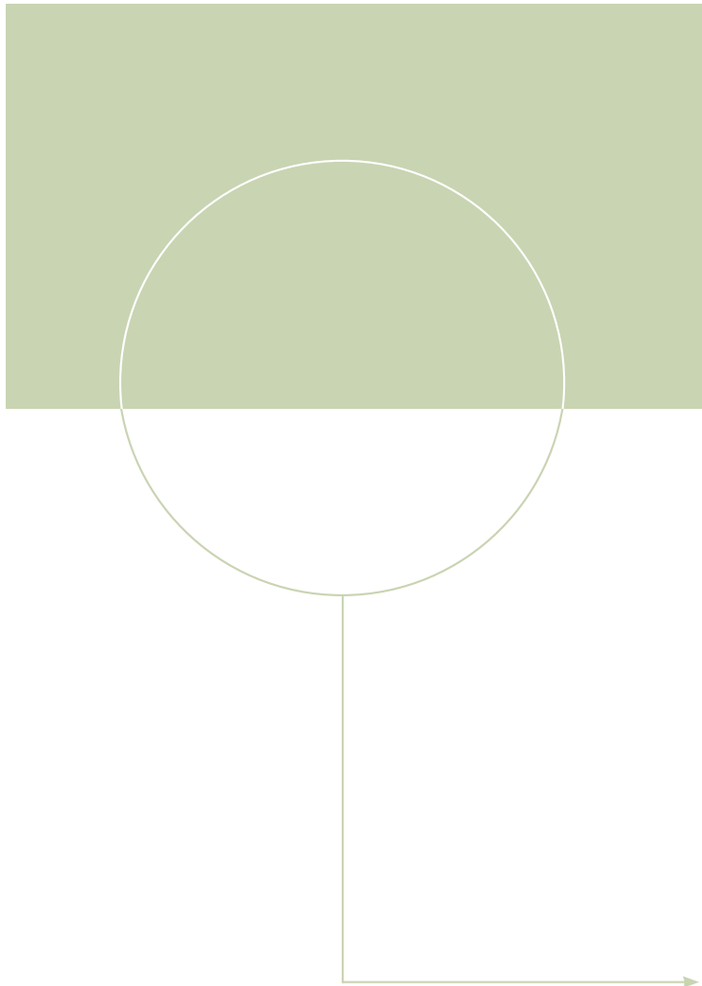

Master's thesis

2019

Simon Alexander Milne

NTNU
Norwegian University of
Science and Technology
Faculty of Information Technology and Electrical
Engineering
Department of Mathematical Sciences

Simon Alexander Milne

Multivariate Quadratic Cryptosystems

December 2019

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of
Science and Technology

Multivariate Quadratic Cryptosystems

Simon Alexander Milne

Master of Science
Submission date: December 2019
Supervisor: Kristian Gjøsteen, IMF

Norwegian University of Science and Technology
Department of Mathematical Sciences

Acknowledgments

This master's thesis is written during the autumn semester of 2019, as the final part of my time as a student at NTNU. This thesis has been both interesting and educational!

First I would like to thank my supervisor, Kristian Gjøsteen, for all the help and patience along the way, it has been very appreciated!

Thank you to my girlfriend, friends and family, for all the love and support during my studies.

A special thanks to my friend Audun, for being an amazing friend and study partner throughout our years as students, and my friend Oskar, for the help in making my years as a student in Trondheim as fun as possible.

Simon Alexander Milne
Trondheim, December 2019

Abstract

We describe the properties of multivariate system of polynomials, and both the properties of Gröbner bases, and how to calculate them. Furthermore, we describe two multivariate quadratic cryptosystems, the Matsumoto-Imai cryptosystem and the HFE cryptosystem. Finally, we describe two attacks against the Matsumoto-Imai cryptosystem, where the first attack is a direct attack based on Gröbner bases, and the second attack takes advantage of how the cryptosystem is constructed.

Sammendrag

Vi beskriver egenskapene til multivariate ligningssystemer, og både egenskapene til Grøbnerbaser, og hvordan man beregner dem. Videre beskriver vi to multivariate kvadratiske kryptosystemer, kryptosystemet Matsumoto-Imai og kryptosystemet HFE. Til slutt beskriver vi to angrep mot kryptosystemet Matsumoto-Imai, der det første angrepet er et direkte angrep basert på Grøbnerbaser, og det andre angrepet utnytter hvordan kryptosystemet er konstruert.

Table of Contents

Chapter 1

Introduction

In todays world cryptography plays an important role in the security of modern communication. As cryptography becomes more and more central in our modern world, we also need the cryptosystems we use to be as secure as possible, so further development of both our current and new cryptosystems are always in progress.

Today some of the most common cryptosystems in use are known as RSA and Diffie-Hellman, which both are public-key cryptosystems, and their security are based on mathematical problems that are so large that current computing technology cannot solve it in reasonable time. Even though our current computing technology cannot break these systems in reasonable time, quantum computers have been said to be able to do so. These quantum computers are not far away, and the introduction of them threathens the security of both the RSA and the Diffie-Hellman. That means we have to develop new cryptosystems, which will remain secure even against quantum computer attacks.

One such candidate for a new cryptosystem is called *multivariate quadratic cryptosystem*. This is a public-key cryptosystem based on multivariate quadratic polynomials over a field k, and is believed to be a good candidate for post-quantum cryptography, because of the difficulity in solving systems of multivariate quadratic polynomial equations. In this thesis we will look at some of the earliest attempts of these type of cryptosystems, along with a few attacks on one of them.

The thesis will be organized as follows: in Chapter 2, we give a quick overview of what a public-key cryptosystem is; in Chapter 3, we study the mathematical properties of multivariate systems of polynomial equations, and we study the so-called Gröbner bases, both the mathematical properties, and how to calculate them given a set of multivariate polynomial equations; in Chapter 4, we study the construction of both the Matsumoto-Imai system, and the HFE system, which are both multivariate quadratic cryptosystem; in Chapter 5, we look at two types of attacks against the Matsumoto-Imai system. The first attack uses Gröbner bases to directly attack the system of polynomials, which makes it possible to recover all of the plaintext for a given ciphertext. The second attack takes advantage of how the cryptosystem is constructed, which makes it possible to recover almost all of the plaintext for a given ciphertext.

Chapter 2

Background

2.1 Public-key cryptography

The classical cryptosystems, such as The Shift Cipher an The Substitution Cipher, are known as **symmetric-key cryptosystems**. Lets say that Alice and Bob are using such a cryptosystem. First they secretly choose a key K together, and from this K they get an encryption rule e_K and a decryption rule d_K. In these cryptosystems, either d_K is the same as e_K, or it is easy to derive from it. So an exposure of either e_K or d_K makes the cryptosystem insecure. One drawback with this system is that it is necessary for Alice and Bob to communicate the key K on a secure channel before they can use the system. So if Alice and Bob for example were to live far away from each other, it could be hard for them to find a secure channel to do this on.

So how can we bypass this? We introduce a new type of cryptosystems, called **assymetric cryptosystems**, or **public-key cryptosystems**. The main idea in this system is that it is supposed to be impossible to compute d_K given e_K. When that is the case, we call the enryption rule e_K the *public key*, which will be available for everyone to use. So now Alice, or anybody else, can send an encrypted message to Bob using the encryption rule e_K, without any prior communication. Bob will now be the only person who knows the decryption rule d_K, known as the *private key*, so only he can decrypt the received ciphertext. One way to imagine the system: Bob has a mailbox which can be locked with a combination lock. Now anybody, for example Alice, would be able to put something inside the mailbox, and then lock it with the combination lock. Then Bob is the only person who can open it, since he is the only one who knows the combination for the lock.

It is also important to notice that this type of cryptosystem is not necessarily fully secure. If Eve were to observe a ciphertext y sent from Alice to Bob, she could encrypt every possible plaintext by using the encryption key e_K, since it is a public key, until she finds the unique x such that $y = e_K(x)$. This x will be the decryption of y, so when studying public-key cryptosystems we study their computational security. We want the public encryption function e_K to be easy to compute, while computing the decryption should be hard for anyone else than Bob.[1]

Chapter 3

Multivariate system of polynomials

3.1 Multivariate system of polynomials

In a multivariate system of polynomials, we have a system consisting of n polynomials in some degree d, in m variables. In Chapter 4 we will look at some public-key cryptosystems where the public key contains a system of n multivariate polynomials of degree 2, in n variables. In this chapter we will look at some properties of general systems of multivariate polynomials. We will mainly follow the structure of [2], with some additions from [3].

3.1.1 Ideals and varieties

When we have a system of polynomials, we define the set of solutions to these polynomials in the following way.

Definition 3.1.1. Let k be a field, and $f_1, ..., f_n$ polynomials in $k[x_1, ..., x_n]$. Then we define

$$\mathbf{V}(f_1, ..., f_n) = \{(a_1, ..., a_s) \in k^n | f_i(a_1, ..., a_s) = 0 \text{ for all } 1 \leq i \leq n\}. \tag{3.1}$$

as the **affine variety** defined by $f_1, ..., f_n$.

This affine variety is the set of all solutions to a system of polynomials, such that $f_1(x_1, ..., x_n) = \cdots = f_n(x_1, ...x_n) = 0$.

Further we will look at how ideals relate to these affine varieties.

Definition 3.1.2. Let $f_1, ..., f_n$ be polynomials in $k[x_1, ..., x_n]$. Then we have

$$\langle f_1, ..., f_n \rangle = \{\sum_{i=1}^{n} h_i f_i | h_1, ..., h_n \in k[x_1, ..., x_n]\}, \tag{3.2}$$

where we note that $\langle f_1, ..., f_n \rangle$ is in fact an ideal.

If there now exist $f_1, ..., f_n \in k[x_1, ..., x_n]$ such that $\langle f_1, ..., f_n \rangle = I$, we say that I is finitely generated. Later, in Section 3.2.2, we will prove the fact that every given ideal in $k[x_1, ..., x_n]$ is finitely generated. We will also see later that even though a given ideal may have many different generating bases, we can choose one useful type of basis called a Gröbner basis.

