

MI-TTM Cryptosystem

Matsumoto-Imai Tame-Transformation-Method Cryptosystem

Master's thesis of Manuel Ribic

Written under the supervision of

Prof. Dr. Joachim Rosenthal

Dr. Jens Zumbrägel

at the Institute of Mathematics at the University of Zurich

August 2009

Acknowledgments

I would like to thank my supervisor Prof. Dr. Joachim Rosenthal for imparting me the knowledge by his lectures and for giving me the chance to treat a very interesting topic in cryptography. A special thank goes to Dr. Jens Zumbrägel who helped me especially with his important support and explanations, which let me recognize mathematical properties long before I would be aware of it by myself, and also for the precise proof reading. Finally, I would like to thank my fellow students, my friends and my parents who encouraged me through this time.

Contents

1	Introduction	3
2	MI Cryptosystem	4
2.1	Construction of a MI Cryptosystem	4
2.2	Mathematical background of MI Cryptosystem construction	7
2.3	Patarin's Attack	11
2.4	Kipnis-Shamir Attack	17
2.5	Practical usage of the MI Cryptosystem	21
3	TTM Cryptosystem	24
3.1	Definitions for the tame automorphism	24
3.2	Construction of TTM Cryptosystem	25
3.3	Mathematical background of TTM Cryptosystem	29
3.4	Minrank Attack of Goubin and Courtois	30
3.5	Attack of Ding and Schmidt	33
3.6	Historical development of TTM Cryptosystem	35
4	Direct Attacks	36
4.1	Gröbner bases method	36
4.2	XL method	46
4.3	Zhuang-Zi method	48
5	MI-TTM Cryptosystem	51
5.1	Construction of the MI-TTM Cryptosystem	51
5.2	Background of the MI-TTM Cryptosystem	53
5.3	Possible attacks	54
5.4	Future of the MI-TTM Cryptosystem	57
	Bibliography	58

Chapter 1

Introduction

Since Diffie and Hellman 1976 introduced the idea of public key cryptography the research in this direction became more and more relevant. Most popular public key cryptosystems nowadays are based on the complexity of the discrete logarithm problem such as ElGamal and on the difficulty of factoring large integers such as RSA. But unfortunately Shor presented in [Sho97] polynomial time quantum algorithms which can be used to solve both problems. A practical quantum computer does not exist yet. But it is necessary to find other mathematical constructions for the future public key cryptosystems.

One approach is the multivariate public key cryptosystem, shortly MPKC. An MPKC can be considered in the following general way [DGS06a, page 139]:

Let K be a finite field. Let K^n denote the n -dimensional vector space over K . Suppose $F : K^n \rightarrow K^n$ is a composition of l invertible maps G_1, \dots, G_l

$$F(x_1, \dots, x_n) = G_1 \circ \dots \circ G_l(x_1, \dots, x_n) = (f_1, \dots, f_n),$$

where each G_i is a map from K^n to K^n and $f_i \in K[x_1, \dots, x_n]$ for $i = 1, \dots, n$. In addition, suppose F has the following properties:

1. Given $(x'_1, \dots, x'_n) \in K^n$, $F(x'_1, \dots, x'_n)$ is efficiently computable;
2. Given only the polynomial components f_1, \dots, f_n of F , it is difficult to recover the composition factors G_1, \dots, G_l .

If we further assume that it is computationally difficult to directly solve $F(x_1, \dots, x_n) = (y'_1, \dots, y'_n)$ with $(y'_1, \dots, y'_n) \in K^n$, then we can use F to build a public key cryptosystem.

The big advantage of this construction is that solving a set of multivariate polynomial equations over a finite field, in general, is proven to be an NP-complete problem, even for quadratic equations. For this see [GJ79, page 251], [PG97, Appendix] or [Wol02, Section 3.2].

MPKCs can be split up in four different groups. We will concentrate on two of them, namely Matsumoto-Imai Cryptosystem and Tame-Transformation-Method Cryptosystem. In Chapter 2 we will see the construction of the Matsumoto-Imai Cryptosystem and we will also have a look at the specific attacks for this system. In Chapter 3 we will study the Tame-Transformation-Method Cryptosystem and its attacks. In Chapter 4 there will be a list of the common attacks which work for all MPKCs. Afterwards in Chapter 5 we will combine the Matsumoto-Imai Cryptosystem and the Tame-Transformation-Method Cryptosystem while we use the advantages of each one.

Chapter 2

Matsumoto-Imai Cryptosystem

Tsutomu Matsumoto and Hideki Imai presented at Eurocrypt '88 a public key cryptosystem which opened a new field for cryptographic research [MI88]. Although there are earlier attempts for finding MPKCs, the Matsumoto-Imai (MI) Cryptosystem was the first system whose ideas were successful. Although Jacques Patarin broke the MI Cryptosystem [Pat95], the idea was exploit further. Hence we first have a look at the MI Cryptosystem for grasping the construction ideas and afterwards at the attack of Patarin and another one for detecting the weak points of the MI Cryptosystem.

2.1 Construction of a Matsumoto-Imai Cryptosystem

This section is based on [MI88] and on [DGS06a, pages 11-20].

Let K be a finite field of characteristic $p = 2$ and cardinality $q = p^m$, where m is a positive integer. Take $g(u) \in K[u]$ to be an irreducible polynomial of degree n , where $n = (2l + 1)2^r$ with $l \geq 1$ and $r \geq 0$. We will show in Section 2.2 why we choose the variables in this way. (We will even see that it is the best if we choose n to be odd.) Define the field $L = K[u]/g(u)$, a degree n extension of K . Let K^n denote the n -dimensional vector space over K .

Let $\phi : L \longrightarrow K^n$ be the standard K -linear isomorphism between L and K^n given by

$$\phi(a_0 + a_1u + \dots + a_{n-1}u^{n-1}) = (a_0, a_1, \dots, a_{n-1}).$$

Let θ be a positive integer such that $\theta = b2^r$ with $1 \leq b \leq l$, where l and r are the same as above. Let us denote $1 + q^\theta$ by h .

Then we construct the invertible map

$$\tilde{f} : L \longrightarrow L, U \longmapsto U^h.$$

As we can see later (see Theorem 2.2.11), the conditions on θ insure that \tilde{f} is an invertible map. If h^{-1} is an integer such that $h^{-1}h \equiv 1 \pmod{(q^n - 1)}$, then \tilde{f}^{-1} is simply

$$\tilde{f}^{-1}(U) = U^{h^{-1}}.$$

At last we choose s and t to be two invertible affine transformations over K^n .

So we can construct a function for the public key:

$$f = t \circ \phi \circ \tilde{f} \circ \phi^{-1} \circ s = (f_0, f_1, \dots, f_{n-1}).$$

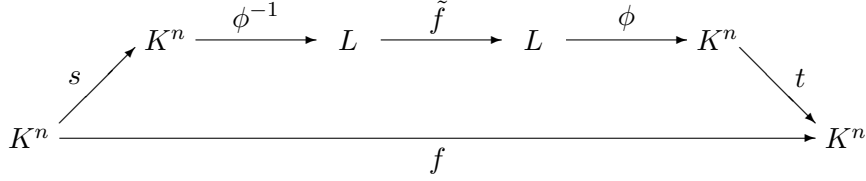


Figure 2.1: composition of maps of the MI Algorithm

the public key

The public key of the Matsumoto-Imai Cryptosystem includes the following:

1. The field K including its additive and multiplicative structure.
2. The n polynomials $f_0, f_1, \dots, f_{n-1} \in K[x_0, x_1, \dots, x_{n-1}]$.

the private key

1. The two invertible affine transformations s and t .
2. The parameter θ . (It can be kept private, though this is not critical. There are fewer than n choices for θ and n is typically not very large. So hiding θ has no substantial effect on attack complexities (only a factor of n).)

encryption

Let $(x'_0, x'_1, \dots, x'_{n-1})$ be a plaintext message. Then it results of it that the associated ciphertext is $(y'_0, y'_1, \dots, y'_{n-1})$, where

$$y'_i = f_i(x'_0, x'_1, \dots, x'_{n-1})$$

for $i = 0, 1, \dots, n - 1$.

decryption

Let $(y'_0, y'_1, \dots, y'_{n-1})$ be a ciphertext message. Then compute in the following way:

1. compute $(\tilde{y}'_0, \tilde{y}'_1, \dots, \tilde{y}'_{n-1}) = t^{-1}(y'_0, y'_1, \dots, y'_{n-1})$
2. compute $(\tilde{x}'_0, \tilde{x}'_1, \dots, \tilde{x}'_{n-1}) = \phi \circ \tilde{f}^{-1} \circ \phi^{-1}(\tilde{y}'_0, \tilde{y}'_1, \dots, \tilde{y}'_{n-1})$
3. compute $(x'_0, x'_1, \dots, x'_{n-1}) = s^{-1}(\tilde{x}'_0, \tilde{x}'_1, \dots, \tilde{x}'_{n-1})$

Theoretically we can also build an inverse f^{-1} . But in practice, the complexity for computing this is too big.

2.1.1 The multiple-branch version

In the original paper of Matsumoto and Imai the algorithm includes a further element. This version is sometimes denoted as the multiple-branch MI Cryptosystem. We want to shortly present this version now.

If we have a look at Figure 2.1, then there exists a difference between the single-branch version and the multiple-branch version for the maps ϕ^{-1} , \tilde{f} and ϕ . Matsumoto and Imai recommended to divide the K^n into d partitions or branches.

So we define an integer $d \geq 1$ and a partition $\sigma = (n_1, n_2, \dots, n_d)$ of the integer n such that

$$n = n_1 + n_2 + \dots + n_d \text{ and } 3 \leq n_1 \leq n_2 \leq \dots \leq n_d,$$

where $n_i = (2l_i + 1)2^{r_i}$ with $l_i \geq 1$ and $r_i \geq 0$ for $i \in \{1, \dots, d\}$.

Now we construct a bijection

$$\mu = (\mu_1, \mu_2, \dots, \mu_d) : K^n \longrightarrow K^{n_1} \times K^{n_2} \times \dots \times K^{n_d},$$

with the projections $\mu_i : K^n \longrightarrow K^{n_i} : (x_0, x_1, \dots, x_{n-1}) \longmapsto (x_{k_i-n_i}, x_{k_i-n_i+1}, \dots, x_{k_i-1})$ in which $k_i = \sum_{j=1}^i n_j$ and $i = 1, \dots, d$.

Further we construct d different field extensions in analogous manner to the single-branch version. Each field extension has the degree n_i with $i = 1, \dots, d$. So we also define the standard K -linear isomorphism $\phi_i : L_i \longrightarrow K^{n_i}$ for $i = 1, \dots, d$ same as before. In the same way are also the invertible maps $\tilde{f}_i : L_i \longrightarrow L_i, U_i \mapsto U_i^{h_i}$ with $h_i = 1 + q^{\theta_i}$ whereas $\theta_i = b_i \cdot 2^{r_i}$ and $1 \leq b_i \leq l_i$.

Then we can construct a function for the public key in the multiple-branch way:

$$\begin{aligned} F &= t \circ \mu^{-1} \circ (\phi_1, \phi_2, \dots, \phi_d) \circ (\tilde{f}_1, \tilde{f}_2, \dots, \tilde{f}_d) \circ (\phi_1^{-1}, \phi_2^{-1}, \dots, \phi_d^{-1}) \circ (\mu_1, \mu_2, \dots, \mu_d) \circ s \\ &= (F_0, F_1, \dots, F_{n-1}) \end{aligned}$$

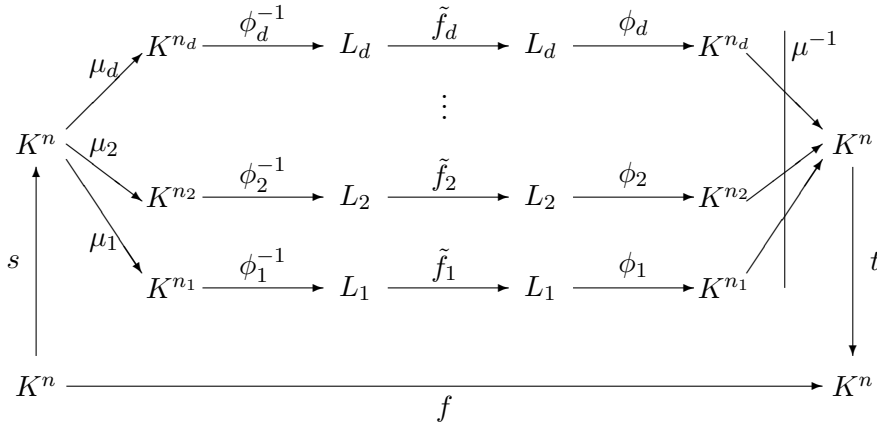


Figure 2.2: composition of maps of the MI Algorithm in the multiple-branch version

Unfortunately, there is no advantage in using the multiple-branch version. Patarin showed in the extended paper of [Pat95], that the complexity, which occurs through the multiple branches, is nullified through the possibility to accelerate the attack [Pat00]. We will have some comments about this in the end of Patarin's Attack. And Felke [Fel05] showed in his paper, that any system with branches can be decomposed into its single branches in polynomial time in average.

2.2 Mathematical background of Matsumoto-Imai Cryptosystem construction

This section is based on Chapter IV of [MI88].

There are some interesting considerations why the MI Cryptosystem is constructed in this way. That is why we will make some pure mathematical thoughts.

We can transform any function of a finite field into a univariate polynomial as follows:

$$\begin{aligned} \{\text{function of a finite field } K\} &\longrightarrow \{\text{univariate polynomial}\}, \\ f &\longmapsto p(x) := \sum_{i \in K} \left(\prod_{j \in K, j \neq i} \frac{x-j}{i-j} \right) f(i). \end{aligned}$$

So we denote the polynomial representation of a function f with $\langle f \rangle$. For better description of our circumstances we insert some notations and definitions.

2.2.1 Notations and definitions

Notation 2.2.1. We describe a polynomial p in the following way:

$$p(x_0, \dots, x_{s-1}) := \sum_i \{p_{i_0 \dots i_{s-1}} x_0^{i_0} \cdots x_{s-1}^{i_{s-1}} \mid i_0, \dots, i_{s-1} \geq 0\}.$$

Definition 2.2.2. Let p be a polynomial. The **degree** $\deg(p)$ of a polynomial is determined by

$$\deg(p) := \begin{cases} \max\{i_0 + \dots + i_{s-1} \mid p_{i_0 \dots i_{s-1}} \neq 0\} & \text{if } p \neq 0 \\ -\infty & \text{if } p = 0. \end{cases}$$

The degree $\deg(P)$ of a polynomial-tuple $P = (p_0, \dots, p_{t-1})$ is defined as $\max\{\deg(p_j) \mid j = 0, \dots, t-1\}$

Definition 2.2.3. Let P be a t -tuple of s -variate polynomials. We define a function τ_{up} by

$$\tau_{up}(P) := t \cdot \binom{s + \deg(P)}{\deg(P)}.$$

$\tau(P)$ denotes the total number of nonzero terms of P .

Remark 2.2.4. $\tau(P)$ is always less than or equal to $\tau_{up}(P)$.

Sketch of proof: Without loss of generality $t = 1$. We introduce a new variable x_s and multiply each term $p(x_0^{i_0} \dots x_{s-1}^{i_{s-1}})$ by a monomial with $x_s^{i_s}$, so that $i_0 + \dots + i_s = \deg(P)$. Thus the number of such monomials is equal to the number of possibilities to establish equalities in the form of $i_0 + \dots + i_s = \deg(P)$, which is $\binom{s + \deg(P)}{\deg(P)}$. \square

Further we have to introduce the weight of an integer and of a polynomial.

Definition 2.2.5. Let a be an integer greater than 1, i a nonnegative integer. Denote the a -ary representation of i by

$$i = \sum \{a^j \cdot i_j \mid 0 \leq i_j < a, j = 0, 1, \dots\}.$$

We define a function W_a on the nonnegative integers as follows:

$$W_a(i) := \sum \{i_j \mid j = 0, 1, \dots\}.$$

This is the sum of digits in a -ary representation. $W_a(i)$ is called the **a -weight** of i .

Definition 2.2.6. For a univariate polynomial $E(u) = E_0 + E_1u + E_2u^2 + \dots + E_du^d$, the **exponential a -weight** $wt_a(E)$ of E is defined by

$$wt_a(E) := \begin{cases} \max\{W_a(i) \mid E_i \neq 0\}; & \text{if } E \neq 0, \\ -\infty & \text{if } E = 0. \end{cases}$$

This is the maximal a -weight of an exponent of a nonzero term.

Finally we have to define a rank for integers.

Definition 2.2.7. Let i be a positive integer. Then we define a function R_a from the positive integers to the nonnegative integers as follows:

$$R_a(i) := \max\{j \geq 0 \mid i \text{ is divisible by } a^j\}.$$

$R_a(i)$ is called the **a -rank** of i .

2.2.2 Mathematical results

Now we can describe two important complexities. Let f be the function which maps the cleartext into the ciphertext. So we can estimate the size of the public key and the complexity of the public transformation of f :

$$\begin{aligned} \{\text{The description length of a public key}\} &= \mathcal{O}(\tau(\langle f \rangle) m \log_2 p) [\text{bit}] \\ \{\text{The complexity of a public transformation}\} &= \mathcal{O}(\tau(\langle f \rangle) m^2) [\text{bit} - \text{operation}], \end{aligned}$$

where p and m are variables of the finite field K , i.e. p is prime and the order $q = p^m$ with $m > 0$.

We see that both depend on the function of $\tau(\langle f \rangle)$. So we have a look at the upper bound $\tau_{up}(\langle f \rangle)$ of $\tau(\langle f \rangle)$. τ_{up} is strongly related to the degree of our polynomial. So if we decrease the degree it follows that we reduce the description length of the public key and the complexity of the public transformation.

On the other side we can have a look at the polynomial representation $\langle f^{-1} \rangle$ of the secret transformation f^{-1} from ciphertext into the cleartext. Here there is the same relation between the degree and τ_{up} as we saw above in connection with f .

Our goal is placing the basic conditions in a way that the degree of $\langle f \rangle$ is small and the degree of $\langle f^{-1} \rangle$ large.

Because we will see in Theorem 2.2.11 that f is bijective, Matsumoto and Imai formulated the following theorem [MI88, Theorem 8] (originally for multiple-branch):

Theorem 2.2.8. *For the bijection f , the following statements are true:*

1. $\deg(\langle f \rangle) = wt_q(\langle \tilde{f} \rangle)$.
2. $\deg(\langle f^{-1} \rangle) = wt_q(\langle \tilde{f}^{-1} \rangle)$.

From this we can derive the following theorem:

Corollary 2.2.9. *For the bijection f , the following statements are true:*

1. $\deg(\langle f \rangle) = W_q(h)$.
2. $\deg(\langle f^{-1} \rangle) = W_q(h^{-1})$.

Proof: By construction of $\langle \tilde{f} \rangle$ we see that this polynomial is a monic monomial in one indeterminate whereas we want to have in mind that $\langle \tilde{f} \rangle$ is a univariate polynomial over L . If we have a look at the definition of the weight for polynomials we can conclude the claim directly from Theorem 2.2.8. \square

Now we can pin the first constraint down. Assume that $\deg(\langle f \rangle) = 1$. Then we have $W_q(h) = 1$. And because $h = 1 + q^\theta$, so only if $h = 1$ then $W_q(h) = 1$ can be fulfilled and so $W_q(h^{-1}) = 1$. This implies that $\deg(\langle f^{-1} \rangle) = 1$, which is not desirable. Hence, it is essential that $\deg(\langle f \rangle) \geq 2$. Due to the simplicity of computing with small degrees, we will study functions with polynomial degree equal to 2.

