CRYPTOSYSTEMS USING ERROR-CORRECTING CODES BASED ON THE RANK METRIC

by

Maximilien Gadouleau

A Thesis

Presented to the Graduate Committee

of Lehigh University

in Candidacy for the Degree of

Master of Science

in

Electrical and Computer Engineering

Lehigh University

September, 2005

Approved and recommended for acceptance as a thesis in partial fulfillment of the requirements for the degree of Master of Science.

_____
Date

_____
Thesis Advisor

_____
Chairperson of Department

Acknowledgments

First, I would like to thank Professor Zhiyuan Yan for all his precious advice. He made me discover the world of cryptosystems using error-correcting codes. It was a real pleasure working with him and I am very thankful to him for being my thesis advisor. I am looking forward to working on the Ph.D. dissertation!

I also wish to thank Professor Kenneth Tzeng, who taught me the theory of error-correcting codes. He introduced me to a world where Evariste Galois (one of my personal heroes) is the king, and which is the melting point of linear algebra and finite field theory.

Professor Richard Decker, who was my academic advisor in my Master of Science degree, advised me to attend Professor Tzeng's class. Thank you for helping me choose the right classes that determined my research field.

My research led me into meeting Professor Pierre Loidreau whom I thank for enlightening me about cryptosystems using Gabidulin codes.

I also would like to thank Brianne Clapp and Jill Tardero who handled all the administrative paperwork, which I have to admit I am pretty afraid of!

I have definitely spent the best time of my life since I came to Bethlehem. I would like to thank Jennifer Humphreys, Alper Uygur, Şirin Ülker, Cyril Guintrand and all my friends at Lehigh University for their love and support.

Finally, thanks a lot to my family and friends for their endless support from either side of the Atlantic ocean.

Contents

List of Tables

List of Figures

Abstract

Cryptosystems using error-correcting codes offer higher security and lower computational costs than the number-theory-based cryptosystems. This thesis is a study of public-key and private-key cryptosystems using error-correcting codes based on the rank metric.

McEliece introduced the first cryptosystem using error-correcting codes. The Gabidulin-Paramonov-Tretjakov (GPT) public-key cryptosystem, based on Gabidulin codes, seems to have some advantages over McEliece's cryptosystem using Goppa codes. Indeed, the GPT cryptosystem can achieve a higher level of security than McEliece's cryptosystem, using smaller keys. We study the security of the GPT cryptosystem. In particular, given a structural attack and a decoding attack, the optimal distortion parameter for the GPT cryptosystem with respect to the two attacks is defined to be the distortion parameter that maximizes the work factor of the better attack. The work factor achieved by the GPT cryptosystem using the optimal distortion parameter thus gives a guaranteed level of security against the two given attacks. It is also shown that under reasonable assumptions, the optimal

distortion parameter always exists and is unique. Furthermore, how to compute and approximate the optimal distortion parameter with respect to two well-known attacks is illustrated and the behavior of the work factors achieved is studied.

Rao and Nam designed a private-key adaptation of McEliece's system. As an extension to our research, a private-key adaptation of the GPT cryptosystem is introduced. For this adaptation, two setups are proposed, one being the direct adaptation and the other using a permutation matrix; and the characteristics of these setups are studied in detail. Furthermore, a new attack against both schemes is designed, and the parameter ranges for which the attack is defeated in each setup are obtained. Concordantly, the robustness of the setups is also shown. Finally, proof that they outperform the private-key adaptations of McEliece's system is given.

Chapter 1

Introduction

This thesis is a study of cryptosystems using error-correcting codes. The purpose of this section is to review the basic concepts of error-correcting codes and cryptography.

1.1 Error-correcting codes

All communication channels contain some degree of noise, namely interference caused by various sources such as electric impulses, neighboring channels, or deterioration of the equipment. Noise can also interfere with data transmission. In order to overcome the detrimental effects caused by noise, one possible solution entails adding redundancy to the data before transmitting it over the channel, which enables error correction at the receiver's end. Error-correcting coding investigates how to efficiently add redundancy to data streams in order to ensure reliable communication.

Due to the multiplicity of channels, error-correcting codes are employed in many applications. Without error-correcting codes, diverse systems, from the CD player to deep-space communications, would not be technically feasible. The main challenge of error-correcting coding is to design the most appropriate codes for given channels, data representations, and applications. For instance, the codes used for deep-space communications focus on providing a high error protection for power limited signals, but the codes used for the CD player focus on performing the coding and decoding schemes at high throughput. Nowadays, advancements in the field of error-correcting codes play an important part in one of the most challenging topics of communication and network theory: building reliable high-speed wireless sensor networks.

1.2 Cryptography

There has always been a need to keep information secret from other parties. Now that global computer networks are thriving, the need for secrecy is more pronounced than ever. Already the exchange of sensitive information over the Internet such as credit card numbers is common practice. Protecting data and electronic systems is crucial to our way of living. Cryptography is the science of designing cryptosystems that are used to keep data secret, and to allow the secure transmission of data.

A basic outline of a communication system, where cryptosystems are implemented, is illustrated as follows. In a given communication system, Alice wants to send a message to Bob over a channel on which Oscar can eavesdrop. She encrypts

her message (called *plaintext*) using an encryption rule depending on an encryption key, and sends the resulting message (*ciphertext*) through the communication channel. The encryption must prevent Oscar from recovering the plaintext. Bob receives the ciphertext and, using his decryption rule and key, recovers the original plaintext.

Two different types of cryptosystems have been developed in the literature. The first category is composed of *private-key* cryptosystems. In this case, only Alice and Bob know which key to use to encrypt a message to Bob. If the encryption and decryption keys are the same, then the cryptosystem is called *symmetric*. On the other hand, in a *public-key* cryptosystem, everyone knows the encryption key used to send messages to Bob. In this setting, Bob is the only individual who knows the decryption key, hence no eavesdropper can decrypt any message sent to Bob.

Cryptography is used to solve real-world problems that require security. Applications of cryptography include online banking, digital watermarking, and secret sharing. To illustrate the importance of cryptography, we consider the example of e-commerce. First, cryptography ensures secure identification by designing efficient password systems, allowing the user to prove his identity while logging on to his machine. Subsequently, secure transactions over open channels need to be carried out and credit card information needs to be protected from fraudulent merchants. Also, if a bill is to be sent from the merchant to the customer, digital signatures guarantee that a third party does not modify the received message.

1.3 Outline

In this thesis, the cryptosystems based on the theory of error-correcting codes are studied. Those systems are interesting alternatives to the RSA or ElGamal's cryptosystems.

In Chapter 2, the first cryptosystem based on error-correcting codes, McEliece's public-key cryptosystem, is reviewed. Its security is based on the NP-complete problem of bounded distance decoding of a linear code. Then, the focus is set to private-key adaptations of McEliece's system. The GPT cryptosystem is also reviewed, with an introduction to the rank metric and Gabidulin codes.

The study of the security of the GPT cryptosystem leads to the definition of an optimal distortion parameter. In Chapter 3, the optimal distortion parameter is defined and its value is determined by an efficient algorithm. The behavior of the overall security of the system with respect to the choices of parameters is then discussed.

An adaptation of the GPT cryptosystem to a private-key cryptosystem is designed in Chapter 4. After a study of its security, it is proved that this private-key cryptosystem can achieve a higher level of security than the private-key adaptations of McEliece's cryptosystem.

Finally, a summary of the findings is presented in Chapter 5.

Chapter 2

Background

This chapter gives an overview of concepts that are critical to the study of public-key and private-key cryptosystems using error-correcting codes. First McEliece's system is studied. Then its private-key adaptations by Rao and Nam are examined. The rank metric and Gabidulin codes, along with the GPT cryptosystem, are also reviewed.

2.1 McEliece's cryptosystem

The principle of public-key cryptosystems using error-correcting codes was established by McEliece in [McE78], two years after Diffie and Hellman [DH76] introduced the concept of public-key cryptography. In [McE78], McEliece proposed a new cryptosystem based on binary irreducible Goppa codes. The security of this system is based on the NP-complete problem of *bounded distance decoding of a linear code*

[BMvT78]. Such a system can be theoretically generalized by replacing Goppa codes with any family of codes for which a polynomial-time decoding algorithm is known. In practice, however, it is difficult to find such a family of codes whose structure can be effectively hidden. In what follows, McEliece's cryptosystem is described, and a summary of the cryptanalysis of McEliece's system is given.

2.1.1 Presentation of McEliece's cryptosystem

Let $\mathcal{G}$ be the family of Goppa codes [Gop70, Ber73] over GF(2) of length $n = 2^l$, dimension k and error correction capability t. Let C be a Goppa code in $\mathcal{G}$, generated by a $k \times n$ matrix $\mathbf{G}$; let $\mathbf{S}$ be a random $k \times k$ invertible matrix over GF(2); and let $\mathbf{P}$ be an $n \times n$ permutation matrix. McEliece proposed to use the triple $(\mathbf{G}, \mathbf{S}, \mathbf{P})$ as a private key for his cryptosystem, with size equal to $nk + k^2 + nl$ bits. The *public key* is composed of t and the matrix $\mathbf{G}' = \mathbf{SGP}$, and its size is equal to nk.

To encrypt a plaintext message $\mathbf{x}$, Alice computes and sends the ciphertext $\mathbf{y} = \mathbf{xG}' + \mathbf{e}$, where $\mathbf{e}$, called the *error vector*, is a random vector of length n over GF(2) with Hamming weight t. At the other end of the channel, the decryption process uses a polynomial-time decoding algorithm for Goppa codes [SKHN76]. To decrypt a ciphertext $\mathbf{y}$, Bob computes $\mathbf{y}' = \mathbf{yP}^{-1} = \mathbf{xSG} + \mathbf{eP}^{-1}$. The Hamming weight of the vector $\mathbf{eP}^{-1}$ is equal to t. It follows that the vector $\mathbf{y}'$ is the sum of a codeword of the private code and a decodable error vector. Using the decoding algorithm for C, Bob retrieves $\mathbf{x}' = \mathbf{xS}$, and in the end recovers the plaintext by

calculating $\mathbf{x} = \mathbf{x}'\mathbf{S}^{-1}$.

The complexity of the encryption process is given by $W_C = nk/2$ binary operations. The computational cost of decrypting a message is roughly equal to the work factor of the appropriate decoding algorithm of Goppa codes, which is approximately given by $W_D = 3lnt + 4l^2t^2$ binary operations [Loi01].

McEliece [McE78] also recommended the following parameters of the private code: $l = 10$, $n = 2^{10} = 1024$, $k = 524$, and $t = 50$. Using these parameters, the public key size is 524 kilobits, and the transmission rate is approximately 51%; the encryption and decryption complexities are respectively given by 512 and 5,101 binary operations per information bit.

2.1.2 Cryptanalysis of McEliece's cryptosystem

In this section, possible attacks against McEliece's system are considered and their efficiencies are discussed. Two types of attacks against McEliece's cryptosystem have been proposed: *structural* (see, for example, [SS92]) and *decoding attacks* (see [CC98]).

