this section of the thesis, we focus on the MQDSS algorithm. Let's start to introduce MQDSS by giving preliminary parameters and functions, the general description, optimized parameter sets, the detailed description, the classical and the quantum algorithms for solving the MQ problem and the security analysis respectively.

6.1 MQDSS PRELIMINARY PARAMETERS AND FUNCTIONS

Parameters of **MQDSS**-q-n [5]:

- $k \in \mathbb{N}^+$: the security parameter,
- $n \in \mathbb{N}^+$: the number of variables and equations of the system F,
- $q \in \mathbb{N}^+$: the order of the finite field $\mathbb{F}_q$ (q is a prime or a prime power),
- $r \in \mathbb{N}^+$: the number of rounds $\left(\text{normally } r = \left\lceil k/\log_2 \frac{2q}{q+1} \right\rceil\right)$.

Auxiliary functions of **MQDSS**-q-n [5]:

- $PRG_{sk} : \{0,1\}^k \rightarrow \{0,1\}^{3k}$: a pseudorandom generator which is utilized to produce three seeds,
- $PRG_s : \{0,1\}^k \rightarrow \{0,1\}^{n\lceil \log_2 q \rceil}$: a pseudorandom generator which is utilized to produce the secret key,
- $PRG_{rte} : \{0,1\}^k \times \{0,1\}^* \rightarrow \{0,1\}^{3rn\lceil \log_2 q \rceil}$: a pseudorandom generator which is utilized to produce pseudorandom values for the signature generation,
- $XOF_F : \{0,1\}^k \rightarrow \{0,1\}^{F_{len}}$: an extendable output function which is utilized to produce a multivariate system F through expanding a seed value that is generated by PRG_{sk} such that:
  $q = 2 \Rightarrow F_{len} = n \cdot \left(\frac{n\cdot(n-1)}{2} + n\right), q > 2 \Rightarrow F_{len} = n \cdot \left(\frac{n\cdot(n+1)}{2} + n\right) \lceil \log_2 q \rceil$,
- $\mathcal{H} : \{0,1\}^* \rightarrow \{0,1\}^{2k}$: a cryptographic hash fuction,
- $H_1 : \{0,1\}^* \rightarrow \mathbb{F}_q^r$: a cryptographic hash fuction,
- $H_2 : \{0,1\}^* \rightarrow \{0,1\}^r$: a cryptographic hash fuction,
- $Com_0 : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \rightarrow \{0,1\}^{2k}$: a string commitment function,
- $Com_1 : \mathbb{F}_q^n \times \mathbb{F}_q^n \rightarrow \{0,1\}^{2k}$: a string commitment function.

6.2 GENERAL DESCRIPTION OF THE MQDSS

Since MQDSS-q-n is a signature algorithm, it can be considered as a triplet of the following algorithms: $(KGen, Sign, Vf)$. Let's describe these 3 algorithms respectively [5].

REMARK

- S_F: a seed of k bits (By using S_F as an input, the MQ function F will be generated.),
- S_s: a seed of k bits (By using S_s as an input, the secret input s of the MQ function F will be generated.),
- S_{rte}: a seed of k bits (By using S_{rte} as an input, parameters $r_0^{(i)}$, $t_0^{(i)}$, $e_0^{(i)}$, where $i \in \{1, \ldots, r\}$ which are needed for the signature generation will be generated.),
- M: the message which will be signed,
- $G : \mathbb{F}_q^n \times \mathbb{F}_q^n \rightarrow \mathbb{F}_q^n$: the polar form of the MQ function F.

Algorithm 10 MQDSS-q-n Key Generation

procedure KGEN( )
  $sk \leftarrow_R \{0,1\}^k$
  $S_F, S_s, S_{rte} \leftarrow PRG_{sk}(sk)$
  $F \leftarrow XOF_F(S_F)$
  $s \leftarrow PRG_s(S_s)$
  $v \leftarrow F(s)$
  $pk := (S_F, v)$
  return (pk, sk)
end procedure

Algorithm 11 MQDSS-q-n Signature Generation

procedure SIGN(sk,M)
  $S_F, S_s, S_{rte} \leftarrow PRG_{sk}(sk)$
  $F \leftarrow XOF_F(S_F)$
  $s \leftarrow PRG_s(S_s)$
  $pk := (S_F, F(s))$
  $R \leftarrow \mathcal{H}(sk||M)$
  $D \leftarrow \mathcal{H}(pk||R||M)$
  $r_0^{(1)}, \ldots, r_0^{(r)}, t_0^{(1)}, \ldots, t_0^{(r)}, e_0^{(1)}, \ldots, e_0^{(r)} \leftarrow PRG_{rte}(S_{rte}, D)$
  for $j \in \{1, \ldots, r\}$ **do**
    $r_1^{(j)} \leftarrow s - r_0^{(j)}$
    $c_0^{(j)} \leftarrow com_0\left(r_0^{(j)}, t_0^{(j)}, e_0^{(j)}\right)$
    $c_1^{(j)} \leftarrow com_1\left(r_1^{(j)}, G\left(t_0^{(j)}, r_1^{(j)}\right) + e_0^{(j)}\right)$
    $com^{(j)} := \left(c_0^{(j)}, c_1^{(j)}\right)$
  end for
  $\sigma_0 \leftarrow \mathcal{H}\left(com^{(1)}||com^{(2)}||\ldots||com^{(r)}\right)$
  $ch_1 \leftarrow H_1(D, \sigma_0)$
  Parse ch_1 as $ch_1 = \left(\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(r)}\right)$, $\alpha^{(j)} \in \mathbb{F}_q$
  for $j \in \{1, \ldots, r\}$ **do**
    $t_1^{(j)} \leftarrow \alpha^{(j)} r_0^{(j)} - t_0^{(j)}, \quad e_1^{(j)} \leftarrow \alpha^{(j)} F\left(r_0^{(j)}\right) - e_0^{(j)}$
    $resp_1^{(j)} := \left(t_1^{(j)}, e_1^{(j)}\right)$
  end for
  $\sigma_1 \leftarrow \left(resp_1^{(1)}||resp_1^{(2)}||\ldots||resp_1^{(r)}\right)$
  $ch_2 \leftarrow H_2(D, \sigma_0, ch_1, \sigma_1)$
  Parse ch_2 as $ch_2 = \left(b^{(1)}, b^{(2)}, \ldots, b^{(r)}\right)$, $b^{(j)} \in \{0,1\}$
  for $j \in \{1, \ldots, r\}$ **do**
    $resp_2^{(j)} \leftarrow r_{b^{(j)}}^{(j)}$
  end for
  $\sigma_2 \leftarrow \left(resp_2^{(1)}||resp_2^{(2)}||\ldots||resp_2^{(r)}||c_{1-b^{(1)}}^{(1)}||c_{1-b^{(2)}}^{(2)}||\ldots||c_{1-b^{(r)}}^{(r)}\right)$
  return $\sigma = (R, \sigma_0, \sigma_1, \sigma_2)$
end procedure

REMARK

- The truth value of $\sigma'_0 == \sigma_0$: means **valid verification**,
- The false value of $\sigma'_0 == \sigma_0$: means **invalid verification**.

