CAPITAL UNIVERSITY OF SCIENCE AND TECHNOLOGY, ISLAMABAD

Digital Signature Based on Matrix Power Function

by

Sundus Iqbal

A thesis submitted in partial fulfillment for the degree of Master of Philosophy

in the

Faculty of Computing
Department of Mathematics

2019

To my mother, sister and brother for their support and love and my late father.

CERTIFICATE OF APPROVAL

Digital Signature Based on Matrix Power Function

by

Sundus Iqbal

(MMT-171004)

THESIS EXAMINING COMMITTEE

S. No.	Examiner	Name	Organization
(a)	External Examiner	Dr. Umar Hayat	QAU, Islamabad
(b)	Internal Examiner	Dr. Muhammad Azhar Iqbal	CUST, Islamabad
(c)	Thesis Supervisor	Dr. Rashid Ali	CUST, Islamabad

Dr. Rashid Ali
Thesis Supervisor
April, 2019

Dr. Muhammad Sagheer
Head
Dept. of Mathematics
April, 2019

Dr. Muhammad Abdul Qadir
Dean
Faculty of Computing
April, 2019

Author's Declaration

I, **Sundus Iqbal** hereby state that my M.Phil thesis titled "**Digital Signature based on Matrix Power Function**" is my own work and has not been submitted previously by me for taking any degree from Capital University of Science and Technology, Islamabad or anywhere else in the country/abroad.

At any time if my statement is found to be incorrect even after my graduation, the University has the right to withdraw my M.Phil Degree.

(Sundus Iqbal)

Registration No: MMT-171004

Plagiarism Undertaking

I solemnly declare that research work presented in this thesis titled "*Digital Signature Based on Matrix Power Function*" is solely my research work with no significant contribution from any other person. Small contribution/help wherever taken has been dully acknowledged and that complete thesis has been written by me.

I understand the zero tolerance policy of the HEC and Capital University of Science and Technology towards plagiarism. Therefore, I as an author of the above titled thesis declare that no portion of my thesis has been plagiarized and any material used as reference is properly referred/cited.

I undertake that if I am found guilty of any formal plagiarism in the above titled thesis even after award of M.Phil Degree, the University reserves the right to withdraw/revoke my M.Phil degree and that HEC and the University have the right to publish my name on the HEC/University website on which names of students are placed who submitted plagiarized work.

(Sundus Iqbal)

Registration No: MMT-171004

Acknowledgements

First of all, I would like to thank **Almighty Allah** for His countless blessings in my life. He has gifted me a loving family and excellent teachers. He supports me in every path of life.

I would like to express my special thanks to my kind supervisor **Dr. Rashid Ali** for his motivation. His unfailing patience and encouragement kept me in good stead. I would never be able to forget his key contribution to one of the most fruitful endeavor of my life. I have appreciated the guidance for my supervisor and feeling proud to be a student of such great teacher.

Also, many thanks are due to all teachers of CUST Islamabad Dr. Muhammad Sagheer, Dr. Abdul Rehman Kashif, Dr. Shafqat Hussain, Dr. M. Afzal and Dr. Rashid Ali for their appreciation and support.

I am grateful to the management staff of Capital University of Science and Technology, Islamabad for providing a friendly environment for studies.

I am grateful to my mother **Qamar Un Nisa** for her prayers, love and motivation. I would like to thank my brother **M. Wajahat Janjua** and sister **Ayesha Janjua** for their support in completing my degree program. They supported and encouraged me throughout my life. I would like to thank my all family members and my husband **Haider Ali** for their continuous support and patience during my research work.

I also feel honored to have such supporting friends. I would like to specially thank my friends Sania Mehmood and Saba Ramazan, Mehwish Seher, Saadia Noor and Sumaira Bibi for providing me the strength to get focused toward my main objectives.

Finally, I am obliged to all people who have shared their knowledge and supported me all along.

Abstract

Digital signature provides authenticity, integrity and non-repudiation. In digital signature the sender signs the message with his/her private key and sends it to the receiver. The receiver verifies the signature by using public key of the sender. A digital signature attaches the identity of the signer to the document and records a binding commitment to the document. We modified the scheme of S. K. Rososhek using matrix power function (MPF) instead of matrix inverses. We proposed to take matrices from $GL_m(\mathbb{Z}_n)$ where the base matrix is defined over a semigroup and power matrices over a semiring. The security of our proposed scheme relies on the solution of matrix multivariate quadratic system of equations over the finite field. Matrix power function (MPF) is a one way function since its inversion is related with the solution of known multivariate quadratic problem which is NP-complete over any field. The scheme is illustrated by various examples. For constructing such examples, the algorithms for various computations related to calculating matrix power function are implemented in Computer Algebra System "Applied Computations in Commutative Algebra" (ApCoCoA).

Contents

List of Figures

List of Tables

Abbreviations

MPF	Matrix Power Function
AES	Advanced Encryption Standard
DES	Data Encryption Standard
2DES	Double Data Encryption Standard
3DES	Triple Data Encryption Standard
RSA	Rivest Shamir Adleman
MMDS	Modular Matrix based Digital Signature
DLP	Discrete Log Problem
IFP	Integer Factorization Problem
GL	General Linear Group
GF	Galois field

Symbols

$\mathcal{M}$	Plaintext or Message
$\mathcal{C}$	Ciphertext
$\mathcal{K}$	Key
E	Encryption Algorithm
D	Decryption Algorithm
R	Ring
$\mathbb{Z}$	Set of integers
$\mathbb{Q}$	Rational numbers
$\mathbb{C}$	Complex numbers
$\mathbb{G}$	Group
$\mathbb{S}$	Semi-Group
S	Semi-Ring
$\mathbb{N}$	Natural numbers
$M_n(R)$	Matrix Ring
H	Hash Function

Chapter 1

Introduction

1.1 Cryptography

Cryptography is the practice and study of techniques for secure communication in the presence of third parties known as adversaries. The security of communication remained a major problem from very beginning. Cryptography is related to creating and evaluating protocols that prevent adversaries or the public from reading private messages [1].

In 1900 BC, Egyptian used the concept of cryptography for the first time in the history. Around 100 BC, Julius Caesar used cryptographic method known as Caesar cipher [2] to convey secret messages to his army generals in world war I. With the passage of time, new methods in cryptography were developed that provided more security and safety. In cryptography, we convert plaintext into ciphertext for the transmission over the public channel. Plaintext is converted into ciphertext with the help of an encryption algorithm, whereas ciphertext is changed into plaintext through the corresponding decryption algorithm. Both sender and receiver use a secret information for encryption and decryption algorithms and this secret information is known as a key. This is known as cryptosystem. Cryptography is divided into two main branches. The Symmetric key cryptography and the Asymmetric key cryptography. In **Symmetric key cryptography** [3] a single key is used in both encryption and decryption algorithm which is known to sender and receiver. Examples are DES [4] and AES [5]. The main drawback in symmetric key cryptography is key distribution. To overcome this problem Diffie and Hellman proposed Public key cryptography or Asymmetric key cryptography in 1976. In **Asymmetric key cryptography** [1], recipient has two different keys for communication one is public key that is made public and the other is private key that is kept secret. For example, RSA [6], ElGamal [7] and Elliptic curve cryptosystems [8].

Cryptography is not only about providing the security of information such as confidentiality, but it also provides data integrity, and data authentication. Where, confidentiality is about hiding information from all but those who are not authorized to see it. Data integrity prevents unauthorized alteration of data. Authentication is related to identification. Two entities participating into a communication should identify each other. Non-repudiation prevents an entity from refusing previous commitments or actions [9, 10].

1.2 Digital Signatures

A cryptographic primitive which is fundamental in authentication, authorization, and non-repudiation is the digital signature [6]. The purpose of a digital signature is to provide a means for an entity to bind its identity to a piece of information. Whitfield Diffee and Martin Hellman invented public key cryptography and digital signature schemes and published in their paper "New Direction in Cryptography" [6]. In their digital signature scheme, every user has identity $i.e$ his/her public key and also holds corresponding secret key. Signature can be verified using private key of the sender. Rivest, Shamir and Adleman constructed the first digital signature scheme. Their scheme is based on assumption that is known as "RSA assumption". Goldwasser *et al.* also work on digital scheme [11] (for more details also see [12]). Rompher showed how to construct a digital signature scheme using one way function. Genaro and Helevi [13] Cramer and Shoup [14] proposed the first

signature schemes whose effectiveness is appropriate for practical use and it is secure against adaptive chosen message attacks.
Now a days, mostly used public key exchange protocols and digital signature schemes such as the RSA signature scheme [15] and Elgamal digital signature scheme [7] are based on the structure of some abelian groups. The hardness of these signature schemes is based on difficulty of solving certain problems over a finite field. Most common of these problems are discret logarithm problem (DLP) [16] and Integer factorization problem (IFP). The security of well recognized signature schemes RSA and ElGamal based respectively on IFP and DLP.

1.3 Current Research

In this research, we focused on modular matrix based digital signature (MMDS) scheme introduced by S. K. Rososhek [17]. He proposed MMDS with matrices from $GL_2(\mathbb{Z}_n)$. His construction is based on finite field $\mathbb{Z}_n$. Also, he uses conjugacy search problem [18] in his scheme. We mainly focused on the modification of digital signature scheme of S. K. Rososhek. For this purpose we use matrix power function (MPF)[19]. Our scheme has become more secure as attacker has to solve multivariate quadratic system of equations and matrix decomposition problem *i.e* $BC = A$, to get access to secret key and to generate signature, which is computationally infeasible. Using matrix power function we constructed examples for the illustration of our modified and improved digital signature scheme. Using computer algebra system ApCoCoA [20] we developed programs for the calculations.

1.4 Thesis layout

The thesis is composed as follows:

1. In **Chapter 2**, we discussed the fundamental ideas and definition of cryptography. Then we discussed the hash functions and their properties. Later on we discussed Matrix power function and its properties.

2. In **Chapter 3**, we presented the review of fast and secure modular matrix digital signature scheme given by S. K. Rososhek [17]. For that purpose we discussed various known digital signature schemes. Furthermore, we described the concept of modular matrix digital signature scheme with the help of an example.

3. In **Chapter 4**, we discussed the modified form of digital signature scheme called fast and secure modular matrix based digital signature scheme given by S. K. Rososhek [17]. In modified scheme we use Matrix Power Function (MPF) to improve the security of scheme. The modified digital signature scheme is illustrated by examples.

Chapter 2

Preliminaries

2.1 Cryptography

Cryptography is a technique used to protect communications and information by means of codes so that only those for whom the information is proposed can read and process it. A system in which we convert data or message (plaintext) into secret codes (ciphertext) using encryption algorithm and convert secret codes back into message using decryption algorithm is known as Cryptosystem [1, 21].
There are five basic components of a cryptosystem:

1. Plaintext Space $\mathcal{M}$

2. Ciphertext Space $\mathcal{C}$

3. Encryption algorithm E

4. Decryption algorithm D

5. Key $\mathcal{K}$

Cryptography is categorized into two types:

- Symmetric key cryptography (Secret Key Cryptography)
- Asymmetric key cryptography (Public Key Cryptography)

2.1.1 Symmetric Key Cryptography

In symmetric or secret key cryptography [3], the sender and receiver of a message have a same key for both encryption and decryption. This key is known as secret key. A model of symmetric key cryptography is shown in Figure 2.1.

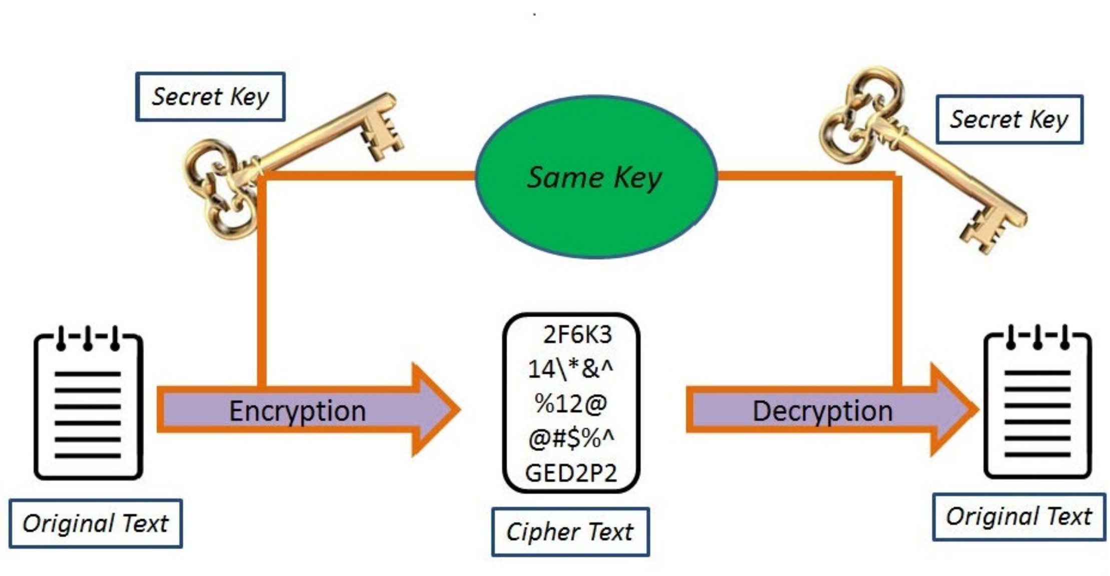

Figure 2.1: Symmetric Key Cryptography

The examples of symmetric key cryptography are Data Encryption Standard (DES) [4], Double Data Encryption Standard (2DES) [1], Triple Data Encryption Standard (TDES) [1] and Advanced Encryption standard (AES) [5].
The main disadvantage of symmetric cryptosystem is key sharing which means that the secret key is to be transmitted to each party involved in communication. Electronic communications used for this purpose is not a secure way of exchanging keys because anyone can tap communication channels. The only protected way of switching keys would be exchanging them personally but it could be a difficult task.