Thinking about this fact results in a next constraint about p , namely $p = 2$. Surely, it is the easiest way to calculate on a computer. But there is also another reason. Look at the following theorem:

Theorem 2.2.10. *Let p be a prime integer, m, n, q and h be integers satisfying $m > 0, n > 0, q = p^m, 0 < h < q^n - 1$, and $\gcd(h, q^n - 1) = 1$. Then $p = 2$ is the necessary condition for $W_q(h) = 2$.*

Proof: Assume q is odd. When $W_q(h) = 2$, h can be written as $h = q^j(1 + q^\theta)$, where j and θ are nonnegative integers. Hence h must be even. Also notice that $q^n - 1$ is apparently even. Thus $\gcd(h, q^n - 1)$ must be divisible by 2, which contradicts the assumption of $\gcd(h, q^n - 1) = 1$. Therefore q must be even. Put it in other words, $p = 2$ is the necessary condition for $W_q(h) = 2$. \square

Let us also spot why the other variables are chosen in this way.

As we just saw in the proof, h can be expressed in the following way:

$$h = q^j(1 + q^\theta),$$

where j and θ are nonnegative.

Since $\phi \circ U^{q^j} \circ \phi^{-1}$ is a linear function, we can consider the functions of evaluating the q^j th power together with the affine transformations s and t . So it suffices to consider the case $j = 0$.

If $\theta = 0$, then $h = 2$. In this case, \tilde{f} is a bijection because $\gcd(2, q^n - 1) = 1$. Since both $p = 2$ and $h = 2$, it is clear f_i ($i = 0, \dots, n - 1$) contains only constant terms and the terms

x_0^2, \dots, x_{n-1}^2 . In this case, one can quickly solve a system of quadratic multivariate polynomial equations. First, taking the system as a system of linear equations in variables x_0^2, \dots, x_{n-1}^2 , one can readily solve the new system and get x_0^2, \dots, x_{n-1}^2 . Then, one can uniquely determine x_i from x_i^2 (note $p = 2$).

After the above discussion, it becomes obvious that we can concentrate our attention upon the case $h = 1 + q^\theta$, where $\theta > 0$.

Now we can formulate the condition for \tilde{f} being bijective.

Theorem 2.2.11. *Let m, q, θ, n and h be integers satisfying $m > 0, q = 2^m, 0 < \theta < n$ and $h = 1 + q^\theta$. We have $\gcd(h, q^n - 1) = 1$ iff $R_2(\theta) \geq R_2(n)$.*

According to the above theorem it is necessary that $n \geq 3$. And it makes sense to choose our variables in the following way:

$$\theta = b \cdot 2^r, \quad 1 \leq b \leq l,$$

where r is a nonnegative integer and l is a positive integer such that

$$n = (2l + 1) \cdot 2^r, \quad r = R_2(n).$$

Now, it is possible to say how large is the degree of f^{-1} , because we can make a statement about the size of $W_q(h^{-1})$.

Theorem 2.2.12. *For integers m, q, θ, n and h satisfying $m > 0, q = 2^m, 0 < \theta < n, h = 1 + q^\theta$ and $\gcd(h, q^n - 1) = 1$, the q -weight of the multiplicative inverse element h^{-1} of h modulo $(q^n - 1)$ is given by:*

$$W_q(h^{-1}) = 1/2\{(q - 1)(n - R_q(h^{-1})) + 1\}.$$

Under special conditions there is an easy way for calculating the q -rank of h^{-1} exactly:

Theorem 2.2.13. *$R_q(h^{-1}) = 2^r - 1$ when $\gcd(b, 2l + 1) = 1$.*

The proof of the last three results can be found in [MI88, Theorem 11-13].

From the last two theorems, we can conclude that, when n is an odd integer ≥ 3 ($r = 0$) and θ is relatively prime to n , the q -rank of h^{-1} becomes zero, and the q -weight of h^{-1} reaches its maximum.

So we can say the parameters are chosen in an optimal way. But we should be nevertheless cautious. We only know $\tau(\langle f \rangle) \leq \tau_{up}(\langle f \rangle)$, and we have shown the right hand side to be large, but the security depends on $\tau(\langle f \rangle)$. There also may be other means to attack the MI Cryptosystem (see Section (2.3) and (2.4)).

2.2.3 Some mathematical thoughts for the multiple-branch version

The derivation for the single-branch version also works for the multiple-branch version. But there are two theorems, which we have to formulate in a slightly other way. Afterwards we can draw some conclusions which only make sense for the multiple-branch version.

Theorem 2.2.8 reads as follows:

Theorem 2.2.14. *For the bijection F , the following statements are true:*

1. $\deg(\langle F \rangle) = \max\{wt_q(\langle \tilde{f}_i \rangle) \mid i = 1, \dots, d\}$.
2. $\deg(\langle F^{-1} \rangle) = \max\{wt_q(\langle \tilde{f}_i^{-1} \rangle) \mid i = 1, \dots, d\}$.

And Corollary 2.2.9 looks in the following way:

Corollary 2.2.15. *For the bijection F , the following statements are true:*

1. $\deg(\langle F \rangle) = \max\{W_q(h_i) \mid i = 1, \dots, d\}$.
2. $\deg(\langle F^{-1} \rangle) = \max\{W_q(h_i^{-1}) \mid i = 1, \dots, d\}$.

So we deduce similarly as for the single-branch version. Assume that $\deg(\langle F \rangle) = 1$. Then we have $W_q(h_i) = 1$ for all i , and also $W_q(h_i^{-1}) = 1$ for all i . This implies that $\deg(\langle F^{-1} \rangle) = 1$, which is not desirable. Hence, it is essential that $\deg(\langle F \rangle) \geq 2$. Due to the simplicity of computing with small degrees, we will study functions with polynomial degree equal to 2.

There is one last comment we want to make for the two theorems above in the multiple-branch version. We only require $W_q(h_i^{-1})$ to be large for some i , whereas almost all $W_q(h_i^{-1})$ may be small. However, Patarin showed in the paper, which also broke the MI Cryptosystem, that $W_q(h_i^{-1})$ should be large for all i (Section 5 of [Pat95]).

2.3 Patarin's Attack

Patarin presented at Crypto '95 an attack which does not yield the secret key but bares the plaintext if you know the ciphertext [Pat95].

We will mainly rely on [Pat00], which is the extended version of [Pat95], and on [DGS06a, pages 20-25].

2.3.1 The general case attack

First, we have to introduce some notations. We adopt the notations from the previous sections of this chapter. Additionally, we need the following:

Notation 2.3.1. *Let us have a look at the degree n extension field L . Let x be the cleartext and y the corresponding ciphertext, which are elements in K^n . We denote by a the element of L affine in x , by b the element of L affine in y such that $b = a^{1+q^\theta}$.*

Remark 2.3.2. *So we can say $a = \phi^{-1}(s(x))$ and $b = \phi^{-1}(t^{-1}(y))$. (see Figure 2.1)*

We have

$$b = a^{1+q^\theta}.$$

By composition on each side of this equation with $g : x \mapsto x^{q^\theta - 1}$, we obtain:

$$b^{q^\theta - 1} = a^{q^{2\theta} - 1}.$$

Now let us multiply each side by $a \cdot b$. We obtain:

$$a \cdot b^{q^\theta} = b \cdot a^{q^{2\theta}}. \quad (2.1)$$

Let $(a_0, a_1, \dots, a_{n-1})$ be the representation of a in L , and $(b_0, b_1, \dots, b_{n-1})$ be the representation of b in L .

Equation (2.1) gives n equations (not necessarily independent) of degree one on the b_k values and of degree one on the a_k values (because $b \mapsto b^{q^\theta}$ is linear, and $a \mapsto a^{q^{2\theta}}$ is linear).

Moreover a is affine in x , and b is affine in y , so that these n equations — when written in terms of $(x_0, x_1, \dots, x_{n-1})$ and $(y_0, y_1, \dots, y_{n-1})$ — give n equations of the form:

$$\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \gamma_{ij} x_i y_j + \sum_{i=0}^{n-1} \alpha_i x_i + \sum_{j=0}^{n-1} \beta_j y_j + \delta = 0. \quad (2.2)$$

These equations are true for all x and y , when x is the plaintext of y . As a result, by choosing some (say v) values for x and computing the value y from x and the public transformation, and then replacing these values x_k and y_k , $k = 0, \dots, n-1$, in (2.2), we will obtain some equations of degree one in the $n^2 + n + n + 1 = (n+1)^2$ variables γ_{ij} , α_i , β_i and δ of K .

In this way, we are able to find quickly, by Gaussian reductions, a basis B of the vector space V of all the equations (2.2). Such a basis B is given by some independent equations. This is the first part of our attack. We may have in V some equations (2.2) which do not come from the equation (2.1) (because V contains all the equations which have the general form of (2.2)), but the important point is that we have found V , and V contains at least all the equations (2.2) which come from (2.1).

During the attack, our aim is to find x from a given y . This is the second part of our attack. For that we insert our ciphertext y in the equations (2.2). But as we wrote above, these equations do not have to be independent. So in order to evaluate the power of this attack, we have to evaluate the number of independent equations (2.2). Let us denote this number by λ . It is also interesting to know how many randomly generated x — our number v — we need to construct our equations (2.2) so that the attack works.

Let us first think about v and afterwards about λ .

Evaluation of v

Let E be the set of all polynomials of the form

$$\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \gamma_{ij} x_i y_j + \sum_{i=0}^{n-1} \alpha_i x_i + \sum_{j=0}^{n-1} \beta_j y_j + \delta.$$

E is a vector space of dimension $n^2 + 2n + 1$ over K . Let V be, as above, the set of all the polynomials of E which are satisfied for all x and y , when y is the ciphertext of x . Let T_v be the set of all the polynomials P of E such that $P(X, Y) = 0$, where Y is the ciphertext of X , for $X = X_1, X = X_2, \dots, X = X_v$, where X_1, X_2, \dots, X_v are the v first random values x of K^n .

Patarin formulated a theorem, which helps the attacker to estimate how many x has to be generated [Pat00, Theorem 6.3]:

Theorem 2.3.3. *The probability that $T_v = V$ is close to 1 when $v \geq \frac{n^2 \log q}{\log \frac{8}{7}}$.*

In this way, we have proved that we have with high probability that every equation in T_v holds for all the cleartexts.

After we found our pairs of plaintext and ciphertext, we detect the variables γ_{ij} , α_i , β_i and δ . But these equations do not have to be independent. So we have to discuss how big λ has to be.

Evaluation of λ

For λ Patarin formulated the following theorem [Pat00, Theorem 6.4]:

Theorem 2.3.4. *For all the practical keys and for most of the ciphertexts y , the number λ of independent equations of degree one in x_0, x_1, \dots, x_{n-1} that we obtain from the equations (2.2) for this given y is $\lambda \geq n - \gcd(\theta, n) \geq \frac{2n}{3}$. Moreover this shows that for many secret keys and for most ciphertexts we have $\lambda \geq n - 1$.*

Sketch of proof: Let us have a look to the equation (2.1). If we transform this equation back to that before, we have got:

$$b^{q^\theta - 1} = a^{q^{2\theta} - 1}$$

This equation has at most $\gcd(q^{2\theta} - 1, q^n - 1)$ solutions in L . Furthermore, because of the condition $\gcd(q^\theta + 1, q^\theta - 1) = 1$, we have that

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

hence the equation (2.1) has at most $\gcd(q^\theta - 1, q^n - 1) + 1$ solutions, including the trivial solution. As it is easily to prove, that $\gcd(q^\theta - 1, q^n - 1) = q^{\gcd(\theta, n)} - 1$, the total number of solutions for (2.1) is at most $q^{\gcd(\theta, n)}$. If λ is the number of linearly independent linear equations that arise from (2.1), then there will be $q^{n-\lambda}$ solutions to the corresponding system of linear equations. Therefore $q^{n-\lambda} \leq q^{\gcd(\theta, n)}$, and so $\lambda \geq n - \gcd(\theta, n)$.

The three largest possible values of $\gcd(\theta, n)$ are n , $n/2$ if n is even, and $n/3$ if 3 divides n , and the rest are of all smaller values. Since $\theta = b2^r \leq l2^r < l2^r + 2^{r-1} = \frac{1}{2}n$ the first two cases are impossible, and we can conclude that

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

So we can deduce that the MI Cryptosystem is not very secure since for a given ciphertext we can most of the times find at least $\frac{2n}{3}$ linear equations satisfied by the plaintext, which is analogous to leaking $\frac{2}{3}$ of the information. More importantly, these equations can be used to eliminate $\frac{2}{3}$ of the variables of the quadratic public equations derived from the public key and the ciphertext, which should then be much easier to solve than before.

But this attack has got one disadvantage. For keys where $\gcd(\theta, n)$ is big, the complexity of this algorithm is not polynomial. So Patarin showed an extension which efficiently attack these very special keys.

2.3.2 An extension of the general case, useful for special keys

In this section, we now try to find some equations of the general form:

$$a^2 \cdot b^v = b^w \quad (2.3)$$

with $W_2(v)$ small and $W_2(w)$ small, whereas we use the definition from Section (2.2). The advantage of $a^2 \cdot b^v = b^w$ over the general case is that for all $b \neq 0$ there exist only one solution in a for this equation. This second attack is not always practical, but when it is, it is sometimes more powerful than the attack above, because it gives immediately one unique solution.

As we know the cardinality is $q = 2^m$. For this second attack, we do not start from $b = a^{1+2^{m\theta}}$, but from $a = b^{h^{-1}}$, where as already used h^{-1} is the multiplicative inverse of $1 + 2^{m\theta} \bmod 2^{mn} - 1$. Hence we need a way to evaluate h^{-1} . For this Patarin presented the following theorem [Pat00, Theorem 4.5]:

Theorem 2.3.5. *Let $\delta = m \cdot \gcd(\theta, n)$, and let α and k be integers such that $\alpha\delta = m\theta$ and $k\delta = mn$. Let h^{-1} be, as usual, the multiplicative inverse of $1 + 2^{m\theta} \bmod 2^{mn} - 1$.*

Then: k is odd and $k \geq 3$.

and: $h^{-1} = 2^{k\delta-1} + \sum_{i=1}^{k-1} (-1)^i \cdot 2^{\alpha\delta i-1}$.

We want to point out that Matsumoto and Imai also presented a formula for calculating h^{-1} in their paper. The interested reader may have a look at [MI88, Theorem 4].

From $a^2 = b^{2h^{-1}}$ and Theorem 2.3.5, we have:

$$a^2 \cdot b^u = b^v, \text{ with } u = \sum_{i=1}^{(k-1)/2} 2^{m\theta(2i-1)} \text{ and } v = 1 + \sum_{i=1}^{(k-1)/2} 2^{2m\theta i}.$$

Let us proof this equation:

Proof: Let $a^2 \cdot b^u = b^v$. Because of $a^2 = b^{2h^{-1}}$ we can transform it to $b^{2h^{-1}} \cdot b^u = b^v$. Now we insert u, v and h^{-1} .

$$b^{2(2^{k\delta-1} + \sum_{i=1}^{k-1} (-1)^i \cdot 2^{\alpha\delta i-1})} \cdot b^{\sum_{i=1}^{(k-1)/2} 2^{m\theta(2i-1)}} = b^{1 + \sum_{i=1}^{(k-1)/2} 2^{2m\theta i}}$$

Let us use on the left side of the equation just one time the basis b and so use only one exponent.

$$b^{2(2^{k\delta-1} + \sum_{i=1}^{k-1} (-1)^i \cdot 2^{\alpha\delta i-1}) + \sum_{i=1}^{(k-1)/2} 2^{m\theta(2i-1)}} = b^{1 + \sum_{i=1}^{(k-1)/2} 2^{2m\theta i}}$$

Because we have on each side the same basis, let us only contemplate the exponents.

$$2(2^{k\delta-1} + \sum_{i=1}^{k-1} (-1)^i \cdot 2^{\alpha\delta i-1}) + \sum_{i=1}^{(k-1)/2} 2^{m\theta(2i-1)} = 1 + \sum_{i=1}^{(k-1)/2} 2^{2m\theta i}$$

From Theorem 2.3.5 we know that $\alpha\delta = m\theta$ and $k\delta = mn$. So we can transform it to

$$2(2^{mn-1} + \sum_{i=1}^{k-1} (-1)^i \cdot 2^{m\theta i-1}) + \sum_{i=1}^{(k-1)/2} 2^{m\theta(2i-1)} = 1 + \sum_{i=1}^{(k-1)/2} 2^{2m\theta i}$$

In the first summand on the left side we can expand the bracket by 2.

$$2^{mn} + \sum_{i=1}^{k-1} (-1)^i \cdot 2^{m\theta i} + \sum_{i=1}^{(k-1)/2} 2^{m\theta(2i-1)} = 1 + \sum_{i=1}^{(k-1)/2} 2^{2m\theta i}$$

Because we work in mod $2^{mn} - 1$ we can substitute 2^{mn} for 1 in the exponent.

$$1 + \sum_{i=1}^{k-1} (-1)^i \cdot 2^{m\theta i} + \sum_{i=1}^{(k-1)/2} 2^{m\theta(2i-1)} = 1 + \sum_{i=1}^{(k-1)/2} 2^{2m\theta i}$$

We can cancel on each side 1.

$$\sum_{i=1}^{k-1} (-1)^i \cdot 2^{m\theta i} + \sum_{i=1}^{(k-1)/2} 2^{m\theta(2i-1)} = \sum_{i=1}^{(k-1)/2} 2^{2m\theta i}$$

We can partition the first sum in two sums whereas one sum contains the positive summands and the other one the negative ones.

$$\sum_{i=1}^{(k-1)/2} 2^{m\theta(2i)} + \sum_{i=1}^{(k-1)/2} (-1)^i \cdot 2^{m\theta(2i-1)} + \sum_{i=1}^{(k-1)/2} 2^{m\theta(2i-1)} = \sum_{i=1}^{(k-1)/2} 2^{2m\theta i}$$

From this one we can easily see that this is an equation. □

So $W_{2^m}(u) = (k-1)/2$ and $W_{2^m}(v) = (k+1)/2$. Therefore, the equation $a^2 \cdot b^u = b^v$, when written as n equations in the components x_i^2 and y_i , gives n equations of degree one in the x_i^2 , and of total degree $(k+1)/2$ in the y_i . We denote this equations by (*).

Remark 2.3.6. *For the general case attack, the most difficult keys were the keys with very small k . Strangely enough, these keys are the easiest keys for this extended version.*

The attack finally consists of two parts:

Part 1: We find all the equations (2.2) and also all the equations (*).

Part 2: Then, for a given y , we put to the power 2^{m-1} the found equations (*). Since we have $(\alpha + \beta)^{2^{m-1}} = \alpha^{2^{m-1}} + \beta^{2^{m-1}}$ in K , and since $x_i^{2^m} = x_i$, the equations (*) will give us like this equations of degree one in the x_i . Therefore we will use both equations (2.2) and (*).