Algorithm 12 MQDSS-q-n Signature Verification

procedure VF(pk,σ,M)
  $F \leftarrow XOF_F\left(S_F\right)$
  $D \leftarrow \mathcal{H}\left(pk||R||M\right)$
  $ch_1 \leftarrow H_1\left(D, \sigma_0\right)$
  Parse ch_1 as $ch_1 = \left(\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(r)}\right)$, $\alpha^{(j)} \in \mathbb{F}_q$
  $ch_2 \leftarrow H_2\left(D, \sigma_0, ch_1, \sigma_1\right)$
  Parse ch_2 as $ch_2 = \left(b^{(1)}, b^{(2)}, \ldots, b^{(r)}\right)$, $b^{(j)} \in \{0,1\}$
  Parse σ_1 as $\sigma_1 = \left(resp_1^{(1)}||resp_1^{(2)}||\ldots||resp_1^{(r)}\right)$
  Parse σ_2 as $\sigma_2 = \left(resp_2^{(1)}||resp_2^{(2)}||\ldots||resp_2^{(r)}||c^{(1)}_{1-b^{(1)}}||c^{(2)}_{1-b^{(2)}}||\ldots||c^{(r)}_{1-b^{(r)}}\right)$
  for $j \in \{1, \ldots, r\}$ **do**
    Parse $resp_1^{(j)}$ as $resp_1^{(j)} = \left(t_1^{(j)}, e_1^{(j)}\right)$
    if $b^{(j)} == 0$ **then**
      $r_0^{(j)} = resp_2^{(j)}$
      $c_0^{(j)} \leftarrow Com_0\left(r_0^{(j)}, \alpha^{(j)} r_0^{(j)} - t_1^{(j)}, \alpha^{(j)} F\left(r_0^{(j)}\right) - e_1^{(j)}\right)$
    else
      $r_1^{(j)} = resp_2^{(j)}$
      $c_1^{(j)} \leftarrow Com_1\left(r_1^{(j)}, \alpha^{(j)}\left(v - F\left(r_1^{(j)}\right)\right) - G\left(t_1^{(j)}, r_1^{(j)}\right) - e_1^{(j)}\right)$
    end if
    $com^{(j)} := \left(c_0^{(j)}, c_1^{(j)}\right)$
  end for
  $\sigma'_0 \leftarrow \mathcal{H}\left(com^{(1)}||com^{(2)}||\ldots||com^{(r)}\right)$
  return $\sigma'_0 == \sigma_0$
end procedure

6.3 OPTIMIZED PARAMETER SETS

According to the general description of the MQDSS, lengthes of important parameters are the following [5]:

- the public-key $pk \rightarrow k + n\left\lceil \log_2 q \right\rceil$ bits,
- the secret key $sk \rightarrow k$ bits,
- the signature $\sigma \rightarrow 4k + \left(2k + 3n\left\lceil \log_2 q \right\rceil\right) r$ bits,

- the round number $r \rightarrow \left\lceil k/\log_2 \frac{2q}{q+1} \right\rceil$ bits.

The creators of the MQDSS specify two reference parameter sets for this algorithm:

- MQDSS-31-48 $(k = 128, q = 31, n = 48, r = 135)$,
- MQDSS-31-64 $(k = 192, q = 31, n = 64, r = 202)$.

Now, it is possible to calculate lengthes of pk, sk, σ for MQDSS-31-48 and MQDSS-31-64:

	Security Category	k	q	n	r	pk (bytes)	sk (bytes)	σ (bytes)
MQDSS-31-48	$1-2$	128	31	48	135	46	16	16534
MQDSS-31-64	$3-4$	192	31	64	202	64	24	34032

Then, lets give the complexity of the best classical and quantum attacks against MQDSS-31-48 and MQDSS-31-64:

	The Best Classical Attack		The Best Quantum Attack		
	algorithm	field operations	algorithm	gates	depth
MQDSS-31-48	Hybrid F5	2^{159}	Crossbread	2^{99}	2^{90}
MQDSS-31-64	Hybrid F5	2^{205}	Crossbread	2^{130}	2^{120}

Notice that: Security categories of NIST can be seperated into 3 parts:
- security category $1-2$,
- security category $3-4$,
- security category $5-6$.

If we consider optimized parameter sets of the MQDSS creators, their preferences are based on the two things: security level and performance analysis. They need to balance these two factors so as to determine their optimized parameter sets. In this manner, they don't prefer to obtain NIST's 5-6 security level since this level of security requires a signature size which is out of the practical range. Furthermore, to reach 3-4 security level, they changed the number of rounds. In addition to all these, the most essential decision is associated with the selection of the parameter q. They prefer to use $\mathbb{F}_{31}$ $(q = 31)$ instead of using $\mathbb{F}_{16}$ $(q = 16)$ and $\mathbb{F}_{32}$ $(q = 32)$ although the performance characteristics of the parameter sets over all these fields are really similar with each other for the same level of security.

Let's focus on the reasons:

- First of all, "31" is a **Mersenne prime** which is a prime number of the form $2^n - 1$. In cryptography, Mersenne primes have an important role since in the process of finding and checking the primality of large prime numbers, the usage of this kind of primes is pretty efficient and practical.

- Secondly, for the MQDSS, we need to consider implementations of field arithmetics. All computational devices include instructions for the multiplication of natural numbers over the

field $\mathbb{F}_{31}$. On the other hand, they don't contain any instruction set for the multiplication over $\mathbb{F}_{16}$ and $\mathbb{F}_{32}$, so specific representations need to be generated for the fast implementations over these fields.

- Finally, the creators of the MQDSS benefit from **AVX2** instructions in their optimized implementation and they realized that this instruction set is a lot more faster over $\mathbb{F}_{31}$ than $\mathbb{F}_{16}$ and $\mathbb{F}_{32}$.

6.4 DETAILED DESCRIPTION OF THE MQDSS

In the section 6.4, the optimized parameter sets of the MQDSS have been given. According to these parameter sets, the parameter q is equal to 31 ($q = 31$). That's why, in this section, the detailed description of the MQDSS will be given for $q = 31$. In other words, the finite field which will be worked on is $\mathbb{F}_{31}$. [5] In this process, all functions are defined in terms of parameters $k, n \in \mathbb{N}$ such that $64|k$ and $8|n$. Thus, any implementation over $\mathbb{F}_{31}$ for several security categories can be performed by analyzing the following descriptive pseudo-codes which contain the byte-level details.

6.4.1 PRELIMINARY FUNCTIONS [5]

REMARK

- SHAKE256$(seed, 136)$: interface for the digest of SHAKE256 on input $seed$ as standardized in FIPS 202, the SHA-3 standard.
- subarray(a, b, c): $[a[b], \ldots, a[c-1]]$.
- The secret key sk whose length is $\frac{k}{8}$ bytes (k bits) will be expanded into $\frac{3k}{8}$ bytes ($3k$ bits).
- Then, the extended form of the secret key which is "$block$" will be seperated into three parts such that its first $\frac{k}{8}$ bytes create S_{F}, second $\frac{k}{8}$ bytes create S_{s} and the remaining $\frac{k}{8}$ bytes create S_{rte}.