2.1.2 Asymmetric Key Cryptography

To solve the problem of symmetric key cryptography, asymmetric cryptosystem is proposed by Diffie-Helman in 1976 [6]. In asymmetric key cryptography [1], there are two keys used for the encryption and decryption of data *i.e* one is known to everybody called public key and the other is kept secret which is called private key. The Asymmetric key cryptography is shown in Figure 2.2. Here sender encrypts original text using public key and encryption algorithm to obtain cipher text. The secret key and decryption algorithm are used to obtain original text.

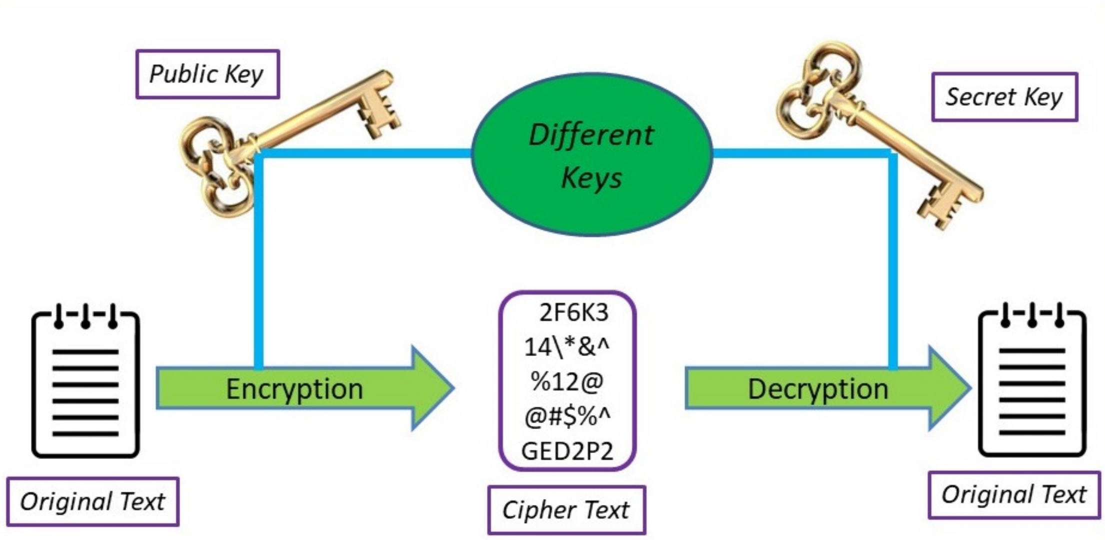

Figure 2.2: Asymmetric key

RSA cryptosystem [15], ElGamal cryptosystem [7] are the examples of asymmetric key cryptography.
Diffie and Hellman vision the cryptosystem based on trapdoor function (which is easy to compute in one direction but difficult to compute in other direction) [22]. Diffie-Hellman protocol relies on the following hard problem.

Definition 2.1.1 (Discrete Logarithm Problem).
Given x, $y \in \mathbb{Z}_p$ such that,

$$x^n = y \mod p$$

then finding n is known as the discrete logarithm problem[23, 24].

The first system which have property of trapdoor function is RSA [15] cryptosystem. RSA uses the difficulty of integer factorization problem as the underlying hard problem.

Definition 2.1.2 (**Integer Factorization Problem**).

Let n be a given number, the problem of decomposition of n to the product of primes p_α and q_α such that $n = p_\alpha q_\alpha$ is called integer factorization problem [23, 25].

Definition 2.1.3 (**Matrix Decomposition Problem**).

Factorization of a matrix into a product of matrices $i.e$ $A = L_1 U_1$ is known as matrix decomposition problem.

2.2 Mathematical Background

In this section, we recall some definitions that are used in the thesis.

Definition 2.2.1 (**Group**).

A group [26] is a non-empty set $\mathbb{G}$ on which there is a binary operation "$*$" such that,

1. **Closure:** For all $p, t \in \mathbb{G}$, $p * t \in \mathbb{G}$
2. **Associativity:** $p * (t * m) = (p * t) * m$ for all $p\,, t, m \in \mathbb{G}$
3. **Identity:** There is an element $e^{'} \in \mathbb{G}$ such that $p * e^{'} = e^{'} * p = p$ for all $p \in \mathbb{G}$
4. **Inverse:** If $p \in \mathbb{G}$, then there is an element $p_1 \in \mathbb{G}$ such that $p * p_1 = p_1 * p = e^{'}$

   Example 2.2.2. Following examples illustrate the above definition:

- Set of integers $\mathbb{Z}$ is group with respect to addition of integers.
- General linear group of order n $i.e$ $GL_n(\mathbb{Z})$ is group of invertible matrices under matrix multiplication where $\mathbb{Z}$ is the set of integers.
- The set of $\mathbb{Z}$ is not a group with respect to subtraction.

Definition 2.2.3 (**Abelian Group**).

A group $\mathbb{G}$ is called abelian if the binary operation "$*$" is commutative that is, $p * t = t * p$ for all $p, t \in \mathbb{G}$ [26].

Definition 2.2.4 (Finite Group).

A group $\mathbb{G}$ is called finite if it contains finitely many elements. The number of elements in a finite group is called its order. Order of finite group $\mathbb{G}$ is denoted by $|\mathbb{G}|$ [27].

Definition 2.2.5 (Semi Group).

A semigroup is a pair $(\mathbb{S}, *)$ in which $\mathbb{S}$ is a non-empty set and " $*$ " is a binary operation on $\mathbb{S}$ and satisfies the associative property. That is,

$$(p * t) * m = p * (t * m)$$

holds for all $p, t, m \in \mathbb{S}$ [28].

Example 2.2.6. Following are the examples of semi group.

- The set $\mathbb{W} = \{0, 1, 2, ...\}$ whole numbers forms a semigroup, under the addition and multiplication.
- The set $\mathbb{N} = \{1, 2, 3, ...\}$ of all natural numbers gives semigroup, under the addition and multiplication.

Definition 2.2.7 (Ring).

A ring [29] is a set R equipped with two binary operations $(+, \cdot)$ must satisfy the following ring axioms.

- $(R, +)$ is an ring if it satisfies the following axioms:

  1. **Closure Law:**

     For all $p, t \in R$, then $p + t \in R$.

  2. **Associative Law:** For all $p, t, m \in R$, then $(p + t) + m = p + (t + m)$.

  3. **Existence of Identity:**

     There exists an element 0 in R, such that for all elements p in R, the equation $0 + p = p + 0 = p$ holds.

4. **Existence of Inverse:**

   For each p in R, there exists an element q in R such that $p + q = q + p = 0$.

5. **Commutative Law:**

   For all $p, t \in R$, then $p + t = t + p$.

- $(R, \cdot)$ is required to be an a semi-group or monoid if it satisfies:

1. **Closure Law:**

   For all $p, t \in R$, then $p \cdot t$ is also in R.

2. **Associative Law:**

   For all $p, t, m \in R$, then $(p \cdot t) \cdot m = p \cdot (t \cdot m)$.

3. **Existence of Identity:**

   There exists an element $e^{'}$ in R, such that for all elements p in R, the equation $e^{'} \cdot p = p \cdot e^{'} = p$ holds.

4. **Distributive Laws for Multiplication over Addition:**

   For all $p, t, m \in R$, the following equations holds.

$$p \cdot (t + m) = (p \cdot t) + (p \cdot m)$$

$$(p + t) \cdot m = (p \cdot m) + (t \cdot m)$$

For more details also see [30].

Example 2.2.8. Following are the examples of Rings.

- The set of integers $\mathbb{Z}$ under usual addition and multiplication.
- The set of rational numbers $\mathbb{Q}$ under usual addition and multiplication.
- The set of complex numbers $\mathbb{C}$ under usual addition and multiplication.
- $M_2(\mathbb{R})$ is the ring of 2×2 matrices with co-efficients from $\mathbb{R}$ under matrix addition and multiplication.

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} + \begin{pmatrix} d & e \\ f & g \end{pmatrix} = \begin{pmatrix} a+d & b+e \\ c+f & d+g \end{pmatrix}$$

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} d & e \\ f & g \end{pmatrix} = \begin{pmatrix} ad+bf & ae+bg \\ cd+df & ce+dg \end{pmatrix}$$

Definition 2.2.9 (**Semiring**).

A semiring is a pair $(S, +, \cdot)$ in which S is a non-empty set "+" and "." are binary associative operations of addition and multiplication on S and satisfies associative property. That is,

$$(p+t)+m = p+(t+m)$$

$$(p\,.\,t)\,.\,m = p\,.\,(t\,.\,m)$$

hold for all $p, t, m \in S$.

Also, multiplication is distributive over addition from either side. That is, the equations

$$p \cdot (t+m) = (p \cdot t) + (p \cdot m)$$

$$(p+t) \cdot m = (p \cdot m) + (t \cdot m)$$

for all $p, t, m \in S$,.

Definition 2.2.10 (**Matrix Ring**).

A collection of all square matrices of any order over ring R, with the operations of matrix addition and matrix multiplication. The matrix ring is denoted by $M_n(R)$ [26].

Definition 2.2.11 (**Field**).

A commutative ring $(R, +, \cdot)$ with multiplicative inverses is called field.

Example 2.2.12. Following are the examples of fields.

- Set of real numbers $\mathbb{R}$ and set of rational numbers $\mathbb{Q}$ under addition $(+)$ and multiplication $(\cdot)$.
- For every prime p, set of integers $\mathbb{Z}_p$ under mod p is a field, also it is denoted by $\mathbb{F}_p$.
- The set of integers with usual addition and multiplication is a ring but it is not a field as integers are not invertible with respect to multiplication.

Definition 2.2.13 (**Finite Field**).
A field having finite number of elements is known as finite field.

2.3 Galois Fields

Galois field is a finite field in which an order of a finite field is a power of a prime p_1^n. Its representation is $GF(p_1^n)$ [31].

Example 2.3.1. Finite field $\mathbb{F}_2$ having elements 0 and 1 with addition and multiplication is defined in Table 2.1 and Table 2.2 below.

+	0	1
0	0	1
1	1	0

Table 2.1: For addition

.	0	1
0	0	0
1	0	1

Table 2.2: For multiplication

Definition 2.3.2 (**Modular Multiplicative Inverse**).
Given two integers r_1 and m_1, find multiplicative inverse of r_1 under modulo m_1. The multiplicative inverse under modulo m_1 is an integer x_1 such that,

$$r_1\, x_1 \equiv 1 \pmod{m_1}$$

The value of x_1 should be in $\{0, 1, 2, \cdots, m_1 - 1\}$, *i.e,* in the ring of integers modulo m_1.

The multiplicative inverse of r_1 modulo m_1 exists if and if r_1 and m_1 are relatively prime that is, $\gcd(r_1, m_1) = 1$.

Algorithm 2.3.3 (**Multiplicative Inverse in Finite Field**).

To find the inverse of $r_1 \mod m_1$, the following steps are to be performed.

Input: An integer r_1 and an irreducible integer m_1.

Output: $r_1^{-1} \mod m_1$.

1. Initialize six integers U_i and V_i for $i = 1, 2, 3$ as
$(U_1, U_2, U_3) = (1, 0, m_1)$
$(V_1, V_2, V_3) = (0, 1, r_1)$

2. If $V_3 = 0$, return $U_3 = \gcd\,(r_1, m_1)$; no inverse of r_1 exists in mod m_1

3. If $V_3 = 1$ then return $V_3 = \gcd(r, m)$ and $V_2 = r^{-1} \mod m$

4. Now divide U_3 with V_3 and find the quotient Q when U_3 is divided by V_3.

5. Set $(T_1, T_2, T_3) = ((U_1 - Q.V_1), (U_2 - Q.V_2), (U_3 - Q.V_3))$

6. Set $(U_1, V_2, U_3) = (V_1, V_2, V_3)$

7. Set $(V_1, V_2, V_3) = (T_1, T_2, T_3)$

8. Goto step 2 [32].

Definition 2.3.4 (**Automorphism**).

An automorphism is simply a bijective homomorphism of an object with itself. Let $(\mathbb{G}, *)$ be a group. Let $\phi : \mathbb{G} \longrightarrow \mathbb{G}$ be a (group) isomorphism from $\mathbb{G}$ to itself. Then ϕ is a group automorphism [26].

Example 2.3.5. Some examples of automorphism are given below.

1. $\phi : C_3 \longrightarrow C_3$ given by $\phi(x) = 3 - x$ for all $x \in C_3$ is an automorphism.

2. The Klein-4 group.

Definition 2.3.6 (Euler's Totient Function).

Euler's totient function is defined as the number of positive integers less than n which are relatively prime to n. It is denoted by $\phi(n)$. For any prime p_α

$$\phi(n) = \phi(p_\alpha) = p_\alpha - 1 \tag{2.1}$$

and for $n = p_\alpha\, q_\alpha$, where p_α and p_α are prime numbers then,

$$\phi(n) = \phi(p_\alpha)\,\phi(q_\alpha \tag{2.2}$$

$$= (p_\alpha - 1) \times (q_\alpha - 1) \tag{2.3}$$

Example 2.3.7. Let prime $p_\alpha = 17$, then using (2.1) we get,

$$\phi(17) = 17 - 1$$

$$\phi(17) = 16$$

If $n = 35$, then by using (2.3) we get,

$$\begin{aligned} \phi(35) &= \phi(5)\,\phi(7) \\ &= (5 - 1) \times (7 - 1) \\ &= 24 \end{aligned}$$

Definition 2.3.8 (Hash Function).