2.3.3 Attack version for the multiple-branch MI Cryptosystem

Most of the results we can adopt with only little change at the degree of the field extension. We have to calculate this for each branch separately. Only Theorem 2.3.4 we want to reformulate:

Theorem 2.3.7. *For all practical keys and for most of the ciphertexts y , the number λ of independent equations of degree one in x_0, x_1, \dots, x_{n-1} that we obtain from the multiple-branch version of the equations (2.2) for this given y is $\lambda \geq \sum_{e=1}^d (n_e - \gcd(\theta_e, n_e)) \geq \frac{2n}{3}$. Moreover this shows that for many secret keys and for most ciphertexts we have $\lambda \geq n - d$.*

We can say that it is not possible to find out so many equations as in the single-branch version. But the lower bound of $2/3$ is still the same.

As we already said above Patarin found ways to improve the algorithm for the multiple-branch version. He presented four algorithms, two for the case when the degrees of the field extensions of the branches are small and two when these are big. Three of these are approaches which use new ideas to attack the MI Cryptosystem. Only one for small degrees is an improvement in the narrower sense. We want to shortly present this. But for all four algorithms the interested reader is requested to consult [Pat00].

First we find as in the general case all equations (2.2). The next step is to find the input \hat{x} such that — for this input — we have $a_e = 0$ for all $1 \leq e \leq d$, where a_e is the element of L_e affine in x . Further we easily construct the ciphertext \hat{y} of \hat{x} .

Then we make the following change of variables: $X = x + \hat{x}$ and $Y = y + \hat{y}$, and we write all the equations (2.2) in the new variables X_i and Y_i . With this change of variables, the secret functions s and t are now linear (instead of affine). From all these equations (2.2), we keep only the equations that give $0 = 0$ when $X = 0$ (i.e. when $x = \hat{x}$).

Let l be the number of independent equations that we can detect with the (\hat{x}, \hat{y}) -pairs. Let (\diamond) be any of these equations. Let l_e be the number of such independent equations that come from the variables of branch number e . It is possible to prove that each of the equations (\diamond) is a linear combination of l_e equations that come from the variables of branch number e , and of $l - l_e$ equations that come from the other variables.

This works because of [Pat00, Theorem 10.1]:

Theorem 2.3.8. *Let (\diamond) be an equation like the following:*

$$\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \mu_{ij} x_i y_j + \sum_{i=0}^{n-1} \nu_i x_i + \sum_{j=0}^{n-1} \xi_j y_j + \omega = 0$$

and let us assume that (\diamond) is always true for all x_i , $1 \leq i \leq n$, when y_i , $1 \leq i \leq n$ is the ciphertext of x_i , $1 \leq i \leq n$.

Let us also assume that we have a MI Algorithm with at least two "branches", i.e. with $d \geq 2$, and with no affine relations between the x_i and their ciphertexts y_i .

Then (\diamond) is the sum of two equations $(\diamond) = (\diamond \diamond) + (\diamond \diamond \diamond)$, where $(\diamond \diamond)$ uses only variables of the first branch and $(\diamond \diamond \diamond)$ uses no variables of the first branch.

Moreover, it is easy to find out whether a particular equation (\diamond) is in the case or not, because — if it is the case — then if we write that the equation should be true for all Y , it gives at most $n - n_e$ independent linear equations $(\diamond \diamond)$ on the X_i , and similarly, if we write that the equation should be true for all X , it give at most $n - n_e$ independent linear equations $(\diamond \diamond \diamond)$ on the Y_i .

Now the equations $(\diamond \diamond)$ and $(\diamond \diamond \diamond)$ give us a change of variables X'_i and Y'_i that allow us to write the equations (\diamond) that do not use any variable of branch number e with only $n - n_e$ variables in X'_i and $n - n_e$ variables in Y'_j .

The variables X'_i , $1 \leq i \leq n - n_e$, are completed with some variables x'_i , $n - n_e < i \leq n$, in order to have a complete basis. Similarly, the variables Y'_i are completed with some variables y'_i , $n - n_e < i \leq n$, in order to have a complete basis.

Now, when we write an equation (2.2) with these variables, we see that all the terms in the x'_i can be written as a linear expression in only n_e new variables Y'_i , $n - n_e < i \leq n$. Similarly, all the terms in the y'_i can be written as a linear expression in only n_e new variables X'_i , $n - n_e < i \leq n$.

This way, we obtain a complete change of variables such that — with this new variables — all the Y'_i , $1 \leq i \leq n - n_e$, depend only on the X'_i , $1 \leq i \leq n - n_e$, and all the Y'_i , $n - n_e < i \leq n$, depend only on the X'_i , $n - n_e < i \leq n$. As a result, we have "separated" the variables.

2.4 Kipnis-Shamir Attack

We want to study a second attack on the MI Cryptosystem. Kipnis and Shamir constructed an attack on the HFE (hidden field equation) Cryptosystem, presented at Crypto '99 [KS99]. Since we can say that the HFE Cryptosystem is in a sense a generalization of the MI Cryptosystem, there is a way to adopt the Kipnis-Shamir Attack to the MI Cryptosystem. The big difference between this attack and Patarin's Attack is that in Patarin's Attack we can just find the plaintext, but in this attack we can also find the private key or something in the same line, which works like the private key.

This section is based on [DGS06a, pages 34-44].

2.4.1 Construction of the Kipnis-Shamir Attack

The construction of the MI Cryptosystem is based on the idea that we can construct a map on a K -vector space from a map on an extension field. The idea of Kipnis and Shamir was to use the structure of the map on the extension field to design the attack on the K -vector space mapping. They call the method relinearization. With this point of view, if $f : K^n \rightarrow K^n$ is a given MI public key mapping, then the first step of the attack is to lift f back to a map over L .

To simplify the exposition we assume that $q > 2$ and that s, t are linear instead of affine, in effect ignoring the constant terms. (The case $q = 2$ we actually should discuss separately. But in our case with characteristic $p = 2$ this is not necessary. For further information look at [DGS06a, Remark 2.4.2].) In other words, we assume that the f_0, \dots, f_{n-1} are degree two homogeneous polynomials in $K[x_0, \dots, x_{n-1}]$. Also, we assume that we know the field L and hence the map $\phi : L \rightarrow K^n$. If we do not have this information, then we will produce s', t' and f' such that $f = t' \circ f' \circ s'$. We will explain this in Section 2.4.3.

From [KS99] and [DGS06a, Lemma A.0.2] we know that

$$\phi^{-1} \circ f \circ \phi(X) = \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} a_{ij} \tilde{X}^{q^i + q^j},$$

for some $a_{ij} \in L$. We also know that

$$\begin{aligned} \phi^{-1} \circ f \circ \phi &= \phi^{-1} \circ (t \circ (\phi \circ \tilde{f} \circ \phi^{-1}) \circ s) \circ \phi \\ &= (\phi^{-1} \circ t \circ \phi) \circ \tilde{f} \circ (\phi^{-1} \circ s \circ \phi), \end{aligned}$$

where the parentheses are added to show the composition consists of three maps defined on L . In particular, we know or can find out by low complexity \tilde{f} (see in Section 2.1 'the private key')

and we can construct $\phi^{-1} \circ f \circ \phi$ from the known f . The Kipnis-Shamir Attack will focus on using the properties of these known maps to find both unknown linear maps $\phi^{-1} \circ s \circ \phi$ and $\phi^{-1} \circ t \circ \phi$. In particular, we will study

$$(\phi^{-1} \circ t^{-1} \circ \phi) \circ (\phi^{-1} \circ f \circ \phi) = \tilde{f} \circ (\phi^{-1} \circ s \circ \phi), \quad (2.4)$$

and the properties of the functions in this formula.

From [DGS06a, Lemma A.0.1] we have the following equations:

$$\begin{aligned} \phi^{-1} \circ t \circ \phi &= \sum_{j=0}^{n-1} t_j X^{q^j} \\ \phi^{-1} \circ t^{-1} \circ \phi &= \sum_{j=0}^{n-1} t_j^{-1} X^{q^j} \\ \phi^{-1} \circ s \circ \phi &= \sum_{j=0}^{n-1} s_j X^{q^j}, \end{aligned}$$

where $s_j, t_j \in L$. So, we can say, our new goal is to find the s_j and t_j , from which we can then construct s and t .

Remark 2.4.1. *We make special note that the notation t_j^{-1} represents the coefficients of X^{q^j} in the polynomial representation of t^{-1} . This is to be distinguished from $(t_j)^{-1}$, the multiplicative inverse of the coefficient of X^{q^j} in the polynomial representation of t . In general these two notations will not refer to the same value in L .*

As we know from linear algebra there is the possibility to translate homogeneous quadratic polynomials into matrices. So for any polynomial $G(X) \in L[X]$ of the form

$$G(X) = \sum_{i=0}^{n-1} \sum_{j=0}^i G_{ij} X^{q^i + q^j},$$

we can associate a unique $n \times n$ symmetric matrix \mathbf{G} defined by

$$[\mathbf{G}]_{ij} = \begin{cases} 2G_{ij} & \text{if } i = j; \\ G_{ij} & \text{if } i > j; \\ G_{ji} & \text{if } i < j. \end{cases}$$

We note that because the characteristic of L is two, the entries on the diagonal of \mathbf{G} are all zero.

With this correspondence between homogeneous quadratic functions on L and $n \times n$ matrices with entries in L we shift from the function point of view to that of matrices.

Now, let \mathbf{F} be the matrix associated with f . Then clearly \mathbf{F} has only two nonzero entries: $[\mathbf{F}]_{0\theta} = 1$ and $[\mathbf{F}]_{\theta 0} = 1$. Further suppose that \mathbf{F}_s is the matrix associated with $\tilde{f} \circ (\phi^{-1} \circ s \circ \phi)$, that \mathbf{F}_ϕ is the matrix associated with $\phi^{-1} \circ f \circ \phi$, and that \mathbf{F}_t is the matrix associated with $(\phi^{-1} \circ t^{-1} \circ \phi) \circ (\phi^{-1} \circ f \circ \phi)$.

From [DGS06a, Lemma 2.4.1] we know that

$$\mathbf{F}_s = \mathbf{S}^T \mathbf{F} \mathbf{S},$$

where the $n \times n$ matrix S is defined by

$$[S]_{ij} = s_{j-i}^{q^i}$$

and $j - i$ is calculated modulo n .

From [DGS06a, Lemma 2.4.2] we know that

$$F_t = \sum_{l=0}^{n-1} t_l^{-1} (F_\phi)_l,$$

where

$$[(F_\phi)_l]_{ij} = [(F_\phi)]_{i-l, j-l}^{q^l},$$

with both $i - l$ and $j - l$ calculated modulo n .

After the translation from Equation (2.4) into matrices we conclude

$$F_s = M = F_t,$$

where M denotes the common value of F_s and F_t .

Clearly, matrix M has rank equal to two. Since S is invertible, we see that $M = S^T F S$ has rank equal to two as well. But this means that the L -linear combination

$$M = \sum_{l=0}^{n-1} t_l^{-1} (F_\phi)_l$$

of the n known matrices $(F_\phi)_0, \dots, (F_\phi)_{n-1}$ has rank two. This construction is common in a more general way:

2.4.2 MinRank problem

Definition 2.4.2. *Given $n \times n$ matrices A_1, A_2, \dots, A_m over a finite field L and $r < n$, find a non-trivial linear combination of*

$$A = \alpha_1 A_1 + \alpha_2 A_2 + \dots + \alpha_m A_m$$

*such that the rank of A is less than or equal to r . This problem is called the **MinRank problem**.*

Kipnis and Shamir presented for this part of the attack a new relinearization method. Later, Courtois proposed a more standard and straightforward method to solve this problem. If the reader is interested in the method of Kipnis and Shamir, he can have a glance at [KS99]. But we will investigate the method of Courtois [Cou01].

Courtois based his method for the MinRank problem on considerations of Coppersmith, Stern and Vaudenay in [CSV93, CSV97].

We treat the A_1, \dots, A_m as known, and the $\alpha_1, \dots, \alpha_m$ as variables. If $A = \alpha_1 A_1 + \alpha_2 A_2 + \dots + \alpha_m A_m$ is to have rank r , then each $(r + 1) \times (r + 1)$ submatrix minor has determinant 0. This means that each $(r + 1) \times (r + 1)$ submatrix yields a total degree $r + 1$ polynomial equation in the m variables $\alpha_1, \dots, \alpha_m$. There are as much as $\binom{n}{r+1}^2$ such equations and we hope that at least about $\binom{n}{r+1}$ of them are linearly independent. We get about $\binom{n}{r+1}$ equations which have $\binom{n}{r+1}$ terms, and are simply linearized and solved by Gaussian reduction.

Shallit, Frandsen and Buss showed in [BFS99], that the MinRank problem in general is NP-complete. But in our case, where $r = 2$, this method above works.

2.4.3 Choice of extension field

In the beginning of the construction of the MI Cryptosystem we defined an extension field $L = K[u]/g(u)$, where we took an irreducible polynomial $g(u) \in K[u]$ of degree n . Now the question is, how we can exactly find this polynomial or if it is necessary to take this one at all. We want to show now, that we can also pick another degree n irreducible polynomial and it works too.

So, suppose we as the attacker have chosen another degree n irreducible polynomial $h(v) \in K[v]$ and construct $L' = K[v]/h(v)$ and $\psi : L' \longrightarrow K^n$. According to each standard book about finite fields, for example [LN86, page 48], any two irreducible polynomials in $K[v]$ of the same degree have isomorphic splitting fields. In the way how we constructed L and L' these are exactly the splitting fields of $g(u)$ respectively $h(v)$. So L and L' are isomorphic, and in fact, a K -linear field isomorphism exists between L and L' . Then we construct such an isomorphism. Let $\alpha \in L'$ be such that

$$g(\alpha(v)) = 0 \pmod{h(v)},$$

and let $\iota : L \longrightarrow L'$ be defined by

$$\iota(p(u)) = p(\alpha(v)) \pmod{h(v)},$$

for $p(u) \in L$.

Let us take the function of the public key of the MI Cryptosystem.

$$f = t \circ \phi \circ \tilde{f} \circ \phi \circ s$$

Now we can describe this function in a slightly other way:

$$\begin{aligned} f &= t \circ \phi \circ \tilde{f} \circ \phi \circ s \\ &= t \circ \phi \circ (\iota^{-1} \circ \iota) \circ \tilde{f} \circ (\iota^{-1} \circ \iota) \circ \phi \circ s \\ &= (t \circ \phi \circ \iota^{-1}) \circ (\iota \circ \tilde{f} \circ \iota^{-1}) \circ (\iota \circ \phi \circ s) \end{aligned}$$

Define $\bar{s} : K^n \longrightarrow L'$, $\bar{t} : L' \longrightarrow K^n$ and $\tilde{f}' : L' \longrightarrow L'$ by

$$\begin{aligned} \bar{s} &= \iota \circ \phi^{-1} \circ s \\ \bar{t} &= t \circ \phi \circ \iota^{-1} \\ \tilde{f}' &= \iota \circ \tilde{f} \circ \iota^{-1}. \end{aligned}$$

Now we consider whether or not there exists s' or t' such that

$$f = t' \circ \psi \circ \tilde{f}' \circ \psi^{-1} \circ s',$$

or equivalently, for a given \bar{s} and \bar{t} , whether or not there exists s' and t' such that

$$\begin{aligned} s' \circ \psi &= \bar{s} \\ \psi^{-1} \circ t' &= \bar{t}. \end{aligned}$$

Solving for s' and t' we see that

$$\begin{aligned} s' &= \bar{s} \circ \psi^{-1} \\ t' &= \psi \circ \bar{t}. \end{aligned}$$

As the attacker we cannot know ι , and thus cannot know \bar{s} and \bar{t} . Therefore we cannot know which s' and t' will be obtained. However, if some s' and t' can be found, then these are just as useful as the original s and t for the attack since we can then easily invert the map $\phi \circ \tilde{f} \circ \phi^{-1}$ anyway as a whole.

2.5 Practical usage of the MI Cryptosystem

There are also variations of the MI Cryptosystem, which improve the security of it. One of this variations was also practically used. Unfortunately, it is cracked in the mean time. Nevertheless we would like to present this system and the attack for it.

Until now we speak only about cryptosystems for encrypting and decrypting the message. Many of the existing cryptosystems can also be used for signing and verifying a message. This version of the MI Cryptosystem we want to present now only works for signing and verifying.

2.5.1 SFLASH^{v2} respectively SFLASH

This section is based on [DGS06a, pages 47-49], [NES04, pages 670-677] and [CGP03].

The New European Schemes for Signatures, Integrity and Encryption project, shortly NESSIE, within the Information Society Technologies Programme of the European Commission made its final selections for cryptographic primitives in 2004. SFLASH^{v2}, a fast multivariate signature scheme, was selected by NESSIE as a security standard for use in low-cost smart cards. We can find this scheme with the additional description *v2* because it exists a previous version. But because it is selected as a standard, often it is just called SFLASH.

SFLASH belongs to the group of the Minus method. The Minus method consists of deleting a few, say r , polynomial components from a given multivariate public key. Let us take the function $f : K^n \rightarrow K^n$ from the MI Cryptosystem. Once we apply the Minus method to f , we will have a new map $f^- : K^n \rightarrow K^{n-r}$ defined by

$$f^-(x_0, x_1, \dots, x_{n-1}) = (f_0, f_1, \dots, f_{n-r-1}).$$

SFLASH uses $\theta = 11$, $n = 37$ and $r = 11$ so that $f^- : K^{37} \rightarrow K^{26}$ is defined by

$$f^-(x_0, x_1, \dots, x_{36}) = (f_0, f_1, \dots, f_{25}),$$

where $f_0, f_1, \dots, f_{25} \in K[x_0, x_1, \dots, x_{36}]$. Further we define the extension L :

$$L = K[u]/(u^{37} + u^{12} + u^{10} + u^2 + 1).$$

The SFLASH scheme has the following structure:

the public key

The public key of SFLASH includes the following:

1. The field $K = \mathbb{F}_{2^7}$, including its additive and multiplicative structure. In particular, $K = \mathbb{F}_2[x]/(x^7 + x + 1)$.
2. The 26 quadratic polynomials $f_0, f_1, \dots, f_{25} \in K[x_0, x_1, \dots, x_{36}]$.

the private key

The following information should be kept private, and is needed in order to generate SFLASH signatures:

1. Δ , a randomly chosen 80-bit long secret key.
2. The two invertible affine transformations s and t .

signature generation

Let m be the message given as a string of bits. The subscripts below refer to the position in the bit string, and \parallel denotes the concatenation of bit strings.

1. Let m_1 and m_2 be two 160-bit strings defined by:

$$\begin{aligned}m_1 &= \text{SHA-1}(m) \\m_2 &= \text{SHA-1}(m_1),\end{aligned}$$

where SHA-1 is a hash standard. For further information about this function, look at [EJ01].

2. Let

$$\begin{aligned}v &= m_1 \parallel (m_{21}, \dots, m_{222}) = (v_1, \dots, v_{182}) \\w &= \text{SHA-1}(v \parallel \Delta) = (w_1, \dots, w_{77}).\end{aligned}$$

3. Let m' be the string of 37 elements of K defined by:

$$\begin{aligned}m' &= (\phi^{-1}(v_1, \dots, v_7), \phi^{-1}(v_8, \dots, v_{14}), \dots, \phi^{-1}(v_{176}, \dots, v_{182}), \\&\quad \phi^{-1}(w_1, \dots, w_7), \phi^{-1}(w_8, \dots, w_{14}), \dots, \phi^{-1}(w_{71}, \dots, w_{77}))\end{aligned}$$

4. Calculate the signature s of m by:

$$s = f^{-1}(m').$$

The pair (m, s) represents the message m with signature s .

signature verification

1. Compute

$$\begin{aligned}m_1 &= \text{SHA-1}(m) \\m_2 &= \text{SHA-1}(m_1) \\v &= m_1 \parallel (m_{21}, \dots, m_{222}) = (v_1, \dots, v_{182})\end{aligned}$$

2. Let

$$n' = (\phi^{-1}(v_1, \dots, v_7), \phi^{-1}(v_8, \dots, v_{14}), \dots, \phi^{-1}(v_{176}, \dots, v_{182}))$$

3. If $n' = f^{-1}(s)$, then accept the signature s as valid; otherwise reject s .

2.5.2 Attack on SFLASH

This section is based on [DFSS07].