Algorithm 13 Secret Key Expansion

1: **procedure** SECRETKEYEXPANSION(sk)
2: **Input:** sk
3: $block \leftarrow$ SHAKE256$(sk, 136)$
4: $S_{\text{F}} \leftarrow$ subarray$(block, 0, k/8)$
5: $S_{\text{s}} \leftarrow$ subarray$(block, k/8, 2k/8)$
6: $S_{\text{rte}} \leftarrow$ subarray$(block, 2k/8, 3k/8)$
7: **Output:** $S_{\text{F}}, S_{\text{s}}, S_{\text{rte}}$
8: **end procedure**

REMARK

- SHAKE256absorb($seed$): interface for the absorb phase of SHAKE256 for extandable output.
- SHAKE256squeeze($state$): interface for the squeeze phase of SHAKE256 for extandable output.
- len(a): the length of a.
- trunc(a, b): $[a[0], a[1], \ldots, a[b-1]]$.
- $a <> b$: logical non-equal test.
- mask(a, b, c): a function which sets the bits $a[b], \ldots, a[c]$ to 0.
- append(a, b): a function which appends the element b to the end of the array a.
- SHAKE256 functions are used in order to extend any input and return extended output.
- The obtained extended output enters into the function of rejection sampling and thus XOF_F, PRG_s, PRG_{rte} are instantiated.
- trunc($cand, 5$) $<>$ 11111 checks whether the least significant 5 bits of the byte array "$cand$" is equal to 31 or not.
- If trunc($cand, 5$) $<>$ 11111 is true, then the function "append" is applied.
- append($array31$, mask ($cand, 5, 7$)) sets the most significant 3 bits of the "$array31$" to 0 and append these 3 bits to the end of the "$array31$". Thus, for each byte array "$cand$", only the least significant 5 bits are taken then, they are reduced in modulo 31 since $\mathbb{F}_q = \mathbb{F}_{31}$ $(q = 31)$.
- At the end, the function "RejectSample" returns an array whose length is equal to input "len".
- The function "RejectSample" is used in order to obtain the function F from S_F, the secret key s from S_s and the random vectors r_0, t_0 and e_0 from S_{rte} in the following three algorithms $(15, 16, 17)$.

Algorithm 14 Rejection Sampling (Expanding S_{F}, S_{s}, S_{rte})

procedure REJECTSAMPLE($seed, len$)
  Input: $seed, len$
  $array31 \leftarrow [\,]$
  $state \leftarrow$ SHAKE256absorb($seed$)
  while len($array31$) $< len$ **do**
    $block \leftarrow$ SHAKE256squeeze($state$)
    $i \leftarrow 0$
    while $i <$ len($block$) $\wedge$ len($array31$) $< len$ **do**
      $cand \leftarrow block\,[i]$
      if trunc($cand, 5$) $<>$ 11111 **then**
        append ($array31$, mask ($cand, 5, 7$))
      end if
      $i \leftarrow i + 1$
    end while
  end while
  Output: $array31$
end procedure

REMARK

- In the following algorithm "MQsystem", the number 15 is substracted from the all elements of the function F so as to obtain signed integers between -15 and $+15$, inclusive.

Algorithm 15 Expanding S_{F}

procedure MQSYSTEM(S_{F})
  Input: S_{F}
  F $\leftarrow$ RejectSample $\left(S_{\mathrm{F}}, n\left(\frac{n(n+1)}{2} + n\right)\right)$
  for $0 \leq i <$ len (F) **do**
    F $[i] \leftarrow$ F $[i] - 15$
  end for
  Output: F
end procedure

Algorithm 16 Expanding S_{s}

procedure SECRETVECTOR(S_{s})
  Input: S_{s}
  s $\leftarrow$ RejectSample (S_{s}, n)
  Output: s
end procedure

Algorithm 17 Expanding S_{rte}

procedure RTEEXPAND(S_{rte}, D)
  Input: S_{rte}
  $array_{rte} \leftarrow$ RejectSample $(S_{\text{rte}}||D, 3rn)$
  $array_r \leftarrow$ subarray $(array_{\text{rte}}, 0, rn)$
  $array_t \leftarrow$ subarray $(array_{\text{rte}}, rn, 2rn)$
  $array_e \leftarrow$ subarray $(array_{\text{rte}}, 2rn, 3rn)$
  r $\leftarrow [[0|\, i = 1 \ldots n]\, |i = 1 \ldots r]$
  t $\leftarrow [[0|\, i = 1 \ldots n]\, |i = 1 \ldots r]$
  e $\leftarrow [[0|\, i = 1 \ldots n]\, |i = 1 \ldots r]$
  for $0 \leq i < rn; i \leftarrow i + n$ **do**
    r$[i] \leftarrow subarray\,(array_{\text{r}}, i, i + n)$
    t$[i] \leftarrow subarray\,(array_{\text{t}}, i, i + n)$
    e$[i] \leftarrow subarray\,(array_{\text{e}}, i, i + n)$
  end for
  Output: r, t, e
end procedure

REMARK

- The function "EvaluateF" first generates the all quadratic terms of F then computes the polynomials which correspond to these terms.
- The function "EvaluateF" computes the value of the function F for given any input vector u.

Algorithm 18 Evaluating F

procedure EVALUATEF(u, F)
  Input: u, F
  $terms \leftarrow [\,]$
  for $0 \leq i < n$ **do**
    for $0 \leq j < i$ **do**
      append$(terms, \text{u}\,[i] \cdot \text{u}\,[j] \mod 31)$
    end for
  end for
  r $\leftarrow [0|j = 0 \ldots n - 1]$
  for $0 \leq i < n; i \leftarrow i + 2$ **do**
    for $0 \leq j < n$ **do**
      r$\,[j] \leftarrow$ r$\,[j] +$ u$\,[i] \cdot$ F$\,[i \cdot n + 2 \cdot j] \mod 31$
      r$\,[j] \leftarrow$ r$\,[j] +$ u$\,[i + 1] \cdot$ F$\,[i \cdot n + 2 \cdot j + 1] \mod 31$
    end for
  end for
  for $0 \leq i < \frac{n \cdot (n+1)}{2}; i \leftarrow i + 2$ **do**
    for $0 \leq j < n$ **do**
      r$\,[j] \leftarrow$ r$\,[j] +$ terms$\,[i] \cdot$ F$\,[n \cdot m + i \cdot m + 2 \cdot j] \mod 31$
      r$\,[j] \leftarrow$ r$\,[j] +$ terms$\,[i + 1] \cdot$ F$\,[n \cdot m + i \cdot m + 2 \cdot j + 1] \mod 31$
    end for
  end for
  Output: r $=$ F (u)
end procedure

REMARK

- The function G is the polar form of the function F (bilinear counterpart of F).

Algorithm 19 Evaluating G

procedure EVALUATEG(u, v, F)
  Input: u, v, F
  $terms \leftarrow [\,]$
  for $0 \leq i < n$ **do**
    for $0 \leq j < i$ **do**
      append$(terms, \mathrm{u}[i] \cdot \mathrm{v}[j] + \mathrm{u}[j] \cdot \mathrm{v}[i] \mod 31)$
    end for
  end for
  $\mathrm{r} \leftarrow [0|j = 1 \ldots n]$
  for $0 \leq i < \frac{n \cdot (n+1)}{2}$; $i \leftarrow i + 2$ **do**
    for $0 \leq j < n$ **do**
      $\mathrm{r}[j] \leftarrow \mathrm{r}[j] + \mathrm{terms}[i] \cdot \mathrm{F}[n \cdot m + i \cdot m + 2 \cdot j] \mod 31$
      $\mathrm{r}[j] \leftarrow \mathrm{r}[j] + \mathrm{terms}[i+1] \cdot \mathrm{F}[n \cdot m + i \cdot m + 2 \cdot j + 1] \mod 31$
    end for
  end for
  Output: $\mathrm{r} = \mathrm{G}(\mathrm{u}, \mathrm{v})$
end procedure

REMARK

- At the beginning, there was an assumption which says $8|n$. The reason for this is not to need any padding for the byte array representations of input and output arrays.
- The function "PackArray31" gets a vector whose elements are expressed with only 5 bits as an input and return its byte array representation.

Algorithm 20 Packing $\mathbb{F}_{31}$ Elements

procedure PACKARRAY31(u)
  Input: u
  $bitstring \leftarrow [\,]$
  for $0 \leq i < \mathrm{len}(\mathrm{u})$ **do**
    $bitstring \leftarrow bitstring || \mathrm{trunc}(\mathrm{u}[i], 5)$
  end for
  $bytearray \leftarrow [\,]$
  for $0 \leq i < \mathrm{len}(bitstring)$; $i \leftarrow i + 8$ **do**
    append$(bytearray, \mathrm{subarray}(bitstring, i, i+8))$
  end for
  Output: $bytearray$
end procedure

REMARK

- The function "UnpackArray31" is the exact inverse of the function "PackArray31".