A hash function maps a variable length message into a fixed-length hash value, or message digest as shown in Figure 2.3. The hash value is representative of the original string of characters, but is normally smaller than the original [33, 34]. Secure Hash Algorithm (SHA) is commonly used hash function. National institute of standards and technology (NIST) devoloped SHA in 1993.

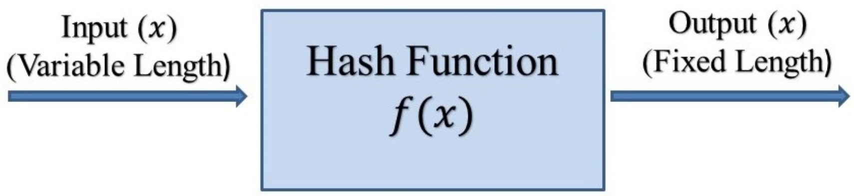

Figure 2.3: Hash Function

A hash value can be used to uniquely identify secret information. Hash function should be collision resistant. Some known cryptograpic hash functions are SHA-1, which produces a hash value of 160 bits and the block size 512 bits, SHA-2 generates 126 bits ouput [35], SHA-512 [36] and MD6 [37].
There are several tools to calculate cryptographic hash functions like Hash tool 1.2, Crypto-precision and DNS.

- **Properties of Hash Function**

Following are the properties of hash function:

1. **Performance**

   It is easy to compute $H(m^{'})$, where $m^{'}$ is the message.

2. **One Way Function**

   If $H(m^{'})$ is given it is in-feasible to find $m^{'}$.

3. **Weak Collision Resistance**

   If $m^{'}$ and $H(m^{'})$ are given it is very difficult to find $m^{''}$ such that $H(m^{''}) = H(m')$.

4. **Strong Collision Resistance**

   It is computationally in-feasible to find two different inputs m_1, m_2 such that $H(m_1) = H(m_2)$.

2.4 Matrix Power Function

"The MPF is based on a matrix powered by another matrix. This function is some generalization of discrete exponent function in cyclic groups by its expansion in matrix set" [19].

Definition 2.4.1.

"The left-sided MPF corresponding to a matrix X powered by a matrix L_s on the left side is equal to matrix $U = u_{ij}$ has the following form "[19]:

$$^{L_s}X = U, \ u_{ij} = \prod_{k=1}^{m} x_{kj}^{l_{ik}}$$

Definition 2.4.2.

"The right-sided MPF corresponding to matrix a X powered by matrix a R_s on the right side is equal to matrix $V = v_{ij}$ has the following form" [19]:

$$X^{R_s} = V, \ v_{ij} = \prod_{k=1}^{m} x_{ik}^{r_{kj}}$$

Note: The matrix which is powered by another matrix in named as base matrix and the matrix that is powering the base matrix are known as power matrix. In general, we define the base matrix over a semigroup and power matrices over a semiring.

• **Representation of Left and Right Matrix Power Function:**

Let us assume that matrices L_s and X have two columns and two rows then matrix U can be expressed in the following way [38]:

$$U = \ ^{L_s}X = {}^{\begin{pmatrix} \ell_{11} & \ell_{12} \\ \ell_{21} & \ell_{22} \end{pmatrix}} \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix} \tag{2.4}$$

$$U = \begin{pmatrix} x_{11}^{\ell_{11}} x_{21}^{\ell_{12}} & x_{12}^{\ell_{11}} x_{22}^{l_{12}} \\ x_{11}^{\ell_{21}} x_{21}^{\ell_{22}} & x_{12}^{\ell_{21}} x_{22}^{\ell_{22}} \end{pmatrix} \tag{2.5}$$

Matrix V can be expressed in the following way:

$$V = X^{R_s} = \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix}^{\begin{pmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{pmatrix}} \tag{2.6}$$

$$V = \begin{pmatrix} x_{11}^{r_{11}} x_{12}^{r_{21}} & x_{11}^{r_{12}} x_{12}^{r_{22}} \\ x_{21}^{r_{11}} x_{22}^{r_{21}} & x_{21}^{r_{12}} x_{22}^{r_{22}} \end{pmatrix}$$

Example 2.4.3. Let $X = \begin{pmatrix} 5 & 2 \\ 3 & 8 \end{pmatrix}$ and $R_s = \begin{pmatrix} 11 & 6 \\ 6 & 11 \end{pmatrix}$, then we compute V over $\mathbb{Z}_{21}$ as:

$$V = X^{R_s} = \begin{pmatrix} 5 & 2 \\ 3 & 8 \end{pmatrix}^{\begin{pmatrix} 11 & 6 \\ 6 & 11 \end{pmatrix}} \mod 21$$

$$= \begin{pmatrix} (5)^{11}.(2)^{6} & (5)^{6}.(2)^{11} \\ (3)^{11}.(8)^{6} & (3)^{6}.(8)^{11} \end{pmatrix}$$

$$V = \begin{pmatrix} 17 & 11 \\ 12 & 15 \end{pmatrix} \mod 21$$

Similarly,
Let $X = \begin{pmatrix} 5 & 2 \\ 3 & 8 \end{pmatrix}$ and $L_s = \begin{pmatrix} 7 & 6 \\ 6 & 7 \end{pmatrix}$, then we compute U over $\mathbb{Z}_{21}$ as:

$$U = {}^{L_s}X = {}^{\begin{pmatrix} 7 & 6 \\ 6 & 7 \end{pmatrix}}\begin{pmatrix} 5 & 2 \\ 3 & 8 \end{pmatrix} \mod 21$$

$$= \begin{pmatrix} (5)^{7}.(3)^{6} & (2)^{7}.(8)^{6} \\ (5)^{6}.(3)^{7} & (2)^{6}.(8)^{7} \end{pmatrix}$$

$$U = \begin{pmatrix} 12 & 11 \\ 12 & 8 \end{pmatrix} \mod 21$$

We implemented codes in ApCoCoA, which calculates MPF.

2.4.1 Properties of MPF:

Following are the properties of matrix power function (MPF) [19].

$$^{R_s}({}^{L_s}X) = {}^{(R_sL_s)}X = {}^{R_sL_s}X \tag{2.7}$$

$$(X^{L_s})^{R_s} = X^{(L_sR_s)} = X^{L_sR_s} \tag{2.8}$$

$$^{L_s}(X^{R_s}) = ({}^{L_s}X)^{R_s} = {}^{L_s}X^{R_s} \tag{2.9}$$

Proof

To prove the Equation 2.7, let X belongs to a semi-group $\mathbb{S}$. Let R_s and L_s belongs to a semi-ring $\mathbb{R}$.

$$X = \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix} \in S$$

$$L_s = \begin{pmatrix} \ell_{11} & \ell_{12} \\ \ell_{12} & \ell_{11} \end{pmatrix} \in R$$

$$R_s = \begin{pmatrix} r_{11} & r_{12} \\ r_{12} & r_{11} \end{pmatrix} \in R$$

Note that,

$$L_sR_s = \begin{pmatrix} \ell_{11}r_{11} + \ell_{12}r_{12} & \ell_{11}r_{12} + \ell_{12}r_{11} \\ \ell_{12}r_{11} + \ell_{11}r_{12} & \ell_{12}r_{12} + \ell_{11}r_{11} \end{pmatrix}$$

$$= \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix}^{\begin{pmatrix} \ell_{11}r_{11}+\ell_{12}r_{12} & \ell_{11}r_{12}+\ell_{12}r_{11} \\ \ell_{12}r_{11}+\ell_{11}r_{12} & \ell_{12}r_{12}+\ell_{11}r_{11} \end{pmatrix}}$$

So,

$$X^{(L_sR_s)} = \begin{pmatrix} x_{11}^{l_{11}r_{11}+l_{12}r_{12}}.x_{21}^{l_{12}r_{11}+l_{11}r_{12}} & x_{11}^{l_{11}r_{12}+l_{12}r_{11}}.x_{12}^{l_{12}r_{12}+l_{11}r_{11}} \\ x_{21}^{l_{11}r_{11}+l_{12}r_{12}}.x_{22}^{l_{12}r_{11}+l_{11}r_{12}} & x_{21}^{l_{11}r_{12}+l_{12}r_{11}}.x_{22}^{l_{12}r_{12}+l_{11}r_{11}} \end{pmatrix} \tag{2.10}$$

Now

$$X^{L_s} = \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix}^{\begin{pmatrix} \ell_{11} & \ell_{12} \\ \ell_{12} & \ell_{11} \end{pmatrix}}$$

$$= \begin{pmatrix} x_{11}^{\ell_{11}}.x_{12}^{\ell_{12}} & x_{11}^{\ell_{12}}.x_{12}^{\ell_{11}} \\ x_{21}^{\ell_{11}}.x_{22}^{\ell_{12}} & x_{21}^{\ell_{12}}.x_{22}^{\ell_{11}} \end{pmatrix}$$

$$(X^{L_s})^{R_s} = \begin{pmatrix} x_{11}^{\ell_{11}}.x_{12}^{\ell_{12}} & x_{11}^{l_{12}}.x_{12}^{\ell_{11}} \\ x_{21}^{\ell_{11}}.x_{22}^{\ell_{12}} & x_{21}^{\ell_{12}}.x_{22}^{\ell_{11}} \end{pmatrix}^{\begin{pmatrix} r_{11} & r_{12} \\ r_{12} & r_{11} \end{pmatrix}}$$

$$= \begin{pmatrix} (x_{11}^{\ell_{11}}.x_{12}^{\ell_{12}})^{r_{11}}.(x_{11}^{\ell_{12}}.x_{12}^{\ell_{11}})^{r_{12}} & (x_{11}^{\ell_{11}}.x_{12}^{\ell_{12}})^{r_{12}}.(x_{11}^{\ell_{12}}.x_{12}^{\ell_{11}})^{r_{11}} \\ (x_{21}^{\ell_{11}}.x_{22}^{\ell_{12}})^{r_{11}}.(x_{21}^{\ell_{12}}.x_{22}^{\ell_{11}})^{r_{12}} & (x_{21}^{\ell_{11}}.x_{22}^{\ell_{12}})^{r_{12}}.(x_{21}^{\ell_{12}}.x_{22}^{\ell_{11}})^{r_{11}} \end{pmatrix}$$

$$(X^{L_s})^{R_s} = \begin{pmatrix} x_{11}^{\ell_{11}r_{11}+\ell_{12}r_{12}}.x_{21}^{\ell_{12}r_{11}+\ell_{11}r_{12}} & x_{11}^{\ell_{11}r_{12}+\ell_{12}r_{11}}.x_{12}^{\ell_{12}r_{12}+\ell_{11}r_{11}} \\ x_{21}^{\ell_{11}r_{11}+\ell_{12}r_{12}}.x_{22}^{l_{12}r_{11}+\ell_{11}r_{12}} & x_{21}^{\ell_{11}r_{12}+\ell_{12}r_{11}}.x_{22}^{\ell_{12}r_{12}+\ell_{11}r_{11}} \end{pmatrix} \tag{2.11}$$

From (2.10) and (2.11), we see that $X^{L_sR_s} = (X^{L_s})^{R_s}$

Similarly, one can prove Equations (2.8) and (2.9).

Corollary 2.4.4.

Let H be a function which is not mutually associative with the MPF [39]. It means that:

$$H({}^{L_s}(X^{R_s})) \neq {}^{L_s}((H(X))^{R_s} \tag{2.12}$$

Here is the counter example.

Example 2.4.5. Let $H = 5$, $L = \begin{pmatrix} 7 & 6 \\ 6 & 7 \end{pmatrix}$, $R = \begin{pmatrix} 11 & 6 \\ 6 & 11 \end{pmatrix}$ and $X = \begin{pmatrix} 5 & 2 \\ 3 & 8 \end{pmatrix}$. From Equation (2.12),

$$
\begin{aligned}
X^{R_s} &= \begin{pmatrix} 5 & 2 \\ 3 & 8 \end{pmatrix}^{\begin{pmatrix} 11 & 6 \\ 6 & 11 \end{pmatrix}} \\
&= \begin{pmatrix} 5^{11}.2^{6} & 5^{6}.2^{11} \\ 3^{11}.8^{6} & 3^{6}.8^{11} \end{pmatrix} \\
&= \begin{pmatrix} 17 & 11 \\ 12 & 15 \end{pmatrix}
\end{aligned}
$$

Now by using (2.5),

$$
\begin{aligned}
{}^{L_s}(X^{R_s}) &= {}^{\begin{pmatrix} 7 & 6 \\ 6 & 7 \end{pmatrix}}\begin{pmatrix} 17 & 11 \\ 12 & 15 \end{pmatrix} \\
&= \begin{pmatrix} (17)^{7}.(12)^{6} & (11)^{7}.(15)^{6} \\ (17)^{6}.(12)^{7} & (11)^{6}.(15)^{7} \end{pmatrix} \\
&= \begin{pmatrix} 3 & 18 \\ 12 & 15 \end{pmatrix} \mod 21 \\
H({}^{L_s}(X^{R_s})) &= \begin{pmatrix} 15 & 6 \\ 18 & 12 \end{pmatrix} \mod 21
\end{aligned}
$$

Similary,

$$
{}^{L_s}((H(X))^{R_s} = {}^{\begin{pmatrix} 7 & 6 \\ 6 & 7 \end{pmatrix}}\begin{pmatrix} 4 & 10 \\ 15 & 19 \end{pmatrix}^{\begin{pmatrix} 11 & 6 \\ 6 & 11 \end{pmatrix}} \mod 21
$$

From Equation (2.5) and Equation (2.6) we get,

$$^{L_s}((H(X))^{R_s} = \begin{pmatrix} 9 & 12 \\ 15 & 3 \end{pmatrix} \mod 21$$

Hence, $H(^{L_s}(X^{R_s})) \neq \ ^{L_s}((H(X))^{R_s}$

Chapter 3

Fast And Secure Modular Matrix Based Digital Signature

In this chapter we discussed digital signature, then we discussed Elgamal signature scheme [40] which is based on DLP and RSA digital signature scheme [41]. Finally, Modular Matrix Based Digital Signature scheme [17] is described.