One useful property we get from ideals that we will prove later, is that when we have a given set of polynomial equations, the variety only depends on the ideal they generate. When we have a basis that generates this ideal, we can also find the variety from this basis. We state the following proposition.

Proposition 3.1.1. Let $f_1, .., f_n$ and $g_1, ..., g_n$ be bases of the same ideal in $k[x_1, ..., x_n]$, such that $\langle f_1, ..., f_n \rangle = \langle g_1, ..., g_n \rangle$. Then $\mathbf{V}(f_1, ..., f_n)$=$\mathbf{V}(g_1, ..., g_n)$.

By this proposition we can determine the varieties by looking at *any* basis for a given ideal. We will later see that the mentioned Gröbner basis is a useful type of basis for determing varieties.

3.1.2 Greatest common divisor

In this section we will look at the use of finding the greatest common divisor of polynomials, and what advantages that gives us. We start by looking at the univariate case.

Proposition 3.1.2. Let k be a field, and g a nonzero polynomial in $k[x]$. Then any polynomial $f \in k[x]$ can be written as

$$f = qg + r, \tag{3.3}$$

where $q, r \in k[x]$, either $r = 0$ or $deg(r) < deg(g)$, and q, r are unique.

This is also known as the division algorithm, where we divide the polynomial f by g. Furthermore, we describe the structure of all ideals on $k[x]$ in the following corollary:

Corollary 3.1.1. For a field k we can write every ideal as $\langle f \rangle$, for any $f \in k[x]$. Now, f is unique up to multiplication by a nonzero constant in k.

This type of ideals, which are generated by one element, are called principal ideals. From Corollary 3.1.1, we say that $k[x]$ is a principal ideal domain, or PID for short. We say that the generator of an ideal $k[x]$ is the nonzero polynomial with the minimum degree contained in the ideal. So when the ideal is generated by one element then that is our generator, but it's not necessarily straightforward to find our generator if the ideal is generated by two or more elements. Then we would have to check the degrees of all the possible polynomials in the ideal, and there could be an infinite amount of polynomials. To make this calculation easier, we start by introducing a definition of the greatest common divisor of two polynomials.

Definition 3.1.3. A **greatest common divisor** of two polynomials $f, g \in k[x]$, is a polynomial h which satisfies

1. h divides f and g

2. If p is another polynomial which also divides f and g, then p divides h. When h has these properties, we write $h = gcd(f, g)$.

The gcd of polynomials also has some useful properties going forward.

Proposition 3.1.3. If we have $f, g \in k[x]$, then:

1. $gcd(f, g)$ exists, and is unique up to multiplication by a nonzero constant in k.
2. $gcd(f, g)$ is a generator of the ideal $\langle f, g\rangle$.

We illustrate the use of this with an example.

Example 3.1.1. What is the generator of the ideal $\langle x^5 - 1, x^3 - 1\rangle \subseteq k[x]$? We calculate the greatest common divisor of the two polynomials, by using the Euclidian algorithm:

$$\begin{aligned} x^5 - 1 &= x^2(x^3 - 1) + x^2 - 1 \\ x^3 - 1 &= x(x^2 - 1) + x - 1 \\ x^2 - 1 &= (x + 1)(x - 1) + 0 \end{aligned}$$

This means that we have

$$\begin{aligned} gcd(x^5 - 1, x^3 - 1) &= gcd(x^3 - 1, x^2 - 1) \\ &= gcd(x^2 - 1, x - 1) = gcd(x - 1, 0) = x - 1. \end{aligned}$$

Now, by using the result from Proposition 3.1.3, we get that the generator of this ideal is given by

$$\langle x^5 - 1, x^3 - 1\rangle = \langle x - 1\rangle.$$

The next question now would be, what do we do if we have an ideal generated by three or more polynomials? We expand the definition we wrote above, such that it includes any number of polynomials.

Definition 3.1.4. The **greatest common divisor** of polynomials $f_1, ..., f_n \in k[x]$ is a polynomial h such that

1. h divides every polynomial $f_1, ..., f_n$.
2. If p is a polynomial which divides $f_1, ...f_n$, then it also divides h.

With these properties, we write $h = gcd(f_1, ..., f_n)$.

This leads to some useful properties, which we give in the following proposition.

Proposition 3.1.4. Let $f_1, ..., f_n \in k[x]$ with $n \geq 2$. Then:

1. $gcd(f_1, ..., f_n)$ exists, and is unique up to multiplication of a nonzero constant in k.
2. $gcd(f_1, ..., f_n)$ is a generator of the ideal $\langle f_1, ..., f_n\rangle$.
3. for s ≥ 3, we have $gcd(f_1, ..., f_n) = gcd(f_1, gcd(f_2, ..., f_n))$.

3.1.3 Monomial ordering

When we are calculating Gröbner bases we will be reducing multivariate polynomials, and so it is important to be consistent in the use of pivoting terms during the calculation. For linear polynomials we choose the pivot to be term of greatest order, using the order $x_1 \succ x_2 \succ ... \succ x_n \succ 1$. In the case of polynomials with only one variable, we would order the pivoting terms by their degree, ${x_1}^d \succ {x_2}^{d-1} \succ ... \succ x^2 \succ x \succ 1$. This is necessary for the division progress to terminate. But how do we order the pivoting terms when we are dealing with more than one variable? We will in this section discuss some of the orderings we use when we are calculating Gröbner bases.

Definition 3.1.5.

1. A polynomial of the form $f = {x_1}^{\alpha_1}{x_2}^{\alpha_2} \cdots {x_n}^{\alpha_n}$ with $\alpha_1, \alpha_2, ..., \alpha_n \in \mathbb{N}$ is called a **monomial**.
2. The monomial ${x_1}^0{x_2}^0 \cdots {x_n}^0$ is usually written as 1.
3. All of the possible monomials in $k[x]$ are contained in the set T^n.
4. For $c \in k$, $f = c{x_1}^{\alpha_1} \cdots {x_n}^{\alpha_n}$ is called a **term**.
5. Every polynomial $f \in k[x]$ can be written as a sum of terms:
$$f = c_1t_1 + \cdots + c_mt_m, \tag{3.4}$$
with $m \in \mathbb{N}, c \neq 0$ and all the monomials t_i distinct from eachother.
6. The set $Supp(f) = \{t_1, ..., t_n\}$ is called the **support** of f. For a subset $F \subset k[x]$ of polynomials, we extend this definition to $Supp(F) = \cup_{f \in F} Supp(f)$.
7. For a monomial $t = {x_1}^{\alpha_1}{x_2}^{\alpha_2} \cdots {x_n}^{\alpha_n}$ with $\alpha_1, \alpha_2, ..., \alpha_n \in \mathbb{N}$ we have that $deg(t) = \alpha_1 + \alpha_2 + \cdots \alpha_n$ is the **degree** of the monomial t.

As we mentioned above, it's not necessarily obvious which ordering we should follow when we are working with polynomials with more than one variable. There are however some properties the ordering should follow, where for example Buchberger's algorithm requires that the set of monomials

$$T^n = \{{x_1}^{\alpha_1} \cdots {x_n}^{\alpha_1} | \alpha_1, ..., \alpha_n \in \mathbb{N}\} \tag{3.5}$$

follow an admissible ordering given in the following definition.

Definition 3.1.6. Let $\sigma \subseteq T^n \times T^n$ be a complete relationship defined on T^n. Instead of $(t_1, t_2) \in \sigma$ we write $t_1 \succ_\sigma t_2$, and an order is admissible if for every $t_i \in T^n$, the following holds:

1. $t_i \succ 1$ for all $t_i \neq 1$ and;
2. If $t_1 \succ t_2$, then for every monomial t_3, $t_1t_3 \succ t_2t_3$.

Now, there are several orderings possible, but there are only three orderings most commonly used, since they are most useful in practice. We present them in the next definitions, but it is helpful to first define the logarithm of a monomial as:

$$\log({x_1}^{\alpha_1} \cdots {x_n}^{\alpha_n}) = (\alpha_1, ..., \alpha_n). \tag{3.6}$$

Definition 3.1.7. For the **lexicographic** (lex) ordering we have

$$t_1 \succ_{lex} t_2 \tag{3.7}$$

for whenever the first non-zero component of $\log(t_1) - \log(t_2)$ is positive.

If we specify the polynomial ring as $k[x_1, x_2, ..., x_n]$, the ordering of variables

$$x_1 \succ_{lex} x_2 \succ_{lex} \cdots \succ_{lex} x_n \tag{3.8}$$

would be implied. The ordering is called lexicographic, since it is the order the terms would be given as in a dictionary.

Definition 3.1.8. For the **graded lexicographic** (glex) ordering, we have:

$$t_1 \succ_{glex} t_2 \tag{3.9}$$

if either $deg(t_1) \succ deg(t_2)$, or both $deg(t_1) = deg(t_2)$ and $t_1 \succ_{lex} t_2$.

In this ordering the monomials are first ordered by their total degree, and the monomials with same total degree are then ordered by the lexicographic ordering. This ordering is not used as much, since the next one often produces a Gröbner basis with fewer terms.

Definition 3.1.9. For the **graded reversed lexicographic** (grevlex) ordering, we have:

$$t_1 \succ_{grevlex} t_2 \tag{3.10}$$

if either $deg(t_1) \succ deg(t_2)$, or both $deg(t_1) = deg(t_2)$ and the last non-zero component of $\log(t_1) - \log(t_2)$ is negative.

Also here the monomials are first graded by their total degrees, and then they are ordered by the negation of the lexicographic ordering of the variables in reverse order. This ordering gives the same results as the graded lexicographic ordering when working with two variables, but they will differ for three or more variables.

We give an example to show how the orderings differ.