Algorithm 21 Unpacking $\mathbb{F}_{31}$ Elements

procedure UNPACKARRAY31($bytearray$)
  Input: $bytearray$
  $bitstring \leftarrow [\,]$
  for $0 \leq i <$ len $(bytearray)$ **do**
    $bitstring \leftarrow bitstring || bytearray\,[i]$
  end for
  u $\leftarrow [\,]$
  for $0 \leq i <$ len $(bitstring)$; $i \leftarrow i + 5$ **do**
    append (u, subarray $(bitstring, i, i + 5)$ $||000)$
  end for
  Output: u
end procedure

REMARK

- In the process of instantiation for the commitments and hash functions, SHAKE256 is used.
- Commitments need byte arrays instead of arrays which consist of $\mathbb{F}_{31}$ elements, as input.
- Inputs of the function "Com0" $\Longrightarrow$ 3 packed byte arrays: PackArray31(r), PackArray31(t), PackArray31(e).
- Inputs of the function "Com1" $\Longrightarrow$ 2 packed byte arrays: PackArray31(r), PackArray31(e).

Algorithm 22 Commitment Function Com_0

procedure COM0(r, t, e)
  Input: r, t, e
  $c_0 \leftarrow [\,]$
  $seed \leftarrow$ (PackArray31 (r) ||PackArray31 (t) ||PackArray31 (e))
  $state \leftarrow$ SHAKE256absorb($seed$)
  $block \leftarrow$ SHAKE256squeeze($state$)
  $c_0 \leftarrow$ subarray($block, 0, k/8$)
  Output: c_0
end procedure

Algorithm 23 Commitment Function Com_1

procedure COM1(r, e)
  Input: r, e
  $c_1 \leftarrow [\,]$
  $seed \leftarrow$ (PackArray31 (r) ||PackArray31 (e))
  $state \leftarrow$ SHAKE256absorb$(seed)$
  $block \leftarrow$ SHAKE256squeeze$(state)$
  $c_1 \leftarrow$ subarray$(block, 0, k/8)$
  Output: c_1
end procedure

Algorithm 24 Hash Function $\mathcal{H}$

procedure HASH$(bytearray)$
  Input: $bytearray$
  $state \leftarrow$ SHAKE256absorb$(bytearray)$
  $block \leftarrow$ SHAKE256squeeze$(state)$
  $digest \leftarrow$ subarray$(block, 0, k/8)$
  Output: $digest$
end procedure

REMARK

- The challenge "ch_1" is obtained by applying "RejectSample" algorithm to the "$seed$" for the length r.
- The function SHAKE256squeeze is used iteratively in order to attain an output of desirable length.

Algorithm 25 Hash Function H_1

procedure HASH1(D, σ_0)
  Input: D, σ_0
  $seed \leftarrow D||\sigma_0$
  $\text{ch}_1 \leftarrow$ RejectSample $[seed, r]$
  Output: ch_1
end procedure

REMARK

- "ch_2" which is a binary challenge is acquired with the method of enumarating the bits of arrays $block\,[0]$, $block\,[1]$, ... whose lengthes are 1-byte.

Algorithm 26 Hash Function H_2

procedure HASH2($D, \sigma_0, \text{ch}_1, \sigma_1$)
  Input: $D, \sigma_0, \text{ch}_1, \sigma_1$
  $seed \leftarrow D||\sigma_0\text{PackArray31}\,(\text{ch}_1)\,||\sigma_1$
  $state \leftarrow \text{SHAKE256absorb}(seed)$
  $block \leftarrow \text{SHAKE256squeeze}(state)$
  $\text{ch}_2 \leftarrow [\,]$
  for $0 \leq i < r$ **do**
    $temp = block\,[\text{floor}\,(i/8)]$
    $\text{append}\,(\text{ch}_2, temp\,[i \mod 8])$
  end for
  Output: ch_2
end procedure

6.4.2 DESCRIPTION OF THE ALGORITHMS: KGen, Sign, Vf

REMARK

- Since all necessary parameters and preliminary functions have been given for the detailed description of the MQDSS over $\mathbb{F}_{31}$, now it is possible to complete this description which benefits from all these auxiliary pseudo codes by giving the following triplet of algorithms [5]: KGen, Sign, Vf.

- The function rand(k) where k is the security parameter of the MQDSS, is used in order to produce a pseudo random secret key sk whose length is k bits.

Algorithm 27 MQDSS Key Generation

procedure KGEN(k)
  Input: k
  $\text{sk} \leftarrow \text{rand}\,(k)$
  $S_{\text{F}}, S_{\text{s}}, S_{\text{rte}} \leftarrow \text{SecretKeyExpansion}\,(\text{sk})$
  $\text{F} \leftarrow \text{MQsystem}\,(S_{\text{F}})$
  $\text{s} \leftarrow \text{SecretVector}\,(S_{\text{s}})$
  $\text{v} \leftarrow \text{EvaluateF}\,(\text{s}, \text{F})$
  $\text{pk} \leftarrow S_{\text{F}}||\text{PackArray31}\,(\text{v})$
  Output: (pk, sk)
end procedure

Algorithm 28 MQDSS Signature Generation

procedure SIGN(sk,M)
  Input: sk, M
  $S_{\mathrm{F}}, S_{\mathrm{s}}, S_{\mathrm{rte}} \leftarrow$ SecretKeyExpansion (sk)
  $\mathrm{F} \leftarrow$ MQsystem (S_{F})
  $\mathrm{s} \leftarrow$ SecretVector (S_{s})
  $\mathrm{v} \leftarrow$ EvaluateF (s, F)
  $\mathrm{pk} \leftarrow S_{\mathrm{F}}||$PackArray31 (v)
  $R \leftarrow$ Hash $(\mathrm{sk}||M)$
  $D \leftarrow$ Hash $(\mathrm{pk}||R||M)$
  $\mathrm{r}_0, \mathrm{t}_0, \mathrm{e}_0 \leftarrow$ RTEexpand (S_{rte}, D)
  $\mathrm{r}_1 \leftarrow [[0|\, i = 1 \ldots n] \,|i = 1 \ldots r]$
  $\mathrm{t}_1 \leftarrow [[0|\, i = 1 \ldots n] \,|i = 1 \ldots r]$
  $\mathrm{e}_1 \leftarrow [[0|\, i = 1 \ldots n] \,|i = 1 \ldots r]$
  $c_0 \leftarrow [[0|\, i = 1 \ldots 2k/8] \,|i = 1 \ldots r]$
  $c_1 \leftarrow [[0|\, i = 1 \ldots 2k/8] \,|i = 1 \ldots r]$
  $\mathrm{com} \leftarrow [\,]$
  for $0 \leq i < r$ **do**
    $\mathrm{r}_1[i] \leftarrow \mathrm{s} - \mathrm{r}_0[i]$
    $c_0[i] \leftarrow$ Com0 $(\mathrm{r}_0[i], \mathrm{t}_0[i], \mathrm{e}_0[i])$
    $c_1[i] \leftarrow$ Com1 $(\mathrm{r}_1[i],$ EvaluateG $(\mathrm{t}_0[i], \mathrm{r}_1[i], \mathrm{F}) + \mathrm{e}_0[i])$
    $\mathrm{com} \leftarrow \mathrm{com}||$PackArray31 $(c_0[i])\,||$PackArray31 $(c_1[i])$
  end for
  $\sigma_0 \leftarrow$ Hash (com)
  $\mathrm{ch}_1 \leftarrow$ Hash1 (D, σ_0)
  $\sigma_1 \leftarrow [\,]$
  for $0 \leq i < r$ **do**
    $\mathrm{t}_1[i] \leftarrow \mathrm{ch}_1[i] \cdot \mathrm{r}_0[i] - \mathrm{t}_0[i]$
    $e_1[i] \leftarrow \mathrm{ch}_1[i] \cdot$ EvaluateF $(\mathrm{r}_0[i], \mathrm{F}) - \mathrm{e}_0[i]$
    $\sigma_1 \leftarrow \sigma_1||$PackArray31 $(\mathrm{t}_1[i])\,||$PackArray31 $(\mathrm{e}_1[i])$
  end for
  $\mathrm{ch}_2 \leftarrow$ Hash2 $(D, \sigma_0, \mathrm{ch}_1, \sigma_1)$
  $\sigma_2 \leftarrow [\,]$
  for $0 \leq i < r$ **do**
    if $\mathrm{ch}_2[i] == 0$ **then**
      $\sigma_2 \leftarrow \sigma_2||$PackArray31 $(\mathrm{r}_0[i])$
    else
      $\sigma_2 \leftarrow \sigma_2||$PackArray31 $(\mathrm{r}_1[i])$
    end if
  end for
  for $0 \leq i < r$ **do**
    if $\mathrm{ch}_2[i] == 0$ **then**
      $\sigma_2 \leftarrow \sigma_2||c_1[i]$
    else
      $\sigma_2 \leftarrow \sigma_2||c_0[i]$
    end if
  end for
  Output: $\sigma = R||\sigma_0||\sigma_1||\sigma_2$
end procedure