Unfortunately Dubois, Fouque, Shamir and Stern achieve in 2007 a total break of SFLASH. Given only the public key a signature for any message can be forged in about one second after a one time computation of several minutes. The asymptotic running time of the attack is $O(\log^2(q)n^6)$ since it only needs standard linear algebra algorithms on $O(n^2)$ variables, and n is typically very small.

Chapter 3

Tame Transformation Method Cryptosystem

The Tame Transformation Method (TTM) Cryptosystem was first presented in April 1997 by Tzuong Tsieng Moh [Moh99c]. The idea of this cryptosystem comes from algebraic geometry. The motivation is based on the difficulty of decomposing a composition of invertible nonlinear polynomial maps.

Additionally, Moh propagates his cryptosystem as very fast [Moh99c, pages 12f]:

The decrypting speed is in general 10 to 20 times faster than the encrypting speed. The PC software TTM 2.5 is faster than a possible hardware implementation for RSA 1024. According to the opinion of certain expert, a couple added instruction about finite field multiplication in the chip architecture would increase the speed of software implementations at least 10-16 times. If it is done, then our software implementations would reach a few million bits per second for the PC and match the speed of the software for secret-key encryptions (DES etc.).

We will present the construction of the TTM Cryptosystem and afterwards the attacks, which exist against the TTM Cryptosystem. But first we have to look at some mathematical definitions.

3.1 Definitions for the tame automorphism

The definitions of this section are based on [DGY99, pages 135f] and [DGS06a, page 138].

Let K be a field and let $K[X] := K[x_1, \dots, x_n]$ be the polynomial ring in n variables x_1, \dots, x_n over K , where n is a fixed positive integer. Let us have a look at the automorphism group $\text{Aut } K[X]$. Let $A(n, K)$ be the subgroup of $\text{Aut } K[X]$ consisting of all affine automorphisms. There also exists another special subgroup of $\text{Aut } K[X]$.

Definition 3.1.1. *Let $a_1, \dots, a_n \in K$ be invertible elements and $h_i \in K[x_{i+1}, \dots, x_n]$, ($i = 1, \dots, n$) be polynomials. (h_n is a polynomial in zero variables, i.e. $h_n \in K$.) An element $\phi \in \text{Aut } K[X]$ is called **de Jonquières automorphism** or sometimes just **triangular automorphism** if the automorphism is in the following way: $\phi(x_i) = a_i x_i + h_i$ ($i = 1, \dots, n$).*

Remark 3.1.2. *The set $J(n, K)$ of all de Jonquières automorphisms of $K[X]$ is a subgroup of $\text{Aut } K[X]$.*

Definition 3.1.3. *The subgroup $T(n, K)$ of $\text{Aut } K[X]$ which is generated by $A(n, K)$ and $J(n, K)$ is called the **group of tame automorphisms**. An element of $T(n, K)$ is called **tame automorphism**.*

*The automorphisms of $K[X]$ which are not in $T(n, K)$ are called **wild**.*

For a long time it was unknown whether a wild automorphism really exists. Nagata formulated in 1972 an automorphism and wrote down as a conjecture that this automorphism is wild [Nag72]. This conjecture was not proven before 2003 [SU03]. Since then we know that wild automorphisms exist.

Let us do some considerations about the group $T(n, K)$. For using a tame automorphism ϕ for a cryptosystem we have to find a way for constructing the inverse of this.

Remark 3.1.4. *The group of tame automorphisms include elements which are composed of affine automorphisms and of de Jonquières automorphisms. So it will do if we only have a look at the components. If it is an affine automorphism, there is no problem to construct an inverse. And because of the form of the de Jonquières automorphisms constructing the inverses work inductively. Let ϕ be a de Jonquières automorphism. We have the inverse $\phi^{-1} = (\phi_1^{-1}, \dots, \phi_n^{-1})$ with $x_n = \phi_n^{-1}(y_1, \dots, y_n) = y_n$ and $x_i = \phi_i^{-1}(y_1, \dots, y_n) = y_i - h_i(\phi_{i+1}^{-1}(y_1, \dots, y_n), \dots, \phi_n^{-1}(y_1, \dots, y_n))$, for $i = n - 1, \dots, 1$.*

This remark we can find in [Moh99c, Proposition 1].

3.2 Construction of the Tame Transformation Method Cryptosystem

As already mentioned Moh presented the TTM Cryptosystem first in 1997. Since then Moh published several papers with different instances of the TTM Cryptosystem. That is why we first introduce the general system and afterwards we present the instances and particularly the similarities of these.

3.2.1 Principle of the algorithm

This subsection is based on [Moh99c, Section 4].

Let $n + r \geq 3$, l a positive integer and K a field of 2^l elements. Let us select k tame automorphisms ϕ_k, \dots, ϕ_2 and ϕ_1 of K^{n+r} . Let $\pi = \phi_k \circ \dots \circ \phi_2 \circ \phi_1 = (\pi_1, \dots, \pi_{n+r})$. Let $\hat{\pi} = (\pi_1(x_1, \dots, x_n, 0, \dots, 0), \dots, \pi_{n+r}(x_1, \dots, x_n, 0, \dots, 0))$, and further it follows $f_i(x_1, \dots, x_n) = \pi_i(x_1, \dots, x_n, 0, \dots, 0)$ for $i = 1, \dots, n + r$.

the public key

The public key of the TTM Cryptosystem includes the following:

1. The field K of 2^l elements.
2. The map $\hat{\pi} = (f_1, \dots, f_{n+r}) : K^n \mapsto K^{n+r}$.

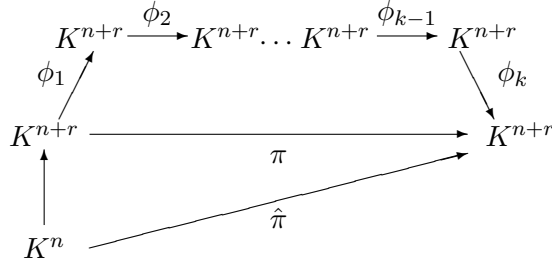


Figure 3.1: composition of maps of the TTM Algorithm

the private key

The private key consists of:

The set of inverse maps $\{\phi_1^{-1}, \dots, \phi_k^{-1}\}$ of the tame automorphisms.

encryption

Given a plaintext $(x'_1, \dots, x'_n) \in K^n$. The sender evaluates $y'_i = f_i(x'_1, \dots, x'_n)$. Then the ciphertext will be $(y'_1, \dots, y'_{n+r}) \in K^{n+r}$.

decryption

In the other way, the legitimate receiver recovers the plaintext by $(x'_1, \dots, x'_n, 0, \dots, 0) = \phi_1^{-1} \circ \dots \circ \phi_k^{-1}(y'_1, \dots, y'_{n+r})$. This works because of Remark 3.1.4.

Moh gave in his papers more concrete instructions how to construct such a TTM Cryptosystem.

3.2.2 Components of the Tame Transformation Method Cryptosystem

First of all we want to think about the k tame automorphisms. k is normally 4, where ϕ_1 and ϕ_4 are affine linear maps. ϕ_2 and ϕ_3 are tame automorphisms, which are not affine linear maps.

About Moh's presentation of ϕ_2 and ϕ_3 in [Moh99b] and [MCY04] we can say they look in the following way:

One considers the plaintext space as the standard affine subspace K^n (i.e., all points with the last r coordinates being zeros) embedded in K^{n+r} which is the cipherspace. Let us select the following tame automorphism $\hat{\phi}_2$,

$$\hat{\phi}_2 = \begin{cases} y_1 = x_1 \\ y_2 = x_2 + g_2(x_1) \\ y_3 = x_3 + g_3(x_1, x_2) \\ \vdots \\ y_n = x_n + g_n(x_1, \dots, x_{n-1}) \\ y_{n+1} = x_{n+1} + g_{n+1}(x_1, \dots, x_n) \\ \vdots \\ y_{n+r} = x_{n+r} + g_{n+r}(x_1, \dots, x_{n+r-1}) \end{cases}$$

Let us restrict $\hat{\phi}_2$ to the subspace K^n , i.e., set $x_{n+1} = \dots = x_{n+r} = 0$ in the above equations, and call it ϕ_2 .

$$\phi_2 = \begin{cases} y_1 = x_1 \\ y_2 = x_2 + h_2(x_1) \\ y_3 = x_3 + h_3(x_1, x_2) \\ \vdots \\ y_n = x_n + h_n(x_1, \dots, x_{n-1}) \\ y_{n+1} = 0 + h_{n+1}(x_1, \dots, x_n) \\ \vdots \\ y_{n+r} = 0 + h_{n+r}(x_1, \dots, x_n) \end{cases}$$

In the ultimate cause, we cannot say anymore ϕ_2 is an automorphism. But if we concretely build in an embedding from K^n to K^{n+r} , then we can speak again about an automorphism. Further we have to say that $p\hat{h}i_2$ and ϕ_2 are not tame strictly speaking. These maps are builded analog to a tame automomorphsm. These maps are lower-triangular in contrast to tame automorphisms which are upper-triangular. But the next automorphism is also strictly speaking a tame automorphism.

We further select suitable polynomials p_1, \dots, p_m with $1 < m < n + r$ (normally m is small, even $m < n$) to construct ϕ_3 ,

$$\phi_3 = \begin{cases} y_1 = x_1 + p_1(x_2, \dots, x_{n+r}) \\ \vdots \\ y_m = x_m + p_m(x_{m+1}, \dots, x_{n+r}) \\ y_{m+1} = x_{m+1} \\ \vdots \\ y_{n+r} = x_{n+r} \end{cases}$$

such that the composition $\phi_3 \circ \phi_2$,

$$\phi_3 \circ \phi_2 = \begin{cases} y_1 = x_1 + p_1(\phi_{2,2}, \dots, \phi_{2,n+r}) \\ y_2 = x_2 + h_2(x_1) + p_2(\phi_{2,3}, \dots, \phi_{2,n+r}) \\ \vdots \\ y_m = x_m + h_m(x_1, \dots, x_{m-1}) + p_m(\phi_{2,m+1}, \dots, \phi_{2,n+r}) \\ y_{m+1} = x_{m+1} + h_{m+1}(x_1, \dots, x_m) \\ \vdots \\ y_n = x_n + h_n(x_1, \dots, x_{n-1}) \\ y_{n+1} = 0 + h_{n+1}(x_1, \dots, x_n) \\ \vdots \\ y_{n+r} = 0 + h_{n+r}(x_1, \dots, x_n) \end{cases}$$

will have the following properties

1. all y_i 's are of the same degree d in x_1, \dots, x_n , normally $d = 2$.
2. all highest degree forms are linearly independent.
3. the degree of ϕ_3 should be high enough and the polynomials $f_2, \dots, f_{n+r}, p_1, \dots, p_m$ should be general enough so that there is no way to recover them by *brute force*.

The public key map π (and therefore $\phi_3 \circ \phi_2$) should be of small degree (ideally 2) for a reasonable key size. It is a very difficult problem to achieve this while maintaining security (see also Section 3.3 'Mathematical background').

Furthermore in [Moh99c, pages 6 and 7] Moh wrote about ϕ_1 and ϕ_4 the following:

An invertible linear transformation $\phi_1 = (\phi_{1,1}, \dots, \phi_{1,n+r})$ has got the following restriction:

$$\phi_{1,i} = \sum_{j=1}^{n+r} a_{i,j} x_j + b_i$$

with the the conditions:

1. For $i = 1, \dots, n$, we always have $b_i \neq 0$, $a_{i,j} = 0$, for $j = n + 1, \dots, n + r$ and at least half of the remaining $a_{i,j}$ are non-zero.
2. For $i = n + 1, \dots, n + r$, we always have $\phi_{1,i} = x_i$.

ϕ_4 should be an invertible transformation satisfying the following condition, where $\pi = (\pi_1, \dots, \pi_{100})$,

$$\pi = \phi_4 \circ \phi_3 \circ \phi_2 \circ \phi_1, \text{ and } \pi_i(0, \dots, 0) = 0.$$

These conditions are general enough to construct many different keys for the TTM Cryptosystem. But the existence of weak keys is not excluded by these conditions.

That is why Moh gives in many of his papers more conditions. Primarily in the beginning he constructed a component Q_8 . (Look at [Moh99c, Section 5].) About this component Moh wrote:

This component does **not** need to vary according to users. It can be made as part of the hardware.

But sometimes it looks like this statement would not be so strong. If someone presented an attack, Moh responded to this with a paper in which he stated that this is only an attack on one implementation and then he presented another Q_8 . (Look at [MC01] and at [MCY04].)

So let us have a look how the component Q_8 is built in principal and where Moh implements it in the TTM Cryptosystem.

Definition 3.2.1. *Let q_1, \dots, q_s be polynomials in variables x_1, \dots, x_t . Let $l(x_1, \dots, x_t)$ be a polynomial. If*

$$Q(q_1(x_1, \dots, x_t), \dots, q_s(x_1, \dots, x_t)) = l(x_1, \dots, x_t),$$

*then Q is called a **generating polynomial** of l (over q_1, \dots, q_s). Furthermore, if Q is of the minimal degree among all possible generating polynomials of l , then it is called a **minimal generating polynomial** of l , and its degree is called the **generating degree** of l , in symbol $\text{gendeg}(l)$. If there is no such polynomial Q , then we define $\text{gendeg}(l) = \infty$.*

Moh constructs s polynomials q_i and then a minimal polynomial of a monomial of degree 8 in q_i , which he calls Q_8 .

Now Moh implements the component Q_8 as the p_j ($1 \leq j \leq m$) of ϕ_3 and varies the p_j by inserting the result ϕ_2 in different orders.

3.2.3 Instances of the Tame Transformation Method Cryptosystem

Over the years Moh presented many different versions of his system. Sometimes it is not so clear which version Moh exhibits in a paper. Most changes of version are a result of the fact that there exists an attack. As we can see in his papers, Moh changes in these versions several components which seemed to be fix in the first version. So he often presents a new Q_8 , he varies the m from ϕ_3 and also the global n and r . And surely he changes the $\phi_{2,i}$.

As it is also the case in the MI Cryptosystem the user has mainly to select the affine maps ϕ_1 and ϕ_4 . It could be possible that the user also has to choose ϕ_2 and ϕ_3 . But this would be a bit more difficult because as we saw these two maps have to fulfill some conditions which are not satisfied a priori.

We will show some of the differences of the versions at the end of this chapter in Section 3.6 'Historical background'.

But here we want to present the concrete parameters of the first version: (The interested reeader who wants to see the polynomials in detail may have a look at [Moh99c] and [DGS06a, pages 142-146].) $n = 64$, $r = 36$ and so $n + r = 100$. $K = \mathbb{F}_{2^8}$. The $\phi_{2,i}$ for $i = 1, \dots, n + r$ are specially chosen quadratic polynomials over K . Further $m = 2$ and the polynomials p_1 and p_2 are chosen in a special way from Moh such that $(\phi_3 \circ \phi_2)_1$ and $(\phi_3 \circ \phi_2)_2$ both are quadratic in the variables x_1, \dots, x_n when the last r variables x_{n+1}, \dots, x_{n+r} are set equal to zero.

3.3 Mathematical background of the Tame Transformation Method Cryptosystem

There are some interesting considerations why the TTM Cryptosystem is constructed in this way.

We use the notations and definitions of Section 2.2.1.

The following thoughts are based on [Moh99b, Section 2].

As we already saw in the introduction and in the chapter about the MI Cryptosystem, we work with polynomials of degree two except when we explicitly choose an affine polynomial.

Let us do again some considerations about tame automorphism. So we can say that the degree of a tame automorphism which is used in a TTM Cryptosystem is 2. But if we directly construct an inverse of such a tame automorphism, then the degree of the inverse map could be as high as 2^{n-1} , where n is the fixed positive integer of Section 3.1. We know from mathematics that a polynomial of degree d in n variables may have $\binom{d+n}{n}$ terms. For example if $n = 100$ and $d = 2^{99}$, this number is 10^{2925} . So it is physically impossible to compute the inverse formula for a tame automorphism. If such a tame automorphism is included in a composition, it offers some protection for the public key system.

But as we saw in Section 3.1, if we know each component of the composition map, we can calculate inductively the input of the tame automorphism in a very fast way.

If we look at the examples of the TTM Cryptosystems in the papers of Moh, we can observe that the composition map is of degree 2. Now we can ask us: How is it possible that the composition of several maps has degree 2?

Moh claims in [MC01] the following:

It can be done for a composition of mappings, as in the case of TTM, that the degrees of mappings (which is defined for every mapping ϕ_i) to be the maximum of the degrees of all coordinate polynomials $\{\phi_{i,j}\}$ may not increase after compositions, i.e., say, if all $\phi_{i,j}$'s are of degree two or less, then their composition $\prod \phi_i$ may have coordinate polynomials of degree two or less.

Further explanation Moh gave in [Moh99b]:

The question is as follows: Can we find suitable polynomials of some degree d ;

$$h_{n+1}(x_1, \dots, x_n), \dots, h_{n+r}(x_1, \dots, x_n)$$

such that $P(h_{n+1}, \dots, h_{n+r}), Q(h_{n+1}, \dots, h_{n+r})$ are of the same suitable degree d ?
From the point of view of Algebra, does there exist P, Q such that

- (a) : $P, Q \in K[h_{n+1}, \dots, h_{n+r}] \subseteq K[x_1, \dots, x_n]$
- (b) : $\deg(P) = \deg(Q) = d = \deg h_i$ for all i ?

Certainly, an Algebraist will answer that there are infinitely many such polynomials.

But this does not explain how we practically construct such polynomials.

Moh found a special way of selecting P, Q and g_i such that $\phi_{2,1}$ and $\phi_{2,2}$ both are quadratic in the variables x_1, \dots, x_n when the last v variables x_{n+1}, \dots, x_{n+r} are set equal to zero.

3.4 Minrank Attack of Goubin and Courtois

Goubin and Courtois presented in 2000 an attack [GC00], which attacks the version of TTM, which was the newest at that time [Moh99c]. We presented in the end of Section 3.2.3 the parameter of this version of TTM.

Goubin and Courtois did not directly show an attack against TTM, but they construct a cryptosystem which contains the TTM Cryptosystem. They call it TPM.

So let us first see why they call it in this way. Then we will have a look how the TPM Cryptosystem would be constructed. Afterwards we understand why the TTM version of that time was a part of TPM. And in the end, we will see how we can attack the TPM Cryptosystem.

This section is based on [GC00].

3.4.1 TPM

By the idea of Fell and Diffie, Goubin and Courtois took the triangular construction, where it uses equations that involve $1, 2, \dots, n$ variables and are solved sequentially [FD86]. We call T this triangular construction. Let TPM (T Plus-Minus) be T with added final u random (fullsize) quadratic polynomials, and with v of the beginning equations removed.

In a general way, we say $\text{TPM}(n, u, v, K)$, with:

- n, u, v integers such that $v \leq n$. We also systematically put $m = n + u - v$.
- K a finite field.