A *structural attack* consists of recovering the private key from the public key. In most cases, it is based on finding an equivalent code to the *public code* for which a polynomial-time algorithm is known for the bounded distance decoding problem. Given such an algorithm, every intercepted message can be decrypted by Oscar, the attacker. It must be noted that this attack depends only on the structure of the *key*

set.

Literature suggests that no efficient structural attack exists against McEliece's system using Goppa codes. However, several attacks have shown that the use of other codes as private codes can lead to a dramatic loss of security. For instance, McEliece's cryptosystem using generalized Reed-Solomon codes has been proved to be insecure against a structural attack by Sidelnikov and Shestakov in [SS92]. Sendrier [Sen98] also proved that concatenated codes are not suitable for McEliece's cryptosystem, and Monico *et al.* [MRS00] demonstrated that LDPC codes should not be used, either. Many researchers suspect that the lack of apparent structure of Goppa codes contributes to their robustness against structural attacks.

Since structural attacks are inefficient, we shall consider *decoding attacks*. The principle of decoding attacks is to identify the plaintext directly from a given ciphertext. Unlike a structural attack that needs to be processed only once, a decoding attack must be performed when every new ciphertext is intercepted. The complexity of such attacks depends only on the length, dimension, and minimum distance of the codes in the key set.

There are two categories of decoding attacks: *passive* and *active attacks*. In a passive attack, Oscar only decodes the ciphertext according to the public code. Conversely, in an active attack, it can operate on several ciphertexts, or even ask Alice to send the same message several times. For further information about passive attacks, see [LB88, CC98]. The attack in [CC98] is the most efficient passive attack known to date and has an average work factor of approximately 2^{66} when the block

size is maintained at 1024. For active attacks, refer to [Ber97, HGS99, KI01].

2.2 Rao-Nam private-key cryptosystem

As the previous section shows, McEliece introduced a public-key cryptosystem based on error-correcting codes which uses large keys. Private-key cryptosystems can offer a higher level of security and use smaller keys than public-key systems. Therefore, several private-key adaptations of this system have been designed to reduce the key sizes.

The first adaptation was proposed by Rao [Rao84]. However, this adaptation was not secure against chosen-plaintext attacks. To overcome this weakness, Rao and Nam [RN86] then introduced a revised private-key cryptosystem based on McEliece's system. Both adaptations, and the security of the scheme, will now be presented and analyzed, with a focus on the ST attack.

2.2.1 PRAC

Rao's initial scheme [Rao84], Private-key Algebraic-coded Cryptosystems (PRAC), is a direct adaptation of McEliece's system: the encryption and decryption processes are exactly the same, but $\mathbf{G}'$ is kept private as well as $\mathbf{S}$, $\mathbf{G}$ and $\mathbf{P}$ to provide a higher level of security. Nevertheless, PRAC is vulnerable against chosen-plaintext attacks.

A chosen-plaintext attack consists of two steps: Step 1 determines $\mathbf{G}'$ from a

large set of given plaintexts and corresponding ciphertexts, and Step 2 retrieves the unknown plaintext $\mathbf{x}$ from the received ciphertext $\mathbf{y}$. Since Step 2 is analogous to a decoding attack against McEliece's system, we only consider Step 1 in detail. A chosen-plaintext attack could proceed as follows. Let $\mathbf{x}_1$ and $\mathbf{x}_2$ be two plaintexts differing in only one position, i.e., $\mathbf{x}_1 - \mathbf{x}_2 = \mathbf{u}^i = (0 \cdots 010 \cdots 0)$ with a single 1 in the i-th position. It follows that $\mathbf{y}_1 - \mathbf{y}_2 = g'_i + (\mathbf{z}_1 - \mathbf{z}_2)$ where g'_i is the i-th row of the $\mathbf{G}'$ matrix. The Hamming weight of $(\mathbf{z}_1 - \mathbf{z}_2)$ is at most $2t$, which is usually much smaller than n. Thus, the majority of the bits of the vector $\mathbf{y}_1 - \mathbf{y}_2$ are those of g'_i and $\mathbf{y}_1 - \mathbf{y}_2$ may be considered as an estimate of g'_i. By repeating the procedure several times, a sufficient number of estimates of g'_i can be obtained. From these estimates, the vector g'_i is correctly determined by majority voting for each position. This procedure, repeated for all $i = 1, 2, \ldots, k$, determines $\mathbf{G}'$. Once $\mathbf{G}'$ is obtained, Oscar may proceed with Step 2, which requires a relatively small work factor because t is small.

2.2.2 The Rao-Nam scheme

In order to overcome the deficiencies of PRAC, Rao and Nam [RN86] introduced an improved private-key system that offers robustness against the attack presented above. Although Rao and Nam only considered a binary system, Struik and van Tilburg [SvT87] generalized the scheme to any finite field GF(q). The generalized scheme is described as follows.

Let C be a linear code over GF(q) with length n, dimension k, minimum distance d, generator matrix $\mathbf{G}$ and parity check matrix $\mathbf{H}$. The encryption matrix $\mathbf{E}$ is constructed by: $\mathbf{E} = \mathbf{SGP}$, where $\mathbf{S}$ is a $k \times k$ non-singular matrix over GF(q) and $\mathbf{P}$ is an $n \times n$ permutation matrix. The matrices $\mathbf{S}$, $\mathbf{G}$, and $\mathbf{P}$ form the secret key.

The encryption of a given message $\mathbf{x}$ of length k over GF(q) differs from the classical McEliece's encryption scheme. Indeed, Alice sends the message $\mathbf{y} = \mathbf{xE} + \mathbf{zP} = (\mathbf{xSG} + \mathbf{z})\mathbf{P}$, where $\mathbf{z}$ is a length-n error vector over GF(q) with a Hamming weight roughly given by $w(\mathbf{z}) = \frac{q-1}{q}n$. This property of $\mathbf{z}$ is used to ensure that majority voting on the rows of $\mathbf{E}$ cannot be applied.

The security of the system depends mostly on the error vector $\mathbf{z}$. The main issue is therefore to define the set of error vectors in order to prevent a chosen-plaintext attack by majority voting on each row of the encryption matrix $\mathbf{E}$. Rao and Nam proposed two methods to select a set of error vectors which satisfy the weight property and allow unique decoding. The first method uses $q = 2$ and i adjacent errors (ATE) for $\mathbf{z}$. The Rao-Nam system using ATE's was cracked by Hin [Hin86], hence we shall not consider this method henceforth. The second technique consists of selecting one error vector from each coset of the standard array of the code C, meaning that each vector corresponds to one syndrome. There are q^{n-k} cosets; thus, the set of error vectors will contain q^{n-k} elements. This pre-defined set of error vectors is part of the secret key.

The decryption consists of three steps. First, Bob calculates $\mathbf{y}' = \mathbf{yP}^{-1} = \mathbf{xSG} + \mathbf{z}$. Then he computes the syndrome $\mathbf{s} = \mathbf{y}'\mathbf{H}^T$ in order to identify the error

vector $\mathbf{z}$. As a result, he calculates $\mathbf{y}'' = \mathbf{y}' - \mathbf{z} = \mathbf{xSG}$. Finally, the plaintext can be determined by computing $\mathbf{x} = \mathbf{y}''(\mathbf{SG})^{-R}$ where $(\mathbf{SG})^{-R}$ is the right inverse of the $(\mathbf{SG})$ matrix.

The decryption process of the Rao-Nam scheme can be described using an equivalent decoding algorithm. Denote the decryption matrix as $\mathbf{D} = \mathbf{HP}$, since $\mathbf{ED}^T = (\mathbf{SGP})(\mathbf{HP})^T = \mathbf{0}$. The equivalent decoding algorithm proceeds as follows. First, compute the syndrome $\mathbf{s} = \mathbf{yD}^T$ and find the corresponding permuted error vector $\mathbf{zP}$. As a result, calculate $\mathbf{y}' = \mathbf{y} - \mathbf{zP} = \mathbf{xE}$. Finally, calculate $\mathbf{c} = \mathbf{y}'\mathbf{E}^{-R}$, where $\mathbf{E}^{-R}$ is the right inverse of the matrix $\mathbf{E}$.

The Rao-Nam scheme is more secure than PRAC against a chosen-plaintext attack based on majority voting. However, two remaining security weaknesses are exploited by the attack described in Section 2.2.3. The first weakness is the small number of different error vectors, which is given by $N = q^{n-k}$ for the syndrome-error table. Oscar has to encrypt on average $N \log N$ times to obtain all the error vectors [SvT87]. The second weakness is the possibility of estimating the rows of the encryption matrix $\mathbf{E}$ by means of constructing unit vectors. The Rao-Nam scheme, while secure against the chosen-plaintext attack previously reviewed, is still vulnerable to more sophisticated attacks which are not based on majority voting.

2.2.3 The ST attack

After introducing the security deficiencies of the Rao-Nam scheme, the most efficient attack to date is reviewed. The ST attack [SvT87] takes advantage of the weaknesses presented above. This section reviews the description of the attack given in [SvT87].

Let $\mathbf{z}_j$ and $\mathbf{z}_l$ be two error vectors and consider $\mathbf{z}_{j,l} = \mathbf{z}_j - \mathbf{z}_l$. The set of all permuted error vector differences will be denoted as $\mathcal{D}_P \stackrel{\text{def}}{=} \{\mathbf{z}_{j,l}\mathbf{P}\}_{j,l=1}^{N}$. Let $\hat{\mathbf{z}}$ denote a guessed error vector; thus, the difference between a guessed error vector and an actual error vector is given by $\tilde{\mathbf{z}}_{j,l} = \hat{\mathbf{z}}_j - \mathbf{z}_l$. For any message $\mathbf{x}$, there are N distinct encryptions $\mathbf{y}_j$ possible, which will be denoted as $\mathcal{E} = \{\mathbf{y}_j\}_{j=1}^{N}$. For any $\mathbf{u}^i$, we define $\mathbf{x}^i = \mathbf{x} + \mathbf{u}^i$. The set of encryptions of the vector $\mathbf{x}^i$ is denoted by $\mathcal{E}^i$.