Algorithm 29 MQDSS Signature Verification

procedure VF(pk, σ, M)
  Input: pk, σ, M
  $R \leftarrow$ subarray $(\sigma, 0, 2 \cdot k/8)$
  $\sigma_0 \leftarrow$ subarray $(\sigma, 2 \cdot k/8, 4 \cdot k/8)$
  $\sigma_1 \leftarrow$ subarray $(\sigma, 4 \cdot k/8, (4 \cdot k + 10 \cdot n \cdot r)\ /8)$
  $\sigma_2 \leftarrow$ subarray $(\sigma, (4 \cdot k + 10 \cdot n \cdot r)\ /8, \text{len}\,(\sigma))$
  $S_{\text{F}} \leftarrow$ subarray $(\text{pk}, 0, k/8)$
  F $\leftarrow$ MQsystem (S_{F})
  $D \leftarrow$ Hash $(\text{pk}||R||M)$
  $\text{ch}_1 \leftarrow$ Hash1 (D, σ_0)
  $\text{ch}_2 \leftarrow$ Hash2 $(D, \sigma_0, \text{ch}_1, \sigma_1)$
  $\text{resp}_1 \leftarrow$ UnpackArray31 (σ_1)
  $\text{resp}_2 \leftarrow$ UnpackArray31 (subarray $(\sigma_2, 0, 5 \cdot n \cdot r/8)$)
  $c \leftarrow$ subarray $(\sigma_2, 5 \cdot n \cdot r/8, \text{len}\,(\sigma_2))$
  com $\leftarrow [\,]$
  for $0 \leq i < r$ **do**
    $\text{t}_1 \leftarrow \text{resp}_1\,[2i]$
    $\text{e}_1 \leftarrow \text{resp}_1\,[2i+1]$
    if $\text{ch}_2\,[i] == 0$ **then**
      $\text{r}_0 \leftarrow \text{resp}_2\,[i]$
      $c_0 \leftarrow$ Com0 $(\text{r}_0, \text{ch}_1\,[i] \cdot \text{r}_0 - \text{t}_1, \text{ch}_1\,[i] \cdot$ EvaluateF $(\text{r}_0, \text{F}) - \text{e}_1)$
      $c_1 \leftarrow$ subarray $(c, i \cdot k/8, (i+1) \cdot k/8)$
    else
      $\text{r}_1 \leftarrow \text{resp}_2\,[i]$
      $c_1 \leftarrow$ Com1 $(\text{r}_1, \text{ch}_1\,[i] \cdot (\text{v} -$ EvaluateF $(\text{r}_1, \text{F})) -$ EvaluateG $(\text{t}_1, \text{r}_1, \text{F}) - \text{e}_1)$
      $c_0 \leftarrow$ subarray $(c, i \cdot k/8, (i+1) \cdot k/8)$
      com $\leftarrow \text{com}||c_0||c_1$
    end if
  end for
  $\sigma_0' \leftarrow$ Hash (com)
  Output: $\sigma_0' == \sigma_0$
end procedure

6.5 CLASSICAL AND QUANTUM ALGORITHMS FOR SOLVING THE MQ PROBLEM

Before the security analysis of the MQDSS, let's give a brief information about the classical and quantum algorithms which are preferred in order to solve the MQ problem.

6.5.1 CLASSICAL ALGORITHMS FOR SOLVING THE MQ PROBLEM

Let $F = (f_1, \ldots, f_m), \quad f_i \in \mathbf{F}_q[x_1, \ldots, x_n]$. The problem which we want to solve, is finding x such that $F(x) = 0$. For solving this problem, it can be benefited from the following classical algorithms [5]: the Exhaustive Search, the HybridF5 algorithm, the BoolenSolve algorithm and the Crossbread algorithm.

6.5.1.1 Exhaustive Search

As is known to all, it is possible to acknowledge the exhaustive search as a brute force attack. Since the aim is solving the problem of $F(x) = 0$ and $x \in \mathbf{F}_{q^n}$, the thing which is needed to do is trying all possible values $x \in \mathbf{F}_{q^n}$ until obtaining $F(x) = 0$.

6.5.1.2 The HybridF5 Algorithm

In the process of solving the problem of $F(x) = 0$, the F5 algorithm obtains the ideal which is produced by the polynomials $f_1, \ldots, f_m$. Then, it computes a Gröbner basis of this ideal. According to Lazard, there is a connection between the Gröbner basis computation and the Gaussian elimination on the Macaulay matrices. That is, Lazard found that by applying row reduction method to a Macaulay matrix, Gröbner basis can be obtained [23]. Macaulay matrices of degree D denoted by $Mac_D(F)$ are generated by setting the coefficients of monomials uf_i of maximal degree D to the rows of it. $Mac_D(F)$ needs to be computed for $D \in 2, \ldots, D_{reg}$ where D_{reg} is the degree of regularity.

Let's move on the hybrid approach of the F5 algorithm. The aim of this approach is reducing the complexity of the F5 algorithm. With this aim, Bettale introduced a combination between the exhaustive search and the Gröbner basis computation. The technique for doing this, is fixing $n - k$ variables of the function F, then applying the reduction operations to the matrix $Mac_{D_{reg}}(\tilde{F})$, where $\tilde{F} = (\tilde{f}_1, \ldots, \tilde{f}_m)$ and $\tilde{f}_i(x_1, \ldots, x_k) = f_1(x_1, \ldots, x_k, a_{k+1}, \ldots, a_n) \quad \forall (a_{k+1}, \ldots, a_n) \in \mathbb{F}_{q^{n-k}}$. Finally, q^{n-k} times Gröbner basis computations of smaller systems are needed as the exhaustive search part [4]. Here, the parameter k is obtained in the result of an optimization process.