We first consider a function $\psi : K^n \longrightarrow K^m$ such that $(y_1, \dots, y_m) = \psi(x_1, \dots, x_n)$ is defined by the following system of equations:

$$\psi = \begin{cases} y_1 = x_1 + g_1(x_{n-v+1}, \dots, x_n) \\ y_2 = x_2 + g_2(x_1, x_{n-v+1}, \dots, x_n) \\ y_3 = x_3 + g_3(x_1, x_2, x_{n-v+1}, \dots, x_n) \\ \vdots \\ y_{n-v} = x_{n-v} + g_{n-v}(x_1, \dots, x_{n-v-1}, x_{n-v+1}, \dots, x_n) \\ y_{n-v+1} = g_{n-v+1}(x_1, \dots, x_n) \\ \vdots \\ y_{n-v+u} = g_{n-v+u}(x_1, \dots, x_n) \end{cases}$$

with each g_i ($1 \leq i \leq m (= n - v + u)$) being a randomly chosen quadratic polynomial.

As we are used by TTM, the user selects a random invertible affine transformation $s : K^n \longrightarrow K^n$, and a random invertible affine transformation $t : K^m \longrightarrow K^m$. Let $F = t \circ \psi \circ s$. By construction, if we denote $(y'_1, \dots, y'_m) = F(x'_1, \dots, x'_n)$, we obtain an explicit set $\{P_1, \dots, P_m\}$ of m quadratic polynomials in n variables, such that:

$$\psi = \begin{cases} y'_1 = P_1(x'_1, \dots, x'_n) \\ \vdots \\ y'_m = P_m(x'_1, \dots, x'_n) \end{cases}$$

the public key

The public key of a $\text{TPM}(n, u, v, K)$ Cryptosystem consists of:

1. The field K .
2. The set $\{P_1, \dots, P_m\}$ of m quadratic polynomials in n variables.

the private key

1. The affine transformations s and t .
2. The functions g_i , $1 \leq i \leq m$.

encryption

Given a plaintext $(x'_1, \dots, x'_n) \in K^n$, the sender computes $y'_i = P_i(x'_1, \dots, x'_n)$ for $1 \leq i \leq m$ and sends the ciphertext $(y'_1, \dots, y'_m) \in K^m$.

decryption

Given a ciphertext $(y'_1, \dots, y'_m) \in K^m$, the legitimate receiver recovers the plaintext by the following method:

1. Compute $(y_1, \dots, y_m) = t^{-1}(y'_1, \dots, y'_m)$.
2. Make an exhaustive search on the v -tuple $(x_{n-v+1}, \dots, x_n) \in K^v$, until the n -tuple (x_1, \dots, x_n) obtained by $x_i = y_i - g_i(x_1, \dots, x_{i-1}; x_{n-v+1}, \dots, x_n)$ (for $1 \leq i \leq n-v$) satisfies the u following equation $g_i(x_1, \dots, x_n) = y_i$ (for $n-v+1 \leq i \leq m$).
3. For the obtained (x_1, \dots, x_n) n -tuple, get $(x'_1, \dots, x'_n) = s^{-1}(x_1, \dots, x_n)$.

3.4.2 TTM belongs to TPM

We can see that the first version of TTM — we saw the parameters in the end of Section 3.2.3 — is a particular case of the general family TPM. ϕ_1 relates to s , ϕ_4 to t and $\phi_3 \circ \phi_2$ together of the TTM to ϕ of the TPM. As a result, TTM belongs to $\text{TPM}(64, 38, 2, \mathbb{F}_{2^8})$.

3.4.3 Attack against TPM

In each equation $y_i = x_i + g_i(x_1, \dots, x_{i-1}; x_{n-v+1}, \dots, x_n)$ ($1 \leq i \leq n-v$), the homogeneous part is given by ${}^t X A_i X$, with ${}^t X = (x_1, \dots, x_n)$, A_i being a (secret) matrix. Similarly, in each public equation $y'_i = P_i(x'_1, \dots, x'_n)$ is given by ${}^t X' M_i X'$, with ${}^t X' = (x'_1, \dots, x'_n)$, M_i being a (public) matrix.

The fact that $(x_1, \dots, x_n) = s(x'_1, \dots, x'_n)$ and $(y'_1, \dots, y'_m) = t(y_1, \dots, y_m)$ implies that there exist an invertible $n \times n$ matrix S and an invertible $m \times m$ matrix T such that:

$$\begin{pmatrix} {}^t(SX')A_1(SX') \\ \vdots \\ {}^t(SX')A_m(SX') \end{pmatrix} = T^{-1} \begin{pmatrix} {}^tX'M_1X' \\ \vdots \\ {}^tX'M_mX' \end{pmatrix}$$

Let $T^{-1} = (t_{ij}^{-1})_{1 \leq i, j \leq m}$. We thus have, for any X' :

$${}^tX'({}^tSA_iS)X' = {}^tX' \left(\sum_{j=1}^m t_{ij}^{-1} M_j \right) X'$$

so that:

$$\forall i, 1 \leq i \leq m, \sum_{j=1}^m t_{ij}^{-1} M_j = {}^tSA_iS.$$

From the construction of $\text{TPM}(n, u, v, K)$, we have $\text{Rank}(A_1) \leq v$. Since S is an invertible matrix, we have $\text{Rank}(A_1) = \text{Rank}({}^tSA_1S)$ and thus $\text{Rank} \left(\sum_{j=1}^m t_{1j} M_j \right) \leq v$. So there exists a Minrank problem as we already met in Section 2.4.2.

With a good probability, we can suppose that:

$$\sum_{j=1}^m \lambda_j M_j = \mu {}^tSA_1S \quad (\mu \in K^*).$$

Then we deduce the vector space $V_0 = S^{-1}(K^{n-v} \times \{0\}^v)$ (corresponding to $x_{n-v+1} = \dots = x_n = 0$) and $W_0 = S^{-1}(\{0\}^{n-v} \times K^v)$ (corresponding to $x_1 = \dots = x_{n-v} = 0$) by simply noticing that $V_0 = \text{Im} \left(\sum_{j=1}^m \lambda_j M_j A_1 \right)$ and $W_0 = \text{Ker} \left(\sum_{j=1}^m \lambda_j M_j A_1 \right)$.

Once we have found V_0 and W_0 , we can easily deduce the vector space $V_1 = S^{-1}(\{0\} \times K^{n-v-1} \times \{0\}^v)$ of dimension $n - v - 1$ (corresponding to $x_1 = x_{n-v+1} = \dots = x_n = 0$) and $W_1 = S^{-1}(K \times \{0\}^{n-v-1} \times K^v)$ (corresponding to $x_2 = \dots = x_{n-v} = 0$): we just look for coefficients $\alpha_1, \dots, \alpha_n, \beta_1, \dots, \beta_m$ such that the following equation:

$$\sum_{j=1}^m \beta_j y_j' = \sum_{i=1}^n \alpha_i x_i + \delta,$$

holds for any element of V_0 . This can be obtained by simple Gaussian reduction. We also obtain the quadratic function g_2 by Gaussian reduction.

By repeating these steps, we obtain two sequences of vector spaces:

$$V_0 \supseteq V_1 \supseteq V_2 \supseteq \dots \supseteq V_{n-v-1}$$

$$W_0 \subseteq W_1 \subseteq W_2 \subseteq \dots \subseteq W_{n-v-1}$$

At the end, we have completely determined the secret transformations s and t , together with the secret functions g_i . As a result, this algorithm completely breaks the TPM family of cryptosystems.

Goubin and Courtois also presented special case attacks. The interested reader can consult [GH00].

3.5 Attack of Ding and Schmidt

This section is based on [DS03].

Ding and Schmidt discovered a defect in many TTM Cryptosystems in 2003. The latest TTM version at that time worked with $m = 4$, where m is the index number of ϕ_3 in Section 3.2.2. The key idea for the attack came from the observation that it is possible to extend the linearization method by Patarin [Pat00] to attack the TTM implementation schemes.

3.5.1 Mathematical background

As we already saw at Patarin's Attack in Section 2.3, we build linearization equations in the form of

$$\sum_{i=1, j=1}^{n, n+r} a_{ij} x_i y_j + \sum_{i=1}^n b_i x_i + \sum_{j=1}^{n+r} c_j y_j + d = 0, \quad (3.1)$$

where $x = (x_1, \dots, x_n)$ is the cleartext and $y = (y_1, \dots, y_{n+r})$ is the corresponding ciphertext calculated by $\hat{\pi}(x)$. In this set of equations we find enough many, say l , linearly independent linear equations satisfied by x_i to perform a substitution of this l linear equations into y_i , which makes quadratic polynomials y_i with l fewer variables, which we denote by $(x_{v_1}, \dots, x_{v_{n-l}})$. The procedure of the substitution of the l linear equations normally eliminates ϕ_3 of the TTM

Cryptosystem, as the interested reader can see in [DS03], because l is bigger than m . For the rest it becomes straightforward because of the triangular form of ϕ_2 and it is accomplished by an iteration of the procedure of first search for linear equations by linear combinations and then linear substitution. Finally the plaintext can be derived by substituting the solution of the values of $(x_{v_1}, \dots, x_{v_{n-l}})$ into the original l linear equations.

So, let V denote the linear space of the linearization equations (3.1).

3.5.2 Construction of the Attack of Ding and Schmidt

Step 1: We look for a basis for the space V , namely the basis of solutions of a_{ij} , b_i , c_j and d for the equations (3.1). As at Patarin's Attack we choose randomly values $(\bar{x}_1, \dots, \bar{x}_n)$, calculate by the public key $\bar{y}_i = \hat{\pi}_i(\bar{x}_1, \dots, \bar{x}_n)$ for $i = 1, \dots, n+r$ and find so a_{ij} , b_i , c_j and d .

Step 2: For a given ciphertext (y'_1, \dots, y'_{n+r}) , we substitute the polynomials of y_i by y'_i into linearly independent solutions of the linearization equations in V and derive l linearly independent linear equations of x_i by the Gaussian elimination method in the form of $x_{u_j} = h_j(x_{v_1}, \dots, x_{v_{n-l}})$, where h_j is a linear function, $A = \{u_1, \dots, u_l\}$, $B = \{v_1, \dots, v_{n-l}\}$, $A \cap B = \emptyset$ and $A \cup B = \{1, \dots, n\}$. We then substitute them into y_i to make it into polynomials with only $n-l$ variables $\{v_1, \dots, v_{n-l}\}$.

Step 3: For the new $n+r$ polynomials with $n-l$ variables, which we denote by \hat{y}_{1i} , we will write down first the $n+r$ equations $\hat{y}_{1i} - y'_i = \tilde{y}_{1i}(x_{v_1}, \dots, x_{v_{n-l}}) = 0$, and they are linearly dependent and the dimension of it we denote by d .

For the equations $\tilde{y}_{1i}(x_{v_1}, \dots, x_{v_{n-l}}) = 0$ for $i = 1, \dots, n+r$, we will use Gaussian elimination method, first on the quadratic terms, to derive d linearly independent equations $\hat{y}_{1i}(x_{v_1}, \dots, x_{v_{n-l}}) = 0$, for $i = 1, \dots, n+r$, and the last one is linear. Then we take the linear equation out and substitute it back into the leftover $d-1$ quadratic equations (the linear is taken out) $\hat{y}_{1i}(x_{v_1}, \dots, x_{v_{n-l}}) = 0$ for $i = 1, \dots, n+r-1$. We denote the new equations $\tilde{y}_{2i}(x_{v_1}, \dots, x_{v_i}, x_{v_{i+2}}, \dots, x_{v_{n-l}}) = 0$. Then we repeat the same process on these new equations, and later again and again for total $n-l$ times. We then collect all the $n-l$ linear equations derived in this process, a set of $n-l$ linearly independent equations in the triangular form. The solution gives us all the values of x_{v_i} , then we plug them back into l linear equations $x_{u_j} = h_j(x_{v_1}, \dots, x_{v_{n-l}})$ in Step 2, which will give us x_{u_i} . We recover the plaintext.

For step 1 Ding and Schmidt found a way to accelerate these computations. The interested reader may have a look at [DS03, bottom of page 7].

Ding and Schmidt also explicate in [DS03], that this attack not always works. Not all the time it can be found enough linear independent equations such that l is bigger than m .

3.6 Historical development of the Tame Transformation Method Cryptosystem

This section is based on [Moh07, Section 2].

As already suggested, there exist different versions of the TTM Cryptosystem. This is because attacks appeared, which could attack the most actual version at that time, and Moh improved afterwards his system again and again. We want to give an overview about these versions:

1999 Moh presented his first version of the TTM Cryptosystem. Correctly we have to say Moh presented the system already in April 1997. But the first paper came out in 1999 [Moh99c]. In this version the number m of ϕ_3 was 2. Moh did not choose 1 because in private communications with Sathaye and Montgomery, they found a rank attack for $m = 1$.

2000 Goubin and Courtois showed an attack, called Minrank Attack [GC00].

2001 Moh answered by a new version. He increased m from 2 to 4 [MC01].

2003 Ding and Schmidt came up with a further attack against the TTM Cryptosystem [DS03].

2004 The reaction of Moh was a further increase of the number m namely > 10 [MCY04].

2006 Nie, Hu, Li, Updegrave and Ding found an attack which also can break the version of 2004. For the attack they have to know the map ϕ_3 [NHL⁺].

2007 Moh presented examples where the attack of 2006 does not work. Moh mainly presented only the map $\phi_3\phi_2$ together and not the maps ϕ_2 and ϕ_3 separately [Moh07].

2008 Nie, Jiang, Hu, Ding and Zhang brought to light that Instance I in paper [Moh07] can be broken and even in an easier way than all other instances of the TTM Cryptosystem which existed before [NJH⁺08].

Chapter 4

Direct Attacks

Alongside the attacks which specifically exist for specific MPKCs there are also such ones which attack all MPKCs. What is the basic concept of these attacks? They use methods that will enable one to 'directly' solve the equation $F(x_1, \dots, x_n) = (y'_1, \dots, y'_n)$, which we already saw in Chapter 1, where $x_1, \dots, x_n \in K$ represent the searched plaintext and $y'_1, \dots, y'_n \in K$ the given ciphertext. So we can say, the attacker automatically has the set of equations

$$\begin{aligned} f_1(x_1, \dots, x_n) &= y'_1 \\ &\vdots \\ f_n(x_1, \dots, x_n) &= y'_n, \end{aligned}$$

which is the same as the set of equations

$$\begin{aligned} f_1(x_1, \dots, x_n) - y'_1 &= 0 \\ &\vdots \\ f_n(x_1, \dots, x_n) - y'_n &= 0. \end{aligned} \tag{4.1}$$

As we mentioned in Chapter 1 the task of solving a set of multivariate polynomial equations over a finite field, in general, is an NP-complete problem. Therefore the attacks respectively algorithms can just efficiently solve a subset of all possible inputs. Whereas it must be pointed out that the actual cryptosystems also use only a subset of all possible equations.

Now, let us see these general attacks.

4.1 Gröbner bases method

Buchberger introduced in 1965 what is known as Buchberger's Algorithm, which result is a special basis. He named it Gröbner basis after his advisor Wolfgang Gröbner [Buc65].

The attack we want to present uses an improved algorithm, which is based on Buchberger's Algorithm. So first we do a little introduction to Gröbner bases and to Buchberger's Algorithm and then we show two other algorithms. Afterwards in Section 4.1.5 we will see why it makes sense to construct such Gröbner bases.

This section is based on [KR05], on [DGS06a, pages 194-218], on [Ste06] and also on [Fau02].

4.1.1 Gröbner bases

Let K be a finite field and let $R = K[x_1, \dots, x_n]$.

This introduction mainly consists of some definitions. For a more elaborate introduction the reader may have a look at [KR05].

Definition 4.1.1. A term $t \in R$ is a monomial where the leading coefficient is equal to 1.

Definition 4.1.2. Let S be a set and $\cdot : S \times S \rightarrow S$ a binary operation. S is a **monoid** if the operation is associative and has a neutral element.

Definition 4.1.3. A relation on S is a subset $\sigma \subseteq S \times S$. If $t_1, t_2 \in S$ and $(t_1, t_2) \in \sigma$, we write $t_1 \geq_\sigma t_2$.

Definition 4.1.4. A monoid ordering σ is a relation on a monoid (S, \cdot) such that for all $t_1, t_2, t_3 \in S$:

1. $t_1 \geq_\sigma t_2$ or $t_2 \geq_\sigma t_1$.
2. $t_1 \geq_\sigma t_1$.
3. If $t_1 \geq_\sigma t_2$ and $t_2 \geq_\sigma t_1$, then $t_1 =_\sigma t_2$.
4. If $t_1 \geq_\sigma t_2$ and $t_2 \geq_\sigma t_3$, then $t_1 \geq_\sigma t_3$.
5. If $t_1 \geq_\sigma t_2$, then $t_1 t_3 \geq_\sigma t_2 t_3$.
6. $t_1 \geq_\sigma 1$.

Remark 4.1.5. (T_n, \cdot) is a monoid whereas $T_n = \{\text{Terms of } R\}$. In the future we will also speak about a **term ordering**. At that time we mean the monoid ordering on T_n .

Remark 4.1.6. There exist different monoid orderings such as lexicographic order (*lex*), degree lexicographic order (*deglex*) and degree reverse lexicographic order (*degrevlex*).

Definition 4.1.7. Let $I \subseteq R$, I ideal. I is a **monomial ideal** if $I = ((m_i)_{i \in J})$ whereas J is an index set and m_i are monomials.

Definition 4.1.8. Let $f \in R$, $f = \sum_{t \in T_n} c_t t$ whereas $c_t \in K$ and $f \neq 0$. And let σ be a term ordering on T_n .

$\text{supp}(f) := \{t \in T_n : c_t \neq 0\}$ is called **support** of f or set of terms of f .

The **leading term** of f is $\text{LT}_\sigma(f) := \max_\sigma \text{supp}(f)$.

The **leading monomial** of f is $\text{LM}_\sigma(f) := c_{\text{LT}_\sigma(f)} \cdot \text{LT}_\sigma(f)$.

Definition 4.1.9. Let $I \subseteq R$ be an ideal and σ a term ordering. $\text{LT}_\sigma(I) := ((\text{LT}_\sigma(f))_{f \in I})$ is the **leading term ideal** of I with respect to σ .

Notation 4.1.10. Let be $G \subseteq R$ a subset. $\text{LT}_\sigma(G) = \{\text{LT}_\sigma(g) : g \in G\}$.

Definition 4.1.11. Let be $I \subseteq R$ an ideal, σ a term ordering, $G \subseteq I$ a subset such that $(\text{LT}_\sigma(G)) = \text{LT}_\sigma(I)$. Then G is a **Gröbner basis** of I with respect to σ .

4.1.2 Buchberger's Algorithm

Before we can present Buchberger's Algorithm, we have to adopt some mathematical thoughts:

Definition 4.1.12. Let $f, g \in R \setminus \{0\}$. The **S-polynomial** of f and g with respect to a term ordering σ is

$$\text{spoly}(f, g) := \frac{\text{lcm}(\text{LT}_\sigma(f), \text{LT}_\sigma(g))}{\text{LM}_\sigma(f)} f - \frac{\text{lcm}(\text{LT}_\sigma(f), \text{LT}_\sigma(g))}{\text{LM}_\sigma(g)} g.$$

And Buchberger formulated the following theorem [Buc76, Theorem 3.3]:

Theorem 4.1.13. $G = \{g_1, \dots, g_s\} \subseteq R$ is a Gröbner basis of (G) with respect to a term ordering σ if and only if $\text{spoly}(g_i, g_j) = 0 \pmod{G}$ for all i, j .