Example 3.1.2. Let $f(x_1, x_2, x_3) = (x_1 + x_2 + x_3)^3 + x_2$ be a polynomial in $k[x_1, x_2, x_3]$ with $k = GF(2)$ and $x_1 \succ_{lex} x_2 \succ_{lex} x_3$ implied. Then the three orderings gives us:

- Lexicographic order: $f(x_1, x_2, x_3) = {x_1}^3 + {x_1}^2 x_2 + {x_1}^2 x_3 + x_1 {x_2}^2 + x_1 {x_3}^2 + {x_2}^4 + {x_2}^3 + {x_2}^2 x_3 + x_2 {x_3}^2 + {x_3}^3$
- Graded lexicographic order: $f(x_1, x_2, x_3) = {x_2}^4 + {x_1}^3 + {x_1}^2 x_2 + {x_1}^2 x_3 + x_1 {x_2}^2 + x_1 {x_3}^2 + {x_2}^3 + {x_2}^2 x_3 + x_2 {x_3}^2 + {x_3}^3$

- Graded reverse lexicographic order: $f(x_1, x_2, x_3) = {x_2}^4 + {x_1}^3 + {x_1}^2x_2 + x_1{x_2}^2 + {x_2}^3 + {x_1}^2x_3 + {x_2}^2x_3 + x_1{x_3}^2 + x_2{x_3}^2 + {x_3}^3$

The last two orderings may look similar, but one notable difference is that in the graded lexicographic ordering, none of the last four monomials contains the variable x_1, while for the graded reverse lexicographic ordering the last four monomials are all divisible by the variable x_3. This makes a difference for both the properties and sizes for the corresponding Gröbner basis.

Furthermore, every polynomial in $k[x]$ contains a term that is maximal with respect to an ordering. We add the following definition.

Definition 3.1.10.
1. Given $f \in k[x]\backslash\{0\}$ with an admissible ordering σ on T^n, there exists a unique representation $f = \sum_{i=1}^{m} c_i t_i$ for f, with $m \in \mathbb{N}^+$, $c_i \in k\backslash\{0\}$, $t_i \in T^n$, and $t_1 \succ_\sigma \cdots \succ_\sigma t_m$
2. The **leading monomial** of f is t_1. This is the maximal monomial in $Supp(f)$ with respect to an ordering σ, which we denote $LM_\sigma(f)$, or for simplicity, $LM(f)$.
3. The **leading coefficient** is c_1, which we for simplicity denote $LC(f)$.
4. The **leading term** is c_1t_1, which we for simplicity denote $LT(f)$. This is the product of the leading coefficient and the leading monomial:

$$LT(f) = LC(f) \cdot LM(f). \tag{3.11}$$

3.1.4 Reduction of multivariate polynomials

Given a set F of polynomials, we can now use the ordering σ we defined above, and reduce one polynomial to another modulo F. Our idea is to expand our algorithm for reducing polynomials with one variable, such that we can divide $f \in k[x_1, ..., x_n]$ by $f_1, ..., f_n \in k[x_1, ..., x_n]$. Then we can express our polynomial f on the form

$$f = q_1f_1 + \cdots q_nf_n + r, \tag{3.12}$$

where both the q_i's and the remainder r lie in $k[x_1, ..., x_n]$. When characterizing the remainder here, we use one of the monomial orderings we discussed above. The algorithm here follows the same idea as in the one-variable case. We want to cancel the leading term of f, with respect to a term ordering, by multiplying some f_i by an appropriate monomial, and then subtract. This monomial will then be given as one of the q_i's in the equation above. We continue this process until we no longer can eliminate the leading term of f. We show this by an example.

Example 3.1.3. We have the polynomial $f = {x_1}^2x_2 + 1$ and want to divide it by $f_1 = x_1x_2 + 1$ and $f_2 = x_1 + 1$, by using the lex order $x_1 \succ x_2$. We start in the same way as for the one-variable case, so first we want to remove the leading term of f, namely $LT(f) = {x_1}^2x_2$. We see that both the leading term of f_1 and the leading term of f_2 divide $LT(f)$, but since f_1 is listed first, we begin by dividing f by f_1.

$$ {x_1}^2x_2 + 1 = x_1 \cdot (x_1x_2 + 1) + (-x_1 + 1).$$

Now we have the remainder $r = (-x_1 + 1)$, which we can reduce further, since $LT(f_2) = x_1$ divides the leading term of the remainder. We get

$$x_1{}^2x_2 + 1 = x_1 \cdot (x_1x_2 + 1) + (-1) \cdot (x_1 + 1) + 2.$$

We now have the remainder $r = 2$, and since none of the leading terms of f_1 and f_2 divides two, we cannot reduce f any further. So f divided by the polynomials f_1 and f_2 is given by

$$f = x_1 \cdot f_1 + (-1) \cdot f_2 + 2.$$

This example shows how the division algorithm works, but one could also meet another occurance when dealing with the algorithm. We illustrate it with another example.

Example 3.1.4. We have $f = x_1{}^2x_2 + x_1x_2{}^2 + x_2{}^2$, and want to divide it by $f_1 = x_1x_2 - 1$ and $f_2 = x_2{}^2 - 1$. We use the lex order $x_1 \succ x_2$, and the first steps proceed as they did in the previous example. First we divide by f_1 to remove $LT(f) = x_1{}^2x_2$, giving us a new leading term for f, which we then remove by dividing by f_1 again

$$\begin{aligned} x_1{}^2x_2 + x_1x_2{}^2 + x_2{}^2 &= x_1 \cdot (x_1x_2 - 1) + x_1x_2{}^2 + x_1 + x_2{}^2 \\ &= x_1 \cdot (x_1x_2 - 1) + x_2 \cdot (x_1x_2 - 1) + x_1 + x_2{}^2 + x_2 \end{aligned}$$

So now we have the remainder $r = x_1 + x_2{}^2 + x_2$, with $LT(r) = x_1$. But neither $LT(f_1) = x_1x_2$ or $LT(f_2) = x_2{}^2$ divides this leading term. However, this is not the final remainder, since $LT(f_2)$ divides $x_2{}^2$, a term in our remainder. So if we now move the term x_1 to the remainder, and let $x_2{}^2 + x_2$ be our intermediate divident, we can continue to divide by this. Every time the leading term of the intermediate divident cannot be reduced, we move it to the remainder, until our intermediate divident is zero. Now we have the leading term $x_2{}^2$ which we can further reduce with f_2

$$\begin{aligned} x_1{}^2x_2 + x_1x_2{}^2 + x_2{}^2 &= (x_1 + x_2) \cdot (x_1x_2 - 1) + (1) \cdot (x_2{}^2 - 1) + x_2 + 1 \qquad |x_1 \\ &= (x_1 + x_2) \cdot (x_1x_2 - 1) + (1) \cdot (x_2{}^2 - 1) \qquad |x_1 + x_2 + 1 \end{aligned}$$

Since $x_2 + 1$ couldn't be reduced any further, we moved it to our remainder. We now have our final remainder, $r = x_1 + x_2 + 1$, where none of the terms can be reduced further by f_1 or f_2. We can then write f as

$$f = (x_1 + x_2) \cdot f_1 + f_2 + x_1 + x_2 + 1$$

These examples illustrate how the division algorithm works. It also shows what kind of properties we want the remainder to have, that none of its terms can be reduced any further. It is also implied by this algorithm that if the remainder is equal to zero after dividing a polynomial f by $F = (f_1, ...f_n)$, then

$$f = q_1f_1 + \cdots q_nf_n \tag{3.13}$$

such that $f \in \langle f_1, ..., f_n \rangle$. Meaning we can use this algorithm to check wether any given $f \in k[x_1, ..., x_n]$ is in $\langle f_1, ..., f_n \rangle$, when $f_1, ..., f_n \in k[x_1, ..., x_n]$. But is this condition a necessity for being in the ideal? Unfortunately not, since the order of how we divide by the polynomials in the set matters.

Example 3.1.5. Let $f = x^2y - y$ be the polynomial we want to divide by $F = (f_1, f_2)$, when $f_1 = x_1x_2 - 1$, $f_2 = {x_1}^2 - 1 \in k[x_1, x_2]$, with lex ordering. If we follow that order, we get

$$ {x_1}^2x_2 - x_2 = x_1 \cdot (x_1x_2 - 1) + 0 \cdot ({x_1}^2 - 1) + (x_1 - x_2), \tag{3.14} $$

but if we now divide by $F = (f_2, f_1)$, we get

$$ {x_1}^2x_2 - x_2 = x_2 \cdot ({x_1}^2 - 1) + 0 \cdot (x_1x_2 - 1) + 0. \tag{3.15} $$

So the second calculation shows that $f \in \langle f_1, f_2 \rangle$, but then the first calculation shows that you don't need a remainder of zero for that to be true.

Looking back at the problem of finding the variety to a given set of polynomials $f_1, ..., f_n \in k[x_1, ..., x_n]$, we said that we can then turn to the ideal I they generate, which further means we can turn to any generating set for I. Does then a generating set where we easier can find the variety exist? With the use of this division algorithm, along with a few extra properties, we will see that we can create Gröbner bases we have mentioned earlier, where this is possible.

3.2 Gröbner basis

3.2.1 Monomial ideals

Before we describe the Gröbner bases, we need to describe a certain type of ideals, namely monomial ideals. We define them as the following:

Definition 3.2.1. An ideal $I \in k[x_1, ..., x_n]$ is called a **monomial ideal**, if there exists a subset $A \subseteq \mathbb{Z}^n$ (possibly infinite), such that I consists of all the polynomials which are finite sums of the form $\sum_{\alpha \in A} h_\alpha x^\alpha$, $h_\alpha \in k[x_1, ..., x_n]$. If such a subset exists, we have $I = \langle x^\alpha | \alpha \in A \rangle$.

Further we characterize all the monomials which lie in such a monomial ideal.