The attack can be described as follows. In the beginning, Oscar encrypts an arbitrary message $\mathbf{x}$ until all the N different ciphertexts $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N$ are obtained and then constructs a directed labeled graph $\Gamma = (\mathcal{E}, \mathcal{D}_P)$. In this graph, the vertices are $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N$, and the edge from vertex $\mathbf{y}_j$ to $\mathbf{y}_l$ has label $\mathbf{y}_j - \mathbf{y}_l = \mathbf{z}_{j,l}\mathbf{P}$. Subsequently, the automorphism group $Aut(\Gamma)$, consisting of all the permutations on $\mathcal{E}$ that leave all the labels $\mathbf{z}_{j,l}\mathbf{P}$ invariant, is constructed. Next, Oscar operates on a message $\mathbf{x}^i = \mathbf{x} + \mathbf{u}^i$. First, the N different ciphertexts $\mathbf{y}_j^i$ are obtained. Afterwards, the graph $\Gamma^i = (\mathcal{E}^i, \mathcal{D}_P)$ is formed. Then an automorphism ϕ from the automorphism group $Aut(\Gamma)$ is chosen at random. The mapping induced by ϕ from Γ^i onto Γ produces N vectors $\mathbf{y}_1^i, \mathbf{y}_2^i, \ldots, \mathbf{y}_N^i$ synchronized in a certain manner with

$\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N$. For any $1 \leq j \leq n$,

$$\mathbf{y}_j^i - \mathbf{y}_j = (\mathbf{x}^i\mathbf{E} + \hat{\mathbf{z}}_j^i\mathbf{P}) - (\mathbf{x}\mathbf{E} + \mathbf{z}_j\mathbf{P}) = \mathbf{e}_i + \tilde{\mathbf{z}}_{j,j}\mathbf{P}.$$

With probability $|Aut(\Gamma)|^{-1}$, the row $\mathbf{e}^i$ of the $\mathbf{E}$ matrix will be correctly estimated as there exists an automorphism ϕ such that $\tilde{\mathbf{z}}_{j,j} = \mathbf{0}$. The correctness of a row cannot be verified independently from the other rows, therefore the entire estimate of the encryption matrix has to be constructed prior to verifying its correctness. On average, we can expect that $|Aut(\Gamma)|^k$ guessed encryption matrices $\hat{\mathbf{E}}$ need to be generated before the correct one can be obtained. Once $\hat{\mathbf{E}}$ is found, Oscar calculates the decryption matrix $\hat{\mathbf{D}}$ and the $\hat{\mathbf{E}}^{-R}$ matrix and builds the syndrome-error table. If the estimated solution is not correct, then another automorphism ϕ is chosen and the procedure is repeated. Below is a summary of the algorithm for the ST attack:

ST attack algorithm

1. Encrypt a message $\mathbf{x}$ until all its N different encryptions $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N$ are obtained.

2. Construct the directed complete graph $\Gamma = (\mathcal{E}, \mathcal{D}_P)$ and the automorphism group $Aut(\Gamma)$.

3. For $1 \leq i \leq k$, select a message $\mathbf{x}^i$ such that $\mathbf{x}^i = \mathbf{x} + \mathbf{u}^i$. Repeat Step 1 for $\mathbf{x}^i$ and construct $\Gamma^i = (\mathcal{E}^i, \mathcal{D}_P)$.

4. For $1 \leq i \leq k$, select a random automorphism ϕ from the automorphism group $Aut(\Gamma)$. Map Γ^i on Γ according to ϕ and calculate $\hat{\mathbf{e}}^i = \mathbf{y}_j^i - \mathbf{y}_j$.

5. Construct the encryption matrix $\hat{\mathbf{E}}$, the decryption matrix $\hat{\mathbf{D}}$, the matrix $\hat{\mathbf{E}}^{-R}$ and the syndrome-error table and verify the solution. If the solution is not correct, repeat Steps 4 and 5.

From the review of the ST attack algorithm, the number of encryptions and the computational costs can be derived. Oscar needs $O(kN \log N)$ encryptions on average to obtain all the necessary ciphertexts and has to run $O(knN^2 \log N + kn|Aut(\Gamma)|^k)$ operations over GF(q). It then appears that $|Aut(\Gamma)|$ is a measure of the computational security of the generalized Rao-Nam scheme. $|Aut(\Gamma)|$ is actually determined by the set of error vectors and is upper bounded by N.

2.3 The GPT cryptosystem

The *rank metric* was introduced by Gabidulin in [Gab85], where the similarities and differences between the Hamming metric and the rank metric were discussed. In [Gab85], the Singleton bound was generalized and a family of codes that achieve the Singleton bound was defined. These codes, called Gabidulin codes, actually are analogous to generalized Reed-Solomon codes. A polynomial-time decoding algorithm for these codes was also proposed. Gabidulin, Paramonov and Tretjakov introduced a public-key cryptosystem, referred to as the GPT cryptosystem henceforth, based

on Gabidulin codes [GPT91]. In this section, the rank metric and Gabidulin codes are reviewed, and the GPT cryptosystem is studied.

2.3.1 The rank metric

Let $\gamma_1, \gamma_2, \ldots, \gamma_m$ be a basis of $\mathrm{GF}(q^m)$ over $\mathrm{GF}(q)$. Every element $b \in \mathrm{GF}(q^m)$ can be expressed as $b = \sum_{i=1}^{m} b_i \gamma_i$ where $b_i \in \mathrm{GF}(q)$. A vector $\mathbf{b}$ of length n over $\mathrm{GF}(q^m)$ can be represented as an $m \times n$ matrix over $\mathrm{GF}(q)$. The rank norm and the rank metric over $\mathrm{GF}(q^m)^n$ can be defined as follows:

Definition 1 (Rank norm) *Let* $\mathbf{a} = (a_1, a_2, \ldots, a_n)$ *be a vector of length* n *over* $GF(q^m)$. *Let* $\mathbf{A}$ *be the* $m \times n$ *matrix consisting of the coordinates* a_i *of* $\mathbf{a}$ *with respect to the basis* $\gamma_1, \gamma_2, \ldots, \gamma_m$. $\mathbf{A} = (a_{ij})_{i=1,j=1}^{m,n}$ *where* $a_j = \sum_{i=1}^{m} a_{ij}\gamma_i$. *The* rank norm *of the vector* $\mathbf{a}$, *denoted as* $rk(\mathbf{a})$, *is defined to be the rank of the matrix* $\mathbf{A}$ *over* $\mathrm{GF}(q)$, *i.e.,* $rk(\mathbf{a}) \stackrel{def}{=} rk(\mathbf{A})$.

Definition 2 (Rank metric) *Let* $\mathbf{a}$ *and* $\mathbf{b}$ *be two vectors of length* n *over* $\mathrm{GF}(q^m)$, *the function* d_{rk} *defined by* $d_{rk}(\mathbf{a}, \mathbf{b}) \stackrel{def}{=} rk(\mathbf{a} - \mathbf{b})$ *is a metric over* $\mathrm{GF}(q^m)^n$, *called the* rank metric.

We can therefore define the minimum rank distance d_{r} of a linear code similarly to its minimum Hamming distance d_{H}. It can be shown that $d_{\mathrm{r}} \leq d_{\mathrm{H}}$. Due to the Singleton bound of linear block codes, the minimum rank distance d_{r} of an (n, k) linear block code over $\mathrm{GF}(q^m)$ satisfies

$$d_{\mathrm{r}} \leq n - k + 1. \tag{2.1}$$

An alternative bound on the minimum rank distance is also given in [Gab85]:

$$d_{\mathrm{r}} \leq \frac{m}{n}(n-k)+1. \tag{2.2}$$

For $n \leq m$, the bound in (2.1) is tighter than that in (2.2). In contrast, the bound in (2.2) is tighter when $n > m$.

2.3.2 Gabidulin codes

The codes which achieve the Singleton bound for the rank metric with equality are called Maximum Rank Distance (MRD) codes. In [Gab85], Gabidulin constructed a family of MRD codes over $\mathrm{GF}(q^m)$ of length $n \leq m$. He also proposed a polynomial-time algorithm for the bounded distance decoding of Gabidulin codes in the rank metric. Below, the formal definition of Gabidulin codes is given.

Definition 3 (Gabidulin codes) *When $n \leq m$, let $\mathbf{g} = (g_1, g_2, \ldots, g_n)$ be linearly independent elements of $\mathrm{GF}(q^m)$. Then the code defined by the following generator matrix*

$$\mathbf{G} = \begin{pmatrix} g_1 & g_2 & \cdots & g_n \\ g_1^{[1]} & g_2^{[1]} & \cdots & g_n^{[1]} \\ \vdots & \vdots & \ddots & \vdots \\ g_1^{[k-1]} & g_2^{[k-1]} & \cdots & g_n^{[k-1]} \end{pmatrix}$$

where $[i] = q^i$ is called a Gabidulin code*, with dimension k, generated by $\mathbf{g} = (g_1, g_2, \ldots, g_n)$.*

The most important feature of Gabidulin codes is the existence of a polynomial-time algorithm for their decoding in the rank metric. Indeed, Gabidulin codes are analogous to Reed-Solomon codes for a special class of polynomials, called linearized polynomials. The extended Euclidean algorithm for linearized polynomials may be used to design a decoding algorithm for Gabidulin codes (see [Gab85] or [Gab91]). To learn more about linearized polynomials, see the review in [LN83] or the fundamental papers by Ore [Ore33, Ore34].

2.3.3 Description of the GPT cryptosystem

Gabidulin, Paramonov and Tretjakov designed in [GPT91] a new public-key cryptosystem, which is quite similar to McEliece's system. Instead of using codes which have a decoding algorithm in the Hamming metric, the GPT system uses Gabidulin codes defined with respect to the rank metric. Due to the properties of the rank metric, this system is more robust against decoding attacks than McEliece's system. Hence, we can, in theory, use smaller key sizes around several kilobits, instead of the several hundreds of kilobits normally used to achieve a good level of security for McEliece's cryptosystem. To produce the public key, the structure of the secret Gabidulin code has to be broken using a specific matrix, called the distortion matrix. The elements of the GPT cryptosystem are described as follows.

The private key consists of the triple $(\mathbf{G}, \mathbf{S}, \mathbf{X})$, where $\mathbf{G}$ is a $k \times n$ generator matrix of a Gabidulin code over GF(q^m), with length $n \leq m$ and error correction

capability t; $\mathbf{S}$ is an invertible $k \times k$ random matrix over GF(q^m); and $\mathbf{X}$ is a $k \times n$ matrix over GF(q^m) such that $\forall\, \mathbf{c} = (c_1, c_2, \ldots, c_k) \in \mathrm{GF}(q^m)^k$, $rk(\mathbf{cX}) \leq t_1$, where $t_1 < t$. $\mathbf{X}$ is called the *distortion matrix* with *distortion parameter* t_1. The public key is given by the distortion parameter t_1 and $\mathbf{G}' = \mathbf{SG} + \mathbf{X}$. The code over GF($q^m$) defined by the generator matrix $\mathbf{G}'$ is called the public code.

To encrypt the plaintext $\mathbf{c}$ of length k, Alice computes and sends the length-n vector $\mathbf{y} = \mathbf{cG}' + \mathbf{e}$ where $\mathbf{e}$ is a random error vector with length n and rank $\leq (t - t_1)$. The decryption of a ciphertext $\mathbf{y}$ relies on the existence of the decoding algorithm of Gabidulin codes in the rank metric. Since $\mathbf{y} = \mathbf{cSG} + (\mathbf{cX} + \mathbf{e})$, where $rk(\mathbf{cX}+\mathbf{e}) \leq t$, Bob applies the decoding algorithm of Gabidulin codes and retrieves the vector $\mathbf{c}' = \mathbf{cS}$, and subsequently computes $\mathbf{c} = \mathbf{c}'\mathbf{S}^{-1}$.