3.1 Digital Signature

A digital signature is the electronic equivalent of a person's physical signature. It is also a guarantee that information has not been modified. A digital signature is an authentication mechanism that enables the sender of a message to attach a code that acts as a signature. Digital signatures are based on asymmetric cryptography. One can generate two keys using public key algorithm. One key is called private key and the other is called public key and both are mathematically linked. To generate a digital signature, one way hash function of data to be signed, is produced. The private key is then used to encrypt the hash. Digital signature consist on three algorithms namely, key generation, signature generation and signature verifcation. A typical model of digital signature using hash function is shown in Figure 3.1.

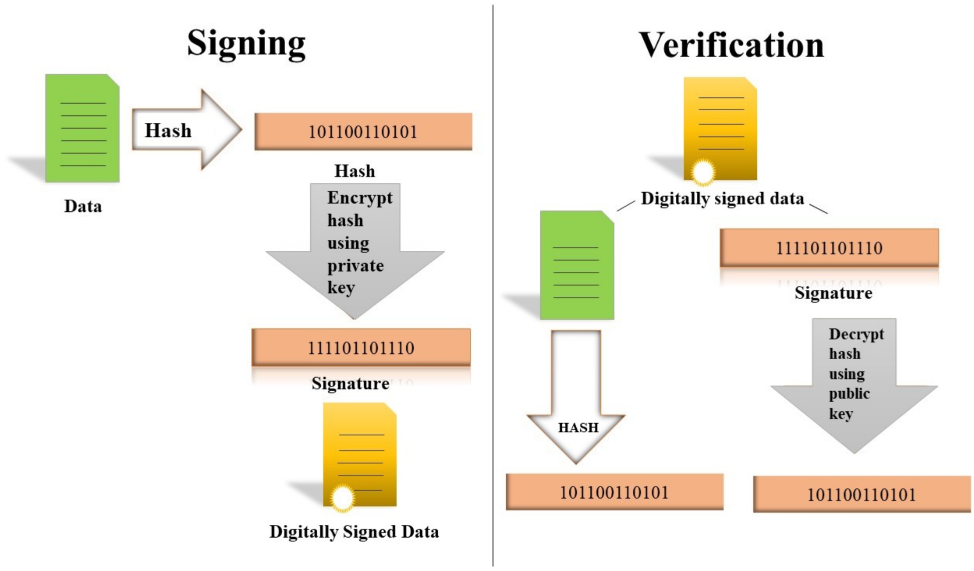

Figure 3.1: Digital Signature

Various known digital signature schemes are given below.

3.2 Elgamal Signature Scheme

The Elgamal signature scheme [40] is a digital signature scheme which is based on difficulty of computing discrete logarithms. The global elements of Elgamal digital signature are a prime p_1, base g_1 and hash H. This scheme is described in the following steps:

Key Generation

Alice generates private and public key pair as:

- Choose randomly a secret key x_1 with $1 < x_1 < p_1 - 1$.
- Compute $y_1 = g_1^{x_1} \mod p_1$
- The public key is (p_1, g_1, y_1).

Signature Generation

To sign message m_1 the signer will perform the following steps:

- Choose a random k_1 such that $0 < k_1 < p_1 - 1$ and gcd $(k_1, p_1 - 1) = 1$.
- Compute $r_1 \equiv g_1^{k_1} (\mod p_1)$

- Compute $s_1 \equiv (H(m_1) - x_1 r_1)\, k_1^{-1} (\mod p_1 - 1)$

  Then the pair (r_1, s_1) is the digital signature of message m_1.

Verification

The signature (r_1, s_1) is verified as:

- $g_1^{H(m_1)} \equiv y_1^{r_1} r_1^{s_1} (\mod p_1)$

  where $0 < r_1 < p_1$ and $0 < s_1 < p_1 - 1$.

The verifier accepts the signature if above conditions are satisfied otherwise reject it.

Correctness

The correctness of the scheme follows from the following steps:

As, $s_1 \equiv (H(m_1) - x_1 r_1) k_1^{-1} (\mod p_1 - 1)$

- $H(m_1) \equiv x_1 r_1 + s_1 k_1 (\mod p_1 - 1)$

  Fermat's little theorem implies

- $g_1^{H(m_1)} \equiv g_1^{x_1 r_1} g_1^{k_1 s_1} (\mod p_1)$

- $g_1^{H(m_1)} \equiv (g_1^{x_1})^{r_1} (g_1^{k_1})^{s_1}$

- $g_1^{H(m_1)} \equiv (y_1^{r_1})(r_1^{s_1}) (\mod p_1)$

3.3 RSA Digital Signature Scheme:

The RSA signature scheme [15] is also based on difficulty of computing discrete logarithms. To sign message m_1 following steps should be performed:

Key Generation

- Choose two large prime numbers p_1 and q_1.

- Compute $n_1 = p_1 \cdot q_1$ where n_1 is modulo.

- Compute $\phi(n_1) = (p_1 - 1)(q_1 - 1)$.

- Take $e_1 = \{1, \cdots, \phi(n_1)\}$ such that $\gcd(e_1, \phi(n_1)) = 1$.

- Compute d_1 such that $d_1 \cdot e_1 = 1(\mod \phi(n_1))$

Signature Generation

To sign message Alice will perform the following steps:

- Compute $S_1 = m_1^{d_1} (\mod n_1)$.

Verification

- Compute $m_1' = S_1^{e_1} (\mod n_1)$
  if $m_1' = m_1$ then accepts the message otherwise reject it.

Correctness

The correctness of the scheme based on following steps:
As, $m_1' = S_1^{e_1} \mod n_1$ and $S_1 = m_1^{d_1} (\mod n_1)$

- $m_1' = (m_1^{d_1})^{e_1} (\mod n_1)$
- $m_1' = (m_1^{d_1 \cdot e_1}) (\mod n_1)$
- $m_1' = m_1$

The signature generation in [17] is based on the following hard problem in group theory.

Definition 3.3.1 (Conjugacy Search Problem).
Given x, $y \in \mathbb{G}$, then

$$y = a^{-1} x\, a$$

for some $a \in \mathbb{G}$. To find an element a is called conjugacy search problem[42].

3.4 Modular Matrix Based Digital Signature Scheme (MMDS)

Recently, S. K. Rososhek proposed "Fast and Secure Modular Matrix Based Digital Signature scheme"[17]. In this scheme Alice generates key from two random invertible matrices. Then she generates signature (r, S_1) and at the end Bob verifies the signature. The whole scheme is described below in three algorithms namely, the key generation, signature generation and the signature verification.

Algorithm 3.4.1 (Key Generation).

Following steps are performed by Alice:

- Generate two random prime numbers p and q where $p \neq q$.
- Compute
$$n = pq$$
- Choose two random invertible matrices B, C in the abelian subgroup $\mathbb{G}$ of the group $GL_2(\mathbb{Z}_n)$.
- Let $\mathbb{G}$ be the set of 2×2 matrices:
$$\mathbb{G} = \left\{ \begin{pmatrix} a_1 & b_1 \\ b_1 & a_1 \end{pmatrix} \mid a_1, b_1 \in \mathbb{Z}_n \text{ and } (a_1^2 - b_1^2) \in \mathbb{Z}_n^*. \right\}$$
Where $\mathbb{Z}_n^*$ is a unit group of the residue ring $\mathbb{Z}_n$ [43].
$\mathbb{G}$ is an abelian subgroup of the $GL_2(\mathbb{Z}_n)$.
- Compute the matrix
$$A = B^{-1}C \tag{3.1}$$
- Master key is (n, A) and secret key is (B, C).

Algorithm 3.4.2 (Digital Signature Generation).

To sign the message m following steps should be performed by Alice:

- Choose a random matrix T in the subgroup $\mathbb{G}$ of the $GL_2(\mathbb{Z}_n)$.
- Choose a random matrix U in the group $GL_2(\mathbb{Z})$.
- Select random elements δ, η in the residue ring $\mathbb{Z}_n$.

- Session private key of Alice is:

$$(\delta, \eta, T, U)$$

- Let χ_A, χ_{CT} and χ_{BT} be the automorphisms of the matrix ring $M_2(\mathbb{Z}_n)$ [44] defined as:

$$\chi_A : D \to \ A^{-1}DA, \tag{3.2}$$

$$\chi_{CT} : D \to (CT)^{-1}D(CT), \tag{3.3}$$

$$\chi_{BT} : D \to (BT)^{-1}D(BT) \tag{3.4}$$

for every $D \ \in \ M_2(\mathbb{Z}_n)$.

- Further matrix r and bit string S_1 are computed as:

$$r_1 = \eta\chi_{BT}(L), \tag{3.5}$$

$$t_1 = \chi_{CT}(L), \delta t_1, \tag{3.6}$$

$$\omega = \eta + \delta \tag{3.7}$$

$$S_1 = H((m)_2 \| (\omega t_1)_2). \tag{3.8}$$

where $(m)_2$ is the bit string binary representation of the message m, $(\omega t_1)_2$ is the bit string obtained after transferring matrix ωt_1 in the string of numbers as follows:

$$\omega t_1 \ = \ \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix} \to a_1 \| a_2 \| a_3 \| a_4$$

.

- Let δt_1 be a session public key (verification key) for the verification of Alice's signature of the message m and (r_1, S_1) is the Alice's signature of the message m.

Algorithm 3.4.3 (Digital Signature Verification).

Alice send message m to Bob and Bob performed following steps:

- Bob obtain master public key (n, A) and session public key δt_1 of Alice.

  - Compute
$$\alpha \;=\; \delta t_1 \;+\; \chi_A(r_1)$$
  - Compute
$$S_1^{'} \;=\; H((m)_2 \| (\alpha)_2) \tag{3.9}$$
  - Bob accept the signature on message m send by Alice if and only if $S_1 = S_1^{'}$.

Correctness Proof:

The correctness of the above scheme is given as follows:

$$\alpha \;=\; \delta t_1 \;+\; \chi_A(r_1)$$

Using Equation (3.1), (3.5) and (3.2) we get,

$$= \delta t_1 \;+\; \eta(B^{-1}C)^{-1}(BT)(B^{-1}W)$$

From Equation (3.6),

$$\begin{aligned} &= \delta t_1 \;+\; \eta t_1 \\ &= (\delta \;+\; \eta)t_1 \;=\; \omega t_1. \end{aligned}$$

Therefore, from Equation (3.9) we get,

$$\begin{aligned} S_1^{'} &= H((m)_2 \| (\alpha)_2) \\ S_1^{'} &= H((m)_2 \| (\omega t_1)_2) \\ S_1^{'} &= S_1. \end{aligned}$$

The scheme [17] is illustrated by the following example.

Example 3.4.4. Let us take matrices of order 2 over Finite field $GL_2(\mathbb{Z}_n)$. All the calculation are performed under (mod n).

Key Generation:

- Alice select random prime numbers *i.e* $p = 5$ and $q = 7$.

- Compute

$$\begin{aligned} n &= pq \\ n &= 35 \end{aligned}$$

- Choose

$$B, C \in \mathbb{G} \subset GL_2(\mathbb{Z}_{35})$$

$$B = \begin{pmatrix} 13 & 5 \\ 5 & 13 \end{pmatrix}, C = \begin{pmatrix} 5 & 6 \\ 6 & 5 \end{pmatrix}$$

Now Alice calculates inverse of B by using $B^{-1} = \text{Adj}(B)/\det(B)$. First she calculates $\det(B) = 4$, she calculates its inverse by using Extended Euclidean Algorithm as given in Table 3.1.