Lemma 3.2.1. Let $I = \langle x^\alpha | \alpha \in A \rangle$ be a monomial ideal. Then a monomial x^β lies in the ideal, if and only if for some $\alpha \in A$, x^β is divisible by x^α.

We now want to show that any polynomial f can be shown to be in a monomial ideal, by looking at the monomials of f.

Lemma 3.2.2. Let I be a monomial ideal, and $f \in k[x_1, ..., x_n]$. Then we have the following equivalences:

1. $f \in I$
2. Every term of f lies in I.
3. f is a k-linear combination of the monomials in I.

From part 3 here we know that a monomial ideal is uniquely determined by its monomials, which leads to the following corollary.

Corollary 3.2.1. Two monomial ideals are the same if and only if they contain the same monomials.

The most important part about monomial ideals that we need further on, is that every monomial ideal in $k[x_1, ..., x_n]$ is finitely generated. We get the following theorem, known as Dickson's lemma.

Theorem 3.2.1. Let $I = \langle x^\alpha | \alpha \in A \rangle \subseteq k[x_1, ..., x_n]$ be a monomial ideal. Then we can write I on the form $I = \langle x^{\alpha_1}, ..., x^{\alpha_s} \rangle$, where $\alpha_1, ..., \alpha_s \in A$. From this, I has a finite basis.

So now we have that every such ideal is finite, and we also know that for $I = \langle x^{\alpha_1}, ..., x^{\alpha_s} \rangle$, that a polynomial f is in I if and only if the remainder of f after dividing by $x^{\alpha_1}, ..., x^{\alpha_s}$ is zero.

3.2.2 Hilbert basis theorem

We are now closing in on the useful generating sets we have previously mentioned. One key feature we will use from now on, is that if we choose a monomial ordering, then each $f \in k[x_1, ..., x_n]$ will have a unique leading term $LT(f)$. We can then describe, for any ideal I, its ideal of leading terms as the following.

Definition 3.2.2. Let $I \subseteq k[x_1, ..., x_n]$ be an ideal, not $\{0\}$, and fix a monomial ordering on $k[x_1, ..., x_n]$. Then we have:

1. We denote the set of leading terms of nonzero elements in I by $LT(I)$, such that

$$LT(I) = \{cx^\alpha | \text{ there exists } f \in I \backslash \{0\} \text{ with } LT(f) = cx^\alpha\}. \tag{3.16}$$

2. We denote the ideal generated by the elements of $LT(I)$ as $\langle LT(I) \rangle$.

We know that the leading terms are important when we are using the division algorithm, so there is one thing worth noting when it comes to $\langle LT(I) \rangle$. If we have a finite generating set for an ideal I, $I = \langle f_1, ..., f_n \rangle$, then $\langle LT(I) \rangle$ and $\langle LT(f_1), ..., LT(f_n) \rangle$ is not necessarily the same ideal. By definition we have $LT(f_i) \in LT(I) \subseteq \langle LT(I) \rangle$, implying $\langle f_1, ..., f_n \rangle \subseteq \langle LT(I) \rangle$, but $\langle LT(I) \rangle$ can still be strictly larger than $\langle LT(f_1), ..., LT(f_n) \rangle$.

We will now show that $\langle LT(I) \rangle$ is a monomial ideal, which allow us to use the properties from above, namely that $\langle LT(I) \rangle$ is generated by a finitely many leading terms.

Proposition 3.2.1. Let $I \subseteq k[x_1, ..., x_n]$ be an ideal, different from $\{0\}$. Then

1. $\langle LT(I) \rangle$ is a monomial ideal.

2. There exists $g_1, ..., g_t \in I$ such that $\langle LT(I) \rangle = \langle LT(g_1), ..., LT(g_t) \rangle$.

Proof. 1. If we have elements $g \in I \backslash \{0\}$, then the leading monomials $LT(g)$ generate the monomial ideal $\langle LM(g) \rangle | g \in I \backslash \{0\}$. Now, $LM(g)$ and $LT(g)$ only differ by a nonzero constant, so we have $\langle LM(g) \rangle | g \in I \backslash \{0\} = \langle LT(I) \rangle$, and so $\langle LT(I) \rangle$ is a monomial ideal.

2. Now $\langle LT(I)\rangle$ is generated by the monomials $g \in I\backslash\{0\}$. From Theorem 3.2.1 we know that $\langle LT(I)\rangle = \langle LM(g_1), ..., LM(g_t)\rangle$ for finitely many $g_1, ..., g_t \in I$. Since $LM(g_i)$ and $LT(g_i)$ only differs in a nonzero constant, we have that $\langle LT(I)\rangle = \langle LT(g_1), ..., LT(g_t)\rangle$.

□

With Proposition 3.2.1 and the division algorithm, we can now prove that there exists a finite generating set for *every* polynomial ideal. The following theorem is known as Hilbert Basis Theorem.

Theorem 3.2.2. Every ideal $I \subseteq k[x_1, ..., x_n]$ has a finite generating set, meaning that $I = \langle g_1, ..., g_t\rangle$ for some $g_1, ..., g_t \in I$.

Proof. If $I = \{0\}$, then we can take $\{0\}$ as the generating set, which is finite. If I consists of some nonzero polynomial, we construct the generating set $g_1, ..., g_t$ for I in the following way. We start by choosing a monomial ordering on the set, and then compute the leading terms by using the division algorithm. Then I will have an ideal of leading terms $\langle LT(I)\rangle$. From Proposition 3.2.1 we then know that $\langle LT(I)\rangle = \langle g_1, ..., g_t\rangle$ for some $g_1, ..., g_t \in I$, and we claim that $I = \langle g_1, ..., g_t\rangle$. Since $g_i \in I$, it is clear that $\langle g_1, ..., g_t\rangle \subseteq I$. We need to prove that $I \subseteq \langle g_1, ..., g_t\rangle$. Let $f \in I$ be any polynomial. Using the division algorithm to divide f by $(g_1, ..., g_t)$, we get an expression on the form

$$f = q_1 g_1 + \cdots + q_t g_t + r, \tag{3.17}$$

where the terms of r are not divisible by any of the $LT(g_1), ..., LT(g_t)$. We claim that $r = 0$, and notice that

$$r = f - q_1 g_1 - \cdots q_t g_t \in I. \tag{3.18}$$

If we now have $r \neq 0$, then $LT(r) \in \langle LT(I)\rangle = \langle LT(g_1), ..., LT(g_t)\rangle$, and thus by Lemma 3.2.1 $LT(r)$ must be divisible by some $LT(g_i)$. This contradicts the meaning of a remainder, so $r = 0$, and we have

$$f = q_1 g_1 + \cdots q_t g_t + 0 \in \langle g_1, ..., g_t\rangle, \tag{3.19}$$

meaning that $I \subseteq \langle g_1, ..., g_t\rangle$. □

3.2.3 Gröbner bases

We are now ready to define the Gröbner bases we have previously mentioned. We give the following definition.

Definition 3.2.3. Fix a monomial ordering on $k[x_1, ..., x_n]$. Then a finite subset $G = \{g_1, ..., g_t\}$ of an ideal $I \subseteq k[x_1, ..., x_n]$, not $\{0\}$, is called a **Gröbner basis** if

$$\langle LT(g_1), ..., LT(g_t)\rangle = \langle LT(I)\rangle. \tag{3.20}$$

Using $\langle\emptyset\rangle = \{0\}$, we say that $\emptyset$ is the Gröbner basis for the zero ideal $\{0\}$.

Equivalently, if we have a set $(g_1, ..., g_t) \subseteq I$, then this set is a Gröbner basis if and only if every leading term of I is divisible by some $LT(g_i)$.

Corollary 3.2.2. Fix a monomial ordering. Then every $I \subset k[x_1, ..., x_n]$ has a Gröbner basis, and every Gröbner basis of an ideal I is a basis of I.

Proof. For a nonzero ideal, the set $G = (g_1, .., g_t)$ we constructed in Theorem 3.2.2 is a Gröbner basis by definition. To prove that every Gröbner basis is a basis, we note that if $\langle LT(I)\rangle = \langle LT(g_1), ..., LT(g_t)\rangle$, then we have from Theorem 3.2.2 that $I = \langle g_1, ..., g_t\rangle$, such that G is a basis for I. □

We end this section with a consequence of Theorem 3.2.2, the Hilbert Basis Theorem. Up until now we have used affine varieties as the solutions to a specific finite set of polynomials:

$$\mathbf{V}(f_1, ..., f_n) = \{(a_1, ..., a_s) \in k^n | f_i(a_1, ..., a_s) = 0, \forall i\}. \tag{3.21}$$

We have previously said that we can turn to the ideal $I \in k[x_1, ..., x_n]$ they generate for finding this affine variety, which we will now prove.

Definition 3.2.4. Let $I \in k[x_1, ..., x_n]$ be an ideal. We denote $\mathbf{V}(I)$ as the set

$$\mathbf{V}(I) = \{(a_1, ..., a_s) \in k^n | f(a_1, ..., a_s) = 0 \text{ for all } f \in I\}. \tag{3.22}$$

A nonzero ideal I will always contain infinitely many different polynomials, but $\mathbf{V}(I)$ can still be defined as a finite set of polynomial equations.

Proposition 3.2.2. Let $\mathbf{V}(I)$ be an affine variety. Now, if $I = \langle f_1, ..., f_n\rangle$, we have $\mathbf{V}(I) = \mathbf{V}(f_1, ..., f_n)$.