Having reviewed the cryptosystem, its performance in terms of computational costs is now studied. The cost of encrypting a message with length k is on average equal to $nk/2$ multiplications in GF(q^m). The computational cost of decrypting a message with length n is equal to the cost of decoding a vector and the cost of a $k \times k$ matrix inversion. The cost of the decryption process is therefore approximately equal to $O(t^3 + nt + k^3)$ multiplications in GF(q^m) [Loi01].

The main drawback of McEliece's system is the size of its public keys. Indeed, if the parameters proposed in [CC98] are used, then the public key size is roughly 628,000 bits. On the other hand, the size of the public key $\mathbf{G}'$ used in the GPT cryptosystem is only $mnk \log_2(q)$ bits. The following parameters were suggested in [GPT91] : $q = 2, m = n = 20, k = 12, t_1 = 1$. Using the recommended

parameters, the public key size, given by 4,800 bits, has been dramatically reduced from McEliece's system.

Note that a large number of distortion matrices can be produced [Loi01], even for small parameters. Hence any attack based on the exhaustive enumeration of the set of distortion matrices is prevented. More precisely, the most efficient structural attack against the GPT system [Gib95] can be overcome for reasonable values of n and m. Finally, the GPT cryptosystem appears to be more robust against decoding attacks than McEliece's system.

2.3.4 Attacks against the GPT cryptosystem

Two types of attacks — decoding and structural attacks — against the GPT cryptosystem have been proposed. In [CS96], a general decoding algorithm in the rank metric is introduced, and a decoding attack is derived. Its work factor is on the order of $O\left((n(t-t_1)+m)^3 q^{(m-t+t_1)(t-t_1-1)}\right)$. More recently, Ourivski and Johansson [OJ02] have proposed two attacks that have similar yet smaller work factors than the attack in [CS96]. The attacks proposed by Ourivski and Johansson are the best decoding attack to our knowledge.

In [Gib96], Gibson proposed a structural attack against the GPT cryptosystem which has a work factor on the order $O\left((n-2t)^3 q^{t_1(n-t_1)}\right)$ [Loi01]. Following the same idea, Overbeck [Ove05] proposed a promising structural attack, but did not give its work factor.

Chapter 3

Optimal distortion parameter for GPT

One of the main problems while designing a cryptosystem is to determine the parameter values that can achieve the highest security. Against the GPT cryptosystem, decoding and structural attacks have been proposed. Since the distortion parameter for the GPT cryptosystem affects its security against structural and decoding attacks in different fashions, given a structural attack and a decoding attack, we define the optimal distortion parameter for the GPT cryptosystem with respect to the two attacks to be the distortion parameter that maximizes the work factor of the better attack. The work factors achieved by the GPT cryptosystem using the optimal distortion parameter thus give a guaranteed level of security against the two given attacks. We also show that under reasonable assumptions, the optimal

distortion parameter always exists and is unique. Furthermore, we illustrate how to compute and approximate the optimal distortion parameter against two well-known attacks and study the behavior of the work factors achieved.

3.1 Definitions and properties

The work factors of the attacks depend on all four parameters — m, n, t_1, and t — of the GPT cryptosystem. The degree of extension m of the field, the block length n, and the error correction capability t of the Gabidulin code depend on the generator matrix $\mathbf{G}$ of the secret Gabidulin code, and are usually given. However, although t_1 determines the robustness against the attacks, how best to choose t_1, to our knowledge, has not been considered in the literature. Thus, we assume that m, n and t are fixed, and try to determine the best value for t_1. Given the triple (m, n, t) and a decoding attack and a structural attack, our goal is to find the optimal value of t_1, denoted as t_o, such that the GPT system achieves the highest level of security against both attacks. Let us define

$$W_m(t_1) \stackrel{\text{def}}{=} \min\left(W_d(t_1),\, W_s(t_1)\right),$$

where $W_d(t_1)$ and $W_s(t_1)$ denote the work factors of the given decoding and structural attacks respectively. Note that W_d, W_s, and hence W_m are all functions of t_1, which is a discrete variable from the set $T = \{1, 2, \ldots, t-1\}$. Below, it is sometimes convenient to treat t_1 as a real-valued variable from the interval $I = [1, t-1]$ and to treat both $W_d(t_1)$ and $W_s(t_1)$, and hence $W_m(t_1)$, as continuous functions on I by

interpolation on T. Intuitively, if t_1 increases, the structure of the secret Gabidulin code is better hidden by the distortion matrix and hence the work factor of any reasonable structural attack increases. In contrast, when t_1 increases, the rank of the error vector $t - t_1$ decreases and a decoding attack will have a smaller work factor. In the discussions below, the decoding and structural attacks are said to *satisfy the monotone conditions* over T (or I respectively) if their respective work factors $W_d(t_1)$ and $W_s(t_1)$ are strictly monotone decreasing and increasing functions with t_1 over T (or I respectively) respectively.

We define the *optimal distortion parameter* t_o with respect to the given structural and decoding attacks as

$$t_o \stackrel{\text{def}}{=} \arg\max_{t_1 \in T} \left(W_m(t_1) \right). \tag{3.1}$$

Clearly, t_o depends on the two given attacks, although to simplify notations such dependence is not reflected in the notation. We can show that

Proposition 1 *If the given structural and decoding attacks satisfy the monotone conditions over T, then the optimal distortion parameter t_o with respect to the two given attacks always exists and is unique.*

It is also instrumental to study the unique value in I, t_1^*, that maximizes W_m. That is,

$$t_1^* \stackrel{\text{def}}{=} \arg\max_{t_1 \in I} \left(W_m(t_1) \right). \tag{3.2}$$

Similarly, we can show that

Proposition 2 *If the given structural and decoding attacks satisfy the monotone conditions over I, then t_1^* always exists and is unique. Furthermore, t_o is either $\lfloor t_1^* \rfloor$ or $\lfloor t_1^* \rfloor + 1$.*

By Proposition 2, if we can find t_1^*, we only need to evaluate $W_m(\lfloor t_1^* \rfloor)$ and $W_m(\lfloor t_1^* \rfloor + 1)$ and to compare them to determine t_o. More generally, we have

Proposition 3 *If the given structural and decoding attacks satisfy the monotone conditions over I and $l \leq t_1^* \leq u$, then $\lfloor l \rfloor \leq t_o \leq \lceil u \rceil$.*

Thus, we only need to evaluate $W_m(t_1)$ and pick t_o from between $\lfloor l \rfloor$ and $\lceil u \rceil$. Proofs of Propositions 1, 2, and 3 are straightforward and hence omitted from this section. Moreover, when the given attacks satisfy both the monotone conditions and $W_d(1) \geq W_s(1)$ and $W_d(t-1) \leq W_s(t-1)$, t_1^* is given by the unique point at which the two functions cross over each other. Finally, if we further assume that $W_d(\cdot)$ has an inverse function $W_d^{-1}(\cdot)$, then $t_1^* = W_d^{-1}(W_s(t_1^*))$. That is, t_1^* is the unique *fixed point* of the function $W_d^{-1}(W_s(\cdot))$.

We remark that the advantage of defining the optimal distortion parameter with respect to the two given attacks is that the optimal distortion parameter can be updated as more efficient attacks emerge. Given a triple (m, n, t), $\text{maxmin}(m, n, t) \stackrel{\text{def}}{=} W_m(t_o)$ represents the work factor guaranteed by the GPT cryptosystem with respect to the two given attacks. However, it is in general difficult to obtain t_o directly. For small values of t, an exhaustive search on all possible values of t_1 can be used to determine t_o. However, as illustrated in Section 3.2, t_o can be computed with

low complexity when t is large. If necessary, t_o can even be approximated with *no computation* as shown in Section 3.2.2.

3.2 Finding the optimal distortion parameter

In this section, we use the work factors of the attacks in [CS96] and [Gib96] to illustrate how to find t_o with low complexity. We remark that the attacks in [CS96] and [Gib96] are used since both are well known and the attack in [Gib96] is the best structural attack whose work factor is known to us. The attack proposed by Overbeck [Ove05] seems promising, but its work factor is unknown to us. The attack in [CS96] is similar to the attacks proposed by Ourivski and Johansson [OJ02], which are the best decoding attacks to our knowledge. Thus, we conjecture that the approach used below can be extended to other similar attacks as well.

3.2.1 Finding the exact value

We consider only the respective dominant terms of the work factors of the attacks in [CS96] and [Gib96]. This approximation should have little effect on t_o in all practical cases. Hence, the work factor of the decoding attack in [CS96] is given by $W_d(t_1) = c_d \left[n(t-t_1)+m\right]^3 \cdot q^{(m-t+t_1)(t-t_1-1)}$. Similarly, the work factor of the structural attack in [Gib96] is given by $W_s(t_1) = c_s(n-2t)^3 \cdot q^{t_1(n-t_1)}$. Note that c_d and c_s are both constants.

We will first consider the case where $\frac{c_s}{c_d} = 1$, and then extend our approach

to the more general case. It is easy to verify that $W_d(t_1)$ and $W_s(t_1)$ are strictly monotone decreasing and increasing over I respectively. Moreover, one can check that $W_d(1) > W_s(1)$ and $W_d(t-1) < W_s(t-1)$. Hence, there is a unique cross-over point between the two functions. This cross-over occurs exactly at t_1^*. It is, however, unnecessary to find a closed formula for t_1^* since, in the end, we only care about the integers surrounding t_1^*. Due to Proposition 3, it suffices to bound t_1^* tight enough so that only a few choices are left for t_o. If we can bound t_1^* between two numbers l_a and u_a which differ by less than 1 but have different floors, then we have only *three* possibilities for t_o: $\lfloor l_a \rfloor$, $\lceil l_a \rceil = \lfloor u_a \rfloor$, or $\lceil u_a \rceil$. If l_a and u_a have the same floor, we have only *two* possibilities for t_o: $\lfloor l_a \rfloor$ or $\lfloor l_a \rfloor + 1$.

Since t_1^* is given by the cross-over point, we have

$$W_d(t_1^*) = W_s(t_1^*).$$

Now let us assume that the characteristic q of the field is 2. Taking $\log_2$ on both sides and assuming that $\log_2\left(\frac{c_s}{c_d}\right) = 0$, we have

$$t_1^*(-m-n+2t-1) + f(t_1^*) + mt - m + t - t^2 = 0, \tag{3.3}$$

where $f(t_1) = 3\log_2\left[\frac{n(t-t_1)+m}{n-2t}\right]$. Rearranging (3.3), we obtain

$$\begin{aligned} t_1^* &= \underbrace{\frac{-t^2+t(m+1)-m}{m+n+1-2t}}_{c_0} + \underbrace{\frac{f(t_1^*)}{m+n+1-2t}}_{g(t_1^*)} \\ &= c_0 + g(t_1^*). \end{aligned} \tag{3.4}$$

Let $\bar{g}(t_1) \stackrel{\text{def}}{=} c_0 + g(t_1)$, then t_1^* is a fixed point of $\bar{g}(\cdot)$ and it can be shown that the

fixed point is unique. Note that c_0 is a constant whose value depends on m, n, and t.