So, $(4)^{-1} \mod 35 = 9$ and multiply inverse of $\det(B)$ with $\text{Adj}(B)$ to gets

Q	A_1	A_2	A_3	B_1	B_2	B_1
-	1	0	35	0	1	4
8	0	1	4	1	-8	3
1	1	-8	3	-1	9	1

Table 3.1: Extended Euclidean Algorithm $(4)^{-1} \mod 35$

inverse of B as

$$B^{-1} = \begin{pmatrix} 12 & 25 \\ 25 & 12 \end{pmatrix} \mod 35$$

- Compute

$$A = B^{-1}C$$

$$A = \begin{pmatrix} 0 & 22 \\ 22 & 0 \end{pmatrix}$$

- Master public key of Alice is:

$$n = 35, \; A = \begin{pmatrix} 0 & 22 \\ 22 & 0 \end{pmatrix}$$

- Master private key of Alice is the pair (B, C)

$$B = \begin{pmatrix} 13 & 5 \\ 5 & 13 \end{pmatrix}, \; C = \begin{pmatrix} 5 & 6 \\ 6 & 5 \end{pmatrix}.$$

Signature Generation:

For signature generation Alice performed following steps:

- Choose the matrices:

$$T = \begin{pmatrix} 11 & 7 \\ 7 & 11 \end{pmatrix} \in G \subset GL_2(\mathbb{Z}_{35}),$$

$$U = \begin{pmatrix} 3 & 5 \\ 7 & 2 \end{pmatrix} \in GL_2(\mathbb{Z}_n).$$

- Select the residues δ, η in $\mathbb{Z}_{35}$ as:

$$\delta = 7, \, \eta = 9.$$

- Session private key of Alice is:

$$\delta = 7, \; \eta = 9, \; T = \begin{pmatrix} 11 & 7 \\ 7 & 11 \end{pmatrix}, \; U = \begin{pmatrix} 3 & 5 \\ 7 & 2 \end{pmatrix}$$

- Alice compute:

$$\chi_A(U) = A^{-1}UA$$

$$\chi_A(U) = \begin{pmatrix} 0 & 8 \\ 8 & 0 \end{pmatrix} \begin{pmatrix} 3 & 5 \\ 7 & 2 \end{pmatrix} \begin{pmatrix} 0 & 22 \\ 22 & 0 \end{pmatrix}$$

$$\chi_A(U) = \begin{pmatrix} 2 & 7 \\ 5 & 3 \end{pmatrix}$$

$$\chi_{BT}(U) = (BT)^{-1}U(BT)$$

The inverse of (BT) is calculated by $\mathrm{Adj}(BT)/\det(BT)$. Alice calculates $\det(BT) = 8$, she calculates its inverse by using Extended Euclidean Algorithm as given in Table 3.2

Q	A_1	A_2	A_3	B_1	B_2	B_1
-	1	0	35	0	1	8
4	0	1	8	1	-4	3
2	1	-4	3	-2	9	2
1	-2	9	2	3	-13	1

Table 3.2: Extended Euclidean Algorithm $(8)^{-1} \mod 35$

$(8)^{-1} \mod 35 = 22$ and multiply inverse of $\det(BT)$ with $\mathrm{Adj}(A)$ to get inverse i.e $(BT)^{-1}$ as

$$(BT)^{-1} = \begin{pmatrix} 31 & 8 \\ 8 & 31 \end{pmatrix} \mod 35$$

$$\chi_{BT}(U) = \begin{pmatrix} 31 & 8 \\ 8 & 31 \end{pmatrix} \begin{pmatrix} 3 & 5 \\ 7 & 2 \end{pmatrix} \begin{pmatrix} 3 & 6 \\ 6 & 3 \end{pmatrix}$$

$$\chi_{BT}(U) = \begin{pmatrix} 3 & 7 \\ 5 & 2 \end{pmatrix}$$

Now, Now Alice calculate inverse of CT by using $(CT)^{-1} = \mathrm{Adj}(CT)/\det(CT)$.

First she calculate $\det(CT) = 13$, she calculate its inverse by using Extended Euclidean Algorithm as given in Table 3.3

Q	A_1	A_2	A_3	B_1	B_2	B_1
-	1	0	35	0	1	13
2	0	1	13	1	-2	9
1	1	-2	9	-1	3	4
2	-1	3	4	3	-8	1

Table 3.3: Extended Euclidean Algorithm $(13)^{-1} \mod 35$

So, $(13)^{-1} \mod 35 = 27$ and multiply inverse of det(13) with Adj(13) to gets inverse *i.e* 13^{-1} as:

$$13^{-1} = \begin{pmatrix} 29 & 3 \\ 3 & 29 \end{pmatrix} \mod 35$$

Also,

$$\chi_{CT}(U) = (CT)^{-1}U(CT)$$

$$\chi_{CT}(U) = \begin{pmatrix} 29 & 3 \\ 3 & 29 \end{pmatrix} \begin{pmatrix} 3 & 5 \\ 7 & 2 \end{pmatrix} \begin{pmatrix} 27 & 31 \\ 31 & 27 \end{pmatrix}$$

$$\chi_{CT}(U) = \begin{pmatrix} 2 & 5 \\ 7 & 3 \end{pmatrix}$$

- Alice compute signature (r_1, s_1)

$$r_1 = \eta \chi_{BT}(U)$$

$$r_1 = 9 \begin{pmatrix} 3 & 7 \\ 5 & 2 \end{pmatrix} = \begin{pmatrix} 27 & 28 \\ 10 & 18 \end{pmatrix}$$

$$t_1 = \chi_{CT}(U)$$

$$t_1 = \begin{pmatrix} 2 & 5 \\ 7 & 3 \end{pmatrix}$$

$$\delta t_1 = 7\begin{pmatrix} 2 & 5 \\ 7 & 3 \end{pmatrix} = \begin{pmatrix} 14 & 0 \\ 14 & 21 \end{pmatrix},$$

- Public session key if Alice is:

$$\delta t_1 = \begin{pmatrix} 14 & 0 \\ 14 & 21 \end{pmatrix}$$

$$\omega = \eta + \delta = 16$$

$$\omega t_1 = 16\begin{pmatrix} 2 & 5 \\ 7 & 3 \end{pmatrix} = \begin{pmatrix} 32 & 10 \\ 7 & 13 \end{pmatrix},$$

$$S_1 = H((m)_2 \| (\omega t)_2)$$

where $(m)_2$ is a bit string binary representation of the message m, $(\omega t_1)_2$ is a bit string obtained after transferring matrix δt_1 in the string of numbers and replace the numbers by their binary representations as follows:

$$32\|10\|7\|13 \rightarrow\ 00100000\|00001010\|00000111\|00001101.$$

Signature Verification:

Bob performed following steps:

- Bob obtain the authentic master public key of Alice.

$$n = 35,\ A = \begin{pmatrix} 0 & 22 \\ 22 & 0 \end{pmatrix},$$

and session public key $\delta t_1 = \begin{pmatrix} 14 & 0 \\ 14 & 21 \end{pmatrix}$

- For verification Bob compute:

$$\alpha = \delta t_1 + \chi_A(r_1).$$

$$\alpha \;=\; \delta t_1 \;+\; A^{-1}\, r_1\, A.$$

Now Bob calculate inverse of A by using $A^{-1} = \mathrm{Adj}(A)/\det(A)$. First he calculate $\det(A) = 6$, he calculates its inverse by using Extended Euclidean Algorithm as given in Table 3.4

Q	A_1	A_2	A_3	B_1	B_2	B_1
-	1	0	35	0	1	6
5	0	1	6	1	-5	5
1	1	-5	5	-1	6	1

Table 3.4: Extended Euclidean Algorithm $(6)^{-1} \mod 35$

So, $(6)^{-1} \mod 35 = 6$ and multiply inverse of $\det(A)$ with $\mathrm{Adj}(A)$ to gets inverse i.e A^{-1} as:

$$A^{-1} = \begin{pmatrix} 0 & 8 \\ 8 & 0 \end{pmatrix} \mod 35$$

also,

$$A^{-1} r_1 A = \begin{pmatrix} 18 & 10 \\ 28 & 27 \end{pmatrix} \mod 35$$

$$\alpha = \begin{pmatrix} 14 & 0 \\ 14 & 21 \end{pmatrix} + \begin{pmatrix} 18 & 10 \\ 28 & 27 \end{pmatrix}$$

$$\alpha = \begin{pmatrix} 32 & 10 \\ 7 & 13 \end{pmatrix} \mod 35.$$

- Since $\alpha \;=\; \omega t_1$, therefore

$$S_1 \;=\; H((m)_2 \| (\omega t)_2) \;=\; H((m)_2 \| (\alpha)_2).$$

Hence $S_1 = S_1^{'}$.

Chapter 4

Digital Signature Based on Matrix Power Function

In this chapter, we will present and discuss an modified form of the digital signature scheme proposed by S. K. Rososhek [17]. For this purpose we aim to use general linear group of matrices over a finite field and matrix power function. The key generation algorithm, the digital signature generation algorithm and the signature verfication algorithm for the new improved digital signature scheme is described below. We implemented our scheme using the computer algebra system ApCoCoA [45]. These implementation are then used to construct various examples to illustrate our scheme.

4.1 Digital Signature Scheme Based on Matrix Power Function

In this section, we will describe the improved form of digital signature scheme that was explained in Chapter 3. For this purpose first we will choose matrices over $GL_m(\mathbb{Z}_n)$ as the platform group that employs computational difficulty of the multivariate quadratic (MQ) equations which forms due to use of matrix power function (MPF). Base matrices are to be taken from $GL_m(\mathbb{Z}_n)$ whereas power matrices are to be taken from the subgroup of $GL_m(\mathbb{Z}_n)$ of right circulant matrices. First recall the following definition of right circulant matrix.

Definition 4.1.1. A matrix α belongs to $M_{m\times m}(\mathbb{R})$ is said to be a right circulant matrix if:

$$\alpha = \begin{pmatrix} \alpha_0 & \alpha_1 & \alpha_2 & \cdots & \alpha_{n-2} & \alpha_{n-1} \\ \alpha_{n-1} & \alpha_0 & \alpha_1 & \cdots & \alpha_{n-3} & \alpha_{n-2} \\ & \vdots & \vdots & \vdots & & \\ \alpha_2 & \alpha_3 & \alpha_4 & \cdots & \alpha_0 & \alpha_1 \\ \alpha_1 & \alpha_2 & \alpha_3 & \cdots & \alpha_{n-1} & \alpha_0 \end{pmatrix} \tag{4.1}$$

In right circulant matrices, each row (column) is a cyclic shift of the first row (column) in a matrix. It is denoted by $(\alpha_0, \alpha_1, \alpha_2, \cdots \alpha_{n-1})$ [46].

Suppose Alice wants to send a signed document to Bob. She has to use matrices over $\mathbb{Z}_n$ of order m to sign the document. Later on Bob uses verification algorithm to verify the document.

Algorithm 4.1.2 (Key Generation).
Following steps are performed by Alice:

- Generate two random prime numbers p and q where $p \neq q$.
- Computes

$$n = pq$$

- Then choose two random matrices B, C in the group $GL_m(\mathbb{Z}_n)$.
- Alice compute the public key as:

$$A = BC \tag{4.2}$$

- Master public key of is (n, A) and secret key is (B, C).

Algorithm 4.1.3 (Signature Generation).
To sign the message m Alice perform the following steps:

- Choose a random matrix T in the $GL_m(\mathbb{Z}_n)$.
- Choose another random matrix U in the group $GL_m(\mathbb{Z}_n)$.
- Select random element δ in the $\mathbb{Z}_n$.
- Session private key of Alice is:

$$(\delta, T, U)$$

- Let χ_A, χ_{B^2CT}, χ_{BT} from the matrix ring $M_2(\mathbb{Z}_n)$ defined as the following:

$$\chi_A = U^A \mod n$$
$$\chi_{B^2CT} = U^{B^2CT} \mod n$$
$$\chi_{BT} = U^{BT} \mod n$$

- Further the matrix r_1 and bit string S_1 are computed as:

$$r_1 = U^{BT} \mod n \tag{4.3}$$
$$t_1 = U^{B^2CT} \mod n,\ \delta t_1 \tag{4.4}$$
$$\omega = \delta + 1 \tag{4.5}$$
$$S_1 = H((m)_2 \parallel (\omega t_1)_2). \tag{4.6}$$

where $(m)_2$ is the bit string binary representation of the message m, $(\omega t_1)_2$ is the bit string obtained after transferring matrix ωt_1 in the string of numbers as follows:

$$\omega t_1 = \begin{pmatrix} a_1^{'} & a_2^{'} \\ a_3^{'} & a_4^{'} \end{pmatrix} \to a_1^{'} \| a_2^{'} \| a_3^{'} \| a_4^{'}.$$

- Let δt_1 be a session public key (verification key) for the verification of Alice's signature of the message m and (r_1, S_1) is the Alice's signature of the message m.

Algorithm 4.1.4 (**Signature Verification**).

Alice send message m to Bob and for verification Bob performed following steps:

- Bob obtain master public key (n, A) and session public key δt_1 of Alice.
- Then he compute:

$$\alpha = \delta t_1 + (r_1)^A \tag{4.7}$$

- Then Bob compute:

$$S_1^{'} = H((m)_2 \parallel (\alpha)_2) \tag{4.8}$$

- Bob accept the signature on message m send by Alice if and only if $S_1 = S_1^{'}$.

Correctness

From Equation (4.7),

$$\alpha \ = \ \delta t_1 \ + \ (r_1)^A$$

Using Equation 4.3 and 4.2 we get,

$$\begin{aligned} &= \delta t_1 \ + \ (U^{BT})^A \\ &= \delta t_1 \ + \ (U^{BT})^{BC} \end{aligned}$$

From Equation 4.4 we get,

$$\begin{aligned} &= \delta t_1 \ + \ (U^{B^2CT}) \\ \alpha &= (\delta \ + \ 1) \ t_1 \ = \ \omega t_1. \end{aligned}$$

Therefore from (4.8),

$$\begin{aligned} S_1^{'} &= H((m)_2 \parallel (\alpha)_2) \\ S_1^{'} &= \ H((m)_2 \parallel (\omega t_1)_2) \\ S_1^{'} &= \ S. \end{aligned}$$

Remark 4.1.5. Using Euler's totient function 2.3.6, one can compute exponents efficiently in the above algorithms by reducing the power matrices by modulo $\phi(n)$.

4.2 Illustrative Examples

In this section we will explain the modified signature scheme based on matrix power function (MPF) through examples.

Example 4.2.1.

Let us take $GL_2(\mathbb{Z}_{77})$ for implementing the improved scheme. Here we take $p = 11$ and $q = 7$ for calculating mod n. All the calculation for power matrix are performed under mod $\phi(n)$ and calculation for base matrix are performed under mod (n).