Proof. By the Hilbert Basis Theorem, $I = \langle f_1, ..., f_n\rangle$ for a finite generating set. We claim that $\mathbf{V}(I) = \mathbf{V}(f_1, ..., f_n)$. Now if we have $f_i \in I$, then $f_i(a_1, ..., a_s) = 0$, since $f(a_1, ..., a_s) = 0$ for any $f \in I$. Thus $\mathbf{V}(I) \subseteq \mathbf{V}(f_1, ..., f_n)$. Now let $(a_1, ..., a_s) \in \mathbf{V}(f_1, ..., f_n)$ and $f \in I$. Since $I = \langle f_1, ..., f_n\rangle$, we have

$$f = \sum_{i=1}^{n} h_i f_i \tag{3.23}$$

for some $h_i \in k[x_1, ..., x_n]$. Now

$$f(a_1, ..., a_s) = \sum_{i=1}^{n} h_i(a_1, ..., a_s) f_i(a_1, ..., a_s) \tag{3.24}$$

$$= \sum_{i=1}^{n} h_i(a_1, ..., a_s) \cdot 0 = 0. \tag{3.25}$$

So $\mathbf{V}(f_1, ..., f_n) \subseteq \mathbf{V}(I)$, and we have $\mathbf{V}(I) = \mathbf{V}(f_1, ..., f_n)$. □

The most important thing we notice from this proposition is that varieties are determined by ideals, which also means that we can turn to any basis for a given ideal, to find the variety.

3.2.4 Properties of Gröbner bases

We now know that every nonzero ideal $I \subseteq k[x_1, ..., x_n]$ has a Gröbner basis. In this section we will look closer at some of the properties of these Gröbner bases. We start by proving that the remainder after dividing by a Gröbner basis is unique.

Proposition 3.2.3. Let $I \subset k[x_1, ..., x_n]$ be an ideal and $G = \{g_1, ..., g_t\}$ a Gröbner basis for this ideal. For a given $f \in k[x_1, ..., x_n]$ there exists an $r \in k[x_1, ..., x_n]$ that have these two properties:

1. No term of r is divisible by any of $LT(g_1), ..., LT(g_t)$.

2. There is $g \in I$ such that $f = g + r$.

Also, r is here the remainder after dividing f by G, and it does not matter how the elements are listed in G when using the division algorithm.

Proof. The division algorithm gives us $f = q_1g_1 + \cdots \cdot q_tg_t + r$, where r satisfies 1. If we now set $g = q_1g_1 + \cdots \cdot q_tg_t \in I$, we satisfy 2 as well. So there exists an r, and we need to prove that it is unique. Suppose that $f = g + r = g' + r'$ satisfy both 1 and 2. Then $r - r' = g' - g \in I$. If we assume $r \neq r'$, then $LT(r - r') \in \langle LT(I)\rangle = \langle LT(g_1), ..., LT(g_t)\rangle$. Then, from Lemma 3.2.1 we have that $LT(r-r')$ must be divisible by some $LT(g_i)$. But by 1 in the proposition, none of the terms of r, r' are divisible by one of $LT(g_1), ..., LT(g_t)$, so we must have $r = r'$, and thus uniqueness is proved. This uniqueness of r also proves the last part of the proposition. □

From this proposition we can also add a corollary, which tells us when a given polynomial f lies in an idea.

Corollary 3.2.3. Let $I \subseteq k[x_1, ..., x_n]$ be an ideal and $G = \{g_1, ..., g_t\}$ its Gröbner basis. Then a polynomial $f \in k[x_1, ..., x_n]$ lies in I if and only if

$$f \in I \iff f \mapsto_G 0, \tag{3.26}$$

where $\mapsto_G$ denotes dividing by the basis G.

Proof. If the remainder is zero, we have already seen that $f \in I$. Now if $f \in I$, then $f = f + 0$ satisfies both of the conditions in Proposition 3.2.3, and so 0 is the remainder of f after dividing by G. □

Generally an arbitrary basis B does not form a Gröbner basis, since if some polynomial in B reduced to a nonzero irreducible polynomial modulo B, then that polynomial would have to be added to B in order to complete the basis. But since every polynomial in the basis has to be reduced to zero with respect to the basis, for it to be a Gröbner basis, the process of finding this basis may be very time inefficient. However, Buchberger showed that it is enough to consider the $S - polynomials$ for finding Gröbner bases, which are given by the following definition.

Definition 3.2.5. The S-polynomial of two polynomials f and g in B is given by:

$$spoly(f,g) = \frac{M}{LT(f)} \cdot f - \frac{M}{LT(g)} \cdot g \tag{3.27}$$

where $M = lcm(LT(f), LT(g))$.

We give an example of how we compute the S-polynomials.

Example 3.2.1. Let $f = 2{x_1}^4x_2 + {x_1}^2{x_2}^3 + x_1x_2$ and $g = 4{x_1}^3{x_2}^2 - {x_1}^2{x_2}^2$ be polynomials in $\mathbb{R}[x_1, x_2]$ with grlex ordering. So $LT(f) = 2{x_1}^4x_2$, $LT(g) = 4{x_1}^3{x_2}^2$ and $M = lcm(LT(f), LT(g)) = 4{x_1}^4{x_2}^2$. Then the S-polynomial is

$$\begin{aligned} S(f,g) &= \frac{4{x_1}^4{x_2}^2}{2{x_1}^4x_2} \cdot f - \frac{4{x_1}^4{x_2}^2}{4{x_1}^3{x_2}^2} \cdot g \\ &= 2x_2 \cdot f - x_1 \cdot g \\ &= {x_1}^3{x_2}^2 + 2{x_1}^2{x_2}^4 + 2x_1{x_2}^2. \end{aligned}$$

With Definition 3.2.5 we can now write the following theorem for when a basis is a Gröbner basis, which is also known as Buchberger's criterion.

Theorem 3.2.3. A basis $G = \{g_1, ..., g_t\}$ is a Gröbner basis if and only if for every pair $i \neq j$

$$spoly(g_i, g_j) \mapsto_G 0, \tag{3.28}$$

for all $g_i, g_j \in G$

So if we now compute the S-polynomial of two polynomials in a given basis G, and that reduces to a nonzero polynomial which can't be reduced any further modulo G, then that polynomial would have to be added to the basis. When every S-polynomial of every pair of polynomials in the basis is reduced to zero modulo G, we have computed our Gröbner basis. In Chapter 4 we will describe one multivariate quadratic cryptosystem called Matsumoto-Imai, and in Chapter 5 we will describe an attack on this system with the use of a Gröbner basis.

We have seen that a Gröbner basis can be found by computing the S-polynomial of every pair of polynomials in a given set of polynomials. However, if the given set contains a large number of polynomials and variables, the process of calculating the S-polynomials can be extremely time consuming. Therefore there has been several computer algorithms developed over the years since the introduction of Gröbner bases, such as Buchberger's algorithm, which one can find an example of in [2]. We will not go into the details of these algorithms in this paper, but they are highly necessary when computing Gröbner bases.

Chapter 4

Multivariate quadratic cryptosystems

We will in this chapter present two multivariate quadratic cryptosystems, and we will mainly follow the structure of [3].

4.1 Matsumoto-Imai

4.1.1 Construction

Let k be a finite field of characteristic two and let $q = 2^m$ be the number of elements in k. If we now let $g(x) \in k[x]$ be any irreducible polynomial of degree n, then we can identify the field $K \cong k[x]/g(x)$ as the extension field of degree n over k. Let $\phi : K \to k^n$ be the standard k-linear isomorphism between K and k^n given by

$$\phi(a_0 + a_1 x + ... + a_{n-1}x^{n-1}) = (a_0 + a_1 + ... + a_{n-1}). \tag{4.1}$$

Now k is a subfield of K, and is embedded in k^n in the standard way:

$$\phi(a) = (a, 0, ..., 0) \qquad \forall a \in k \tag{4.2}$$

Further, we now choose a variable θ such that both $0 < \theta < n$, and

$$gcd(2^{m\theta} + 1, 2^{mn} - 1) = 1, \tag{4.3}$$

and define the map $\tilde{F}$ over K to be

$$\tilde{F}(X) = X^{1+2^{m\theta}}. \tag{4.4}$$

and with the conditions we had on the variable θ we get that $\tilde{F}$ is an invertible map, such that we can easily find the inverse map. If we let $\hbar$ be an integer such that

$$\hbar(1 + 2^{m\theta}) \equiv 1 \bmod (2^{mn} - 1), \tag{4.5}$$

then we have that the inverse map $\tilde{F}^{-1}$ is given by

$$\tilde{F}^{-1}(X) = X^{\hbar}. \tag{4.6}$$

Furthermore, we define the map $\bar{F}$ over k^n to be

$$\bar{F}(x_1, ..., x_n) = \phi \circ \tilde{F} \circ \phi^{-1}(x_1, ..., x_n) = \bar{f}_1(x_1, ..., x_n), ..., \bar{f}_n(x_1, ..., x_n), \tag{4.7}$$

where $\bar{f}_1(x_1, ..., x_n), ..., \bar{f}_n(x_1, ..., x_n) \in k[x_1, ..., x_n]$. Finally, to fully describe the MI public key cryptosystem, we choose T and S to be two invertible affine transformations over k^n, such that the map over k^n is defined as

$$F(x_1, ..., x_n) = (T \circ \bar{F} \circ S)(x_1, ..., x_n) = f_1(x_1, ..., x_n), ..., f_n(x_1, ..., x_n), \tag{4.8}$$

where $f_1(x_1, ..., x_n), ..., f_n(x_1, ..., x_n) \in k[x_1, ..., x_n]$ are quadratic polynomials. This is because in the map $\tilde{F}(X) = X \cdot X^{2^{m\theta}}$, both $X \mapsto X$ and $X \mapsto X^{2^{m\theta}}$ (because $X \mapsto X^2$ is k-linear) are k-linear transformations, such that $X \mapsto X \cdot X^{2^{m\theta}}$ is quadratic over k. These polynomials are now the public polynomials, which anyone can use to encrypt messages.

We can now fully describe the cryptosystem.

Public key

The public key for the MI cryptosystem contains:

1. The field k with its multiplicative and additive strucure;
2. The n polynomials $f_1(x_1, ..., x_n), ..., f_n(x_1, ..., x_n) \in k[x_1, ..., x_n]$ of degree 2.