Now, let us bound the value of t_1^*. Since $g(t_1^*) > 0$ over I,

$$c_0 \leq t_1^* \leq t - 1. \tag{3.5}$$

Note that $\bar{g}(t_1)$ is a decreasing function in t_1 on I and $\bar{g}(t_1) > 0$, so applying the function $\bar{g}(\cdot)$ to all the terms in (3.5) leads to

$$\bar{g}(t-1) \leq \bar{g}(t_1^*) = t_1^* \leq \bar{g}(c_0). \tag{3.6}$$

If we go on and recursively apply the function $\bar{g}$ to all the terms in (3.5), we obtain a sequence of lower and upper bounds on t_1^*. The sequence of lower and upper bounds of t_1^* is given by

$$u_0 = t - 1 \qquad l_0 = c_0$$

$$u_{i+1} = \bar{g}(l_i) \qquad l_{i+1} = \bar{g}(u_i).$$

Clearly, $l_i \leq t_1^* \leq u_i$ for $i = 0, 1, \ldots$. In fact, it can be shown that the series u_i and l_i both converge to t_1^* (proof is given in Appendix A.1). However, we will show next that the bounds in (3.6) are sufficient.

Let us denote the difference between the upper and lower bounds in (3.6) as

$$\begin{aligned} h(m,n,t) &\stackrel{\text{def}}{=} \bar{g}(c_0) - \bar{g}(t-1) \\ &= \frac{3}{m+n+1-2t} \log_2 \left[\frac{n(t-c_0)+m}{n+m} \right]. \end{aligned}$$

We can show that

Proposition 4 $h(m, n, t) < 1$.

The proof of Proposition 4 is given in Appendix A.2. Hence, it is only necessary to check at most three choices to obtain t_o. In fact, $h(m, n, t)$ often is so small that both bounds have the same floor; hence, only two integers have to be checked to obtain t_o.

We have assumed $\frac{c_s}{c_d} = 1$ in the discussions above. This assumption is justified when the order of the work factor is much more important than the constant coefficient in the work factor. Hence, the distortion parameter obtained under the assumption that $\frac{c_s}{c_d} = 1$ is optimized with respect to the orders of the work factors for the two attacks. When c_s and c_d are far apart, it is necessary to take the constant coefficients in the work factors into account. When $\frac{c_s}{c_d}$ has to be accounted for, c_0 in (3.5) becomes

$$c_0 = \frac{-t^2 + t(m+1) - m - \log_2\left(\frac{c_s}{c_d}\right)}{m + n + 1 - 2t}.$$

If $c_0 \geq t - 1$, it is clear that $t_o = t - 1$. Otherwise, we redefine $l_0 = \max\{1, c_0\}$, and the recursive algorithm proposed above is still applicable.

In Table 3.1, we show the optimal distortion parameter t_o and its corresponding maxmin(m, n, t) with respect to the attacks in [CS96] and [Gib96] for $(m, n) = (32, 32)$ and variable t. We assumed that $\frac{c_s}{c_d} = 1$. We note that t_o is selected between two possibilities in all the cases.

t	u_0	l_0	u_1	l_1	t_o	$\log_2$(maxmin)
2	1	0.4918	0.5665	0.5505	1	18
3	2	0.9831	1.0793	1.0491	1	45
4	3	1.4737	1.5912	1.5482	2	49
5	4	1.9636	2.1029	2.0477	2	73
6	5	2.4528	2.6149	2.5478	3	79
7	6	2.9412	3.1276	3.0488	3	99
8	7	3.4286	3.6415	3.5510	4	105
9	8	3.9149	4.1573	4.0548	4	123
10	9	4.4000	4.6758	4.5610	5	130
11	10	4.8837	5.1983	5.0706	5	144
12	11	5.3659	5.7268	5.5854	6	153
13	12	5.8462	6.2648	6.1088	6	163
14	13	6.3243	6.8203	6.6486	7	174
15	14	6.8000	7.4173	7.2286	7	178

Table 3.1: The optimal distortion parameter and its corresponding maxmin(m, n, t) with respect to the attacks in [CS96] and [Gib96] for $(m, n) = (32, 32)$ and variable t

3.2.2 Approximation of t_o

We note that $t_o = \lfloor t/2 \rfloor$ for all the cases in Table 3.1. We show in this section that this is not a coincidence and that, assuming $n > 4$ and moderate m, $\lfloor t/2 \rfloor$ is a good approximation for t_o. Thus, for real-time applications, t_o can be obtained with no computation if necessary. Since $t_1^* = \frac{t-1}{2} + a(m,n,t) + g(t_1^*)$, where

$$a(m,n,t) = \frac{(m-n-1)(t-1)}{2(m+n+1-2t)}, \tag{3.7}$$

we will in turn show that t_1^* is close to $\frac{t-1}{2}$ by illustrating that the two correcting terms $a(m,n,t)$ and $g(t_1^*)$ are both small.

In Appendix A.3, we first show that

Lemma 1 *For all t, n, and m, $a(m,n,t)$ satisfies*

$$-\frac{m-2}{4(m+1)} \leq a(m,n,t) \leq \frac{(m-3)^2}{16(m+1)}. \tag{3.8}$$

Both bounds in (3.8) are small for moderate m. Indeed, for $m = 32$, (3.8) becomes $-0.23 \leq a(m,n,t) \leq 1.6$.

In Appendix A.4, we also show that

Lemma 2 *For all t, n, and m, $g(t_1^*)$ satisfies*

$$0 < g(t_1^*) \leq \frac{3}{m+1} \log_2 \left[m \frac{m^2+8m+4}{4m+4} \right]. \tag{3.9}$$

For moderate m, the upper bound in Lemma 2) is small. For $m = 32$, the upper bound in (3.9) is 0.76.

Due to Lemmas 1 and 2, the value of t_1^* is close to $\frac{t-1}{2}$ and hence $t_o \approx \lfloor t/2 \rfloor$ for $n > 4$ and moderate m.

3.3 Discussions

Now that we have obtained the optimal distortion parameter and its corresponding maxmin(m, n, t) with respect to the attacks in [CS96] and [Gib96], we compare maxmin(m, n, t) with the security of McEliece's cryptosystem. We then study the behavior of maxmin(m, n, t) when the parameters of the GPT cryptosystem vary. This study provides guidelines on how to select the parameters for the GPT cryptosystem.

3.3.1 Comparison with McEliece's cryptosystem

The public key size of McEliece's cryptosystem is roughly nk [Loi01], and the work factor of the best attack against it is given in [CC98]. Setting $n = 1024$, the optimal parameter for McEliece's system given in [CC98] — $k = 614$ and $t = 41$ — leads to a public key size of roughly $629,000$ bits and a work factor of 2^{66} [CC98]. To maintain the same block length, we use $m = n = 32$ for the GPT cryptosystem. Furthermore, $k = 20$ ($t = 6$) is selected so as to achieve approximately the same rate as McEliece's system above. From Table 3.1, this set of parameters leads to the optimal distortion parameter $t_o = 3$ and a work factor of at least 2^{79} against the attacks in [CS96] and [Gib96]. Using these parameters, the public key size, roughly nkm [Loi01], is reduced to approximately $20,000$ bits. Thus, in comparison to McEliece's cryptosystem, the GPT public-key cryptosystem achieves a higher security level with much smaller key sizes.

3.3.2 The behavior of maxmin(m, n, t)

In this section, we study the behavior of maxmin(m, n, t) when n and t vary. Thus, we will use $W_d(t_1; m, n, t)$ and $W_s(t_1; m, n, t)$ to denote the work factors of the given decoding and structural attacks to avoid confusion. In Figure 3.1, we fix $m = 32$ and show maxmin(m, n, t) as a function of t with n varying from 20 to 32. Note that maxmin(m, n, t) is well-defined only on $t \in \left\{0, 1, \ldots, \left\lfloor \frac{n-1}{2} \right\rfloor\right\}$. The curves for maxmin$(m, n, t)$ shown in Figure 3.1 are obtained simply by connecting the points with straight lines.

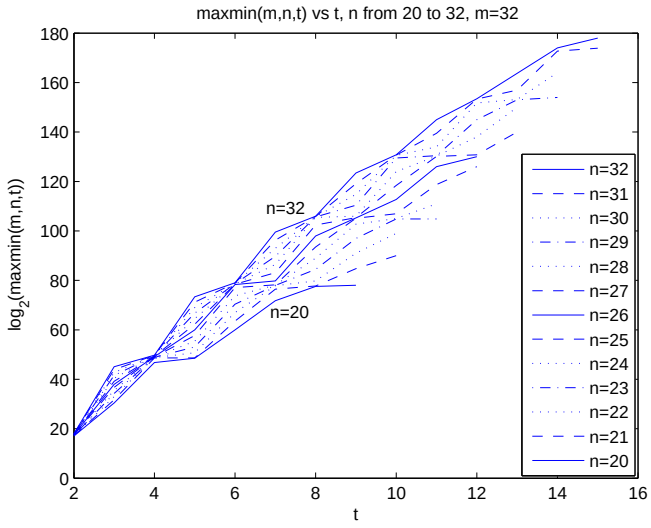

Figure 3.1: The guaranteed work factors maxmin(m, n, t) as a function of t, $m = 32$ and n varies from 20 to 32

As we can see, there are some points where a range of values of n lead to approximately the same maxmin value. We refer to these points as *tradeoff points*,

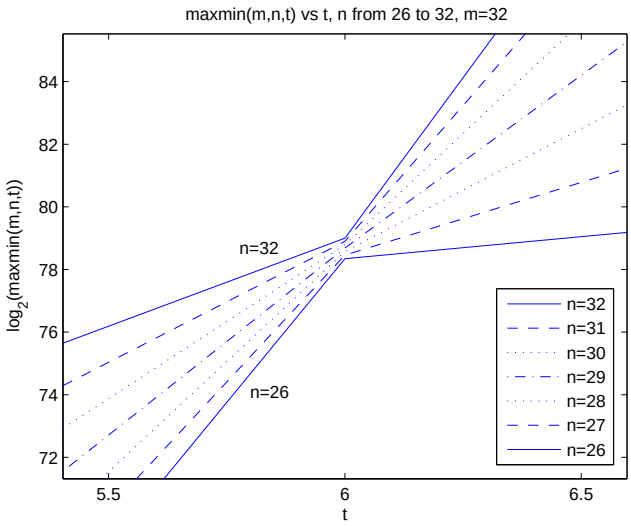

Figure 3.2: The guaranteed work factors maxmin(m, n, t) as a function of t, $m = 32$ and n varies from 26 to 32

where we can choose either a larger transmission rate (n large) or a smaller key size (n small) with roughly the same level of security. One of these points is magnified in Figure 3.2. In Figure 3.2, when $t = 6$, the values of maxmin$(32, n, 6)$ for n from 26 to 32 are quite close. At this tradeoff point, if $n = 26$, then the transmission rate is only approximately 54%, but the public key size is roughly $11,000$ bits; on the other hand, if $n = 32$, the transmission rate is increased to 62.5%, but the key size is nearly doubled to $20,000$ bits. Thus, the appropriate value for n should be chosen carefully. We also observe that, for some values of n in Figures 3.1 and 3.2, increasing t by 1, which leads to a smaller transmission rate, affects maxmin(m, n, t) very little. For example, the values of maxmin$(32, 26, 7)$ and maxmin$(32, 26, 6)$ are very close. Clearly, these cases (for example $(m, n, t) = (32, 26, 7)$) should be avoided.