6.5.1.3 The BooleanSolve Algorithm

The BooleanSolve algorithm focuses on solving the problem of $F(x) = 0$ for $q = 2$. This algorithm is similar to the hybrid approach. First of all, as a result of the optimization process, the value of the parameter k is determined. After fixing $n - k$ variables, the problem is transformed into the consistency testing problem of the following linear system:

$$u \cdot Mac_{D_{reg}}(\tilde{F}) = (0, \ldots, 0, 1), \tag{6.1}$$

where $\tilde{F} = (\tilde{f}_1, \ldots, \tilde{f}_m)$ and $\tilde{f}_i(x_1, \ldots, x_k) = f_1(x_1, \ldots, x_k, a_{k+1}, \ldots, a_n)$ $\forall (a_{k+1}, \ldots, a_n) \in \mathbb{F}_{2^{n-k}}$.

Consistency of the equation (6.1) means that the problem of $F(x) = 0$ doesn't have a solution and vice versa [17]. For the remaining part, exhaustive search operations are applied. That is, since $a \in \mathbb{F}_{2^{n-k}}$, 2^{n-k} times consistency testing operation is repeated.

6.5.1.4 The Crossbread Algorithm

The Crossbread algorithm which was generated by Joux and Vitse, focuses on solving the problem of $F(x) = 0$. Unlike the previous ones, this algorithm first applies some algebraic operations on the matrix $Mac_D(F)$ where $D \geq D_{reg}(k, n)$, then it moves on to the fixing process. In the second part, after two new matrices which are denoted by $Mac^{(k)}_{D,d}(F)$ and $M^{(k)}_{D,d}(F)$, were derived from the matrix $Mac_D(F)$, the operation of attaining sufficent number of linearly independent elements which exist in the kernel of the matrix $M^{(k)}_{D,d}(F)$ but not in the kernel of the matrix $Mac^{(k)}_{D,d}(F)$, is performed. Then the multiplication of these linearly independent elements with the matrix $Mac^{(k)}_{D,d}(F)$ gives us the set of polynomials, say P. The rest of it is similar to the previous algorithms. That is, the system of

$$\tilde{P}(x_1, \ldots, x_k) = P(x_1, \ldots, x_k, a_{k+1}, \ldots, a_n) \quad \forall (a_{k+1}, \ldots, a_n) \in \mathbb{F}_q^{n-k} \tag{6.2}$$

is generated. Finally, for solving the system of $\tilde{P}$, the method of linearization operation is peformed [21].

6.5.2 QUANTUM ALGORITHMS FOR SOLVING THE MQ PROBLEM

For solving the problem of $F(x) = 0$, it is not possible to find any dedicated quantum algorithms. Instead of this, Grover's algorithm running on a quantum computer is applied to the classical algorithms. For using this algorithm, there exists a need for the existence of an oracle whose usage is associated with the evaluation of the quadratic equations at a superposition of all possible x values. Thus, the cost of the exhaustive search which is amplified by the Grover's algorithm, is decreased into $\Theta(2^{\frac{n}{2}})$ from $\Theta(2^n)$ in the case of n variables, according to the work of Westerbaan and Schwabe. Now, the aim is applying the Grover's quantum search algorithm to the classical algorithms for solving the MQ problem [36].

The logic behind the Grover's quantum search algorithm which is in a superposition of states, is the process of simultaneous examination among the elements of an unordered list whose size is 2^k. In order to explain a little bit more, assume that there exists a quantum black box function: $f : \{0,1\}^k \to \{0,1\}$ and its unitary circuit: $U_f : |x\rangle y\rangle \to |x\rangle |x \oplus f(y)\rangle$ which is called as an oracle. The result x_0 which is desired to find, gives 1 when it is putting into the function f as an input. The rest of the x values gives 0. Then an operator G is determined as a function of U_f. After that, the optimal number of times, G is is applied on a state which is in

an equal superposition of all possible input values. Finally, the desired result can be obtained really faster than the exhaustive search [20].

Now, let's remember the classical algorithms for solving the MQ problem: the Exhaustive Search, the BooleanSolve, the HybridF5 and the Crossbread algorithms. By using these classical algorithms and the Grover's quantum search algorithm together, the Grover enhanced quantum algorithms whose names are the MQ oracle, the BooleanSolve oracle, the HybridF5 oracle and the Crossbread oracle respectively, can be obtained for solving the problem of $F(x) = 0$. Actually, for the Grover's qauntum search algorithm, the number of solutions of the MQ problem is a significant parameter. Since we know that the MQ problem has exactly one solution, it is possible to assume that the number of all solutions for this problem is equal to 1.

6.5.2.1 The MQ Oracle

Since this method explains the Grover enhanced exhaustive search algorithm for solving the MQ problem, the corresponding oracle will be U_{MQ}. The logic behind the MQ oracle is the following: if $F(a) = 0$, $U_{MQ}(a) = 1$ for $a \in \mathbb{F}_{q^n}$. For solving the binary MQ problem meaning that the MQ problem for $\mathbb{F}_q = \mathbb{F}_2$, Westerbaan and Schwabe prefered to use two oracles for the MQ polynomials which are defined over the field $\mathbb{F}_{2^n}$ [36].

6.5.2.2 The BooleanSolve Oracle

For this case, the oracle will be U_{Bool}. As we remember the classical BoolenSolve algorithm, after the variable fixing process, it focuses on solving the consistency testing problem of the linear system (6.1). If the equation (6.1) is inconsistent and after the inconsistency testing, the remaining part of the classical BooleanSolve algorithm outputs $b \in \mathbb{F}_{q^k}$ so that $\tilde{F}(b) = 0$, then $U_{Bool}(a) = 1$ for $a \in \mathbb{F}_{2^{n-k}}$.

6.5.2.3 The HybridF5 Oracle

For this case, the oracle will be U_{HybF5}. The logic behind the HybridF5 oracle is the following: if the classical F5 algorithm outputs $b \in \mathbb{F}_{q^k}$ so that $\tilde{F}(b) = 0$, then $U_{HybF5}(a) = 1$ for $a \in \mathbb{F}_{q^{n-k}}$.

6.5.2.4 The Crossbread Oracle

For this case, the oracle will be U_{Cross}. The logic behind the Crossbread oracle is the following: if the classical Crossbread algorithm outputs $b \in \mathbb{F}_{q^k}$ so that $\tilde{P}(b) = 0$, then $U_{Cross}(a) = 1$ for $a \in \mathbb{F}_{q^{n-k}}$.

6.6 SECURITY ANALYSIS

As we explained before, the security level of the MQDSS depends on:

- the MQ problem,
- the computationally hiding and binding properties of the commitments,
- the hash functions,
- the pseudo-random generators.

For the commitment functions, pseudo-random generators and extendable output functions, they benefit from **SHAKE-**256 (as standardized in FIPS 202, the SHA-3 standard) whose security properties of the preimage and the second preimage resistance enable to a pretty large of security level.

In the process of determining parameters of the MQDSS, the creators of this algorithm pay attention to the following points for providing high-level of security and performance:

- choosing equal number of variables and equations in F so as to obtain the hardest MQ problem,
- determining the lower bound n' for the number of variables in F within a specific security level against classical field operations of the best classical attacks,
- determining the lower bound n'' for the number of variables in F within a specific security level against quantum circuit size and depth of the best quantum attacks,
- determining the number of variables n in F such that $n = \max\{n', n''\}$,
- choosing the security parameter k such that the auxiliary hash functions $\mathcal{H}$, H_1 and H_2 have the security property of collision resistance,
- choosing the number of rounds r such that the Fiat-Shamir transform of the SSH-5-pass IDS according to r satisfy the security property of the soundness with soundness error $< \frac{1}{2^k}$.