Private key

The private key contains the two invertible affine transformations S and T. It is also possible to include the variable θ in the private key, but since it is lesser than n choices for θ and n is rarely quite large, it does not make any attack significantly harder by keeping it a secret.

Encryption

Given a plaintext $(x_1, ..., x_n)$, we have the associated ciphertext $(y_1, ..., y_n)$, where

$$y_i = f_i(x_1, ..., x_n), \tag{4.9}$$

for $i = 1, .., n$. Since the public key is available to everyone, anyone can encrypt a message.

Decryption

When we have a ciphertext $(y_1, ..., y_n)$, we can decrypt it by computing

$$\begin{aligned} \bar{F}^{-1}(y_1, ..., y_n) &= (S^{-1} \circ F^{-1} \circ T^{-1})(y_1, ..., y_n) \\ &= (S^{-1} \circ \phi \circ \tilde{F}^{-1} \circ \phi^{-1} \circ T^{-1})(y_1, ..., y_n). \end{aligned}$$

We can decrypt a ciphermessage by doing a step-by-step computation in the following way:

1. Start by computing $(z_1, ..., z_n) = T^{-1}(y_1, ..., y_n)$;
2. Then compute $(\bar{z}_1, ..., \bar{z}_n) = (\phi \circ \tilde{F}^{-1} \circ \phi^{-1})(z_1, ..., z_n)$;
3. And finally compute $(x_1, ..., x_n) = S^{-1}(\bar{z}_1, ..., \bar{z}_n)$.

An overview of the construction is given in figure 4.1. One can see that that deciphering is straightforward as long as you have the private key, as it should be. In chapter (5) we will look at some ways to attack this cryptosystem. One of the attacks focuses on the public polynomials and their corresponding ciphertext, while the other attack takes advantage of how the map $\tilde{F}$ is constructed. The second attack led Patarin to propose a new cryptosystem inspired by the MI-cryptosystem, which we will look at in the next section.

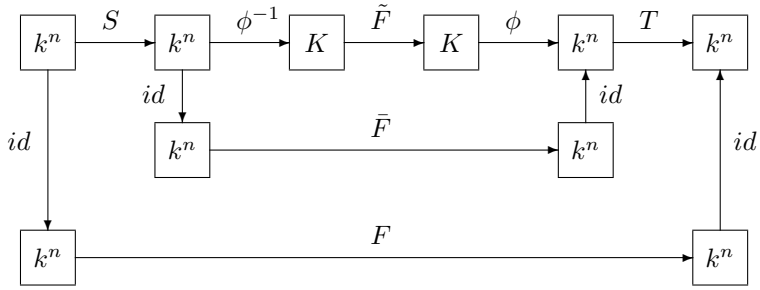

Figure 4.1: Construction of the MI-cryptosystem

4.2 Hidden field equations

In 1996 Patarin proposed a new cryptosystem called the Hidden Field Equations cryptosystem, or HFE for short. It is a further development of the Matsumoto-Imai cryptosystem we saw above, where the difference is that he wanted to replace the map $\tilde{F}$ with a new map that would make the system more secure.

4.2.1 Construction

The construction of HFE is very similar to the Matsumoto-Imai cryptosystem. We have that k is a finite field with q elements, and $g(x) \in k[x]$ an irreducible polynomial of degree n. Then we can identify $K \cong k[x]/g(x)$ as a degree n field extension of k. Also, k does not need to have characteristic two here. Also here ϕ is the standard k-linear map that identifies K with k^n, $\phi : K \to k^n$, where

$$\phi(a_0 + a_1 x + ... + a_{n-1}x^{n-1}) = (a_0 + a_1 + ... + a_{n-1}). \tag{4.10}$$

As we mentioned above, the construction is very similiar to the MI-system, but there is one main difference. In the MI-cryptosystem we described above, we used the map $\tilde{F}(X) = X^{1+q^\theta}$, but in the HFE-cryptosystem we replace it with the following map

$$\tilde{F}(X) = \sum_{i=0}^{r_2-1} \sum_{j=0}^{i} a_{ij} X^{q^i+q^j} + \sum_{i=0}^{r_1-1} b_i X^{q^i} + c, \tag{4.11}$$

where we choose the coefficients $a_{ij}, b_i, c \in K$ randomly, and we choose r_1, r_2 such that the degree of $\tilde{F}(X)$ is less than some parameter d. The public key polynomials are now given by

$$F(x_1, ..., x_n) = (T \circ \bar{F} \circ S)(x_1, ..., x_n) = f_1(x_1, ..., x_n), ..., f_n(x_1, ..., x_n), \tag{4.12}$$

where as in the previous section $\bar{F} = \phi \circ \tilde{F} \circ \phi^{-1}$, and T, S are the secret invertible affine transformations on k^n. As in the MI construction, the polynomials $f_1(x_1, ..., x_n), ..., f_n(x_1, ..., x_n) \in k[x_1, ..., x_n]$ are here quadratic, since both $X \mapsto X^{q^i}$ and $X \mapsto X^{q^j}$ are k-linear transformations, such that $X \mapsto X^{q^i} \cdot X^{q^j}$ is quadratic over k.

Public key

The public key consists of

1. The field k, with its multiplicative and additive structure;
2. The n polynomials $f_1(x_1, ..., x_n), ..., f_n(x_1, ..., x_n) \in k[x_1, ..., x_n]$ of total degree 2.

Private key

The private key consists of

1. The map $\tilde{F}$;
2. The two invertible affine transformations T and S.

In the construction of the MI-cryptosystem we noticed that if we kept the parameter θ secret, we could still find the map $\tilde{F}$ by guessing. Here, it is nearly impossible to guess the map.

Encryption

A given plaintext message $(x_1, ..., x_n)$ corresponds to a ciphertext in the following way:

$$(y_1, ..., y_n) = F(x_1, ..., x_n), \tag{4.13}$$

or equivalently

$$y_i = f_i(x_1, ..., x_n) \qquad \text{for } i = 1, ..., n. \tag{4.14}$$

Decryption

Given a ciphertext $(y_1, ..., y_n)$, we can decrypt it in the following way:

1. Compute $(\bar{y}_1, ..., \bar{y}_n) = T^{-1}(y_1, ..., y_n)$.
2. Let $Y = \phi^{-1}(\bar{y}_1, ..., \bar{y}_n)$. Compute the set

   $$\mathcal{Z} = \{Z \in K | \tilde{F}(Z) = Y\} \tag{4.15}$$

   We can compute this $\mathcal{Z}$ by using a variant of the Berlekamp algorithm, suitable for use over the field K. If $d = deg\tilde{F}(X)$, then the complexity of this step is $O(nd^2 \log d + d^3)$. From this we see that $\tilde{F}$ cannot have too large degree, since the decryption process would become very inefficient, thus making the cryptosystem itself inefficient. Equivalently, we cannot choose r_1, r_2 to be too large.
3. For each element $Z_i \in \mathcal{Z}$, compute

   $$(x_{i1}, ..., x_{in}) = {L_2}^{-1} \circ \phi(Z_i) \tag{4.16}$$

   Optimally, we would like that the map $\tilde{F}$ is a one-to-one map as it were in the MI-cryptosystem, such that one would get unique solutions for the elements in $\mathcal{Z}$. It is however possible that $\mathcal{Z}$ contains multiple solutions, and in that case we may have to search a while to find the corresponding plaintext $(x_1, ..., x_n)$ to the ciphertext $y_1, ..., y_n)$.

Chapter 5

Attacking multivariate quadratic cryptosystems

5.1 Gröbner basis attack

In this section we will look at how we can attack the MI-cryptosystem by using a Gröbner basis, which we discussed in Chapter 3. This attack only needs the public polynomials and the ciphertext that was sent, such that we do not really need to know how the MI-cryptosystem is constructed. So we have the system of polynomials and the associated ciphertext such that

$$\begin{aligned} f_1(x_1, ..., x_n) &= y_1 \\ &\vdots \\ f_n(x_1, ..., x_n) &= y_n. \end{aligned}$$

Since the values for $y_1, ..., y_n$ are known, we can move them over to the left hand side, such that we get the system

$$\begin{aligned} f_1(x_1, ..., x_n) - y_1 &= 0 \\ &\vdots \\ f_n(x_1, ..., x_n) - y_n &= 0. \end{aligned}$$

In this system we now have n polynomials in n variables, so we can find the solution by Gaussian elimination, but for a secure MI-cryptosystem this would be very time consuming. Therefore we try to solve the system using a Gröbner basis instead. From chapter 3 we know that $\mathbf{V}(f_1(x_1, ..., x_n) - y_1, ..., f_n(x_1, ..., x_n) - y_n)$ will give us the solutions for all the unknown variables in our system. We also know that if we have an ideal $I = \langle g_1, ..., g_n \rangle$, then $\mathbf{V}(I) = \mathbf{V}(g_1, ..., g_n)$, meaning that solutions for the ideal generated by some polynomials will be the same as the solutions for that system of polynomials. We

let $I = \langle f_1(x_1, ..., x_n) - y_1, ..., f_n(x_1, ..., x_n) - y_n \rangle$. Further, we also know from Chapter 3 that it is sufficient to look at any basis for an ideal to find the solutions for this ideal, and we also know that every ideal has a Gröbner basis. So by finding a Gröbner basis for our system

$$\begin{aligned} f_1(x_1, ..., x_n) - y_1 &= 0 \\ &\vdots \\ f_n(x_1, ..., x_n) - y_n &= 0, \end{aligned}$$

then the solutions we find in that basis will be the same solutions as for the original system. Calculating the Gröbner basis G can be done as in Chapter 3, where we find the S-polynomial of every pair of polynomials in the basis, and then reduce it with respect to G. Calculating G would be the most time consuming part of our attack, so optimally we would like to have a computer algorithm which could calculate this efficiently. An advantage of this attack is that when we find G, we will have at least one polynomial which only contains one variable, let us say x_n for example. If it's only one polynomial, finding the solution(s) for x_n is straightforward. If there are several polynomials containing only one variable, we can calculate the greatest common divisor for these polynomials, and then find the solution(s) from this resulting polynomial. A method for calculating the gcd of several polynomials was presented in Chapter 3.