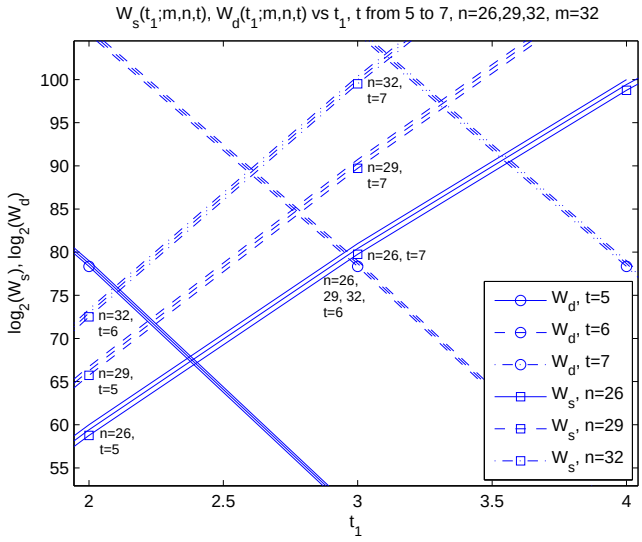

Figure 3.3: W_s and W_d and values of maxmin. The small circles or squares not only help to differentiate the curves, but also mark the locations of t_o.

The two observations above are due to the different properties of $W_d(t_1; m, n, t)$ and $W_s(t_1; m, n, t)$: $W_d(t_1; m, n, t)$ is sensitive to the parameter t but not to n, while $W_s(t_1; m, n, t)$ is sensitive to the parameter n but not to t. The work factors $W_d(t_1; 32, n, t)$ and $W_s(t_1; 32, n, t)$ for $n = 26, 29,$ and 32 and $t = 5, 6,$ and 7 are shown in Figure 3.3. Due to the above properties, in Figure 3.3 the 18 curves fall into 6 groups, whose legends are also given. The tradeoff point at $t = 6$ shown in Figure 3.2 occurs because when $t = 6$ and $26 \leq n \leq 32$, t_1^* falls between 2.5 and 3 and $t_o = 3$ for these cases. In these cases, $\text{maxmin}(32, n, 6) = W_d(3; 32, n, 6)$. Since $W_d(t_1; m, n, t)$ is not sensitive to n, $\text{maxmin}(32, n, 6)$ are roughly equal for $26 \leq n \leq 32$. Note that when $t = 6$ and $n = 26$, t_1^* is close to 3. Thus, $W_s(3; 32, 26, 6) \approx W_d(3; 32, 26, 6)$. Hence, $t_o = 3$ and $\text{maxmin}(32, 26, 6) = W_d(3; 32, 26, 6)$. When

$t = 7$ and $n = 26$, t_1^* is close to 3.5 and $t_o = 3$. In this case, $\text{maxmin}(32, 26, 7) = W_s(3; 32, 26, 7) \approx W_s(3; 32, 26, 6) \approx W_d(3; 32, 26, 6) = \text{maxmin}(32, 26, 6)$. Thus, when $n = 26$ there is little improvement in $\text{maxmin}(m, n, t)$ when t is increased from 6 to 7.

3.4 Conclusions

In this section, the optimal distortion parameter is defined to be the one that maximizes the work factor of two given structural and decoding attacks. That is, the GPT cryptosystem with the optimal distortion parameter achieves the largest work factor with respect to the two given attacks. The optimal distortion parameter also has useful properties under reasonable assumptions. It is then shown how to compute and approximate with low complexity the optimal parameter with respect to two well-known attacks. Finally, the behavior of the work factors achieved using the optimal distortion parameters is studied so as to facilitate the choices of the parameters for the GPT system.

Chapter 4

Private-key cryptosystems

The Gabidulin-Paramonov-Tretjakov (GPT) public-key cryptosystems, based on Gabidulin codes, seem to have some advantages over McEliece's public-key cryptosystems using Goppa codes. In this section, we introduce a private-key adaptation of the GPT cryptosystem. Two setups are proposed, one being the direct adaptation and the other using a permutation matrix, then their characteristics are studied. A new attack is given against these schemes, from which some values to avoid for the parameters of the system are found. Their robustness against other attacks is also shown. Finally, proof is given that they outperform the private-key adaptations of McEliece's system. This section expands the ideas introduced in [GY05b].

4.1 New private-key cryptosystems based on the rank metric

4.1.1 Description and considerations of the new systems

The GPT cryptosystem is now adapted to private-key ciphers. First, consideration is given to a direct adaptation of the GPT system. That is, the encryption and decryption matrices and the way of selecting the error vectors (all error vectors have a rank of $t - t_1$) remain the same. However, the encryption matrix $\mathbf{E}$ is not published.

An alternative is to use a permutation matrix in the encryption as in the Rao-Nam scheme. The encryption matrix then becomes $\mathbf{E} = \mathbf{SGP} + \mathbf{XP}$, where $\mathbf{P}$ is an $n \times n$ permutation matrix. Note that the introduction of the permutation $\mathbf{P}$ does not affect the definition of $\mathbf{E}$: the product $\mathbf{GP}$ is now a generator matrix of a different Gabidulin code [Loi01] with minimum rank distance d, and the $\mathbf{XP}$ matrix is still a valid distortion matrix with parameter t_1. As shown below, using a permutation matrix does not increase the level of security of the scheme against the attacks considered herein.

As stated in Section 2.2, an attack against a private-key cryptosystem consists of two steps. The goal of the first step is to discover the encryption matrix, and the second step is used to recover the key or the plaintext. Note that with the knowledge of the encryption matrix, the second step is similar to an attack against

the GPT cryptosystem.

There are two types of attacks against the GPT system: structural attacks and decoding attacks. Gibson [Gib96] discovered two structural attacks against the GPT scheme, and Chabaud and Stern [CS96] discovered an efficient decoding attack against the GPT cryptosystem. So far, both types of attacks against the GPT system have exponential complexities. In our paper [GY05a], a way to determine the optimal distortion parameter against both decoding and structural attacks was already given.

In the following, the focus is on the security against the first step of the attack. A discussion of the efficiency of two previously proposed attacks ensues, followed by the design of a new attack.

4.1.2 Previously proposed attacks

The first chosen-plaintext attack was based on majority voting, taking advantage of the low weight of the error vectors to estimate the rows of the encryption matrix. Here it is shown that this attack is not effective against the new cryptosystems.

Let $\mathbf{c}^1$ and $\mathbf{c}^2$ be two plaintexts differing in only one position, i.e., $\mathbf{c}^1 - \mathbf{c}^2 = \mathbf{u}^i$ for $1 \leq i \leq k$. The difference between their corresponding ciphertexts is given by $\mathbf{y}^1 - \mathbf{y}^2 = \mathbf{e}^i + (\mathbf{z}_1 - \mathbf{z}_2)\mathbf{P}$ where $\mathbf{e}^i$ is the i-th row of the encryption matrix $\mathbf{E}$. The Hamming weight of the difference $(\mathbf{z}_1 - \mathbf{z}_2)\mathbf{P}$ is not bounded. Hence, majority voting cannot be applied here and Oscar cannot derive a good estimate of $\mathbf{e}^i$. Note

that the permutation matrix $\mathbf{P}$ does not make a difference in the discussion above. Thus, the permutation does not enhance the security.

The second attack is the ST attack. This attack is applicable to the new system; therefore, its robustness against this attack will be studied in Section 4.2.

4.1.3 New chosen-plaintext attack against the systems

The set of error vectors of rank weight $t-t_1$ will be denoted as Z. That is, $Z = \{\mathbf{z} : rk(\mathbf{z}) = t - t_1\}$. Furthermore, the set of encryptions of one message $\mathbf{c}$, represented as $\mathbf{y}_j = \mathbf{cE} + \mathbf{z}_j\mathbf{P} = \mathbf{y} + \mathbf{z}_j\mathbf{P}$, will be written as Y. $N = |Z|$ is assumed not to be a multiple of the characteristic p of the field. More precisely, the remainder of the division of N by p is defined as $\nu \equiv N \bmod p$.

The idea behind this attack is that once all the elements in Y are recovered, Oscar can sum them up.

$$\begin{aligned}
\sum_{\mathbf{y}_j \in Y} \mathbf{y}_j &= \sum_{\mathbf{z}_j \in Z} (\mathbf{y} + \mathbf{z}_j\mathbf{P}) \\
&= \nu\mathbf{y} + \sum_{\mathbf{z}_j \in Z} \mathbf{z}_j\mathbf{P} && (4.1) \\
&= \nu\mathbf{y} + \sum_{\mathbf{z}_j \in Z} \mathbf{z}_j && (4.2) \\
&= \nu\mathbf{y}. && (4.3)
\end{aligned}$$

Equation (4.2) comes from the fact that a permutation does not change the rank of a vector. Note that if the conditions do not respect our assumption, i.e. if $\nu = 0$, the $\mathbf{y}$ term would disappear and the attack would not be applicable. Equation (4.3)

holds because, for any $t - t_1$, the sum of all vectors having a given rank $t - t_1$ is equal to $\mathbf{0}$. The vector $\mathbf{y}$ is thus recovered: $\mathbf{y} = \nu^{-1} \sum_{\mathbf{y}_j \in Y} \mathbf{y}_j$.