Step 1: Key Generation

- As mentioned above, Alice selects random prime numbers *i.e* $p = 11$ and $q = 7$.
- Then she computes

$$n = pq$$

$$n = 77$$

Also, by using 2.3.6 we get,

$$\begin{aligned}\phi(n) &= \phi(77) \\ &= \phi(7 \times 11) = 60\end{aligned}$$

- Choose right circulant matrices:

$$B, C \in GL_2(\mathbb{Z}_n)$$

$$B = \begin{pmatrix} 49 & 30 \\ 30 & 49 \end{pmatrix}, C = \begin{pmatrix} 60 & 52 \\ 52 & 60 \end{pmatrix} \in GL_2(\mathbb{Z}_{77})$$

- Now we compute BC to get A. *i.e*

$$A = B\,C$$

$$A = \begin{pmatrix} 4500 & 4348 \\ 4348 & 4500 \end{pmatrix} \quad \bmod \phi(n)$$

$$A = \begin{pmatrix} 0 & 28 \\ 28 & 0 \end{pmatrix} \quad \bmod 60.$$

- Master pubic key of Alice is:

$$n = 77,\ A = \begin{pmatrix} 0 & 28 \\ 28 & 0 \end{pmatrix},$$

- Master private key of Alice is:

$$B = \begin{pmatrix} 49 & 30 \\ 30 & 49 \end{pmatrix},\ C = \begin{pmatrix} 60 & 52 \\ 52 & 60 \end{pmatrix}$$

Step 2: Signature Generation

For signature generation Alice should perform following steps:

- Choose the right circulant matrix T belongs to $GL_2(\mathbb{Z}_{77})$ and a random matrix $U \in GL_2(\mathbb{Z}_n)$:

  Let

$$T = \begin{pmatrix} 52 & 28 \\ 28 & 52 \end{pmatrix} \ \in\ GL_2(\mathbb{Z}_{77}),$$

$$U = \begin{pmatrix} 25 & 31 \\ 41 & 10 \end{pmatrix} \ \in\ GL_2(\mathbb{Z}_n).$$

- Select the residue:

$$\delta\ \in\ \mathbb{Z}_{77}$$

$$\delta = 41,$$

- Session private key of Alice is:

$$\delta = 41,\ T = \begin{pmatrix} 52 & 28 \\ 28 & 52 \end{pmatrix},\ U = \begin{pmatrix} 25 & 31 \\ 41 & 10 \end{pmatrix}.$$

- Alice computes:

$$\chi_A(U) = U^A \mod n$$

$$\chi_A(U) = \begin{pmatrix} 25 & 31 \\ 41 & 10 \end{pmatrix}^{\begin{pmatrix} 0 & 28 \\ 28 & 0 \end{pmatrix}}$$

$$= \begin{pmatrix} (25)^0.(31)^{28} & (25)^{28}.(31)^0 \\ (41)^0.(10)^{28} & (41)^{28}.(10)^0 \end{pmatrix}$$

$$\chi_A(U) = \begin{pmatrix} 25 & 60 \\ 67 & 71 \end{pmatrix} \mod 77$$

Similarly,

$$\chi_{BT}(U) = U^{BT} \mod n$$

First calculate BT and takes the power of U

$$BT = \begin{pmatrix} 3388 & 2932 \\ 2932 & 3388 \end{pmatrix} \mod \phi(n)$$

$$= \begin{pmatrix} 28 & 52 \\ 52 & 28 \end{pmatrix} \mod 60$$

$$\chi_{BT}(U) = U^{BT} = \begin{pmatrix} 25 & 31 \\ 41 & 10 \end{pmatrix}^{\begin{pmatrix} 28 & 52 \\ 52 & 28 \end{pmatrix}} \mod 77$$

$$\chi_{BT}(U) = \begin{pmatrix} 9 & 16 \\ 60 & 53 \end{pmatrix} \mod 77$$

Similarly, compute

$$\chi_{B^2CT}(U) = U^{B^2CT} \mod n$$

First we compute B^2CT which is given as:

$$B^2CT = \begin{pmatrix} 27994336 & 27925024 \\ 27925024 & 27994336 \end{pmatrix} \mod \phi(n)$$

$$= \begin{pmatrix} 16 & 14 \\ 4 & 16 \end{pmatrix} \mod 60$$

Now, compute U^{B^2CT} using MPF

$$\chi_{B^2CT}(U) = \begin{pmatrix} 25 & 31 \\ 41 & 10 \end{pmatrix}^{\begin{pmatrix} 16 & 14 \\ 4 & 16 \end{pmatrix}} \mod 77$$

We get,

$$\chi_{B^2CT}(U) = \begin{pmatrix} 37 & 58 \\ 25 & 4 \end{pmatrix}$$

- Alice computes signature (r_1, S_1):

$$r_1 = \chi_{BT}(U)$$

$$r_1 = \begin{pmatrix} 9 & 16 \\ 60 & 53 \end{pmatrix}$$

$$t_1 = \chi_{B^2CT}(U)$$

$$t_1 = \begin{pmatrix} 37 & 58 \\ 25 & 4 \end{pmatrix}$$

$$\delta t_1 = 41 \begin{pmatrix} 37 & 58 \\ 25 & 4 \end{pmatrix} = \begin{pmatrix} 54 & 68 \\ 24 & 10 \end{pmatrix},$$

- Public session key of Alice is:

$$\delta t_1 = \begin{pmatrix} 54 & 68 \\ 24 & 10 \end{pmatrix}$$

$$\omega = \delta + 1 \ = 42$$

$$\omega t_1 = 42 \begin{pmatrix} 37 & 58 \\ 25 & 4 \end{pmatrix} = \begin{pmatrix} 14 & 49 \\ 49 & 14 \end{pmatrix},$$

$$S_1 = H((m)_2 \| (\omega t_1)_2)$$

where $(m)_2$ is a bit string binary representation of the message m, $(\omega t_1)_2$ is a bit string obtained after transferring matrix δt_1 in the string of numbers and replace the numbers by their binary representations as follows:

$$14 \parallel 49 \parallel 49 \parallel 14 \ \rightarrow \ 00001110 \parallel 00110001 \parallel 00110001 \parallel 00001110.$$

Step 3: Signature Verification

Bob should perform following steps:

- Bob obtain the authentic master public key of Alice.

$$n = 77, A = \begin{pmatrix} 0 & 28 \\ 28 & 0 \end{pmatrix}$$

and session public key $\delta t_1 \; = \begin{pmatrix} 54 & 68 \\ 24 & 10 \end{pmatrix}$

- For verification Bob computes:

$$\alpha = \delta t_1 + \chi_A(r_1)$$

$$\alpha = \delta t_1 + (r_1)^A$$

as,

$$(r_1)^A = \begin{pmatrix} 37 & 58 \\ 25 & 4 \end{pmatrix}$$

$$\alpha = \begin{pmatrix} 54 & 68 \\ 24 & 10 \end{pmatrix} + \begin{pmatrix} 37 & 58 \\ 25 & 4 \end{pmatrix}$$

$$\alpha = \begin{pmatrix} 14 & 49 \\ 49 & 14 \end{pmatrix} \quad \bmod 77$$

- Since $\alpha = \omega t_1$, therefore

$$S_1 = H((m)_2 \parallel (\omega t)_2) = H((m)_2 \parallel (\alpha)_2) = S_1^{'}.$$

Example 4.2.2.

Let us take $GL_3(\mathbb{Z}_{85})$ for implementing the modified scheme. Here we take $p = 17$ and $q = 5$ for calculating mod n. All the calculation for power matrix are performed under mod $\phi(n)$ and calculation for base matrix are performed under mod n.

Step 1: Key Generation

Alice performs the following steps:

- Chooses two random prime numbers $p = 17$ and $q = 5$, then calculates $n = pq = 85$.

- Computes $\phi(n)$ as:

$$\begin{aligned}\phi(n) &= \phi(85)\\ &= \phi(17 \times 5)\\ &= \phi(17)\phi(5)\\ &= (17-1)(5-1) = 64\end{aligned}$$

- Chooses right circulant matrices:

$$B, C \in GL_3(\mathbb{Z}_n)$$

$$B = \begin{pmatrix} 21 & 16 & 32 \\ 32 & 21 & 16 \\ 16 & 32 & 21 \end{pmatrix}, C = \begin{pmatrix} 42 & 19 & 50 \\ 50 & 42 & 19 \\ 19 & 50 & 42 \end{pmatrix} \in GL_3(\mathbb{Z}_{85})$$

- Now computes BC to get A.

$$\begin{aligned} A &= B\,C \\ A &= \begin{pmatrix} 2290 & 2671 & 2698 \\ 2698 & 2290 & 2671 \\ 2671 & 2698 & 2290 \end{pmatrix} \quad \mod \phi(n)\end{aligned}$$

$$A = \begin{pmatrix} 50 & 47 & 10 \\ 10 & 50 & 47 \\ 47 & 10 & 50 \end{pmatrix} \quad \mod 64.$$

- Master pubic key is:

$$n = 85,\ A = \begin{pmatrix} 50 & 47 & 10 \\ 10 & 50 & 47 \\ 47 & 10 & 50 \end{pmatrix},$$

- Master private key is:

$$B = \begin{pmatrix} 21 & 16 & 32 \\ 32 & 21 & 16 \\ 16 & 32 & 21 \end{pmatrix}, \ C = \begin{pmatrix} 42 & 19 & 50 \\ 50 & 42 & 19 \\ 19 & 50 & 42 \end{pmatrix}$$

Step 1: Signature Generation

Alice performs the following steps:

- Chooses the right circulant matrix $T \in GL_3(\mathbb{Z}_{85})$ and a random matrix $U \in GL_3(\mathbb{Z}_{85})$:

Let

$$T = \begin{pmatrix} 13 & 25 & 61 \\ 61 & 13 & 25 \\ 25 & 61 & 13 \end{pmatrix} \ \in \ GL_3(\mathbb{Z}_{85}),$$

$$U = \begin{pmatrix} 15 & 29 & 37 \\ 33 & 45 & 59 \\ 11 & 42 & 28 \end{pmatrix} \ \in \ GL_3(\mathbb{Z}_{85}).$$

- Selects the random element:

$$\delta \ \in \ \mathbb{Z}_{85}$$
$$\delta = 41,$$

- Session private key is:

$$\delta = 41, \ T = \begin{pmatrix} 13 & 25 & 61 \\ 61 & 13 & 25 \\ 25 & 61 & 13 \end{pmatrix}, \ U = \begin{pmatrix} 15 & 29 & 37 \\ 33 & 45 & 59 \\ 11 & 42 & 28 \end{pmatrix}.$$

- Computes

$$\chi_{BT}(U) = U^{BT} \quad \mod n$$

First calculates BT under mod $\phi(n)$.

$$BT = \begin{pmatrix} 1 & 61 & 49 \\ 49 & 1 & 61 \\ 61 & 49 & 1 \end{pmatrix} \mod 64$$

$$\chi_{BT}(U) = U^{BT} = \begin{pmatrix} 15 & 29 & 37 \\ 33 & 45 & 59 \\ 11 & 42 & 28 \end{pmatrix}^{\begin{pmatrix} 1 & 61 & 49 \\ 49 & 1 & 61 \\ 61 & 49 & 1 \end{pmatrix}} \mod 85$$

$$= \begin{pmatrix} (15)^1.(29)^{49}.(27)^{61} & (15)^{61}.(29)^1.(27)^{49} & (15)^{49}.(29)^{61}.(27)^1 \\ (33)^1.(45)^{49}.(59)^{61} & (33)^{61}.(45)^1.(59)^{49} & (33)^{49}.(45)^{61}.(59)^1 \\ (11)^1.(42)^{49}.(28)^{61} & (11)^{61}.(42)^1.(28)^{49} & (11)^{49}.(42)^{61}.(28)^1 \end{pmatrix}$$

$$\chi_{BT}(U) = \begin{pmatrix} 35 & 55 & 35 \\ 20 & 65 & 80 \\ 21 & 21 & 1 \end{pmatrix} \mod 85$$

We implemented code in ApCoCoA which calculates MPF.