The MQDSS which is a kind of $q2$-signature scheme, is **EU-CMA** secure in the **ROM** according to the following theorem:

Theorem 8. *The MQDSS is EU-CMA secure in the ROM if the following conditions are satisfied:*

- *the MQ search problem is intractable in the average case,*
- *the hash functions $\mathcal{H}$, H_1, H_2 and the function XOF_F are modeled as random oracles,*

- *the commitment functions Com_0 and Com_1 have the properties of computationally binding and hiding also have $O(k)$ bits of output entropy which means the number of states where a system can be found,*
- *the pseudo-random generators PRG_{sk}, PRG_s and PRG_{rte} have outputs which are computationally indistinguishable from random for any polynomial time adversary.*

That's why, the creators of the MQDSS have a security proof in the **ROM** but not in the **QROM** in which the security proof of the Fiat-Shamir transform for the MQDSS is based on the following techniques: the rewinding technique of the adversary and the adaptively programming of the random oracle in the QROM. However, on 20 February 2019, Jelle Don, Serge Fehr, Christian Majenz and Christian Schaffner submitted a paper under the title of "Security of the Fiat-Shamir Transformation in the QROM". Since MQDSS is the Fiat-Shamir transform of a kind of SSH 5-pass IDS, it is natural to generalize their results also for the MQDSS. Therefore, it is possible to acknowledge the MQDSS as secure both in the ROM and in the QROM [5, 15].

CHAPTER 7

CONCLUSION

In this thesis, one specific solution of the security problem which arises from quantum computers has been analyzed. Since quantum computers will be generated and substitute with classical ones in the near future, then all symmetric and asymmetric (public-key) cryptosystems will be invalid. Thus, the need for the quantum-resistant algorithms occur. Associated with this need, people started to work on post-quantum algorithms all around the world. In this process, NIST has a crucial role. By organizing a competition among the quantum-resistant algorithms, NIST has made easier to follow the improvements in this area and taken them a step forward [25]. The submissions for the NIST competition are composed of different kind of post-quantum algorithms. One kind of them is the multivariate public-key cryptosystems [5, 25]. In order to explain the root idea behind this kind of cryptosystems, as a starting and descriptive example, the Matsumoto-Imai cryptosystem has been studied together with its linearization equations attack. Then, we have constructed our own specific toy example for illustrating the construction of both the single-branch MI cryptosystem and its linearization equations attack. As well as these, Matsumoto-Imai variants which were developed with the aim of increasing the security of the original MI cryptosystem have been examined. In this thesis, the name of the specific solution which was scrutinised is the MQDSS that belongs to the algorithm family of the multivariate public-key cryptosystems. Since MQDSS is a digital signature scheme whose tools are identification schemes, we analyzed identification schemes, signature schemes and their security properties step by step. In this process, we realized that by using the Fiat-Shamir transform, a secure identification scheme can be transformed into a secure digital signature scheme.

For the security proof of the cryptographic protocols, the means which we need are ROM and QROM. That's why, at the beginning, we have investigated these two mechanisms in detailed. Since our crucial aim is the security analysis of the MQDSS, we have focused on the security sources of this algorithm. Accordingly, the security of the MQDSS is based on the intractability of the MQ problem and the hardness of the commitment schemes which satisfy computationally hiding and binding properties instead of the other problems in multivariate cryptography such as MinRank [7], IP [29], Extended IP [13] and IP with partial knowledge [33]. To construct an intractable MQ problem, MQDSS transforms a specific kind of canonical 5-pass identification scheme whose name is Sakumoto-Shirai-Hiwatari (SSH) 5-pass IDS

into a digital signature scheme. In this transformation, canonical 5-pass IDS is a specific kind of canonical $2n + 1$-pass IDS in which the prover and the verifier seperate their processes into $n + 1$ groups: $\mathcal{P} = (P_0, P_1, \ldots, P_n)$, $\mathcal{V} = (ChS_1, \ldots, ChS_n, Vf)$ and all the work is done iteratively over these groups. In this concept, "$2n + 1$" represents the number of the messages exchanged between the prover and the verifier for the identification process and the most common ones of these scehemes is obtained by putting 1 or 2 instead of n.

If we move on the SSH 5-pass IDS, in this algorithm Sakumoto, Shirai and Hiwatari use the idea of splitting the secret key and the output of the function F into 2 parts. After the first splitting process is completed, some parts will further split [35]. For both operations, they benefit from the polar form $G(x, y)$ of $F(x)$ since the polar form is bilinear . By using the multiple splitting process of the secret key and the polar form, parts of the secret key are getting away from each other so as to keep it secure [5].

Finally with the help of all these structural tools, the algorithm MQDSS is constructed. For generating MQDSS, specific commitment functions, pseudo-random generators and parameter sets are used inside of the defining framework of it. Therefore, we have analyzed all preliminary parameters and functions, general description (KGen, Sign, Vf), optimized parameter sets [5] $(k = 128, q = 31, n = 48, r = 135; k = 192, q = 31, n = 64, r = 202)$, detailed description, the classical and the quantum algorithms for solving the MQ problem and the security analysis of the MQDSS.

If we take all these points into consideration, we can say that for the problem of information security which occurs with the expectation of quantum computers, the algorithm MQDSS which belongs to the family of multivariate public-key cryptosystem seems like really secure and efficient in both ROM and QROM. That's why, in order to construct a secure signature scheme belongs to this family, we can benefit from the power of commitment schemes which are one-way functions and the MQ problem. Furthermore, the Fiat-Shamir transform has an essential role for maintaining security in the process of obtaining a signature scheme from an identification scheme. For these reasons, by using the tools of commitment schemes, MQ problem, canonical $2n + 1$-pass IDS, splitting idea [35] which comes from the SSH 5-pass IDS and the Fiat-Shamir transform and selecting optimized parameter sets, it is possible to improve more secure and efficient cryptographic schemes against both classical and quantum attacks.

REFERENCES

[1] M.-L. Akkar, N. T. Courtois, R. Duteuil, and L. Goubin, A fast and secure implementation of sflash, in Y. G. Desmedt, editor, *Public Key Cryptography — PKC 2003*, pp. 267–278, Springer Berlin Heidelberg, Berlin, Heidelberg, 2002, ISBN 978-3-540-36288-3.

[2] S. AKLEYLEK and M. Soysaldi, A novel 3-pass identification scheme and signature scheme based on multivariate quadratic polynomials, TURKISH JOURNAL OF MATHEMATICS, 43, pp. 241–257, 01 2019.

[3] G. Alagic, J. M. Alperin-Sheriff, D. C. Apon, D. A. Cooper, Q. H. Dang, C. A. Miller, D. Moody, R. C. Peralta, R. A. Perlner, A. Y. Robinson, D. C. Smith-Tone, and Y.-K. Liu, Status report on the first round of the nist post-quantum cryptography standardization process.

[4] L. Bettale, J.-C. Faugère, and L. Perret, Solving polynomial systems over finite fields: Improved analysis of the hybrid approach, pp. 67–74, 07 2012.

[5] M. Chen, A. Hülsing, J. Rijneveld, S. Samardjiska, and P. Schwabe, Mqdss specifications version 1.1., In: NIST's First PQC Standardization Conference, 2018.

[6] M.-S. Chen, A. Hülsing, J. Rijneveld, S. Samardjiska, and P. Schwabe, From 5-pass mq-based identification to mq-based signatures, Cryptology ePrint Archive, Report 2016/708, 2016, `https://eprint.iacr.org/2016/708`.

[7] N. T. Courtois, Efficient zero-knowledge authentication based on a linear algebra problem minrank, in C. Boyd, editor, *Advances in Cryptology — ASIACRYPT 2001*, pp. 402–421, Springer Berlin Heidelberg, Berlin, Heidelberg, 2001, ISBN 978-3-540-45682-7.