The process is then repeated for any $\mathbf{c}^i = \mathbf{c} + \mathbf{u}^i$. The set of ciphertexts obtained from $\mathbf{c}^i$: $\mathbf{y}_j^i = \mathbf{c}^i\mathbf{E} + \mathbf{z}_j\mathbf{P} = \mathbf{y}^i + \mathbf{z}_j\mathbf{P}$, where $\mathbf{y}^i = \mathbf{y} + \mathbf{e}^i$, will be defined as Y^i. If the sum is computed, it produces $\sum_{\mathbf{y}_j^i \in Y^i} \mathbf{y}_j^i = \sum_{\mathbf{z}_j \in Z} (\mathbf{y}^i + \mathbf{z}_j\mathbf{P}) = \nu\mathbf{y}^i + \sum_{\mathbf{z}_j \in Z} \mathbf{z}_j\mathbf{P} = \nu\mathbf{e}^i + \nu\mathbf{y}$. Hence $\mathbf{e}^i$ can be computed using $\mathbf{e}^i = \nu^{-1} \sum_{\mathbf{y}_j^i \in Y^i} \mathbf{y}_j^i - \mathbf{y}$. If this operation is performed for every i between 1 and k, then the encryption matrix $\mathbf{E}$ is recovered. The attack can be summarized as follows:

1. Encrypt a message $\mathbf{c}$ until all the N different cryptograms $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N$ are obtained.

2. Compute $\mathbf{y} = \nu^{-1} \sum_{\mathbf{y}_j \in Y} \mathbf{y}_j$.

3. For $1 \leq i \leq k$, select a message $\mathbf{c}^i$ such that $\mathbf{c}^i = \mathbf{c} + \mathbf{u}^i$. Repeat step 1 for $\mathbf{c}^i$.

4. For $1 \leq i \leq k$, recover the i-th row of the encryption matrix by computing $\mathbf{e}^i = \nu^{-1} \sum_{\mathbf{y}_j^i \in Y^i} \mathbf{y}_j^i - \mathbf{y}$.

It is noted that the attack can only proceed when $t - t_1 = 1$. Indeed, the number of vectors of length n and rank r over $\mathrm{GF}(q^m)$ is given by [Loi01]

$$R(r) = \prod_{i=0}^{r-1} \frac{(q^m - q^i)}{(q^r - q^i)} \cdot \prod_{i=0}^{r-1} (q^n - q^i). \tag{4.4}$$

If $t - t_1 = 1$, there are $R(1) = (q^m - 1)(q^n - 1)$ vectors with rank 1. This number is not a multiple of q, therefore in this case, $\nu \neq 0$, allowing Oscar to perform the

attack. On the other hand, when $t - t_1 > 1$, $R(t - t_1)$ becomes a multiple of q, and the attack cannot be performed.

In the discussion above, Z is the set of error vectors with rank weight $t - t_1$. If a permutation matrix is not used, then the attack can be generalized to other sets of error vectors. Let Z_0 be a subset of Z and let $\bar{Z}_0$ be its complement over Z, i.e. $Z = Z_0 \cup \bar{Z}_0$. Y_0 is defined as the set of encryptions of $\mathbf{c}$ using an error vector from Z_0. The attack detailed above is applicable for this new Z_0 if Z_0 verifies the two following properties: $N_0 = |Z_0|$ must not be a multiple of p, and given Z_0, the elements of Y_0 can be easily discriminated from the elements of Y .

4.2 Robustness of our scheme

Three different attacks in Section 4.1 were considered. Since only the ST attack can be performed for any parameter value, the work factors of the ST attack against the scheme are now evaluated.

In the evaluation below, it is supposed that the automorphism group of the graph Γ is restricted to the identity. Therefore, the worst-case situation is considered, and the lower bounds on the work factor of the attack are derived. The structure of the set of error vectors defined as $Z = \{\mathbf{z} : rk(\mathbf{z}) = t - t_1\}$ suggests that the study's assumption holds.

The work factor of the ST attack against the scheme is given by $O(m^2knN^2 \log N)$.

In particular, when $n = m$ the work factor becomes $W = m^3 k N^2 \log N$. First, suppose m and t are fixed. This leaves two possible values for k: $k_0 = m - 2t$ or $k_1 = m - 2t - 1$. Choosing k_0 over k_1 would slightly increase the work factor of the attack, and most importantly would enhance the transmission rate. Therefore, k should always satisfy $k = m - 2t$. Secondly, given that N increases with $t - t_1$, W can be optimized by setting $t_1 = 1$. It must be noted that this does not contradict our results from [GY05a] since we are now interested in the security against chosen-plaintext attacks. The values of W for m ranging from 5 to 15 are given in Table 4.1.

$m\backslash t$	2	3	4
5	30.091	0	0
6	36.243	0	0
7	41.767	63.78	0
8	46.976	73.639	0
9	51.99	82.953	108.19
10	56.867	92.004	121.86
11	61.642	100.89	135.03
12	66.338	109.67	147.97
13	70.971	118.36	160.78
14	75.551	126.99	173.48
15	80.087	135.56	186.11

Table 4.1: Work factor of the ST Attack ($\log_2$ scale)

Assume that an attack is feasible if its work factor is less than 2^{65}. In such a case, the desired level of security can be achieved even when small parameters are selected. As an illustration, $m = 8, t = 3$ gives a work factor of 2^{73}, which

exceeds the threshold of 2^{65}. However, the transmission rate using this setting is not sufficient. Using the new values $m = 12, t = 3, k = 6$ offers a sufficient level of security and an acceptable transmission rate. These results demonstrate that the ST attack can be defeated even when small parameters are used.

4.3 Number of error vectors

Struik and van Tilburg have found an efficient way of performing the first step of the chosen-plaintext attack. In his rebuttal, Rao [Rao87] indicates that Oscar needs to obtain the entire set of encryptions of a given plaintext to proceed with the attack. He then concludes that if the set of error vectors is large enough, the chosen-plaintext attack is infeasible. Discussion occurs over the number of error vectors that can be achieved using the new system.

The number of vectors over GF(q^m) with length n and rank r is given by Equation (4.4). If $n = m$, the number of vectors with rank r becomes

$$R(r) = \prod_{i=0}^{r-1} \frac{(q^n - q^i)^2}{(q^r - q^i)}.$$

Let $H(r)$ denote the number of vectors with Hamming weight r. For any $r < n$, the inequality $R(r) \geq H(r)$ holds, hence the rank metric produces a larger number of error vectors.

If, for example, $q = 2, n = m = 20, k = 14, t = 3, t_1 = 1$ are used, the number of available error vectors is approximately $N = 2^{77.4}$. In comparison, using a $(25, 20)$

Reed-Solomon code over GF(2^{16}) generates $N = 2^{40}$ error vectors. If the standard-array table technique is used, the number of error vectors jumps to $N = 2^{80}$, leading to a memory outage. Indeed, Alice and Bob need to store the entire list of possible error vectors, which by far exceeds any memory capacity available nowadays. Using the standard-array with large parameters brings a paradox, because it produces a large amount of data which has excessive storage requirements.

Let us assume that it takes 1μs to receive a message and compare it with the other received messages. Then, using our scheme and supposing that $n = m = 12, k = 6, t - t_1 = 2$ are chosen, more than 400 years are required to obtain all the necessary encryptions, which need a storage capability of 760 Tera bytes.

In conclusion, it can be argued that with the new cryptosystem, an extremely large number of error vectors can be generated, which protects the system from any chosen-plaintext attack.

4.4 Conclusion

Our system was shown to be secure against chosen-plaintext attacks, which are the most efficient attacks known against private-key cryptosystems using error-correcting codes. Our system outperforms the alternative private-key cryptosystems based on error-correcting codes. Indeed, our adaptation can produce a higher number of error vectors, using smaller keys than Denny's system. Moreover, unlike the Rao-Nam scheme using the standard-array lookup technique, our system does not

need to store the entire set of error vectors.

Chapter 5

Conclusion and discussions

In this thesis, the characteristics and performances of cryptosystems using error-correcting codes based on the rank metric were investigated. The analysis of the security of the GPT cryptosystem against structural and decoding attacks led to defining the optimal distortion parameter for which the overall security of the scheme is maximized. The value of the optimal distortion parameter given two well-known attacks was determined using an efficient algorithm. The behavior of the optimal level of security was also discussed. In addition to the study of the optimal distortion parameter, the GPT public-key cryptosystem was adapted into a private-key scheme. After a discussion of its security, this new system seems to outperform the private-key adaptations of McEliece's cryptosystem.

McEliece's system seems to be highly secure against structural attacks if Goppa codes are used as the key set, yet it seems weak against decoding attacks. Indeed,

the best attack known is nearly feasible. On the other hand, the GPT cryptosystem appears to be insecure against structural attacks: Overbeck [Ove05] recently designed a structural attack that seems capable of breaking the GPT system and many of its variants.

Nowadays, cryptosystems using error-correcting codes based on the rank metric are facing serious attacks. However, it seems that these systems can still be interesting alternatives to cryptosystems based on the Hamming metric. Indeed, the GPT cryptosystem is based on Gabidulin codes, which are the analogue to Reed-Solomon codes, which have been proved to be insecure in McEliece's case. We know that the structural security of McEliece's system relies on Goppa codes. One question arises: can structurally secure codes based on the rank metric be constructed? If so, then cryptosystems using structurally secure rank distance codes may be designed.

Regarding both private and public-key cryptosystems using error-correcting codes, it is too early to decide between the Hamming metric and the rank metric. Indeed, the rank metric seems more secure against decoding attacks, and it allows the use of small key sizes; however, structurally secure codes based on the rank metric are still unknown, and the cryptanalysis against cryptosystems based on the rank metric is not mature yet.

Appendix A

Proofs

A.1 Proof of convergence of the series u_i and l_i

Before proving that the series converge towards t_1^*, a lemma is shown.

Lemma 3 *For any* $a \geq b \geq \frac{1}{2\ln(q)}$, $\log(a) - \log(b) \leq 2(a-b)$.

Proof of Lemma 3

The function $f(x) = \log(x) - 2x$ has the following derivative

$$f'(x) = \frac{1}{x\ln(q)} - 2$$

which is negative if $x \geq \frac{1}{2\ln(q)}$. Hence, $\forall (a,b) \geq \frac{1}{2\ln(q)}, a \geq b$,

$$\begin{aligned} \log(a) - 2a &\leq \log(b) - 2b \\ \log(a) - \log(b) &\leq 2(a-b). \quad \blacksquare \end{aligned}$$

Proof of Convergence

Let the series x_i and y_i be defined by $x_{2p} = u_{2p}$, $x_{2p+1} = l_{2p+1}$ and $y_{2p} = l_{2p}$, $y_{2p+1} = u_{2p+1}$. x_i and y_i then satisfy $x_{i+1} = \bar{g}(x_i)$ and $y_{i+1} = \bar{g}(y_i)$.