- Similarly, computes

$$\chi_{B^2CT}(U) = U^{B^2CT} \mod n$$

$$B^2CT = \begin{pmatrix} 19 & 3 & 15 \\ 15 & 19 & 3 \\ 3 & 15 & 19 \end{pmatrix} \mod 64$$

- Now, computes U^{B^2CT} using ApCoCoA

$$\chi_{B^2CT}(U) = \begin{pmatrix} 15 & 29 & 37 \\ 33 & 45 & 59 \\ 11 & 42 & 28 \end{pmatrix}^{\begin{pmatrix} 19 & 3 & 15 \\ 15 & 19 & 3 \\ 3 & 15 & 19 \end{pmatrix}} \mod 85$$

Alice gets,

$$\chi_{B^2CT}(U) = \begin{pmatrix} 50 & 50 & 30 \\ 40 & 10 & 75 \\ 1 & 21 & 21 \end{pmatrix}$$

- Computes signature (r_1, s_1):

$$r_1 = \chi_{BT}(U)$$

$$r_1 = \begin{pmatrix} 35 & 55 & 35 \\ 20 & 65 & 80 \\ 21 & 21 & 1 \end{pmatrix}$$

$$t_1 = \chi_{B^2CT}(U)$$

$$t_1 = \begin{pmatrix} 50 & 50 & 30 \\ 40 & 10 & 75 \\ 1 & 21 & 21 \end{pmatrix}$$

$$\delta t_1 = 41 \begin{pmatrix} 50 & 50 & 30 \\ 40 & 10 & 75 \\ 1 & 21 & 21 \end{pmatrix} = \begin{pmatrix} 10 & 10 & 40 \\ 25 & 70 & 15 \\ 41 & 11 & 11 \end{pmatrix},$$

- Public session key is:

$$\delta t_1 = \begin{pmatrix} 10 & 10 & 40 \\ 25 & 70 & 15 \\ 41 & 11 & 11 \end{pmatrix}$$

$$\omega = \delta + 1 \ = \ 42$$

$$\omega t_1 = 42 \begin{pmatrix} 50 & 50 & 30 \\ 40 & 10 & 75 \\ 1 & 21 & 21 \end{pmatrix} = \begin{pmatrix} 60 & 60 & 70 \\ 65 & 80 & 5 \\ 42 & 32 & 32 \end{pmatrix},$$

$$s_1 = H((m)_2 \| (\omega t_1)_2)$$

Step 1: Verification

Bob should perform the following steps:

- Obtain the authentic master public key of Alice.

$$n = 65,\ A = \begin{pmatrix} 50 & 47 & 10 \\ 10 & 50 & 47 \\ 47 & 10 & 50 \end{pmatrix}$$

and session public key $\delta t_1 \ = \begin{pmatrix} 10 & 10 & 40 \\ 25 & 70 & 15 \\ 41 & 11 & 11 \end{pmatrix}$

- Computes following for verification:

$$\alpha = \delta t_1 + \chi_A(r_1)$$

$$\alpha = \delta t_1 + (r_1)^A$$

as,

$$(r_1)^A = \begin{pmatrix} 50 & 50 & 30 \\ 10 & 10 & 75 \\ 1 & 21 & 21 \end{pmatrix}$$

$$\alpha = \begin{pmatrix} 10 & 10 & 40 \\ 25 & 70 & 15 \\ 41 & 11 & 11 \end{pmatrix} + \begin{pmatrix} 50 & 50 & 30 \\ 10 & 10 & 75 \\ 1 & 21 & 21 \end{pmatrix} \quad \text{mod } 85$$

$$\alpha = \begin{pmatrix} 60 & 60 & 70 \\ 65 & 80 & 5 \\ 42 & 32 & 32 \end{pmatrix} \quad \text{mod } \ 85.$$

- Since $\alpha = \omega t_1$, therefore

$$S_1 = H((m)_2 \parallel (\omega t)_2) = H((m)_2 \parallel (\alpha)_2) = S_1^{'}.$$

Example 4.2.3.

Let us take $GL_3(\mathbb{Z}_{65})$ for implementing the improved scheme. Here we take $p = 13$ and $q = 5$ for calculating mod n.

Step 1: Key Generation

- As mentioned above, Alice selects random prime numbers *i.e* $p = 13$ and $q = 5$.

- Then she computes

$$n = pq$$
$$n = 65$$

  Also, by using 2.3.6, we get

$$\begin{aligned}\phi(n) &= \phi(65) \\ &= \phi(13 \times 5) = 48\end{aligned}$$

- Choose commutative matrices

$$B, C \in GL_3(\mathbb{Z}_n)$$

$$B = \begin{pmatrix} 21 & 16 & 32 \\ 32 & 21 & 16 \\ 16 & 32 & 21 \end{pmatrix}, C = \begin{pmatrix} 42 & 19 & 50 \\ 50 & 42 & 19 \\ 19 & 50 & 42 \end{pmatrix} \in GL_3(\mathbb{Z}_{65})$$

- Now we compute BC to get A *i.e*

$$A = B\,C$$

$$A = \begin{pmatrix} 2290 & 2671 & 2698 \\ 2698 & 2290 & 2671 \\ 2671 & 2698 & 2290 \end{pmatrix} \mod \phi(n)$$

$$A = \begin{pmatrix} 34 & 31 & 10 \\ 10 & 34 & 31 \\ 31 & 10 & 34 \end{pmatrix} \mod 48$$

- Master pubic key of Alice is:

$$n = 65,\ A = \begin{pmatrix} 34 & 31 & 10 \\ 10 & 34 & 31 \\ 31 & 10 & 34 \end{pmatrix}$$

- Master private key of Alice is:

$$B = \begin{pmatrix} 21 & 16 & 32 \\ 32 & 21 & 16 \\ 16 & 32 & 21 \end{pmatrix},\ C = \begin{pmatrix} 42 & 19 & 50 \\ 50 & 42 & 19 \\ 19 & 50 & 42 \end{pmatrix}$$

Step 2: Signature Generation

For signature generation Alice should perform following steps:

- Choose the commutative matrix T belongs to $GL_2(\mathbb{Z}_{65})$ and a random matrix $U \in GL_2(\mathbb{Z}_n)$:

  Let,

$$T = \begin{pmatrix} 13 & 25 & 61 \\ 61 & 13 & 25 \\ 25 & 61 & 13 \end{pmatrix} \in GL_3(\mathbb{Z}_{65}),$$

$$U = \begin{pmatrix} 15 & 29 & 37 \\ 33 & 45 & 59 \\ 11 & 42 & 28 \end{pmatrix} \in GL_3(\mathbb{Z}_n).$$

- Select the random element:

$$\delta \in \mathbb{Z}_{65}$$
$$\delta = 35,$$

- Session private key of Alice is:

$$\delta = 35,\ T = \begin{pmatrix} 13 & 25 & 61 \\ 61 & 13 & 25 \\ 25 & 61 & 13 \end{pmatrix},\ U = \begin{pmatrix} 15 & 29 & 37 \\ 33 & 45 & 59 \\ 11 & 42 & 28 \end{pmatrix}.$$

- Alice computes:

$$\chi_{BT}(U) = U^{BT} \mod n$$

First calculate BT and takes the power of U

$$BT = \begin{pmatrix} 2049 & 2685 & 2097 \\ 2097 & 2049 & 2685 \\ 2685 & 2097 & 2049 \end{pmatrix} \mod \phi(n)$$
$$= \begin{pmatrix} 33 & 45 & 33 \\ 33 & 33 & 45 \\ 45 & 33 & 33 \end{pmatrix} \mod 48$$

$$\chi_{BT}(U) = U^{BT} = \begin{pmatrix} 15 & 29 & 37 \\ 33 & 45 & 59 \\ 11 & 42 & 28 \end{pmatrix}^{\begin{pmatrix} 33 & 45 & 33 \\ 33 & 33 & 45 \\ 45 & 33 & 33 \end{pmatrix}}$$

Now, using MPF Alice gets,

$$\chi_{BT}(U) = \begin{pmatrix} 40 & 40 & 40 \\ 60 & 60 & 60 \\ 1 & 1 & 1 \end{pmatrix} \mod 65$$

Similarly, compute

$$\chi_{B^2CT}(U) = U^{B^2CT} \mod n$$

First we compute B^2CT which is given as:

$$B^2CT = \begin{pmatrix} 17537427 & 17279235 & 17501967 \\ 17501967 & 17537427 & 17279235 \\ 17279235 & 17501967 & 17537427 \end{pmatrix} \mod \phi(n)$$

$$= \begin{pmatrix} 3 & 3 & 15 \\ 15 & 3 & 3 \\ 3 & 15 & 3 \end{pmatrix} \mod 48$$

Now, compute U^{B^2CT} using MPF.

$$\chi_{B^2CT}(U) = \begin{pmatrix} 15 & 29 & 37 \\ 33 & 45 & 59 \\ 11 & 42 & 28 \end{pmatrix}^{\begin{pmatrix} 3 & 3 & 15 \\ 15 & 3 & 3 \\ 3 & 15 & 3 \end{pmatrix}} \mod 65$$

Alice gets,

$$\chi_{B^2CT}(U) = \begin{pmatrix} 40 & 40 & 40 \\ 5 & 5 & 5 \\ 1 & 1 & 1 \end{pmatrix}$$

- Alice computes signature (r_1, S_1) as:

$$r_1 = \chi_{BT}(U)$$

$$r_1 = \begin{pmatrix} 40 & 40 & 40 \\ 60 & 60 & 60 \\ 1 & 1 & 1 \end{pmatrix}$$

$$t_1 = \chi_{B^2CT}(U)$$

$$t_1 = \begin{pmatrix} 40 & 40 & 40 \\ 5 & 5 & 5 \\ 1 & 1 & 1 \end{pmatrix}$$

$$\delta t_1 = 35 \begin{pmatrix} 40 & 40 & 40 \\ 5 & 5 & 5 \\ 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 40 & 40 & 40 \\ 5 & 5 & 5 \\ 1 & 1 & 1 \end{pmatrix},$$

- Public session key of Alice is:

$$\delta t_1 = \begin{pmatrix} 35 & 35 & 35 \\ 45 & 45 & 45 \\ 35 & 35 & 35 \end{pmatrix}$$

$$\omega = \delta + 1 \ = \ 36$$

$$\omega t_1 = 36 \begin{pmatrix} 40 & 40 & 40 \\ 5 & 5 & 5 \\ 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 10 & 10 & 10 \\ 50 & 50 & 50 \\ 36 & 36 & 36 \end{pmatrix},$$

$$S_1 = H((m)_2 \| (\omega t_1)_2),$$

where $(m)_2$ is a bit string binary representation of the message m, $(\omega t_1)_2$ is a bit string obtained after transferring matrix δt_1 in the string of numbers and replace the numbers by their binary representations as follows:

$10 \parallel 10 \parallel 10 \parallel 50 \parallel 50 \parallel 50 \parallel 36 \parallel 36 \parallel 36 \ \rightarrow \ 00001010 \parallel 00001010 \parallel 00001010 \parallel$ $00110010 \parallel 00110010 \parallel 00110010 \parallel 00100100 \parallel 00100100 \parallel 00100100.$

Step 3: Signature Verification

Bob should perform following steps:

- Bob obtain the authentic master public key of Alice.

$$n = 65,\ A = \begin{pmatrix} 34 & 31 & 10 \\ 10 & 34 & 31 \\ 31 & 10 & 34 \end{pmatrix}$$

and session public key $\delta t_1 \ = \begin{pmatrix} 35 & 35 & 35 \\ 45 & 45 & 45 \\ 35 & 35 & 35 \end{pmatrix}$

- For verification Bob computes:

$$\alpha = \ \delta t_1 + \chi_A(r_1)$$

$$\alpha = \ \delta t_1 + (r_1)^A$$

as,

$$(r_1)^A = \begin{pmatrix} 40 & 40 & 40 \\ 5 & 5 & 5 \\ 1 & 1 & 1 \end{pmatrix}$$

$$\alpha = \begin{pmatrix} 35 & 35 & 35 \\ 45 & 45 & 45 \\ 35 & 35 & 35 \end{pmatrix} + \begin{pmatrix} 40 & 40 & 40 \\ 5 & 5 & 5 \\ 1 & 1 & 1 \end{pmatrix}$$

$$\alpha = \begin{pmatrix} 10 & 10 & 10 \\ 50 & 50 & 50 \\ 36 & 36 & 36 \end{pmatrix} \mod 65.$$

- Since $\alpha = \omega t_1$, therefore

$$S_1 = H((m)_2 \parallel (\omega t)_2) = H((m)_2 \parallel (\alpha)_2) = S_1^{'}.$$

4.3 Security Analysis

In $GL_m(\mathbb{Z}_n)$, for a large key space n should be kept very large. Use of matrix power function MPF increases the security of scheme. Due to large key space and matrices of large order it is difficult to solve matrix decomposition problem which is the underlying hard problem of our scheme. The security of proposed scheme relies on the complexity of solution of matrix MQ problem. Hence the security of our scheme is increased.

4.3.1 Key-Recovery Attack

In this attack an adversary knows only the Alice master public key $A = BC$ and aim of hacker is to find the Alice's private key B, C. Critical problem is the Matrix modular factorization problem. B and C are unknown matrices from the group $GL_m(\mathbb{Z}_n)$.

Finding the unknown power matrices B and C is known as matrix decomposition problem. By letting matrices B and C as:

$$B = \begin{pmatrix} b_{11} & b_{12} \\ b_{12} & b_{11} \end{pmatrix} \text{ and } C = \begin{pmatrix} c_{11} & c_{12} \\ c_{12} & c_{11} \end{pmatrix}$$

In $BC = A$, the matrix $A = \begin{pmatrix} a_{11} & a_{12} \\ a_{12} & a_{11} \end{pmatrix}$ is known and we get the following equations.

$$b_{11}c_{11} + b_{12}c_{12} = a_{11} \tag{4.9}$$

$$b_{11}c_{12} + b_{12}c_{11} = a_{12} \tag{4.10}$$

There are two equations and four unknowns which means there are infinitely many solutions. Therefore, finding Private keys B and C is clear from Equation (4.9) and Equation (4.10) is not possible. We have matrix decomposition problem $BC = A$. A is the public key of Alice which is equal to $\begin{pmatrix} 2 & 9 \\ 9 & 2 \end{pmatrix}$.

Now let

$$A = \begin{pmatrix} b_{11} & b_{12} \\ b_{12} & b_{11} \end{pmatrix} \begin{pmatrix} c_{11} & c_{12} \\ c_{12} & c_{11} \end{pmatrix} \tag{4.11}$$

We get,

$$b_{11}c_{11} + b_{12}c_{12} = 2 \tag{4.12}$$

$$b_{11}c_{12} + b_{12}c_{11} = 9 \tag{4.13}$$

$$b_{12}c_{11} + b_{11}c_{12} = 9 \tag{4.14}$$

$$b_{12}c_{12} + b_{11}c_{11} = 2 \tag{4.15}$$

By solving (4.12), (4.13), (4.14) and (4.15) we get following equations.

$$b_{11}c_{11} + b_{12}c_{12} = 2 \tag{4.16}$$

$$b_{11}c_{12} + b_{12}c_{11} = 9 \tag{4.17}$$

From Equations (4.16) and (4.17) it is clear that, there are two equations and four unknown this means there are infinitely many solutions. So finding B and C is in-feasible.