[8] N. T. Courtois, Algebraic attacks over gf(2k), application to hfe challenge 2 and sflash-v2, in F. Bao, R. Deng, and J. Zhou, editors, *Public Key Cryptography – PKC 2004*, pp. 201–217, Springer Berlin Heidelberg, Berlin, Heidelberg, 2004, ISBN 978-3-540-24632-9.

[9] Ö. Dagdelen, M. Fischlin, and T. Gagliardoni, The fiat–shamir transformation in a quantum world, in K. Sako and P. Sarkar, editors, *Advances in Cryptology - ASIACRYPT 2013*, pp. 62–81, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, ISBN 978-3-642-42045-0.

[10] I. Damgård, *Commitment Schemes and Zero-Knowledge Protocols*, pp. 63–86, Springer Berlin Heidelberg, Berlin, Heidelberg, 1999, ISBN 978-3-540-48969-6.

[11] A. Diene, J. Ding, J. E. Gower, T. J. Hodges, and Z. Yin, Dimension of the linearization equations of the matsumoto-imai cryptosystems, in Ø. Ytrehus, editor, *Coding and Cryptography*, pp. 242–251, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006, ISBN 978-3-540-35482-6.

[12] J. Ding, J. Gower, and D. Schmidt, *Multivariate Public Key Cryptosystems*, Advances in Information Security, Springer US, 2006, ISBN 9780387369464.

[13] J. Ding, L. Hu, B.-Y. Yang, and J.-M. Chen, Note on design criteria for rainbow-type multivariates, IACR Cryptology ePrint Archive, 2006, p. 307, 2006.

[14] J. Ding and B.-Y. Yang, *Multivariate Public Key Cryptography*, pp. 193–241, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, ISBN 978-3-540-88702-7.

[15] J. Don, S. Fehr, C. Majenz, and C. Schaffner, Security of the fiat-shamir transformation in the quantum random-oracle model, Cryptology ePrint Archive, Report 2019/190, 2019, `https://eprint.iacr.org/2019/190`.

[16] E. Eaton, Signature schemes in the quantum random-oracle model, pp. 1–14, UWSpace, 2017.

[17] M. Giesbrecht, A. Lobo, and D. Saunders, Certifying inconsistency of sparse linear systems, Proceedings of the International Symposium on Symbolic and Algebraic Computation, ISSAC, 08 1999.

[18] H. Gilbert and M. Minier, Cryptanalysis of sflash, in L. R. Knudsen, editor, *Advances in Cryptology — EUROCRYPT 2002*, pp. 288–298, Springer Berlin Heidelberg, Berlin, Heidelberg, 2002, ISBN 978-3-540-46035-0.

[19] S. Goldwasser, S. Micali, and R. L. Rivest, A digital signature scheme secure against adaptive chosen-message attacks, SIAM J. Comput., 17, pp. 281–308, 1988.

[20] L. K. Grover, A fast quantum mechanical algorithm for database search, in *ANNUAL ACM SYMPOSIUM ON THEORY OF COMPUTING*, pp. 212–219, ACM, 1996.

[21] A. Joux and V. Vitse, A crossbred algorithm for solving boolean polynomial systems, in J. Kaczorowski, J. Pieprzyk, and J. Pomykała, editors, *Number-Theoretic Methods in Cryptology*, pp. 3–21, Springer International Publishing, Cham, 2018.

[22] S. Katsumata, S. Yamada, and T. Yamakawa, Tighter security proofs for gpv-ibe in the quantum random oracle model, in T. Peyrin and S. Galbraith, editors, *Advances in Cryptology – ASIACRYPT 2018*, pp. 253–282, Springer International Publishing, Cham, 2018, ISBN 978-3-030-03329-3.

[23] D. Lazard, Gröbner bases, gaussian elimination and resolution of systems of algebraic equations, in J. A. van Hulzen, editor, *Computer Algebra*, pp. 146–156, Springer Berlin Heidelberg, Berlin, Heidelberg, 1983, ISBN 978-3-540-38756-5.

[24] T. Matsumoto and H. Imai, Public quadratic polynomial-tuples for efficient signature-verification and message-encryption, in D. Barstow, W. Brauer, P. Brinch Hansen, D. Gries, D. Luckham, C. Moler, A. Pnueli, G. Seegmüller, J. Stoer, N. Wirth, and C. G. Günther, editors, *Advances in Cryptology — EUROCRYPT '88*, pp. 419–453, Springer Berlin Heidelberg, Berlin, Heidelberg, 1988, ISBN 978-3-540-45961-3.

[25] D. Moody, L. Chen, S. Jordan, Y.-K. Liu, D. Smith, R. Perlner, and R. Peralta, Nist report on post-quantum cryptography, 04 2016.

[26] L. G. Nicolas T. Courtois and J. Patarin, Sflashv3, a fast asymmetric signature scheme, Cryptology ePrint Archive, Report 2003/211, 2003, `https://eprint.iacr.org/2003/211`.

[27] K. Okeya, T. Takagi, and C. Vuillaume, On the importance of protecting in sflash against side channel attacks., IEICE Transactions, 88-A, pp. 123–131, 01 2005.

[28] J. Patarin, Cryptanalysis of the matsumoto and imai public key scheme of eurocrypt'88, in D. Coppersmith, editor, *Advances in Cryptology — CRYPT0' 95*, pp. 248–261, Springer Berlin Heidelberg, Berlin, Heidelberg, 1995, ISBN 978-3-540-44750-4.

[29] J. Patarin, Hidden fields equations (hfe) and isomorphisms of polynomials (ip): Two new families of asymmetric algorithms, in U. Maurer, editor, *Advances in Cryptology — EUROCRYPT '96*, pp. 33–48, Springer Berlin Heidelberg, Berlin, Heidelberg, 1996, ISBN 978-3-540-68339-1.

[30] J. Patarin, Cryptanalysis of the matsumoto and imai public key scheme of eurocrypt'98, Designs, Codes and Cryptography, 20(2), pp. 175–209, Jun 2000, ISSN 1573-7586.

[31] J. Patarin, N. Courtois, and L. Goubin, Flash, a fast multivariate signature algorithm, in D. Naccache, editor, *Topics in Cryptology — CT-RSA 2001*, pp. 298–307, Springer Berlin Heidelberg, Berlin, Heidelberg, 2001, ISBN 978-3-540-45353-6.

[32] J. Patarin, L. Goubin, and N. Courtois, C -+* and hm: Variations around two schemes of t. matsumoto and h. imai, in K. Ohta and D. Pei, editors, *Advances in Cryptology — ASIACRYPT'98*, pp. 35–50, Springer Berlin Heidelberg, Berlin, Heidelberg, 1998, ISBN 978-3-540-49649-6.

[33] L. Perret and A. Bayad, A differential approach to a polynomial equivalence problem, in *International Symposium onInformation Theory, 2004. ISIT 2004. Proceedings.*, pp. 142–, June 2004.

[34] P. Rogaway and T. Shrimpton, Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance, 2004.

[35] 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, pp. 706–723, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, ISBN 978-3-642-22792-9.

[36] P. Schwabe and B. Westerbaan, Solving binary mq with grover's algorithm, in C. Carlet, M. A. Hasan, and V. Saraswat, editors, *Security, Privacy, and Applied Cryptography Engineering*, pp. 303–322, Springer International Publishing, Cham, 2016.

[37] A. Shamir, Efficient signature schemes based on birational permutations, in *CRYPTO*, 1993.

[38] P. W. Shor, Polynomial time algorithms for discrete logarithms and factoring on a quantum computer, in L. M. Adleman and M.-D. Huang, editors, *Algorithmic Number Theory*, pp. 289–289, Springer Berlin Heidelberg, Berlin, Heidelberg, 1994, ISBN 978-3-540-49044-9.

[39] P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM Review, 41, 05 1997.