The theorem can be translated immediately into an algorithm:

Algorithm 4.1.1: Buchberger's Algorithm

Input: $F = \{f_1, \dots, f_n\}$, σ term ordering

Output: G Gröbner basis of F

$m \leftarrow n$;

$g_i \leftarrow f_i$ for $1 \leq i \leq m$;

$P \leftarrow \{(g_i, g_j) \mid 1 \leq i < j \leq m\}$;

while $P \neq \emptyset$ **do**

 pick $(p, q) \in P$;

$P = P \setminus \{(p, q)\}$;

$s \leftarrow \text{spoly}(p, q)$;

$h \leftarrow \text{NormalForm}(s, \{g_1, \dots, g_m\})$;

if $h \neq 0$ **then**

$m = m + 1$;

$g_m \leftarrow h$;

$P = P \cup \{(g_i, g_m) : 1 \leq i < m\}$;

end

end

return $G = \{g_1, \dots, g_m\}$

Remark 4.1.14. With the operation `NormalForm` we denote the operation that s is reduced modulo $\{g_1, \dots, g_m\}$ to a polynomial which is not more reducible modulo $\{g_1, \dots, g_m\}$.

This algorithm was a milestone. But such a Gröbner basis was not unique. So let us state the following theorem for finding a unique basis:

Theorem 4.1.15. Let $I \subseteq R$ be an ideal and σ be the term ordering on R . Then there exists exactly one reduced Gröbner basis G of I with respect to σ .

Buchberger already advert to this in 1965 [Buc65].

To understand this theorem we need two further definitions:

Definition 4.1.16. Let $I \subseteq R$, σ a term ordering, G a Gröbner basis for I with respect to σ . G is **minimal**, if $\text{LT}_\sigma(G)$ are a minimal system of generators for $\text{LT}_\sigma(I)$.

Definition 4.1.17. Let $G = \{g_1, \dots, g_s\} \subseteq R$ be a Gröbner basis of I with respect to σ . G is **reduced** if the following conditions are satisfied:

1. $\text{LC}_\sigma(g_i) = 1$ for all i .
2. G is minimal.
3. $\text{supp}(g_i) \cap \text{LT}_\sigma(G \setminus \{g_i\}) = \emptyset$ for all i .

With the theorem above we can rebind a first improvement in Buchberger's Algorithm:

Algorithm 4.1.2: improved Buchberger's Algorithm

Input: $F = \{f_1, \dots, f_n\}$, σ term ordering

Output: G reduced Gröbner basis of F

$G \leftarrow \text{Reduce}(F)$;

$n \leftarrow \text{length}(G)$;

$P \leftarrow \{(g_i, g_j) \mid 1 \leq i < j \leq n\}$;

while $P \neq \emptyset$ **do**

pick $(p, q) \in P$;

$P = P \setminus \{(p, q)\}$;

$s \leftarrow \text{spoly}(p, q)$;

$h \leftarrow \text{NormalForm}(s, \{g_1, \dots, g_n\})$;

if $h \neq 0$ **then**

$g_{n+1} \leftarrow h$;

$P = P \cup \{(g_i, g_{n+1}) : 1 \leq i \leq n\}$;

$G = \text{Reduce}(G)$;

$n = \text{length}(G)$;

Update(P);

end

end

return $G = \{g_1, \dots, g_n\}$

Each time a new polynomial h is added to the basis G , it must be checked whether or not any polynomials in G can be reduced further or perhaps reduced to zero with the help of h . When this happens the set of pairs of functions in P must be updated.

But there still is a big problem in Buchberger's Algorithm. It needs many computations. In Buchberger's Algorithm there are too many idle times. Buchberger presented some rules and criteria to improve his algorithm. But we want to jump over this and go further to two improved algorithms which were presented in 1999 and 2002 by Faugère, [Fau99] and [Fau02]. The main difference to Buchberger's Algorithm is that instead of going through all pairs in $P \leftarrow \{(g_i, g_j) \mid 1 \leq i < j \leq n\}$, Faugère proposes to select a subset $Q \subset P$ of critical elements. Faugère called these two algorithms F_4 and F_5 , just algorithm number 4 and 5. Algorithms 1 till 3 are not really common.

We want to present the two algorithms consecutively and also accentuate the advantage in each case over Buchberger's Algorithm.

4.1.3 The Algorithm F_4

Faugère proposes to select a subset $Q \subset P$, where P is the set of the pairs of Buchberger's Algorithm. So, let us first see the Algorithm F_4 and afterwards we will have a more precise look at the several parts of the algorithm.

Algorithm 4.1.3: Algorithm F_4

Input: $F = \{f_1, \dots, f_n\}$, σ term ordering

Output: G reduced Gröbner basis of F

$G \leftarrow \tilde{F}$;

$P \leftarrow \{(g_i, g_j) \mid g_i \neq g_j \in G\}$;

$d \leftarrow 0$;

while $P \neq \emptyset$ **do**

$d \leftarrow d + 1$;

$Q_d \leftarrow \text{Select}(P)$;

$P \leftarrow P \setminus Q_d$;

$L_d \leftarrow \text{Left}(Q_d) \cup \text{Right}(Q_d)$;

$\tilde{F}^+ \leftarrow \text{Reduction}(L_d, G)$;

for $h \in \tilde{F}^+$ **do**

$P \leftarrow P \cup \{(h, g) \mid g \in G\}$;

$G \leftarrow G \cup \{h\}$;

end

end

return G

The first question is how the subset $Q_d \subset P$ is selected.

Remark 4.1.18. *Faugère has got the strategy to select all pairs in P simultaneously whose degree of $\text{lcm}(\text{LT}(f_i), \text{LT}(f_j))$ is minimal. Let*

$$d = \min\{\deg(\text{lcm}(\text{LM}(f_i), \text{LM}(f_j))) \mid \forall (f_i, f_j) \in P\}.$$

Then the selected subset is

$$\text{Select}(P) = \{(f_i, f_j) \in P \mid \deg(\text{lcm}(\text{LM}(f_i), \text{LM}(f_j))) = d\}.$$

And the algorithm needs some more explanations.

Definition 4.1.19. *Let $F = \{f_1, \dots, f_s\}$ be a set of polynomials in $K[x_1, \dots, x_n]$. Denote by $X = \{t_1, \dots, t_m\}$ the monomials in $\text{Supp}(F)$ listed in decreasing order in accordance with the ordering σ . The matrix representation of the polynomials is:*

$$F = AX,$$

where F and X are now to be treated as column vectors and the $s \times m$ matrix A has entries from K .

Let \tilde{A} be the unique row echelon form of A , with rows of zeros removed; i.e., $\bar{s} \times m$ matrix, where \bar{s} is the rank of A . Then \tilde{F} is the corresponding set of polynomials coming from

$$\tilde{F} = \tilde{A}X.$$

Define

$$\tilde{F}^+ = \{f \in \tilde{F} \mid \text{LM}(f) \notin \text{LM}(F)\},$$

so that \tilde{F}^+ consists of those polynomials in \tilde{F} whose leading monomials are not leading monomials in F .

Definition 4.1.20. Let $F = \{f_1, \dots, f_s\} \subset K[x_1, \dots, x_n]$ and let $p = (f_i, f_j)$ be a pair of polynomials with $i \neq j$. Let t_i and t_j be the two monomials such that

$$t_i \cdot \text{LM}(f_i) = t_j \cdot \text{LM}(f_j) = \text{lcm}(\text{LM}(f_i), \text{LM}(f_j)).$$

Define $\text{Left}(p) := (t_i, f_i)$ and $\text{Right}(p) := (t_j, f_j)$; that is, as a pair consisting of a monomial and a polynomial. If $Q \subset P$ is a set of pairs, then extend the definition as follows:

$$\text{Left}(Q) := \bigcup_{p \in Q} \text{Left}(p) \text{ and } \text{Right}(Q) := \bigcup_{p \in Q} \text{Right}(p)$$

And then we want to present the reduction procedure of the Algorithm F_4 :

Algorithm 4.1.4: Reduction

Input: L_d a finite set of monomials in R , G a finite subset of R

Output: \tilde{F}^+ a finite subset of R

$F_d \leftarrow \{t \cdot f \mid (t, f) \in L_d\};$

$D \leftarrow \text{LM}(F_d);$

while $D \neq \text{Supp}(F_d)$ **do**

 pick $(m) \in \text{Supp}(F_d) \setminus D;$

$D \leftarrow D \cup \{m\};$

if $\exists g \in G : m = m' \cdot \text{LM}(g)$ **then**

$F_d \leftarrow F_d \cup \{m' \cdot g\};$

end

end

$\tilde{F}_d \leftarrow \text{ReductionToRowEchelonForm}(F_d);$

$\tilde{F}^+ \leftarrow \{f \in \tilde{F}_d \mid \text{LM}(f) \notin \text{LM}(F_d)\};$

return \tilde{F}^+

We want to give one remark to the interested reader about F_4 . Allan Steel implemented F_4 for Magma such that the algorithm runs in a very fast way.

4.1.4 The Algorithm F_5

Another approach to improve Buchberger's Algorithm is the following:

The reduction of an S-polynomial may lead to an element that must be added to the basis. At that time more S-polynomials are added to the list for checking later. Since the algorithm terminates, the portion of S-polynomials that reduce to zero increases as we get further into the computation. After some time, a large amount of work is carried out with little earning. Long before the algorithm terminates, the desired Gröbner basis has likely already been found. From this point on, the computations could be considered useless. Unfortunately, we do not know when the basis is complete until the end of the computations.

For this reason it is desirable to have rules for minimizing the number of S-polynomials, or to have criteria that allow us to decide *a priori* if an S-polynomial reduces to zero.

But first we have to give some definitions.

Notation 4.1.21. The set of terms of $R = K[x_1, \dots, x_n]$ will be denoted by T .

Definition 4.1.22. A **labeled polynomial** is an element (te_i, p) , where $t \in T$, $p \in R$, and e_i is some standard basis vector in R^m . The set of all labeled polynomials is denoted by L .

Definition 4.1.23. For a labeled polynomial $r = (ue_k, p)$, we define the **polynomial** of r by $\text{poly}(r) := p$, the **signature** of r by $S(r) := ue_k$, the **leading term** of r by $\text{LT}(r) := \text{LT}(p)$, the **leading coefficient** of r by $\text{LC}(r) := \text{LC}(p)$, and the **leading monomial** of r by $\text{LM}(r) := \text{LM}(p)$.

Notation 4.1.24. Let $F \subseteq R$ be a sequence of m polynomials. v_F denotes the **evaluation homomorphism** with respect to F , defined by

$$v_F : R^m \longrightarrow R, (h_1, \dots, h_m) \mapsto \sum_{i=1}^m h_i f_i.$$

Definition 4.1.25. Let $g \in R^m$. The **leading term** of g is defined in the following way

$$\text{LT}(g) := \text{LT} \left(\sum_{i=1}^m g_i e_i \right).$$

Definition 4.1.26. Let $F \subseteq R$ be a sequence of m polynomials. A labeled polynomial r is called **admissible** with respect to F if there exists a $g \in R^m \setminus \{0\}$ such that $v_F(g) = \text{poly}(r)$ and $\text{LT}(g) = S(r)$.

Definition 4.1.27. We define the following operations on labeled polynomials. Let $r = (ue_k, p)$ be a labeled polynomial. If $t \in T$, then $tr = (tue_k, tp)$. If $\lambda \in K$, then $\lambda r = (ue_k, \lambda p)$.

Definition 4.1.28. A labeled polynomial $r = (ue_k, p)$ is called **normalized** with respect to F if $u \notin \text{LT}((f_{k+1}, \dots, f_m))$. A pair $(v, r) \in T \times L$ is called **normalized** if the labeled polynomial vr is normalized.

Notation 4.1.29. Suppose $g = (g_1, \dots, g_m)$ and $h = (h_1, \dots, h_m)$ are in $R^m \setminus \{0\}$. The **index** of g , written $\text{index}(g)$, is the smallest $i \in \mathbb{N}$ such that g_i is nonzero. Suppose $\text{index}(g) = i$, $\text{index}(h) = j$. We write $g \preceq h$ if and only if either

1. $i > j$, or
2. $i = j$ and $\text{LT}(g_i) \leq \text{LT}(h_i)$.

Definition 4.1.30. A pair (r_i, r_j) of labeled polynomials is called a **normalized pair** if, for $\tau_{ij} = \text{lcm}(\text{LT}(r_i), \text{LT}(r_j))$, $u_i = \frac{\tau_{ij}}{\text{LT}(r_i)}$, the pairs (u_i, r_i) , (u_j, r_j) are normalized and $u_j S(r_j) \prec u_i S(r_i)$.

Definition 4.1.31. Let P be a finite set of labeled polynomials, and s and t labeled polynomials with nonzero polynomial part. We say that

$$\text{poly}(s) = \sum_{p \in P} \mu_p \text{poly}(p), \mu_p \in R$$

is a t -**representation** of s with respect to P if for all $p \in P$ such that $\text{poly}(p) \neq 0$

$$\text{LT}(\mu_p \text{poly}(p)) = \text{LT}(\mu_p) \text{LT}(p) \leq \text{LT}(t) \text{ and } S(\mu_p \text{poly}(p)) = \text{LT}(\mu_p) S(p) \preceq S(s).$$

If this is the case, we write $s = \mathcal{O}_P(t)$. Similarly, we write $s = o_P(t)$ if there exists a labeled polynomial t' satisfying $S(t') \preceq S(t)$ and $\text{LT}(t') < \text{LT}(t)$ such that $s = \mathcal{O}_P(t')$.

The idea behind the Algorithm F_5 is the following [Fau02]:

Buchberger's Algorithm can be considered as a triangularisation of a submatrix of the sylvester matrix. The reduction of a polynomial to zero can be interpreted as a linear dependence of the rows of this matrix. Since each row of the matrix is a product tf where $t \in T$ and $f \in F$ (F is a sequence of m polynomials in R), a linear dependence is $\sum \lambda t f = 0$ or by grouping terms: $\sum_{i=1}^m g_i f_i = 0$. In other words, g_1, \dots, g_m is a syzygy. The strategy in the paper where F_5 is explained is to take into account only the trivial syzygies $f_i f_j - f_j f_i = 0$ but not to compute the module of syzygies.

Let us have a look at the main part of the algorithm and afterwards give an idea of why this algorithm works.

Algorithm 4.1.5: Incremental F_5

global r (array of labeled polynomials);

global Rules (array of simplification rules);

Input: A sequence $F = (f_1, \dots, f_m)$ of nonzero homogeneous polynomials in $K[x_1, \dots, x_n]$

Output: a Gröbner basis of F

$m \leftarrow |F|;$

Rules $\leftarrow (())_{j=1}^m;$

$r \leftarrow ();$

$r_m \leftarrow (e_m, \text{LC}(f_m)^{-1} f_m);$

$G \leftarrow (\emptyset)_{i=1}^m;$

$G_m \leftarrow \{m\};$

for $i = m - 1, \dots, 1$ **do**

$G_i \leftarrow \text{AlgorithmF5}(f_i, i, G);$

if $\exists k \in G_i : \text{poly}(r_k) = 1$ **then**

return 1

end

end

return $\{\text{poly}(r_k) \mid k \in G_1\}$

As we can see in Algorithm 'Incremental F_5 ' because of the for-loop, the Algorithm F_5 computes a Gröbner basis in increments; that is, for $(f_m), (f_m, f_{m-1}), \dots, (f_m, \dots, f_1)$.

And now let us see the proper part of the algorithm.

Algorithm 4.1.6: Algorithm F_5

Input: $f \in \mathbb{F}[x_1, \dots, x_n]$, $i \in \mathbb{N}$, G a sequence of sets of r -indices

Output: set G' of r -indices with $i \in G'$

$r_i \leftarrow (e_i, \text{LC}(f)^{-1}f)$;

$G' \leftarrow G_{i+1} \cup \{i\}$;

$P \leftarrow \emptyset$;

for $j \in G_{i+1}$ **do**

$P \leftarrow P \cup \text{CritPair}(i, j, i, G)$;

end

while $P \neq \emptyset$ **do**

$d \leftarrow \min\{\text{deg}(p) \mid p \in P\}$;

$P_d \leftarrow \{p \in P \mid \text{deg}(p) = d\}$;

$P \leftarrow P \setminus P_d$;

$S_d \leftarrow \text{SPols}(P_d)$;

$R_d \leftarrow \text{Reduction}(S_d, G', G)$;

for $k \in R_d$ **do**

$P \leftarrow P \cup \bigcup\{\text{CritPair}(k, l, i, G) \mid l \in G'\}$;

$G' \leftarrow G' \cup \{k\}$;

end

end

return G'

First it will get assembled a set of critical pairs. This set is called P . For that the method CritPair passes through the labeled polynomials and checks whether such a pair is normalized.

Then the algorithm takes a subset of the set P whereas the elements of this subset has all minimal degree as to P . By the elements of this subset it will built S-polynomials. Thereby it will post rules which are used in the next step.

Because the next step realizes a reduction in the set of S-polynomials. In this reduction process the point is to prove whether this polynomial is normalized to the set of polynomials which is already a part of the searched Gröbner basis, whether previous computation can be used and whether it is possible to remove identical rows in the visualized matrix. As a result the reduction of one polynomial by a list of polynomials may be several polynomials. In doing so the saved labeled polynomials help because the process just often has to use only the signature.

The Algorithm F_5 works because of the following theorem:

Theorem 4.1.32. *Let $F = \{f_1, \dots, f_n\}$ be a list of polynomials. Let $G = \{r_1, \dots, r_n\} \in \mathbb{L}^{nG}$ such that*

1. $F \subset \text{poly}(G)$, let $g_i = \text{poly}(r_i)$ and $G_1 = \{g_1, \dots, g_{n_G}\}$;
2. all the r_i are admissible with respect to F ($i = 1, \dots, n_G$);
3. for all $(i, j) \in \{1, \dots, n_g\}^2$, such that the pair (r_i, r_j) is normalized then $\text{spoly}(g_i, g_j) = o_G(u_i r_i)$ (or 0) where

$$u_i = \frac{\text{lcm}(\text{LT}(g_i), \text{LT}(g_j))}{\text{LT}(r_i)}.$$

Then G_1 is a Gröbner basis of the generated ideal of F .

For a proof see either at [Fau02, Theorem 1] or for a proof in detail at [Ste06].

4.1.5 Attack of Faugère and Joux

This section is based on [DGS06a, pages 218f] and on [FJ03].

Faugère and Joux concentrate their attack on a group of MPKCs which we do not present in this paper. (This group of MPKCs is called Hidden Field Equations. The interested reader may have a look in the paper [Pat96].) So we want to give an overlook how the attack would work against MPKCs in general.

Let K be a field and $R = K[x_1, \dots, x_n]$ the ring of multivariate polynomials. To a system of equations

$$f_1(x_1, \dots, x_n) = \dots = f_m(x_1, \dots, x_n) = 0$$

we associate the ideal I generated by f_1, \dots, f_m . As we know a Gröbner basis is a generating system of I that "behaves nicely" with respect to some order on monomials.