These two series will converge if $\bar{g}$ is a contracting function on I, i.e., $\exists\, a \in [0,1[$ such that $\forall (c,d) \in I^2$, $|\bar{g}(c) - \bar{g}(d)| \leq a|c-d|$. Without loss of generality, let $c \leq d$. It comes that:

$$\begin{aligned}
\bar{g}(c) - \bar{g}(d) &= \frac{3}{m+n+1-2t} \log\left(\frac{n(t-c)+m}{n(t-d)+m}\right) \\
&\leq \frac{3}{m+3} \log\left(\frac{(t-c)+m/n}{(t-d)+m/n}\right) \\
&\leq \frac{3}{m+3} \log\left(\frac{t-c}{t-d}\right).
\end{aligned}$$

By definition, $(t-c)$ and $(t-d)$ are both greater than 1, therefore greater than $\frac{1}{2\ln(2)}$. Thus, by applying Lemma 3,

$$|\bar{g}(c) - \bar{g}(d)| \leq \frac{6}{m+3}|(t-c)-(t-d)| \leq \frac{6}{7}|c-d|$$

because $m \geq 4$. Consequently, $\bar{g}$ is contracting, hence x_i and y_i converge to one of the fixed points of $\bar{g}$. Yet $\bar{g}$ is monotonically decreasing, meaning it has only one fixed point, which is t_1^*. We conclude that x_i and y_i converge to t_1^* and, by construction, u_i and l_i also converge to t_1^*. ■

A.2 Proof of Proposition 4

We first show that $h(m,n,t)$ is an increasing function of t, and hence $h(m,n,t) \leq h(m,n,\frac{n}{2})$ since $t \leq n/2$. Furthermore, we can show that $h(m,n,\frac{n}{2})$ is an increasing

function of n. Thus, we have $h(m,n,t) \leq h\left(m,m,\frac{m}{2}\right) = \frac{3}{m+1}\log_2\left[\frac{m^2+8m+4}{8(m+1)}\right]$, which achieves a maximum of 0.2918 at $m = 9$. ■

A.3 Proof of Lemma 1

The behavior of $a(m,n,t)$ as a function of t depends on the value of n. Indeed, $a(m,n,t)$ is equal to zero when $n = m-1$; it is a positive increasing function of t when $n < m-1$; and it is a negative decreasing function of t when $n = m$. Thus the upper and lower bounds for $a(m,n,t)$ are given by $a(m, n < m-1, \frac{n}{2})$ and $a(m,m,\frac{m}{2})$ respectively. Let us consider

$$b(m,n) \stackrel{\text{def}}{=} a(m,n,\frac{n}{2}) = \frac{(m-n-1)(n-2)}{4(m+1)}.$$

When $n < m-1$, $b(m,n)$ reaches its maximum,

$$c_1(m) \stackrel{\text{def}}{=} \max_{n \leq m}\left(b(m,n)\right) = \frac{(m-3)^2}{16(m+1)},$$

when $n = \frac{m+1}{2}$. The minimum of $b(m,n)$, given by

$$c_2(m) \stackrel{\text{def}}{=} \min_{n \leq m}\left(b(m,n)\right) = -\frac{m-2}{4(m+1)},$$

is reached when $n = m$. ■

A.4 Proof of Lemma 2

As stated earlier,

$$0 < g(t-1) \leq g(t_1^*) \leq g(c_0) = g(t-1) + h(m,n,t).$$

A.4. PROOF OF LEMMA 2

We already showed that $h(m,n,t)$ was bounded by $\frac{3}{m+1}\log_2\left[\frac{m^2+8m+4}{8m+8}\right]$ for all m. Since $g(t-1) \leq \frac{3}{m+1}\log_2(2m)$, Lemma 2 follows naturally. ■

Bibliography

[Ber73] E.R. Berlekamp, "Goppa codes," *IEEE Transactions on Information Theory*, vol. 19, no. 5, pp. 590–592, Sept. 1973.

[BMvT78] E.R. Berlekamp, R.J. McEliece and H.C. van Tilborg, "On the inherent intractability of certain coding problems," *IEEE Transactions on Information Theory*, vol. 24, no. 3, pp. 384–386, May 1978.

[Ber97] T. Berson, "Failure of the McEliece Public-Key Cryptosystem Under Message-Resend and Related-Message Attack," *Proceedings of Advances in Cryptology–CRYPTO'97*,pp. 213–220, 1997.

[CC98] A. Canteaut and F. Chabaud, "A new algorithm for finding minimum-weight words in a linear code: Application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511," *IEEE Transactions on Information Theory*, vol. 44, no. 1, pp. 367–378, Jan. 1998.

[CS96] F. Chabaud and J. Stern, "The cryptographic security of the syndrome decoding problem for rank distance codes," *Proceedings of Advances in*

Cryptology–ASIACRYPT'96, LNCS, vol. 1163, pp. 368–381, 1996.

[Den88] W.F. Denny, "Encryptions using linear and non-linear codes: Implementation and security considerations," Ph.D. dissertation, The Center for Advanced Computer Studies, University of Southwestern Louisiana, Lafayette, 1988.

[DH76] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[Gab85] E.M. Gabidulin, "Theory of codes with maximum rank distance," *Problems on Information Transmission*, vol. 21, no. 1, pp. 1–12, Jan. 1985.

[Gab91] E.M. Gabidulin, "A fast matrix decoding algorithm for rank-error correcting codes," *Algebraic coding*, pp. 126–133, 1991.

[Gab95] E.M. Gabidulin, "Public-key cryptosystems based on linear codes over large alphabets: efficiency and weakness," *Codes and Cyphers*, P.G. Farrell, Ed., Formara Limited, Southend-on-sea Essex, pp. 17–31, 1995.

[GPT91] E.M. Gabidulin, A.V. Paramonov and O.V. Tretjakov, "Ideals over a non-commutative ring and their application in cryptology," *LNCS*, vol. 573, pp. 482–489, 1991.

[GY05a] M. Gadouleau and Z. Yan, "Optimal Distortion Parameter for the GPT Public-Key Cryptosystem," *Proceedings of the 2005 IEEE Sarnoff Symposium*, pp. 130–133, Princeton NJ, April 2005.

[GY05b] M. Gadouleau and Z. Yan, "A Private-Key Cryptosystem Based on the Rank Metric," to appear in *Proceedings of the 2005 Algebraic Methods in Cryptography Workshop*, Beijing China, July 2005.

[Gib95] J.K. Gibson, "Severely denting the Gabidulin version of the McEliece public-key cryptosystem," *Designs, Codes, and Cryptography*, vol. 6, pp. 37–45, 1995.

[Gib96] J.K. Gibson, "The security of the Gabidulin public-key cryptosystem," *Proceedings of Advances in Cryptology–EUROCRYPT'96*, pp. 212–223, 1996.

[Gop70] V.D. Goppa, "A new class of linear error-correcting codes," *Problemy Peredachi Informatsii*, vol. 6, no. 3, pp. 207–212, 1970.

[HGS99] C. Hall, I. Goldberg and B. Schneider, "Reaction Attacks Against Several Public-Key Cryptosystems," *Proceedings of Information and Communication Security (ICICS'99)*, pp. 2–12, 1999.

[Hin86] P.J.M. Hin, "Channel-Error-Correcting Privacy Cryptosystems," Thesis, Delft University of Technology, 1986. In Dutch.

[KI01] K. Kobara and H. Imai, "Semantically secure McEliece public-key cryptosystems - conversions for McEliece public-key cryptosystem," *Proceedings of the International Worshop on Practice and Theory in Public Key Cryptography*, 2001.

[LB88] P.J. Lee and E.F. Brickell, "An observation on the Security of a Public-Key Cryptosystem," *Proceedings of Advances in Cryptology–EUROCRYPT'88*, Davos, 1988.

[LN83] R. Lidl and H. Niederreiter, "Finite Fields," G.C. Rota, Ed., *Encyclopedia of Mathematics and its Apllications*, vol. 20, 1983.

[Loi01] P. Loidreau, "Étude et Optimisation de Cryptosystèmes à Clé Publique Fondés sur la Théorie des Codes Correcteurs," Ph.D. dissertation, École Polytechnique, Paris France, May 2001. In French.

[McE78] R.J. McEliece, "A public-key cryptosystem based on algebraic coding theory," *Jet Propulsion Lab. DSN Progress Report 42–44*, pp. 114–116, 1978.

[MRS00] C. Monico, J. Rosenthal and A. Shokrollahi, "Using Low Density Parity Check Codes in the McEliece Cryptosystem," *Proceedings of IEEE International Symposium on Information Theory*, Sorrento Italy, p. 215, June 2000.

[Ore33] Ö. Ore, "On a special class of polynomials," *Transactions of the American Mathematical Society*, vol. 35, pp. 559–584, 1933.

[Ore34] Ö. Ore, "Contribution to the theory of finite fields," *Transactions of the American Mathematical Society*, vol. 36, pp. 243–274, 1934.

[OJ02] A. V. Ourivski and T. Johansson, "New Technique for decoding codes in the rank metric and its cryptography applications," *Problems on Information Transmission*, vol. 38, no. 3, pp. 237–246, 2002.

[Ove05] R. Overbeck, "Extending Gibson's attacks on the GPT cryptosystem," *Proceedings of the 2005 International Workshop on Coding and Cryptography*, Bergen, Norway, March, 2005.

[Rao84] T.R.N. Rao, "Cryptosystems Using Algebraic Codes," *International Conference on Computer Science and Signal Processing*, Bangalore India, Dec. 1984.

[Rao87] T.R.N. Rao, "On Struik-Tilburg cryptanalysis of Rao-Nam scheme," *Proceedings of Advances in Cryptology–CRYPTO'87*, pp. 448–459, Santa Barbara CA, 1987.

[RN86] T.R.N. Rao and K.H. Nam, "Private-key algebraic cryptosystems," *Proceedings of Advances in Cryptology–CRYPTO'86*, pp. 458–461, 1986.

[RN89] T.R.N. Rao and K.H. Nam, "Private-key Algebraic-Code Encryptions," *IEEE Transactions on Information Theory*, vol. 35, no. 4, pp. 829–833, July 1989.

[Sen98] N. Sendrier, "On the concatenated structure of a linear code," *AAECC*, vol. 9, no. 3, pp. 221–242, 1998.

[SS92] V.M. Sidelnikov and S.O. Shestakov, "On cryptosystems based on generalized Reed-Solomon codes," *Discrete Mathematics and Applications*, vol. 2, no. 4, pp. 439–445, 1992.

[SvT87] R. Struik and J. van Tilburg, "The Rao-Nam scheme is insecure against a chosen-plaintext attack," *Proceedings of Advances in Cryptology–CRYPTO'87*, pp. 445–457, 1987.

[SKHN76] Y. Sugiyama, M. Kasahara, S. Hirasawa and T. Namekawa, "An Erasures-and-Errors Decoding Algorithm for Goppa Codes," *IEEE Transactions on Information Theory*, pp. 238–241, Mar. 1976.

Vita

Maximilien Rémy Gadouleau was born on December 28th, 1983, in Mont-Saint-Aignan, France. He is the son of Mr. Rémy Jean Gadouleau and Mrs. Viviane Simone Gadouleau, born Revert. He attended middle and high school in Institution du Sacré-Cœur, Rouen, France. He graduated from high school in June 1999, with honors. He then enrolled in École Supérieure d'Ingénieurs en Génie Électrique (ESIGELEC), Mont-Saint-Aignan, France. Since August 2003, he has been a Master of Science student in the Electrical and Computer Engineering department of Lehigh University, Bethlehem, Pennsylvania.