Further, we will also have some polynomials in G containing only this variable x_n and another variable, lets say x_{n-1}. So by inserting the solution(s) we found for x_n, we can follow the same procedure to find the solution(s) to x_{n-1}. Following this procedure step-by-step backwards, we can find all of the solutions to our original system. One important notice however, is that we don't necessarily find only one solution for each variable in each step, so we might have to check for several solutions each time. In the end though, by how the MI-cryptosystem is constructed, we will find the unique solution to every variable $x_1, ..., x_n$, which would be the plaintext we seek.

This attack will work on **any** set of polynomials with an associated ciphertext, but if we want this to be an efficient attack we would need a good computer algorithm for computing G, since this computation will be the most time consuming part of our attack.

5.2 Patarins attack

In 1995 Patarin[4] proposed an attack on the Matsumoto-Imai cryptosystem, where the idea is to take advantage of how the map $\tilde{F}$ is constructed. As in Chapter 4, we have a field k with $q = 2^m$ elements chosen, and n is both the degree of the extension field K over k, and the number of components in k that we have in each message. If $g(x)$ is an irreducible polynomial of degree n in k, we can identify the extension field as $K_n \cong k[x]/g(x)$. As in Chapter 4, we choose θ such that

$$\tilde{F} : X \mapsto X^{1+2^{m\theta}} \tag{5.1}$$

is an invertible map, where $\tilde{F}^{-1} = X^{\hbar}$ is the inverse map, when $\hbar$ is the multiplicative inverse of $X^{1+2^{m\theta}}$ modulo $X^{2^{mn}} - 1$. If now B is a basis of K_n, we can express $\tilde{F}$ in B as:

$$\tilde{F}(x_1, ..., x_n) = (f_1(x_1, ..., x_n), ..., f_n(x_1, ..., x_n)), \tag{5.2}$$

where $f_1, ..., f_n$ are n polynomials in n varibles of degree 2. Again, as in Chapter 4, this comes from the fact that both $X \mapsto X$ and $X \mapsto X^{2^{m\theta}}$ are linear functions, so $\tilde{F} = X \cdot X^{2^{m\theta}}$ is a quadratic function, and its components can be expressed quadratic in B.

Unlike in Chapter 4, we now split our message of length n such that $n = n_1 + ... + n_d$. So now we have d integers $n_1 + ... + n_d$, and each of those integers will need an extension of k, $K_{n_1}, ..., K_{n_d}$ of degree respectively $n_1, ..., n_d$. So if we now look at an element of K_{n_e}, where $1 \leq e \leq d$, this element can be seen as a "word" of length n_e. Then we will use some quadratic functions $\tilde{f}_1, ..., \tilde{f}_d$, such that we get d words, then these d words will be recombined into a word of length n.

Now, if we look at Figure 4.1 in Chapter 4, Patarins attack takes place in $K \to K$. The notation we will use from now on, is that we have the secret split $n = n_1 + ...n_d$, e an index such that $1 \leq e \leq d$, and we let x be the plaintext with the associated ciphertext y. Now when K_{n_e} is the extension field of degree n_e over k, we denote a_e as an element of K_{n_e} such a_e is affine in x, and we denote b_e as an element of K_{n_e} such that b_e is affine in y. Then by choosing the parameter θ_e, we get

$$b_e = {a_e}^{1+2^{m\theta_e}}, \tag{5.3}$$

which is quadratic in a. This leads to a simplification of Figure 4.1 in Chapter 4, which we can see in Figure 5.1. For simplicity, we will mostly denote a by a_e, b by b_e and θ by θ_e in this text.

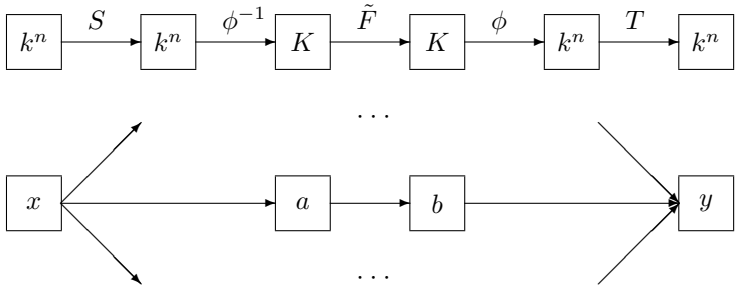

Figure 5.1: Simple MI construction

5.2.1 Weak keys

Before we look at the general, we start by looking at some weak keys in the Matsumoto-Imai algorithm that Patarin discovered. They are not necessarily a problem, as they can easily be avoided. We can write equation (5.3) as $a = b^{\hbar_e}$, where $\hbar_e$ is the multiplicative inverse of $(1 + 2^{m\theta_e})$ modulo $(2^{mn_e} - 1)$. If α is an integer, we denote by $HW(\alpha)$ the number of 1 in the expression of α in base 2 (HW stands for "Hamming weight in base 2"). Now let x_i be the bits of x, and y_i the bits of y, and $1 \leq i \leq mn$. Now each value y_j, $1 \leq j \leq n$, has a quadratic expression in the values x_i, and similarly for each value x_j, $1 \leq j \leq n$, we can express each value as a polynomial of degree $\sup HW(\hbar_e)$ in the values y_i. So if we now want to make the expression of x_j as polynomials in y_i impractible for an attack, we need to have at least one variable e such that $HW(\hbar_e)$ isn't too small. Further we will see that it is actually better if $HW(\hbar_e)$ isn't too small for all the variables e. Lets assume the opposite, that there is a variable e such that $HW(\hbar_e)$ is small, and

$$a = b^{\hbar_e}. \tag{5.4}$$

Let now a_i be bits of a and b_i bits of b in a basis, with $1 \leq i \leq n_e m$. Then since a is affine in x, there are some values α_{1i} such that $a_1 = \alpha_{10} + \sum_{i=1}^{nm} \alpha_{1i} x_i$. By equation (5.4) we know that a has a polynomial expression of total degree $HW(\hbar_e)$ in $b_1, ..., b_{n_e m}$. All these values $b_1, ..., b_{n_e m}$ are affine in $y_1, ..., y_{nm}$, so a_1 has a polynomial expression of total degree $HW(\hbar_e)$ in $y_1, ..., y_{nm}$. So now we have a polynomial P of total degree $HW(\hbar_e)$ such that

$$\alpha_{10} + \sum_{i=1}^{nm} \alpha_{1i} x_i = P(y_1, ..., y_{nm}). \tag{5.5}$$

This will also be the same for $a_2, a_3, ..., a_{nm}$, so there are at least $n_e m$ equations similar to equation (5.5), which are of degree 1 in the x_i, and of total degree $HW(\hbar_e)$ in the y_i. So if now $HW(\hbar_e)$ is very small for a given e, we can find these $n_e m$ equations which are similar to (5.5). To do that, we start by writing out the general form of such an equation with degree $HW(\hbar_e)$, which contains some unknown coefficients. Further, since we have the public polynomials, we can generate values for x and y by inserting values for x into those polynomials, and then calculate the associated ciphertext y. After calculating enough values for the x, y-pairs, we can insert these values into our equation on the general

form, such that the only unknowns are the coefficients. By Gaussian elimination on those equations we will be able to find these coefficients, and we will then have at least $n_e m$ similar equations that are independent from (5.5). So now, with these equations and a given ciphertext y, we immediatly get $n_e m$ equations of degree 1 in the x_i bits of the plaintext. So now one can do an exhaustive search in $2^{(n-n_e)m}$ instead of 2^{nm} on the plaintext, which we do not want. So we conclude that we must choose the values n_e and θ_e such that $HW(\hbar_e)$ isn't too small.

5.2.2 General attack

In this section we will look at the general attack on the MI-cryptosystem proposed by Patarin. We start by looking at the general case.

As above, we use the notation

$$b = a^{1+2^{m\theta}}. \tag{5.6}$$

By doing the composition $g : x \mapsto x^{2^{m\theta-1}}$ on each side, we get

$$b^{2^{m\theta-1}} = a^{2^{2m\theta-1}}, \tag{5.7}$$

and by multiplying each side by $a \cdot b$ we get

$$a \cdot b^{2^{m\theta}} = b \cdot a^{2^{2m\theta}}. \tag{5.8}$$

Let $(a_1, ..., a_{n_e})$ be the representation of a in the extension field K_{n_e}, and $(b_1, ..., b_{n_e})$ the representation of b in K_{n_e}. Then the equation (5.8) gives us n_e equations of degree 1 in both the b_j values **and** the a_j values. This is because both $b \mapsto b^{2^{m\theta}}$ and $a \mapsto a^{2^{2m\theta}}$ are linear functions, meaning that in a basis, the n_e components of $b^{2^{m\theta}}$ can be written as a polynomial of degree 1 in the components of b, where the coefficients are in $k = GF(2^m)$. Further, we have that a is affine in x, and b is affine in y, so if we now write these n_e equations by the components $(x_1, ..., x_n)$ and $(y_1, ..., y_n)$ of x and y, we get equations on the form

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\gamma_{ij}x_i y_j + \sum_{i=1}^{n}\alpha_i x_i + \sum_{i=1}^{n}\beta_i y_i + \delta_0 = 0. \tag{5.9}$$

These equations are true for all x, y, when x is the plaintext of y. Since we have the public polynomials, we can now choose some values for x, and then compute the associated value for y. Then, by inserting those values for x_i and y_i in equation (5.9), we will get some equations of degree 1 in the $n^2 + n + n + 1 = (n + 1)^2$ variables $\gamma_{ij}, \alpha_i, \beta_i$ and δ_i. So if we computed enough values for x and y, we can by Gaussian elimination find the values for all the variables, and we have then found all of the equations on the form of equation (5.9). This is part 1 of the attack, we have now found every equation on the form of (5.9) which comes from equation (5.8), but we may also have some equations which doesn't come from (5.8). In part 2 of the attack, given a y for which we want to find x, these equations will give us some equations of degree 1 on the values $x_1, ..., x_n$. By Gaussion

elimination on these equations, we will find λ independent equations on the form of (5.9), where $x_1, ..., x_n$ are the unknown variables, since we can replace $y_1, ..., y_n$ with the values we know. We denote λ as the number of variables $x_1, ..., x_n$ we are able to find from the others, so in order to figure out how powerful this attack is, we need to evaluate λ further.