4.3.2 Forgery Attack

If attacker has got access to intended key he/she that is, he has got access of private keys B and C but still he will not able to recover signature. Attacker will have to solve the following equations to generate a signature.

$$r_1 = U^{BT} \mod n \tag{4.18}$$

$$t_1 = U^{B^2CT} \mod n \tag{4.19}$$

$$\omega = \delta + 1 \tag{4.20}$$

$$S_1 = H((m)_2 \parallel (\omega t_1))_2 \tag{4.21}$$

Here $(m)_2$ is binary representation of message and $(\omega t_1)_2$ is binary representation of ωt_1.

Therefore, it is infeasible to solve above equations without knowledge of T, U and δ. In (4.19) the matrices T and U are unknown matrices. In (4.20) δ is unknown, without these parameters attacker will not be able to recover t_1 and ω. To forge the signature of Alice, attacker select $r_1 = U^{B'T'}$. He computes $r_1 = U^{B'T'}$ from $\alpha = \delta t_1 + (r_1)^A$. For this purpose he needs δt_1 which is public key of Alice and multiplier of matrix δt_1 can not be known to attacker. Without knowledge of δ and t_1 he will limit himself to random selection of U and T but for a large key space it is infeasible.

For instance, let the Equation (4.19) is the form $X = Y^Z$. From $X = Y^Z$ we get,

$$X = \begin{pmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{pmatrix}^{\begin{pmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{pmatrix}} \tag{4.22}$$

By solving (4.22) using MPF we get the system of equations.

$$y_{11}^{z_{11}} \cdot y_{12}^{z_{12}} = x_{11} \tag{4.23}$$

$$y_{11}^{z_{12}} \cdot y_{12}^{z_{11}} = x_{12} \tag{4.24}$$

$$y_{21}^{z_{11}} \cdot y_{22}^{z_{12}} = x_{21} \tag{4.25}$$

$$y_{22}^{z_{12}} \cdot y_{22}^{z_{11}} = x_{22} \tag{4.26}$$

(4.23), (4.24), (4.25) and (4.26) forms the system of multivariate quadratic (MQ) equations which are also called the matrix MQ (MMQ) problem. Cryptanalysis of proposed scheme is based on the solution of matrix multivariate quadratic (MQ) system of equations which is NP-complete. So, finding correct soluctions of (4.23), (4.24), (4.25) and (4.26) one has to solve the above system of equations which is not possible. Hence it is infeasible to generate signature S_1 given in (4.21).

4.3.3 Brute Force Attack

In Brute force attack, the attacker tries every possible key to get success. Attacker tries to obtain the key from given data.
For large values of p and q, m will be very large. For instance, if p, q are of the size of 64-bits. For every entry of the matrix there are 2^{64} possibilities. If we fix $m = 2$ then for the matrix of order 2, we get approximate $(2^{64})^4 = 2^{256}$ which is very large space and hence brute-force attack is not feasible.

4.4 Conclusion

In this thesis, we review the article " Fast And Secure Modular Matrix Based Digital Signature" proposed by S. K. Rososhek [17]. This scheme is based on matrices over finite field of integer $\mathbb{Z}_n$ and conjugacy search problem. We modified this scheme by using matrix power function (MPF) [39] which increases the security. In fact, an attacker has to solve multivariate quadratic (MQ) equations and matrix decomposition problem for matrices that is, $C = AB$, it is hard to find A and B from the knowledge of just C. For the implementation, we create computer programs for computation of MPF using Computer Algebra system ApCoCoA [20]. We give examples of proposed scheme and compute digital signature over a finite field using MPF. The overall security of the scheme is increased by using MPF. We give security analysis of our scheme. One can extend our work by checking the possibilities of extended Galois fields of the form $GF(p^q)$.

Appendix A

Elements of Finite Field

A.1 ApCoCoA Code for Calculation of Elements of Finite Field

This section contain the ApCoCoA code for calculation of elements of finite field $GL_2(Z_n)$.

RMPFMod(A,B,M calculate the right matrix power function over finite field. It require the input A, B and M where A and B are the matrices from $GL_2(Z_n)$ and M is modulo.

```
Define RMPFMod(A,B,M)
        Prod:=1;
        Rows:=NumRows(A);Cols:=NumCols(A);
        C:=NewMat(Rows,Cols,1);

        For K:=1 To Rows Do
                For J:=1 To Rows Do
                        For I:=1 To Rows Do
                                Prod:=Mod(Prod*A[K][I]^B[I][J],M);
                        EndFor;
                        C[K][J]:=Prod;
                        Prod:=1;
                EndFor;
        EndFor;
```

```
        Return C;
EndDefine;
```

ReducedMat(A,M) gives the matrix A that is reduced on some integer mod M.

```
Define ReducedMat(A,M)
Num:=NumRows(A);
For I:=1 To Num Do
        For J:=1 To Num Do
                A[I][J]:=Mod(A[I][J],M);
        EndFor;
EndFor;
Return A;
EndDefine;
```

ModInv calculate the inverse of a number under the mod . It require the input Q, M where Q is number and M is mod . This function uses the extended euclidean inverse algorithm.

```
Define ModInv(Q,M);
A1:=1;A2:=0;A3:=M;
B1:=0;B2:=1;B3:=Q;
While B3<0 Do
B3:=B3+M;
EndWhile;
While B3<>1 Do
Q:=Div(A3,B3);
--If Q=0 Then Error(" Q is 0");EndIf;
T1:=A1-Q*B1;T2:=A2-Q*B2;T3:=A3-Q*B3;
A1:=B1;A2:=B2;A3:=B3;
B1:=T1;B2:=T2;B3:=T3;
If B2<0 Then B2:=B2+M; EndIf;
If B3=1  Then Return B2;EndIf;
If B3=0 Then Return("Not Invertible!"); EndIf;
EndWhile;
--If B2<0 Then B2:=B2+M; EndIf;
Return B2;
EndDefine;
```

Bibliography

[1] W. Stallings, *Cryptography and network security: principles and practices.* Pearson Education India, 2006, vol. 7, no. 1.

[2] C. Cipher and M. Cipher, "Introduction to cryptography," *EEC*, vol. 484, no. 3-4, pp. 1–584, 2004.

[3] M. S. Iqbal, S. Singh, and A. Jaiswal, "Symmetric key cryptography: Technological developments in the field," *International Journal of Computer Applications*, vol. 117, no. 15, pp. 23–26, 2015.

[4] W. G. Barker, *Introduction to the analysis of the Data Encryption Standard (DES).* Aegean Park Press, 1991, vol. 55, no. 1.

[5] M. A. Musa, E. F. Schaefer, and S. Wedig, "A simplified aes algorithm and its linear and differential cryptanalyses," *Cryptologia*, vol. 27, no. 2, pp. 148–177, 2003.

[6] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[7] R. Singh and S. Kumar, "Elgamal's algorithm in cryptography," *International Journal of Scientific and Engineering Research*, vol. 3, no. 12, pp. 1–4, 2012.

[8] D. Hankerson, A. J. Menezes, and S. Vanstone, *Guide to elliptic curve cryptography.* Springer Science & Business Media, 2006, vol. 1, no. 1.

[9] J. Katz, A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography.* CRC press, 1996, vol. 2, no. 1.

[10] W. Stallings, *Cryptography and Network Security,*. Pearson Education India, 2006, vol. 4, no. 1.

[11] S. Goldwasser, S. Micali, and R. L. Rivest, "A digital signature scheme secure against adaptive chosen-message attacks," *SIAM Journal on Computing*, vol. 17, no. 2, pp. 281–308, 1988.

[12] O. Goldreich, "Two remarks concerning the goldwasser-micali-rivest signature scheme," in *Conference on the Theory and Application of Cryptographic Techniques*, vol. 263, no. 1. Springer, 1986, pp. 104–110.

[13] R. Gennaro, S. Halevi, and T. Rabin, "Secure hash-and-sign signatures without the random oracle," in *International Conference on the Theory and Applications of Cryptographic Techniques*, vol. 1592, no. 1. Springer, 1999, pp. 123–139.

[14] R. Cramer and V. Shoup, "Signature schemes based on the strong rsa assumption," *ACM Transactions on Information and System Security (TISSEC)*, vol. 3, no. 3, pp. 161–185, 2000.

[15] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[16] P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Proceedings 35th Annual Symposium on Foundations of Computer Science*, vol. 1, no. 1. IEEE, 1994, pp. 124–134.

[17] S. Rososhek, "Fast and secure modular matrix based digital signature," *British Journal of Mathematics and Computer Science*, vol. 13, no. 1, pp. 20–40, 2016.

[18] V. Shpilrain and A. Ushakov, "The conjugacy search problem in public key cryptography: unnecessary and insufficient," *Applicable Algebra in Engineering, Communication and Computing*, vol. 17, no. 3-4, pp. 285–289, 2006.

[19] E. Sakalauskas, "Enhanced matrix power function for cryptographic primitive construction," *Symmetry*, vol. 10, no. 2, pp. 1–43, 2018.

[20] A. Team, "Apcocoa: Applied computations in commutative algebra:."

[21] D. R. Stinson, *Cryptography: theory and practice.* CRC press, 2005, vol. 3, no. 1.

[22] M. Naor and M. Yung, "Universal one-way hash functions and their cryptographic applications," in *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, vol. 89, no. 1. ACM, 1989, pp. 33–43.

[23] S. M. Kevin, "The discrete logarithm problem," *Cryptology and computational number theory*, vol. 42, no. 1, pp. 49–74, 1990.

[24] A. Odlyzko, "Discrete logarithms: The past and the future," in *Towards a Quarter-Century of Public Key Cryptography.* Springer, 2000, vol. 19, no. 2, pp. 56–75.

[25] A. K. Lenstra, "Integer factoring," in *Towards a quarter-century of public key cryptography.* Springer, 2000, vol. 19, no. 1, pp. 101–128.

[26] J. J. Rotman, *A first course in abstract algebra.* Pearson College Division, 2000, vol. 7, no. 1.

[27] M. Aschbacher, *Finite group theory.* Cambridge University Press, 2000, vol. 10, no. 1.

[28] E. Hille and R. S. Phillips, *Functional analysis and semi-groups.* American Mathematical Soc., 1996, vol. 31, no. 1.

[29] P. M. Cohn, *Basic algebra: groups, rings and fields.* Springer Science & Business Media, 2012, vol. 1, no. 1.

[30] J. B. Fraleigh, *A first course in abstract algebra.* Pearson Education India, 2003, vol. 7, no. 1.

[31] C. J. Benvenuto, "Galois field in cryptography," *University of Washington*, vol. 1, no. 1, pp. 1–11, 2012.

[32] J. Kerl, "Computation in finite fields," *Arizona State University and Lockheed Martin Corporation*, vol. 1, no. 1, pp. 1–84, 2004.

[33] B. Preneel, "Cryptographic hash functions," *European Transactions on Telecommunications*, vol. 5, no. 4, pp. 431–448, 1994.

[34] H. Krawczyk, M. Bellare, and R. Canetti, "Hmac: Keyed-hashing for message authentication," Tech. Rep. 1, 1997.

[35] G. M. Wolrich, K. S. Yap, J. D. Guilford, V. Gopal, and S. M. Gulley, "Instruction set for message scheduling of sha256 algorithm," Sep. 16 2014, uS Patent 8,838,997 B2.

[36] S. Gueron, S. Johnson, and J. Walker, "Sha-512/256," in *Information Technology: New Generations (ITNG), 2011 Eighth International Conference on*, vol. 2, no. 3. IEEE, 2011, pp. 354–358.

[37] R. L. Rivest, B. Agre, D. V. Bailey, C. Crutchfield, Y. Dodis, K. E. Fleming, A. Khan, J. Krishnamurthy, Y. Lin, L. Reyzin *et al.*, "The md6 hash function–a proposal to nist for sha-3," *Submission to NIST*, vol. 2, no. 3, pp. 1–234, 2008.

[38] E. Sakalauskas and K. Luksys, "Matrix power function and its application to block cipher s-box construction," *Int. J. Inn. Comp., Inf. Contr*, vol. 8, no. 4, pp. 2655–2664, 2012.

[39] E. Sakalauskas and K. Luksys, "Matrix power cipher," *Information Technology and Control*, vol. 41, no. 4, pp. 349–355, 2012.

[40] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE transactions on information theory*, vol. 31, no. 4, pp. 469–472, 1985.

[41] B. Kaliski, *RSA Digital Signature Scheme.* Springer US, 2011, vol. 2, no. 1, pp. 1061–1064.

[42] K. H. Ko, D.-H. Choi, M. S. Cho, and J.-W. Lee, "New signature scheme using conjugacy problem." *IACR Cryptology ePrint Archive*, vol. 2002, no. 168, pp. 1–13, 2002.

[43] R. Zippel, "Residue rings," in *Effective Polynomial Computation*. Springer, 1993, vol. 1, no. 1, pp. 85–106.

[44] P. Krylov and A. Tuganbaev, "Modules over formal matrix rings," *Journal of Mathematical Sciences*, vol. 171, no. 2, pp. 248–295, 2010.

[45] A. I. Moldenhauer, G. Rosenberger, and K. Rosenthal, "On the Tits alternative for a class of finitely presented groups with a special focus on symbolic computations," *Algebra and Computer Science*, vol. 677, p. 145, 2016.

[46] A. C. F. Bueno, "Right circulant matrices with geometric progression," *International Journal of Applied Mathematical Research*, vol. 1, no. 4, pp. 593–603, 2012.