Let M be a field containing K , we can define the set of solutions of f_1, \dots, f_m in M which is the algebraic variety:

$$V_M = \{(z_1, \dots, z_n) \in M \mid f_i(z_1, \dots, z_n) = 0, i = 1, \dots, m\}$$

which is in fact the set of roots of the system of equations. Normally we want to compute solutions of algebraic systems in \mathbb{F}_2 . Faugère and Joux presented the following proposition which tell us how to use Gröbner bases in order to solve a system over \mathbb{F}_2 :

Proposition 4.1.33. *The Gröbner basis G of $\{f_1, \dots, f_m, x_1^2 - x_1, \dots, x_n^2 - x_n\}$, in $\mathbb{F}_2[x_1, \dots, x_n]$, describes all the solutions in $V_{\mathbb{F}_2}$. Particular useful cases are:*

1. $V_{\mathbb{F}_2} = \emptyset$ (no solution) iff $G = \{1\}$.
2. $V_{\mathbb{F}_2}$ has exactly one solution iff $G = \{x_1 - a_1, \dots, x_n - a_n\}$ where $a_i \in \mathbb{F}_2$. Then (a_1, \dots, a_n) is the solution in \mathbb{F}_2 of the algebraic system.

This proposition tells us that we have to add the "field equations" $x_i^2 = x_i$ to the list of equations that we want to solve. Consequently we have to compute a Gröbner basis of $m + n$ polynomials and n variables. In fact, the more equations we have the more able we are to compute a Gröbner basis.

With this considerations we can execute Bucherberger's Algorithm or better the Algorithm F_4 or the Algorithm F_5 and receive so a Gröbner basis of our system which result in the discovering of the plain text.

Fortunately these algorithms have an upper bound of feasibility. So we can assume that there can exist MPKCs which are resistible against Gröbner bases method. Faugère and Joux write in the end of their paper [FJ03, page 59] that it depends on the largeness of the dimension of the vector space we work and of the degree of the public key polynomial. But it must be pointed out that a statement about the degree of the public key polynomial is especially given to the group of MPKCs which we do not observe in this paper. For using in a practical way against all MPKCs the attack of Faugère and Joux should be studied more. This would go beyond of the scope of this thesis.

4.2 eXtended Linearization method

This section is based on [CKPS00] and on [DGS06a, pages 220-223].

The idea for this attack is based on a paper of Kipnis and Shamir [KS99]. In Section 2.4 we already met the idea of this paper. In the second part there was to solve a MinRank problem (see Section 2.4.2). Kipnis and Shamir presented a relinearization method to solve this, which we did not show. But Courtois, Klimov, Patarin and Shamir took exactly this method and develop a new method, the eXtended Linearization (XL) method. The idea of relinearization is to add to the given system of linear equations additional nonlinear equations which express the fact that these variables are related rather than independent. The basic idea of the XL method is to generate from each polynomial equation a large number of higher degree variants by multiplying it with all the possible monomials of some bounded degree, and then to linearize the expanded system. Courtois, Klimov, Patarin and Shamir proved that the relinearization is a subcase of the XL Algorithm [CKPS00, page 11].

First we want to present some mathematical thoughts to the XL Algorithm and afterwards the algorithm itself.

4.2.1 Mathematical background

Let K be a field, and let \mathcal{A} be a system of multivariate quadratic equations $l_k = 0$ with $1 \leq k \leq m$ where each l_k is a multivariate polynomial of the set of equations (4.1).

From Algebraic Geometry we know the following:

Theorem 4.2.1. *If a set of polynomial equations*

$$\begin{aligned} l_1(x_1, \dots, x_n) &= 0 \\ &\vdots \\ l_m(x_1, \dots, x_n) &= 0 \end{aligned}$$

has a unique solution

$$x_1 = a_1, \dots, x_n = a_n,$$

then the ideal generated by l_1, \dots, l_m contains the elements $x_1 - a_1, \dots, x_n - a_n$, and in fact is $(x_1 - a_1, \dots, x_n - a_n)$.

So, there exists a set of functions h_{ij} , $i = 1, \dots, n$ and $j = 1, \dots, m$ such that

$$x_i - a_i = \sum_{j=1}^m h_{ij} l_j.$$

If we can find these h_{ij} , then we find the solution. However, we do not know how to find the h_{ij} directly, so instead we generate the ideal gradually until we have the solution.

Let $k, D \in \mathbb{N}$. Let D be fixed bigger than 2. Let M_k be the set of monomials in x_1, \dots, x_n with degree less than or equal to k . As we will see immediately $k < D$.

We say that the equations of the form $\prod_{i=1}^k h_{ij} l_j = 0$ are of type $h^k l$, and we call $h^k l$ the set of all these equations.

We also denote by h^k the set of all terms of degree exactly k , $\prod_{i=1}^k h_{ij}$. So we can say h^k is an element of M_k .

We consider all the polynomials $\prod_j h_{ij} l_j = 0$ of total degree $\leq D$.

Let \mathcal{I}_D be the set of equations they span. \mathcal{I}_D is the linear space generated by all the $h^k l$, $0 \leq k \leq D - 2$. $\mathcal{I}_D \subset \mathcal{I}$, \mathcal{I} being the ideal spanned by the l_j (could be called \mathcal{I}_∞).

The idea of the XL Algorithm is to find in some \mathcal{I}_D a set of equations which is easier to solve than the initial set of equations $\mathcal{I}_0 = \mathcal{A}$.

4.2.2 XL Algorithm

Step 1: Fix an integer $D > 2$.

Step 2: Generate all the products $\sum_{i=1}^k h_{ij} l_j$ with $k \leq D - 2$.

Step 3: Consider each monomial in the x_i of degree $\leq D$ as a new variable and perform Gaussian elimination on the equation obtained in 1.

The ordering on the monomials must be such that all the terms containing on a variable (say x_1) are eliminated last.

Step 4: Assume that Step 3 yields at least one univariate equation in the powers of x_1 . Solve this equation over the finite fields.

Step 5: Simplify the equations by substituting in the value of the variable x_1 .

Step 6: Repeat the process to find the values of the other variables.

4.2.3 Feasibility

Courtois, Klimov, Patarin and Shamir stated in their paper [CKPS00], it is not clear for which values of n and m it ends successfully and what is its asymptotic complexity. But nevertheless they make the following statement:

We analyze the time complexity of the XL technique, and provide strong theoretical and practical evidence that the expected running time of this technique is:

- Polynomial when the number m of (random) equations is at least ϵn^2 , and this for all $\epsilon > 0$.
- Subexponential if m exceeds n even by a small number.

But in 2004 Diem published a paper in which he shows that the second statement of above is not subexponential but more complex [Die04].

4.3 Zhuang-Zi method

Ding, Gower and Schmidt presented in 2006 a new algorithm which attacks multivariate public key cryptosystems [DGS06b]. We will rely our explanation upon this paper.

The basic idea for this algorithm we already met earlier. We pick up the idea of the extension field which we already saw in Chapter 2.

Ding, Gower and Schmidt named this algorithm after Zhuang-Zi, an ancient Chinese philosopher who they believe was one of the first to propose the idea of shifting from a local view of problems to a global view.

First we want to present some mathematical thoughts to the Zhuang-Zi Algorithm and afterwards the algorithm itself.

4.3.1 Mathematical background

Let K be a finite field with q elements and suppose we have m polynomials $f_0, f_1, \dots, f_{m-1} \in K[x_0, x_1, \dots, x_{n-1}]$. We wish to find all $(a_0, a_1, \dots, a_{n-1}) \in K^n$ such that

$$\begin{aligned} f_0(a_0, a_1, \dots, a_{n-1}) &= 0 \\ f_1(a_0, a_1, \dots, a_{n-1}) &= 0 \\ &\vdots \\ f_{m-1}(a_0, a_1, \dots, a_{n-1}) &= 0. \end{aligned}$$

We may as well work in the ring

$$K[x_0, x_1, \dots, x_{n-1}]/(x_0^q - x_0, x_1^q - x_1, \dots, x_{n-1}^q - x_{n-1}),$$

though for convenience we will abuse notation and write $K[x_0, x_1, \dots, x_{n-1}]$.

To simplify matters, let us assume that $m = n$. We will say later some comments to the other cases. As we saw in Section 2.1, choose any irreducible polynomials $g(y) \in K[y]$ of degree n . Then $L = K[y]/(g(y))$ is a degree n field extension of K . Let K^n denote the n -dimensional vector space over K . Let $\phi : L \rightarrow K^n$ be the standard K -linear isomorphism between L and K^n given by

$$\phi(a_0 + a_1u + \dots + a_{n-1}u^{n-1}) = (a_0, a_1, \dots, a_{n-1}).$$

Let $f : K^n \rightarrow K^n$ be the polynomial map defined by $f = (f_0, f_1, \dots, f_{n-1})$. We can lift f up to the extension field L using ϕ to create a map $F : L \rightarrow L$ defined by

$$F = \phi^{-1} \circ f \circ \phi.$$

Using the Lagrangian interpolation formula, we can think of F as a polynomial in $L[X]$, where X is an indeterminate. In fact, F has a unique representation in the quotient space $K[X]/(X^{q^n} - X)$. For any given f , the corresponding F can be calculated by solving a set of linear equations. The following theorem tells us the exact form of this representation.

Theorem 4.3.1. *Using the notation as we defined above, for a linear polynomial map $f = (f_0, f_1, \dots, f_{n-1})$ we have*

$$F(X) = \sum_{i=0}^{n-1} \beta_i X^{q^i} + \alpha \pmod{(X^{q^n} - X)},$$

for some $\beta_i, \alpha \in L$. If f is a quadratic polynomial map, then

$$F(X) = \sum_{i=0}^{n-1} \sum_{j=i}^{n-1} \gamma_{ij} X^{q^i+q^j} + \sum_{i=0}^{n-1} \beta_i X^{q^i} + \alpha \pmod{(X^{q^n} - X)},$$

for some $\gamma_{ij}, \beta_i, \alpha \in L$. Representations for higher order polynomials maps are similarly described. In the case of $q = 2$, the formulas are slightly different.

4.3.2 Zhuang-Zi Algorithm

We will start the standard case of $m = n$, where we have the same number of variables and equations.

Input

1. the polynomials $f_0, f_1, \dots, f_{n-1} \in K[x_0, x_1, \dots, x_{n-1}]$
2. a positive number D , where D is the upper bound on the degree of a polynomial equation which can be solved efficiently

Output

When successful,

all n -tuples $(a_0, a_1, \dots, a_{n-1}) \in K^n$ such that $f_i(a_0, a_1, \dots, a_{n-1}) = 0$, for $i = 0, 1, \dots, n - 1$.

Step 1: Choose any degree n irreducible polynomial $g(y) \in K[x]$ and define $L = K[y]/(g(y))$. Let $\phi : L \rightarrow K^n$. Define $f = (f_0, f_1, \dots, f_{n-1})$, lift this to L by $F = \phi^{-1} \circ f \circ \phi$, and compute the polynomial representation of $F(X)$ modulo $X^{q^n} - X$. If the $\deg(F(X)) \leq D$, then go to the last step; otherwise continue to the next step.

Step 2: Let $G = \text{Gal}(L/K)$ be the Galois group of L over K consisting of the Frobenius maps $G_i(X) = X^{q^i}$, for $i = 0, 1, \dots, n - 1$. Calculate

$$F_i(X) = G_i \circ F(X) = F(X)^{q^i} \pmod{(X^{q^n} - X)},$$

for $i = 0, 1, \dots, n-1$. Note that $F_0(X) = F(X)$. If there exists an F_i such $\deg(F_i(X)) \leq D$, then go to the last step; otherwise continue to the next step.

Step 3: Let N be the number of monomials that appear in any $F_i(X)$. For each $F_i(X)$ create a row vector in K^N , where the entries are the coefficients of $F_i(X)$ listed in decreasing order, and construct an $n \times N$ matrix using these row vectors. Then use Gaussian elimination to produce a new set of t basis polynomials $S = \{S_0(X), S_1(X), \dots, S_{t-1}(X)\}$. In other words eliminate the monomials in the order of the highest degree first. Label the elements of S so that $S_{t-1}(X)$ is the element of lowest degree. If $\deg(S_{t-1}(X)) \leq D$, then go to the last step; otherwise continue to the next step.

Step 4: It must be that the polynomial of minimal degree in S has degree greater than D . For each $i = 0, 1, \dots, t - 1$ and $j = 0, 1, \dots, n - 1$ compute

$$X^{q^j} S_i(X) \bmod (X^{q^n} - X).$$

As before, apply Gaussian elimination to the matrix associated with this set of polynomials to produce a set S' of new basis polynomials. Let $S'_{t'-1}(X)$ be the polynomial in S' of minimal degree. If the $\deg(S'_{t'-1}(X)) \leq D$, then go to the last step; otherwise replace S with S' and repeat this step.

Step 5: At this point we have a polynomial $G(X)$ with $\deg(G(X)) \leq D$. Solve $G(X) = 0$, to obtain a set $W = \{\alpha \in L \mid G(\alpha) = 0\}$. The solutions of $F(X) = 0$ will be the subset $\{\alpha \in W \mid F(\alpha) = 0\}$.

Let us see some remarks to this algorithm.

Remark 4.3.2. *Each application of Step 4 produces monomials of the form $X^{a_i q^i}, X^{a_i q^i + a_j q^j}, X^{a_i q^i + a_j q^j + a_k q^k}$, etc. Clearly then all possible monomials will have been generated after nq steps, since there are only nq possible monomials in $L[X]/(X^{q^n} - X)$.*

Remark 4.3.3. *When we have more variables than equations ($n > m$) then we can augment the given equations with $n - m$ trivial equations of $0 = 0$. Similarly, when $n < m$ we can simply introduce additional variables x_n, \dots, x_{m-1} to make up for the shortfall.*

Ding, Gower and Schmidt establish that the Zhuang-Zi Algorithm requires that we work in a finite field, whereas Gröbner bases do not. There exist many examples that can be solved easily by the Zhuang-Zi Algorithm, but only with great difficulties via Gröbner bases. But it is not yet clear how the complexity of the Zhuang-Zi Algorithm is for a set of n nonlinear equations in n variables. In general the complexity will be exponential in n , though for certain types of equations the Zhuang-Zi Algorithm will work much better.

Chapter 5

Matsumoto-Imai Tame-Transformation-Method Cryptosystem

In [Moh99b] Moh said the following:

In the beginning they try to modify polynomials of one variable to make them polynomials of several variables. [...] They have a reasonably simple polynomial in one variable to begin with, and then express it in a basis of the field extension to achieve an expression of several variables.

The second approach is to apply the theory of genuine several variables to achieve a higher security and faster speed.

Amongst others this statement guides to the idea that we can combine the MI Cryptosystem and the TTM Cryptosystem.

So let us first show how the combination of these two cryptosystems looks like and why we choose that in this way. And in a second part we will see how this new cryptosystem resists against the attacks in stock.

5.1 Construction of the Matsumoto-Imai Tame-Transformation-Method Cryptosystem

The Matsumoto-Imai Tame-Transformation-Method (shortly MI-TTM) Cryptosystem is a system which takes the common elements of the MI Cryptosystem and the TTM Cryptosystem and embed in these the peculiarities of the two systems side by side. This should lead to the goal that the advantages of both cryptosystem improve the security of the combined system.

Let K be a finite field of characteristic $p = 2$ and cardinality $q = p^m$, where m is a positive integer. Let K^N denote the N -dimensional vector space over K . We define a map $\xi : K^N \rightarrow K^N$ which is a composition of seven maps $\kappa_1, \dots, \kappa_7$

$$\xi = \kappa_7 \circ \dots \circ \kappa_1.$$

The two maps κ_1 and κ_7 are invertible affine transformations over K^N . κ_1 has got the restriction of the ϕ_1 of the TTM Cryptosystem. (See Section 3.2.2.)

The two maps κ_2 and κ_3 adopt the properties of ϕ_2 and ϕ_3 . Hence $N = n + r$ of the TTM Cryptosystem. (See Section 3.2.1.)

The three maps κ_4 , κ_5 and κ_6 are used to adopt the middle part of the MI Cryptosystem. So let us take a $g(u) \in K[u]$ which is an irreducible polynomial of degree N , where $N = (2l + 1)2^r$ of the MI Cryptosystem. Define the field $L = K[u]/g(u)$, a degree n extension of K . Then $\kappa_6 = \phi$ where $\phi : L \rightarrow K^N$ is the standard K -linear isomorphism between L and K^N . And analogous $\kappa_4 = \phi^{-1}$ of the MI Cryptosystem. Consequently $\kappa_5 = \hat{f} : L \rightarrow L, U \mapsto U^h$, where h has got the condition which we saw at the MI Cryptosystem. (See Section 2.1.)

It is obvious that N now has got two conditions to fulfill, the condition of the MI Cryptosystem and the one of the TTM Cryptosystem. But that should not be a problem because the condition of the TTM Cryptosystem is only that $N \geq 3$ (see Section 3.2.1) and this is fulfilled by the condition of the MI Cryptosystem unless we construct a toy example.

If we think about ϕ_2 and $\hat{\phi}_2$ of the TTM Cryptosystem and also how the conditions of ϕ_1 are, we can construct a map $\hat{\xi}$ from K^{N-r} to K^N where r satisfy the r of the TTM Cryptosystem. (See Section 3.2.2.)

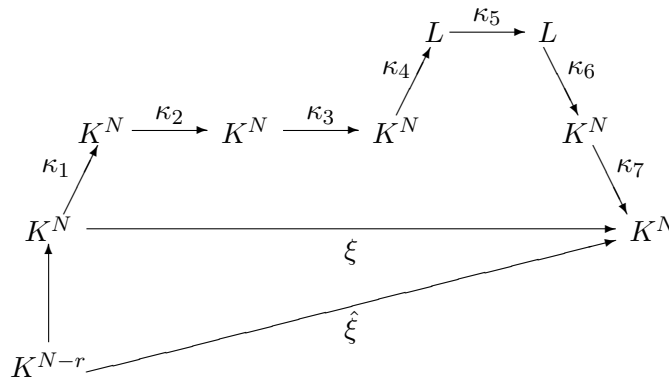


Figure 5.1: composition of maps of the MI-TTM Algorithm

the public key

The public key of the MI-TTM Cryptosystem includes the following:

1. The field K of p^m elements including its additive and multiplicative structure.
2. The map $\hat{\xi} : K^{N-r} \rightarrow K^N$.

the private key

The private key consists of:

1. The two affine transformations κ_1 and κ_7 .
2. The inverse maps κ_2^{-1} and κ_3^{-1} of the automorphisms.

3. The composition of h which is $1 + q^\theta$ as we saw in Section 2.1. Naturally we should also know the field L .

encryption

Let $(x'_1, \dots, x'_{N-r}) \in K^{N-r}$ be the plaintext. Then the associated ciphertext is $(y'_1, \dots, y'_N) \in K^N$, where

$$y'_i = \hat{\xi}_i(x'_1, \dots, x'_{N-r})$$

for $i = 1, \dots, N$.

decryption

Let $(y'_1, \dots, y'_N) \in K^N$ be the ciphertext. Then we compute in the following way:

1. compute $(y_1, \dots, y_N) = \kappa_7^{-1}(y'_1, \dots, y'_N)$
2. compute $(z_1, \dots, z_N) = \kappa_4^{-1} \circ \kappa_5^{-1} \circ \kappa_6^{-1}(y_1, \dots, y_N)$
3. compute $(x_1, \dots, x_{N-r}, 0, \dots, 0) = \kappa_2^{-1} \circ \kappa_3^{-1}(z_1, \dots, z_N)$
4. compute $(x'_1, \dots, x'_{N-r}, 0, \dots, 0) = \kappa_1^{-1}(x_1, \dots, x_{N-r}, 0, \dots, 0)$

Theoretically we can also build an inverse $\hat{\xi}^{-1}$. But in practice, the complexity for computing this is too big.