Evaluation of λ

Theorem 5.2.1. For all the practical keys and most of the ciphertexts y, the number of λ independent equations of degree 1 in $x_1, ..., x_n$ that we will get from equation (5.9) by this given y, is given by $\lambda \geq \sum_{e=1}^{d}(n_e - gcd(\theta_e, n_e)) \geq \frac{2n}{3}$. From this we also see that for a lot of secret keys for most of the ciphertext, we will have $\lambda \geq n - d$.

We will need four lemmas to prove this theorem.

Lemma 5.2.1. Let L be a finite field with q elements. Now let p be an integer, and y an element of L. Then the equation $x^p = y$ has at most $gcd(p, q-1)$ solutions for x.

Proof. If $y = 0$, then $x = 0$ is the only solution, and the lemma is true.
Assume $y \neq 0$. Then $x = 0$ isn't a solution, so we assume $x \neq 0$, and that $x^{q-1} = 1$.
Let $\mu = gcd(p, q-1)$.
Then we know from Bézout's identity that there exists two integers α and β such that $\alpha p - \beta(q-1) = \mu$. So now

$$\begin{aligned} x^p = y &\Rightarrow x^{\alpha p} = y^{\alpha} \\ &\Rightarrow x^{\mu} \cdot (x^{q-1})^{\beta} = y^{\alpha} \\ &\Rightarrow x^{\mu} = y^{\alpha}. \end{aligned}$$

And since every equation of degree k in a field has at most k solutions, we get that $x^{\mu} = y^{\alpha}$ has at most μ solutions, and then also for $x^p = y$, as claimed. □

Lemma 5.2.2. For all integers m, α and β, we have

$$gcd(2^{m\alpha} - 1, 2^{m\beta} - 1) = 2^{m(gcd(\alpha,\beta))} - 1. \tag{5.10}$$

Lemma 5.2.3. In the extension field K_{n_e} with 2^{mn_e} elements, equation (5.8) has at most $2^{m(gcd(\theta,n_e))}$ solutions in a, for each given $b \neq 0$.

Proof. For $b \neq 0$, equation (5.8) has two sets of solutions for a:

1. $a = 0$.
2. a such that

$$(a^{2^{m\theta}-1})^{2^{m\theta}+1} = b^{2^{m\theta}-1} \tag{5.11}$$

The function $g : x \mapsto x^{2^{m\theta}+1}$ is a bijection in K_{n_e}. This is because of how we construct the MI-cryptosystem, where we choose the variables θ and n_e such that g satisfy this property. Therefore, a is a solution of (5.11) if and only if

$$a^{2^{m\theta}-1} = g^{-1}(b^{2^{m\theta}-1}). \tag{5.12}$$

We know from Lemma 5.2.1 then, that the equation (5.12) for a given b has at most $gcd(2^{m\theta}-1, 2^{mn_e}-1)$ solutions a. Further, by Lemma 5.2.2 we then know that (5.11) has at most $2^{m(gcd(\theta,n_e))}-1$ solutions in a. Now, by adding the solution $a=0$, we get that for $b \neq 0$ that equation (5.8) has at most $2^{m(gcd(\theta,n_e))}$ solutions in a, as claimed. □

We add a corollary to Lemma 5.2.3:

Corollary 5.2.1. Given a $b \neq 0$, and if we write the equation (5.8) in a basis on the components of a, we will get at least $n_e - gcd(\theta, n_e)$ independent equations of degree 1, in the components of a.

Proof. We have already seen that the equations we get are of degree 1 in the components of a. If we let λ_e be the number of independent equations, then we know that we will have exactly $2^{m(n_e-\lambda_e)}$ solutions. But from Lemma 5.2.3 we know that we will have at most $2^{m(gcd(\theta,n_e))}$ solutions, so $\lambda \geq n_e - gcd(\theta, n_e)$, as claimed. □

Lemma 5.2.4. $\forall e, 1 \leq e \leq d$, let $\delta_e = gcd(\theta_e, n_e)$, and let $k_e = \frac{n_e}{\delta_e}$. Then k_e is always odd, and $k_e \geq 3$.

We can now write out the proof for Theorem 5.2.1.

Proof. Let y be a ciphertext such that for this y we have:
$\forall e, 1 \leq e \leq d, b_e \neq 0$. (And therefore $a_e \neq 0$ as well, since $b_e = {a_e}^{1+2^{m\theta_e}}$).

Further, we may assume that mn_e isn't too small, since then we would have $a = b^{\hbar_e}$ with a small $\hbar$, which would give a small $HW(\hbar)$, and we have already seen that this gives the weak keys we want to avoid. So with mn_e not too small, most of the ciphertext y will be such that $\forall e, 1 \leq e \leq d, b \neq 0$. We saw earlier that for the equations on the form of equation (5.9), we will find at least all of the equations from (5.8), when we write these equations with the components of x and y instead of a and b, for all the values of e, $1 \leq e \leq d$. From Corollary 5.2.1, we know that we will get at least $\sum_{e=1}^{d}(n_e - gcd(\theta_n, n_e))$ independent equations of degree 1 in the components of x. This completes the proof for part 1 of the theorem. For part 2, we have proved that $\lambda \geq \sum_{e=1}^{d}(n_e - gcd(\theta_n, n_e))$. From Lemma 5.2.4 we then have $\lambda \geq \sum_{e=1}^{d}(n_e - \frac{n_e}{3}) = \frac{2}{3}\sum_{e=1}^{d} n_e = \frac{2n}{3}$.
And since we will have $gcd(\theta, n_e) = 1$ for a lot of keys, we also have $\sum_{e=1}^{d}(n_e - 1) = n - d$, so $\lambda \geq n - d$. This completes our proof. □

This attack we now have described proceeds in two parts:

1. We find all of the equations on the form of equation (5.9).
2. For each specific ciphertext y, we try to find x with the help of equation (5.9). This would have to be done for every x we try to find from a given y.

5.2.3 An example

As above, let x be the plaintext, y the ciphertext, a affine in x, b affine in y and $b = a^{1+2^{m\theta}}$. We are looking at what happens in the extension field K_{e_n}, and we will in this example look at the case when $m = 1$ and $\theta = 1$, such that

$$b = a^3. \tag{5.13}$$

Now let $(b_1, ..., b_{n_e})$ be the representation of b in K_{n_e}, and $(a_1, ..., a_{n_e})$ the representation of a. From equation (5.13) we have that all the b_j have a quadratic representation in $(a_1, ..., a_{n_e})$, since $b = a \cdot a^2$, and a^2 is linear because of $m = 1$. We want to find an expression which gives us the a_j values from the b_j values, and one could think of trying

$$a = b^{\hbar}. \tag{5.14}$$

But the $HW(\hbar)$ values would most likely be too big, so we probably would not get a useful expression for the a_j values. Instead, we try to multiply both sides of (5.13) by a:

$$b \cdot a = a^4. \tag{5.15}$$

Since $m = 1$ we have that a^4 is linear in a here, so this equation will give us n_e equations of degree 1 of both the b_j values **and** the a_j values. Now if for any $b \neq 0$, we will have exactly two solutions: the solution $a = 0$ and the solution $a = b^{\hbar}$. Since equation (5.14) probably is not useful, we sort of lose one equation over $GF(2)$, but even in that case, equation (5.15) will still be useful. Since a is affine in x, b is affine in y and a^4 is linear in a, we know from equation (5.15) that we have some equations on the form:

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\gamma_{ij}x_iy_j + \sum_{i=1}^{n}\alpha_i x_i + \sum_{i=1}^{n}\beta_i y_i + \delta_0 = 0. \tag{5.16}$$

These equations are true for all x, y, when x is the plaintext of y. If $b = 0$, there is only one solution for equation (5.15), so there is necessarily at least n_e formally independent solutions like (5.16). (By formally we mean here that the vector-space of the solutions for the values $\gamma_{ij}, \alpha_i, \beta_i$ and δ_i is at least n_e). But since (5.15) has two solutions in a when $b \neq 0$, we can say that for a given value y, we have at least $n_e - 1$ independent equations on the form of (5.16). Now we choose some values for x, and by using the public polynomials we compute the associated values for y. Then when we insert these values x_i and y_i into equation (5.16), we will get some equations of degree 1 in the $(n+1)^2$ values for $\gamma_{ij}, \alpha_i, \beta_i$ and δ_i. By Gaussian elimination we can now find all of the unknown variables as long as we have enough values for x and y, and then we will have all of the equations on the form of (5.16). By inserting the values for the variables and the y_i values for which we want to find the plaintext x, we have $n_e - 1$ independent equations of degree 1, where only the values $x_1, ..., x_n$ are unknown. So by Gaussian elimination we will be able to find $n_e - 1$ variables $x_1, ..., x_m$ from the others.

Bibliography

[1] Stinson D.R. *Cryptography theory and practice*. Taylor & Francis group, LLC, 2006.

[2] O'Shea D. Cox D.A., Little J. *Ideals, Varieties, and Algorithms*. Springer, Cham, 2015.

[3] Schmidt D.S. Ding J, Gower J.E. *Multivariate Public Key Cryptosystems*. Springer, Boston, MA, 2006.

[4] Patarin J. *Cryptanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt'88. In: Coppersmith D. (eds) Advances in Cryptology — CRYPT0' 95.* Springer, Berlin, Heidelberg, 1995.