For the inverses κ_2^{-1} and κ_3^{-1} we want to point to Remark 3.1.4. This explains how the inverses work.

5.2 Background of the Matsumoto-Imai Tame-Transformation-Method Cryptosystem

Now we want to see why this cryptosystem works and why we built it in this way.

So, let $x' = (x'_1, \dots, x'_{N-r})$ be the plaintext. If we compute $\hat{\xi}(x')$ we achieve the ciphertext $y' = (y'_1, \dots, y'_N) = (\hat{\xi}_1(x'_1, \dots, x'_{N-r}), \dots, \hat{\xi}_N(x'_1, \dots, x'_{N-r}))$. By construction $\hat{\xi} = \kappa_7 \circ \dots \circ \kappa_3 \circ \hat{\kappa}_2 \circ \hat{\kappa}_1$, where $\hat{\kappa}_2$ relates to $\hat{\phi}_2$ and $\hat{\kappa}_1$ relates to $\hat{\phi}_1$ as already mentioned in Section 5.1. From this it follows that $\kappa_1^{-1}(\kappa_2^{-1}(\kappa_3^{-1}(\kappa_4^{-1}(\kappa_5^{-1}(\kappa_6^{-1}(\kappa_7^{-1}(y'))))))$ is equal to $(x'_1, \dots, x'_{N-r}, 0, \dots, 0) \in K^N$ so after restriction to K^{N-r} we get the plaintext.

And why we built it in this way? It makes sense that the two affine maps are in the beginning and in the end of the composition. This was the great idea of Matsumoto and Imai which Moh also used for his cryptosystem.

ϕ_2 of the TTM Cryptosystem is constructed in a special way. It assumes that $x_{N-r+1} = \dots = x_N = 0$ or that we do an embedding from K^{N-r} to K^N . As we saw that already has consequences for κ_1 : we have to restrict this map as we saw above. So it does not make sense to place this map after further maps. We take this map just after the first affine transformation.

κ_4 and κ_6 have to enclose κ_5 because κ_5 is a map in L and κ_4 and κ_6 transfer from K^N to L and back again.

So, the question comes up whether we can interchange the order of κ_3 and $\kappa_6 \circ \kappa_5 \circ \kappa_4$. And the answer is yes, we can do this. But it is unnecessary. κ_3 is anyway an automorphism in K^N . Further is κ_5 according to the construction of it also an automorphism in L . κ_4 and κ_6 are the standard K -linear isomorphism between L and K^N and its inverse. So it follows that $\kappa_6 \circ \kappa_5 \circ \kappa_4$ is an automorphism in K^N . From linear algebra we know that automorphisms build a group and hence both combinations work. Although the result may differ for the two combinations, for the efficiency and the security of the MI-TTM Cryptosystem it is irrelevant whether κ_3 or $\kappa_6 \circ \kappa_5 \circ \kappa_4$ is first. We decide in favor of this order because in this way the middle part of the TTM Cryptosystem stays together.

Finally we comment on the degree of the polynomial.

5.2.1 Disadvantage of higher degree

If we construct the MI-TTM Cryptosystem in this general way, it can happen that we find maps with a bigger degree than quadratic. The maps κ_1 and κ_7 are affine. So an affine map does not have an influence to the degree of the combination with another map with higher degree. The combination has the same degree as the map with higher degree than 1. So we have to study the maps κ_2 , κ_3 and $\kappa_6 \circ \kappa_5 \circ \kappa_4$. As we already mentioned in Section 3.3 the combination $\kappa_3 \circ \kappa_2$ has quadratic degree because Moh constructed it in this way. The map $\kappa_6 \circ \kappa_5 \circ \kappa_4$ by itself has quadratic degree. So we can formulate the following remark:

Remark 5.2.1. *The degree of the MI-TTM Cryptosystem is at most 4.*

So it possibly takes more computation to encrypt a message by the MI-TTM Cryptosystem than by the MI Cryptosystem or the TTM Cryptosystem.

But there is a second disadvantage of this remark above. Because there occur terms with degree at most 4 there also occur more terms. So the size of the public key itself increases in fact up to an order of N^4 terms.

It stays open whether it is possible to reduce this upper bound. Maybe it would be possible to construct κ_2 and κ_3 in a way that the combination of the κ s results in a map with degree smaller than 4. This would be an open problem for further research.

5.3 Possible attacks

Let us have a look at each attack we discussed in this paper and see how these attacks possibly work against the MI-TTM Cryptosystem.

5.3.1 Combination of Patarin's Attack and of the Attack of Ding and Schmidt

As we saw in Section 3.5, Ding and Schmidt took the idea of Patarin's Attack against the MI Cryptosystem and adopt it against the TTM Cryptosystem.

In Section 2.3.1 we saw Patarin's idea is to lift up the elements of K^n to the degree n extension field L . The goal is to find equations in the following form:

$$\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \gamma_{ij} x_i y_j + \sum_{i=0}^{n-1} \alpha_i x_i + \sum_{j=0}^{n-1} \beta_j y_j + \delta = 0,$$

where γ_{ij} , α_i , β_i and δ are in K . With these equations we can build a vector space. So we can find the plaintext from the ciphertext.

Further Ding and Schmidt showed that we can also build such analog linearization equations as above. In Section 3.5 we saw Ding and Schmidt build such linear equations to eliminate ϕ_3 of the TTM Cryptosystem. Afterwards because of the form of ϕ_2 of the TTM Cryptosystem we can solve the system by iterated Gaussian reduction.

In view of these considerations the idea suggests itself that an adapted version of these two attacks can theoretically also attack the MI-TTM Cryptosystem. But such an adapted version is more complex.

As we saw in Section 3.5, ϕ_2 and ϕ_3 are attacked consecutively. In the MI-TTM Cryptosystem there are three parts to attack, namely κ_2 , κ_3 and $\kappa_6 \circ \kappa_5 \circ \kappa_4$. But this constitutes a problem. If we try to attack the parts consecutively, we could try this in the following ways:

- First we attack $\kappa_6 \circ \kappa_5 \circ \kappa_4$ in the art of Patarin's Attack. So we want to produce $n + r$ linear independent equations by constructing (x, y) -pairs with the aid of ξ . But this is not possible because the plaintext only consists of elements of K^n and we can not find $n + r$ linearly independent linear equations. So let us try it the other way around.
- First we attack κ_3 by using Step 1 of the Attack of Ding and Schmidt, where we construct linearly independent linear equations with the aid of ξ , and by using Step 2. Then we use Step 3 of the attack to eliminate the influence of κ_2 . But now we do not know what we really found. We get an element of K^n which is not the plaintext because $\kappa_6 \circ \kappa_5 \circ \kappa_4$ is still applied to this element whereas $\kappa_6 \circ \kappa_5 \circ \kappa_4$ is a map in K^{n+r} .

So, it does not really work to construct a consecutive linear equation attack.

Maybe, we can construct an attack against the MI-TTM Cryptosystem as a whole. But there is to question how this should with the complexity. We remind of the end of Section 3.5. There exists instances of the TTM Cryptosystem where the Attack of Ding and Schmidt fails because it is impossible to find l linearly independent linear equations, where l should be bigger than m . For an attack against the whole MI-TTM Cryptosystem we have to find n linear independent equations, which is more than m and l . But based on the construction of the MI-TTM Cryptosystem with more security it gets more difficult to find these equations.

5.3.2 Kipnis-Shamir Attack

As we saw in Section 2.4, the idea to recover the private key is to lift the map in K^n to a map over L , where L is a degree n extension field of K . In principle this attack would also work against the MI-TTM Cryptosystem. But Moh comes to our aid. He wrote a paper about the Attack of Kipnis and Shamir applied to the TTM Cryptosystem [Moh99a]. Kipnis and Shamir construct a system of quadratic polynomials to attack the cryptosystem. But Moh could show

that the system which is necessary to attack a TTM Cryptosystem is too big for solving this system.

If we now construct a cryptosystem in which the security is better or at least equal to the TTM Cryptosystem alone, then the Attack of Kipnis and Shamir should not be able to find the private key of a MI-TTM Cryptosystem.

5.3.3 Minrank Attack of Goubin and Courtois

If we have a look at the Minrank Attack of Goubin and Courtois which is presented in Section 3.4, we see that Goubin and Courtois introduce the TPM Cryptosystem. This clearly show us that the idea which is behind it is the form of a triangular construction. But in the MI-TTM Cryptosystem there appears $\kappa_6 \circ \kappa_5 \circ \kappa_4$ which does not have a triangular form. So it is not possible to adopt this attack against the whole MI-TTM Cryptosystem.

5.3.4 Attack based on Gröbner bases

Attacks based on Gröbner bases get faster. Buchberger's Algorithm is more a theoretical study but not so good in practice. The Algorithm F_4 and the Algorithm F_5 are much better and hold a risk to attack against the MI-TTM Cryptosystem. But as we saw in Section 4.1.5 there is only one a practical attack based on Gröbner bases against an MPKC, and this one is just tailored against one group of MPKCs. So we have to wait for a more precise study about attacks based on Gröbner bases, in particular the complexity of it.

5.3.5 XL Attack

As we already saw in Section 4.2.3, the XL Algorithm is not so efficient as thought at first. Based on the conclusion of Diem we can deduce that the XL Algorithm cannot attack the TTM Cryptosystem. In the MI-TTM Cryptosystem we adopt the circumstance that we use N equations with $N - r$ variables. Hence there is exactly the situation which Diem studied.

5.3.6 Zhuang-Zi Attack

According to the construction of the MI-TTM Cryptosystem we engage in a system of equations which is the basic requirement for the Zhuang-Zi Algorithm. As we saw in Theorem 4.3.1 the Zhuang-Zi Algorithm also works for higher order polynomial maps than quadratic. But it results in a bigger complexity. Before we can study the application of the Zhuang-Zi Algorithm to the MI-TTM Cryptosystem we should wait until the complexity of the Zhuang-Zi Algorithm is clearer in general.

5.4 The Future of the Matsumoto-Imai Tame-Transformation-Method Cryptosystem

We saw in this chapter how the MI-TTM Cryptosystem works in a theoretical way. There are some features which could be improved in the future. But the advantage is that the MI-TTM Cryptosystem is resistant against the attacks which exist. Whether this cryptosystem is also usable in a practical way will show us the future.

Bibliography

- [BFS99] Jonathan F. Buss, Gudmund S. Frandsen, and Jeffrey O. Shallit. The computational complexity of some problems of linear algebra. *J. Comput. System Sci.*, 58(3):572–596, 1999.
- [Buc65] Bruno Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal*. PhD thesis, University of Innsbruck, 1965.
- [Buc76] Bruno Buchberger. A theoretical basis for the reduction of polynomials to canonical forms. volume 10, pages 19–29, New York, NY, USA, 1976. ACM.
- [CGP03] Nicolas Courtois, Louis Goubin, and Jacques Patarin. SFLASH, a fast asymmetric signature scheme for low-cost smartcards primitive specification and supporting documentation. 2003.
- [CKPS00] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *Advances in cryptology—EUROCRYPT 2000 (Bruges)*, volume 1807 of *Lecture Notes in Comput. Sci.*, pages 392–407. Springer, Berlin, 2000.
- [Cou01] Nicolas T. Courtois. The security of hidden field equations (HFE). In *Topics in cryptology—CT-RSA 2001 (San Francisco, CA)*, volume 2020 of *Lecture Notes in Comput. Sci.*, pages 266–281. Springer, Berlin, 2001.
- [CSV93] Don Coppersmith, Jacques Stern, and Serge Vaudenay. Attacks on the birational permutation signature schemes. In *Proceedings of CRYPTO’93, number 773 in LNCS*, pages 435–443. Springer-Verlag, 1993.
- [CSV97] Don Coppersmith, Jacques Stern, and Serge Vaudenay. The security of the birational permutation signature schemes. *J. Cryptology*, 10(3):207–221, 1997.
- [DFSS07] Vivien Dubois, Pierre-Alain Fouque, Adi Shamir, and Jacques Stern. Practical cryptanalysis of sflash. In *Cryptology ePrint Archive, Report 2007/141*, 2007.
- [DGS06a] Jintai Ding, Jason E. Gower, and Dieter S. Schmidt. *Multivariate Public Key Cryptosystems*, volume 25 of *Advances in Information Security*. Springer, New York, 2006.
- [DGS06b] Jintai Ding, Jason E. Gower, and Dieter S. Schmidt. Zhuang-zi: A new algorithm for solving multivariate polynomial equations over a finite field, cryptology eprint archive, report 2006/038, 2006. 2006.

- [DGY99] Vesselin Drensky, Jaime Gutierrez, and Jie-Tai Yu. Gröbner bases and the Nagata automorphism. *J. Pure Appl. Algebra*, 135(2):135–153, 1999.
- [Die04] Claus Diem. The XL-algorithm and a conjecture from commutative algebra. In *Advances in cryptology—ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Comput. Sci.*, pages 323–337. Springer, Berlin, 2004.
- [DS03] Jintai Ding and Dieter Schmidt. A defect of the implementation schemes of the TTM cryptosystem. Cryptology ePrint Archive, Report 2003/085, 2003.
- [EJ01] D. Eastlake, 3rd and P. Jones. US Secure Hash Algorithm 1 (SHA1). United States, 2001. RFC Editor.
- [Fau99] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases (F_4). *J. Pure Appl. Algebra*, 139(1-3):61–88, 1999. Effective methods in algebraic geometry (Saint-Malo, 1998).
- [Fau02] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). In *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation*, pages 75–83 (electronic), New York, 2002. ACM.
- [FD86] Harriet Fell and Whitfield Diffie. Analysis of a public key approach based on polynomial substitution. In *Advances in cryptology—CRYPTO '85 (Santa Barbara, Calif., 1985)*, volume 218 of *Lecture Notes in Comput. Sci.*, pages 340–349. Springer, Berlin, 1986.
- [Fel05] Patrick Felke. On the Affine Transformations of HFE-Cryptosystems and Systems with Branches. In *WCC*, pages 229–241, 2005.
- [FJ03] Jean-Charles Faugère and Antoine Joux. Algebraic cryptanalysis of hidden field equation (HFE) cryptosystems using Gröbner bases. In *Advances in cryptology—CRYPTO 2003*, volume 2729 of *Lecture Notes in Comput. Sci.*, pages 44–60. Springer, Berlin, 2003.
- [GC00] Louis Goubin and Nicolas T. Courtois. Cryptanalysis of the TTM cryptosystem. In *Advances in cryptology—ASIACRYPT 2000 (Kyoto)*, volume 1976 of *Lecture Notes in Comput. Sci.*, pages 44–57. Springer, Berlin, 2000.
- [GH00] J. Goldman and J. Haglund. Generalized Rook Polynomials. *J. Combin. Theory Ser. A*, 91(1-2):509–530, 2000. In memory of Gian-Carlo Rota.
- [GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.
- [KR05] Martin Kreuzer and Lorenzo Robbiano. *Computational commutative algebra*. Springer-Verlag, Berlin, 2005.
- [KS99] Aviad Kipnis and Adi Shamir. Cryptanalysis of the HFE public key cryptosystem by relinearization. In *Advances in cryptology—CRYPTO '99 (Santa Barbara, CA)*, volume 1666 of *Lecture Notes in Comput. Sci.*, pages 19–30. Springer, Berlin, 1999.

- [LN86] R. Lidl and H. Niederreiter. *Introduction to Finite Fields and their Applications*. Cambridge University Press, Cambridge, London, 1986.
- [MC01] Tzuong Tsieng Moh and Jiun-Ming Chen. On the Goubin-Courtois Attack on TTM. *Cryptology ePrint Archive, Report 2001/072*, 2001.
- [MCY04] Tzuong Tsieng Moh, Jiun-Ming Chen, and Boyin Yang. Building Instances of TTM Immune to the Goubin-Courtois Attack and the Ding-Schmidt Attack. In *Cryptology ePrint Archive, Report 2004/168*, 2004.
- [MI88] Tsutomu Matsumoto and Hideki Imai. Public quadratic polynomial-tuples for efficient signature-verification and message-encryption. In *Advances in cryptology—EUROCRYPT '88 (Davos, 1988)*, volume 330 of *Lecture Notes in Comput. Sci.*, pages 419–453, Berlin, 1988. Springer.
- [Moh99a] Tzuong Tsieng Moh. The Method of "Relinearization" of Kipnis and Shamir And It's Applications to TTM. 1999.
- [Moh99b] Tzuong Tsieng Moh. On Tame Transformation Method (TTM). In *Midwest Arithmetical Geometry in Cryptography Workshop at University of Illinois*, 1999.
- [Moh99c] Tzuong Tsieng Moh. A public key system with signature and master key functions. *Comm. Algebra*, 27(5):2207–2222, 1999.
- [Moh07] Tzuong Tsieng Moh. Two New Examples of TTM. *Cryptology ePrint Archive, Report 2007/144*, 2007.
- [Nag72] Masayoshi Nagata. *On automorphism group of $k[x, y]$* . Kinokuniya Book-Store Co. Ltd., Tokyo, 1972. Department of Mathematics, Kyoto University, Lectures in Mathematics, No. 5.
- [NES04] Information Society Technologies Programme of the European Commission. *Final report of European project number IST-1999-12324, named New European Schemes for Signatures, Integrity, and Encryption*, 2004.
- [NHL⁺] Xuyun Nie, Lei Hu, Jianyu Li, Crystal Updegrave, and Jintai Ding. Breaking a New Instance of TTM Cryptosystems. In Jianying Zhou, Moti Yung, and Feng Bao, editors, *ACNS*, volume 3989 of *Lecture Notes in Computer Science*, pages 210–225. Springer Berlin / Heidelberg.
- [NJH⁺08] Xuyun Nie, Xin Jiang, Lei Hu, Jintai Ding, and Fengli Zhang. Breaking Instance I of New TTM Cryptosystems. *Communications, Circuits and Systems, 2008. ICCAS 2008. International Conference on*, pages 493–497, May 2008.
- [Pat95] Jacques Patarin. Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocrypt '88. In *Advances in cryptology—CRYPTO '95 (Santa Barbara, CA, 1995)*, volume 963 of *Lecture Notes in Comput. Sci.*, pages 248–261, Berlin, 1995. Springer.
- [Pat96] Jacques Patarin. Hidden Field Equations HFE and Isomorphisms of Polynomials IP: two new Families of Asymmetric Algorithms. pages 33–48. Springer-Verlag, 1996.
- [Pat00] Jacques Patarin. Cryptanalysis of the Matsumoto and Imai public key scheme of Eurocrypt '88. *Des. Codes Cryptogr.*, 20(2):175–209, 2000.

- [PG97] Jacques Patarin and Louis Goubin. Trapdoor one-way permutations and multivariate polynomials. In *ICICS '97: Proceedings of the First International Conference on Information and Communication Security*, pages 356–368, London, UK, 1997. Springer-Verlag.
- [Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.
- [Ste06] Till Stegers. Faugere’s f5 algorithm revisited. Cryptology ePrint Archive, Report 2006/404, 2006.
- [SU03] Ivan P. Shestakov and Ualbai U. Umirbaev. The Nagata automorphism is wild. *Proc. Natl. Acad. Sci. USA*, 100(22):12561–12563 (electronic), 2003.
- [Wol02] Christopher Wolf. ”Hidden Field Equations” (HFE) - Variations and Attacks. Master’s thesis, Universität Ulm, 2002.